When Is GRPO Worth It? A Compute- and Risk-Aware Study of Tiny Language Models for Banking Intent Routing

Aryan Golbaghi, Independent Researcher (aryangolbaghi@gmail.com)

November 2026

Abstract

Group Relative Policy Optimization (GRPO) is widely used to post-train language models with verifiable rewards. It is less clear whether GRPO helps when labels are available, compute is limited, and some errors are much costlier than others. We study this in a CPU-only setting: routing BANKING77 customer messages to eight intents (four designated, by assumption, as high-severity) or to manual review, with the 14M-parameter Pythia language model as a single-token constrained-action policy. We compare TF-IDF/logistic regression, supervised fine-tuning (SFT), cost-weighted SFT, single-step GRPO with binary and cost-based rewards, and rejection-sampling fine-tuning. Every verifier call counts against the label budget, and CPU time is measured. Across 255 runs (five seeds per configuration), SFT outperformed GRPO on the same labelled examples in accuracy, macro-F1, expected cost and critical errors: 0.787 vs. 0.608 accuracy and 0.909 vs. 1.711 expected cost with all 1,106 labels. TF-IDF/logistic regression outperformed both (0.860 accuracy) in under one CPU second. In every GRPO configuration, 77.9%–99.7% of sampled groups had identical rewards and produced no gradient. A cost-based reward reduced GRPO's expected cost (−0.31, 95% CI [−0.44, −0.18]) and critical-error rate (0.265 → 0.204) relative to a binary reward, partly through more abstention, but remained worse than SFT. A minimum-expected-cost decision rule applied to supervised models reduced critical errors more than any training-time cost shaping, and no GRPO configuration beat sending every request to review under the assumed costs. A reduced replication with Pythia-31M (45 runs) left these conclusions unchanged: SFT still beat GRPO at equal labels (0.887 vs. 1.278 expected cost). The costs are hypothetical, the model is tiny, and the shifts are simulated. Within these limits, a linear model with a cost-aware decision layer is the practical default.

1 Introduction

Reinforcement learning with verifiable rewards, and Group Relative Policy Optimization (GRPO) in particular (Shao et al. 2024; Guo et al. 2025), has become a standard tool for post-training language models. GRPO avoids a learned value function by comparing several sampled outputs for the same prompt and pushing probability toward those that score above the group mean. Its appeal is clearest when correct outputs are easy to check but hard to demonstrate, as in mathematics or code.

Many practical uses of small language models are not like that. Routing customer messages to the right team is a classification problem with gold labels, modest compute budgets, and asymmetric stakes: sending a report of a stolen card to a fee-enquiry queue is far worse than confusing two routine questions. For such problems the practical question is whether GRPO earns its cost relative to supervised fine-tuning or a classical model, and whether a reward that encodes the asymmetric stakes reduces the errors that matter.

Three features of this setting complicate a fair answer. First, GRPO's verifier compares sampled actions with gold labels, so GRPO is not label-free. In a small action space each verifier call reveals label information, and a fair comparison must charge those labels to GRPO. Second, GRPO's efficiency depends on groups

containing a mix of rewards. When every sample in a group receives the same reward, the group-relative advantage is zero and the verifier call buys no gradient (He et al. 2026). Third, abstention changes everything. When a request can be sent to manual review at a modest cost, a policy that abstains often can look good on expected cost while automating little. Any claim about cost reductions must therefore be checked against coverage and against the trivial always-review policy.

We address these issues in a deliberately small, fully reproducible study on a single laptop CPU. The task is nine-action routing on eight intents of BANKING77 (Casanueva et al. 2020), four of them designated high-severity by assumption, plus a manual-review action that is also the correct response to out-of-scope requests. The policy is Pythia-14M (Biderman et al. 2023) restricted to nine single-token actions, so that SFT, cost-weighted SFT, GRPO, and rejection-sampling fine-tuning all train the same parameters with different objectives. We ask four questions:

- **RQ1:** Under identical labelled-example budgets, how does GRPO compare with SFT in accuracy, expected operational cost, CPU time, and verifier efficiency?
- **RQ2:** Does a risk-sensitive GRPO reward reduce high-severity errors relative to standard GRPO and cost-weighted SFT?
- **RQ3:** How do group size and the share of homogeneous-reward groups affect performance under a fixed verifier budget?
- **RQ4:** Does any benefit persist under simulated prevalence shifts, synthetic wording perturbations, and unseen out-of-scope intents?

Contributions. The contributions are empirical and limited in scope:

1. A budget-accounted comparison of TF-IDF/logistic regression, SFT, two cost-weighted SFT variants, GRPO with binary and cost-based rewards, and online rejection-sampling fine-tuning on one task and model. Every run reports distinct labels (counting verifier-scored prompts), warm-start data, verifier calls, sampled actions, optimiser steps, forward/backward passes, CPU time, and peak memory.
2. A decision-theoretic evaluation: expected cost under a declared hypothetical cost matrix, critical-error and false-escalation rates, coverage at validation-calibrated targets, and sensitivity to nine cost matrices. Every method is also scored under a Bayes minimum-expected-cost rule, so that training-time and decision-time cost sensitivity can be separated.
3. Measurement of homogeneous-reward groups across group sizes and budgets, with ablations of advantage normalisation, training-time cost scale, warm starting, and rejection sampling, plus a reduced replication of the core comparisons with Pythia-31M.
4. Negative and nuanced findings. In this regime GRPO trails SFT, which trails a linear model. A cost reward helps GRPO relative to a binary reward but does not close the gap. A post-hoc decision rule reduces critical errors more than any training objective.

We do not propose a new algorithm. GRPO-versus-SFT comparisons for intent detection (Feng et al. 2025), cost- and abstention-aware GRPO rewards (Zuo et al. 2026; Wei et al. 2026; Mohamadi et al. 2025), and measurement of advantage collapse (He et al. 2026) all precede this work. Our contribution is to evaluate these ideas together under explicit label, compute, and cost accounting, at a scale where every method can be rerun on commodity hardware.

2 Related Work

GRPO and its lineage. GRPO was introduced in DeepSeekMath (Shao et al. 2024) as a variant of PPO (Schulman et al. 2017) that replaces the learned value function with the mean reward of a group of samples for the same prompt, itself a refinement of likelihood-ratio policy gradients with baselines (Williams 1992). Its use for reinforcement learning with verifiable rewards was popularised by DeepSeek-R1 (Guo et al. 2025). Subsequent work modified the recipe for large-scale reasoning (DAPO, which removes the KL term and filters prompts whose samples are all right or all wrong (Yu et al. 2025)), corrected an optimisation bias related to response length (Liu et al. 2025), identified collapse of advantages when multiple rewards are summed (Liu et al. 2026), and named and measured *advantage collapse*, the failure mode in which every member of a group receives the same reward and the gradient vanishes (He et al. 2026). Low-budget GRPO has been studied for a 1.5B reasoning model on four GPUs (Dang and Ngo 2025). Our model is roughly a hundred times smaller, our episodes are single tokens, and we run on a CPU.

Is reinforcement learning needed? Rejection-sampling fine-tuning (Yuan et al. 2023) and reward-ranked fine-tuning (Dong et al. 2023) train on filtered self-generated samples. Xiong et al. (2025) report that such filtering is competitive with GRPO for mathematical reasoning and attribute much of GRPO's benefit to discarding prompts whose samples are all incorrect. We include an online RFT variant with the same verifier budget.

GRPO for intent detection, abstention, and asymmetric costs. The study closest to ours compares GRPO and SFT for intent detection with a 7B instruction-tuned model and finds SFT equal or slightly better in-domain, with GRPO stronger on unseen or restructured intents (Feng et al. 2025); it has no cost matrix, abstention action, or compute accounting. Ternary rewards that separate abstention from error have been used with GRPO to reduce hallucination (Wei et al. 2026; Mohamadi et al. 2025; Pan et al. 2026), and an asymmetric 2×2 cost-matrix reward has been used for a binary decision in an industrial system, where the authors argue that dividing by the group standard deviation erases cost magnitude and therefore use mean-centring only (Zuo et al. 2026). We extend the cost-reward idea to a nine-action cost matrix with an explicit review action and test the normalisation question directly with an ablation.

Cost-sensitive learning, imbalance, and selective classification. Under a known cost matrix the optimal decision minimises expected cost under the posterior, and cost changes can be emulated by re-weighting or thresholding (Elkan 2001); re-weighting and resampling are standard tools for imbalanced data (He and Garcia 2009; Buda et al. 2018). Abstention has a long history as a reject option (Chow 1970), selective classification with risk–coverage trade-offs (El-Yaniv and Wiener 2010; Geifman and El-Yaniv 2017), and confidence scoring with the maximum softmax probability (Hendrycks and Gimpel 2017). These give strong post-hoc alternatives to training-time cost shaping, and we evaluate every method under them. Intent detection with explicit out-of-scope queries was popularised by CLINC150 (Larson et al. 2019); BANKING77 has no OOS class, so we construct one from held-out intents. Optimising tail risk such as CVaR (Tamar et al. 2015) is a different objective from the expected cost studied here.

Efficiency and baselines. Reporting the computational cost of results alongside accuracy has been advocated as a matter of practice (Schwartz et al. 2019). Parameter-efficient methods such as LoRA (Hu et al. 2021) reduce adaptation cost for large models; at 14M parameters we fine-tune all weights. Linear models over n-

gram features remain strong, inexpensive text classifiers (Wang and Manning 2012; Joulin et al. 2017), and BANKING77 was itself introduced with detectors trainable on a CPU in minutes (Casanueva et al. 2020).

Positioning. No new algorithm is proposed. The comparison of GRPO with SFT for intent detection (Feng et al. 2025), cost- and abstention-aware GRPO rewards (Zuo et al. 2026; Wei et al. 2026; Mohamadi et al. 2025), and measuring homogeneous-reward groups (He et al. 2026) all have precedents. What we add is a joint, budget-accounted evaluation of all of them in one tiny-model, CPU-only setting, including classical baselines and post-hoc decision rules applied uniformly across methods.

3 Problem Formulation

3.1 Data, labels, and actions

Let x be a customer message and $y \in \mathcal{Y} = \{0, \ldots, 8\}$ its label. Labels 0–3 are the designated *critical* intents $\mathcal{Y}_{\text{crit}}$ (`compromised_card`, `lost_or_stolen_card`, `card_payment_not_recognised`, `cash_withdrawal_not_recognised`), labels 4–7 the *routine* intents $\mathcal{Y}_{\text{rout}}$ (`card_arrival`, `card_payment_fee_charged`, `pending_transfer`, `exchange_rate`), and $y = 8$ denotes an out-of-scope (OOS) request drawn from one of the other 69 BANKING77 intents. The action space $\mathcal{A} = \{0, \ldots, 8\}$ contains the eight automatic routes and $a = 8$, `MANUAL_REVIEW`. The correct action for an OOS request is manual review. The severity split is an assumption of this study, not a property of BANKING77.

3.2 Policy

All neural methods share one parameterisation. A fixed prompt `Banking customer message: {x}\nRouting code:` is fed to Pythia-14M, and the final hidden state $h_\theta(x) \in \mathbb{R}^{128}$ is scored against the rows w_a of the language-model output embedding that correspond to nine verified single-token strings " A"…" I":

$$\pi_\theta(a \mid x) = \frac{\exp\left(h_\theta(x)^\top w_a\right)}{\sum_{b \in \mathcal{A}} \exp\left(h_\theta(x)^\top w_b\right)}, \qquad a \in \mathcal{A}. \tag{1}$$

This is exactly the language model's next-token distribution renormalised over the nine permitted tokens (verified numerically against the full LM head; maximum absolute logit difference 5×10^{-6}). All $14,067,712$ parameters are trained; no new head is added.

3.3 Supervised objectives

Standard and weighted SFT minimise

$$\mathcal{L}_{\text{SFT}}(\theta) = -\frac{1}{N} \sum_{i=1}^{N} \omega_{y_i} \log \pi_\theta(y_i \mid x_i), \tag{2}$$

with $\omega \equiv 1$ (SFT), $\omega_y \propto \max_{a \neq y,\, a < 8} C(y, a)$ (cost-proportionate WSFT: 10 for critical, 2 for routine, 3 for OOS) or $\omega_y \propto N/(9N_y)$ (inverse-frequency WSFT), normalised to mean 1 over the training set.

3.4 Decision cost

The hypothetical cost $C(y, a)$ is assigned by the first matching rule:

$$C(y,a) = \begin{cases} c_{\text{rev}} = 0.5 & a = 8 \text{ (manual review, including correct review of OOS)} \\ c_{\text{oos}} = 3 & y = 8,\ a \neq 8 \\ 0 & a = y \\ c_{\text{crit}} = 10 & y \in \mathcal{Y}_{\text{crit}} \text{ (any wrong automatic route)} \\ c_{\text{esc}} = 2 & y \in \mathcal{Y}_{\text{rout}},\ a \in \mathcal{Y}_{\text{crit}} \\ c_{\text{rr}} = 1 & y \in \mathcal{Y}_{\text{rout}},\ a \in \mathcal{Y}_{\text{rout}} \backslash \{y\} \end{cases} \tag{3}$$

The values are unit-free study assumptions. The operational objective is the expected cost of a decision rule δ,

$$\text{EC}(\delta) = \frac{1}{n} \sum_{i=1}^{n} C\left(y_i, \delta(x_i)\right). \tag{4}$$

Sensitivity analyses use $c_{\text{crit}} \in \{5, 10, 20\}$ and $c_{\text{rev}} \in \{0.25, 0.5, 1\}$.

3.5 Decision rules

Every trained model yields probabilities $p(\cdot \mid x)$ over $\mathcal{A}$ and is evaluated under the same three rules: (i) *argmax*, $\delta(x) = \text{argmax}_a p(a \mid x)$; (ii) the *Bayes* minimum-expected-cost rule (Elkan 2001), which treats $p(8 \mid x)$ as the probability of OOS,

$$\delta_{\text{B}}(x) = \arg\min_{a \in \mathcal{A}} \sum_{y \in \mathcal{Y}} p(y \mid x)\, C(y, a); \tag{5}$$

and (iii) a *selective* rule (Chow 1970; Geifman and El-Yaniv 2017) that routes to $\text{argmax}_{a<8} p(a \mid x)$ only when $s(x) = \max_{a<8} p(a \mid x) \geq \tau$ (and $\text{argmax}_a p \neq 8$), otherwise to manual review, with τ fitted on the validation split to reach automation coverage $\kappa \in \{0.7, 0.6, 0.5\}$ of all traffic (an oracle router automates the in-scope 2/3 of test traffic). Coverage is the fraction of requests not sent to manual review.

3.6 Single-step constrained-action GRPO

For a batch of P prompts, the policy before the update, $\pi_{\theta_{\text{old}}}$, is evaluated once (its log-probabilities are detached and frozen) and G actions $a_{i1}, \ldots, a_{iG}$ are sampled per prompt. Each sampled action is scored by the verifier, $r_{ij} = R(y_i, a_{ij})$, with either the *binary* reward $R(y, a) = \mathbb{1}[a = y]$ or the *cost* reward $R(y, a) = -C(y, a)/c_{\text{crit}}$. Group-relative advantages are

$$\widehat{A}_{ij} = \frac{r_{ij} - \bar{r}_i}{\sigma_i + \epsilon}, \qquad \bar{r}_i = \frac{1}{G} \sum_j r_{ij}, \quad \sigma_i = \sqrt{\frac{1}{G} \sum_j (r_{ij} - \bar{r}_i)^2}, \tag{6}$$

with $\epsilon = 10^{-6}$; groups with $\sigma_i < 10^{-8}$ (all rewards identical) are assigned $\widehat{A}_{ij} = 0$ and counted as *homogeneous*. An ablation uses mean-centring only, $\widehat{A}_{ij} = r_{ij} - \bar{r}_i$. With $\rho_{ij}(\theta) = \pi_\theta(a_{ij} \mid x_i)/\pi_{\theta_{\text{old}}}(a_{ij} \mid x_i)$, the policy minimises, for $\mu = 2$ gradient steps per batch,

$$\mathcal{L}_{\mathrm{GRPO}}(\theta) = -\frac{1}{PG}\sum_{i=1}^{P}\sum_{j=1}^{G}\min\Big(\rho_{ij}\widehat{A}_{ij},\ \mathrm{clip}(\rho_{ij}, 1-\varepsilon, 1+\varepsilon)\,\widehat{A}_{ij}\Big) + \beta\,\bar{\mathrm{KL}}, \tag{7}$$

$$\bar{\mathrm{KL}} = \frac{1}{P}\sum_{i=1}^{P}\sum_{a\in\mathcal{A}}\pi_\theta(a \mid x_i)\log\frac{\pi_\theta(a \mid x_i)}{\pi_{\mathrm{ref}}(a \mid x_i)}, \tag{8}$$

with $\varepsilon = 0.2$, $\beta = 0.04$, and π_{ref} the frozen warm-start policy. Because the action is a single token, the KL in Eq. (8) is computed exactly over the nine actions rather than estimated from samples. This is the GRPO objective of Shao et al. (2024) (with the PPO clipping of Schulman et al. (2017)) specialised to a one-step, nine-action episode; it is not multi-token reasoning. The label enters training only through R; unit tests verify that the update follows the reward even when the reward is deliberately decoupled from the label, that homogeneous groups produce zero policy gradient, that $\rho \equiv 1$ at the first inner step, and that clipped samples receive no gradient.

Online rejection-sampling fine-tuning (RFT) (Yuan et al. 2023; Dong et al. 2023) uses the same sampling and verifier budget but minimises $-\frac{1}{PG}\sum_{ij}\mathbb{1}[r_{ij}=1]\log\pi_\theta(a_{ij} \mid x_i)$.

3.7 Budgets

For every run we record separately: the number of distinct labelled examples L whose label was used by any part of training (for GRPO, the union of the warm-start set and every prompt passed to the verifier); the warm-start examples; the number of reward evaluations $V = G\times$ (prompts processed); sampled actions; optimiser steps; forward and backward sequence passes; single-thread CPU seconds (process time, excluding periodic validation monitoring); and peak resident memory. Because GRPO's verifier compares a sampled action with the gold label, each prompt it scores is counted as a labelled example. For a nine-way action space a single binary verifier call reveals whether the sampled action is the label; repeated calls on the same prompt can reveal it fully. GRPO is therefore never treated as label-free.

4 Experimental Methodology

4.1 Data and splits

We use BANKING77 (Casanueva et al. 2020) as released on the Hugging Face Hub (`PolyAI/banking77`; 10,003 training and 3,080 test messages). Test data come exclusively from the official test split: all 320 test messages of the eight selected intents (40 each) and 160 OOS messages sampled from 34 *test-only* OOS intents, giving 480 test requests with equal thirds of critical, routine, and OOS traffic. Training and validation data come exclusively from the official training split: in-scope messages are split 85/15 by intent (stratified), and OOS messages are drawn from 35 *different* OOS intents (238 train and 80 validation after filtering). Exact duplicates (after lower-casing and punctuation removal) and near-duplicates (character-trigram TF-IDF cosine ≥ 0.90) of any test message were removed from the training pool (1 exact and 64 near-duplicates), as were 6 exact duplicates within the pool, and training messages near-duplicating a validation message were removed from training (13). The resulting splits contain 1,106 training (404 critical / 464 routine / 238 OOS), 234 validation, and 480 test requests. Split files, intent lists, counts, and SHA-256 hashes are in the repository manifest. The test split was not used for any design or hyperparameter decision.

For each seed s, the training pool is arranged into a seeded, class-proportional stream; the first n elements form the labelled set of size n. SFT with n examples, TF-IDF with n examples, and GRPO whose verifier has seen n distinct prompts therefore use *the same* labelled examples (the label-matched design). GRPO consumes prompts from the start of the same stream, cycling when the stream is exhausted.

Robustness sets are deterministic synthetic rewrites of the test set: punctuation removal and lower-casing (464/480 messages changed), adjacent-character swaps in about one word in four (480/480), and sentence/clause reordering (141/480; single-clause messages are unchanged). These are controlled perturbations, not natural distribution shift. Prevalence shifts are *simulated* by re-weighting test examples so that critical intents form 5%, 10%, 25%, 50%, or 75% of in-scope traffic while OOS stays at one third.

4.2 Model and hardware

The language model is `EleutherAI/pythia-14m` (Biderman et al. 2023) at Hub revision `cf967c0a9a04383db6f7b1108d86b2962634b4ac` (tokenizer from the same revision): 6 layers, width 128, 14,067,712 parameters, of which the 50,304 × 128 output embedding accounts for most. The published Pythia paper describes the 70M–12B models; the 14M checkpoint was released later with the same training setup. All experiments ran on one laptop CPU (AMD64 Family 25 Model 116, 6 physical / 12 logical cores, 32 GB RAM, Windows 11) with PyTorch 2.13.0 (CPU build), Transformers 5.14.1, and scikit-learn 1.9.0. Each run used one thread; five runs executed concurrently. A batch-16 forward/backward pass took about 0.19 s on one thread.

4.3 Methods and hyperparameters

- **TF-IDF + logistic regression**: word 1–2-gram and character 2–5-gram TF-IDF features (sublinear TF), multinomial logistic regression with $C = 10$; unweighted and inverse-frequency class weights.
- **SFT**: Eq. (2) with $\omega \equiv 1$; AdamW, learning rate 10^{-4}, no weight decay, batch 16, 10 epochs with at least 60 steps, gradient-norm clipping at 1.0, final checkpoint (no early stopping). Labelled-set sizes $n \in \{64, 128, 256, 512, 1024, 1106\}$.
- **WSFT**: as SFT with cost-proportionate or inverse-frequency weights; $n \in \{512, 1106\}$.
- **Warm start**: SFT on the first 64 stream examples (60 steps), shared by all warm-started GRPO and RFT runs of the same seed. Its data and compute are added to those runs' budgets.
- **GRPO**: Eqs. (6)–(8); 16 prompts per batch, $\mu = 2$, $\varepsilon = 0.2$, $\beta = 0.04$, sampling temperature 1, AdamW with learning rate 10^{-5}, gradient clipping at 1.0. Main grid: reward $\in$ {binary, cost} × $G \in \{2, 4, 8\}$ × reward-evaluation budget $V \in \{512, 2048, 8192\}$, plus $V = 32,768$ at $G = 4$. With $G = 4$, $V = 2048$ corresponds to 512 distinct prompts and $V \geq 8192$ to the whole training pool (1,106).
- **Ablations**: mean-centred advantages (cost reward; $V \in \{2048, 8192\}$); RFT ($G = 4$, $V \in \{512, 2048, 8192\}$); GRPO without warm start (from the pretrained model; $G = 4$, $V = 8192$); cost reward trained with $c_{\text{crit}} \in \{5, 20\}$ ($G = 4$, $V = 8192$).

Learning rates were selected in a validation-only pilot with a seed (100) not used in the main study, by final validation expected cost under the argmax rule: SFT from $\{10^{-4}, 3 \times 10^{-4}, 10^{-3}\}$ (both $n = 64$ and $n = 1106$ preferred 10^{-4}) and GRPO, with the binary reward, $G = 4$, $V = 2048$, from $\{3 \times 10^{-6}, 10^{-5}, 3 \times 10^{-5}, 10^{-4}, 3 \times 10^{-4}\}$ (validation cost 1.41, **1.17**, 1.55, 1.78, 2.45; the grid was extended to 3×10^{-6} after 10^{-5} was best at its edge). The cost-reward GRPO variant reuses the binary-reward learning

rate, so it receives no additional tuning. All other settings were fixed in advance. The main study uses five seeds (0–4) for every configuration; the seed controls the labelled-subset stream, initial batch order, and GRPO sampling.

4.4 Evaluation and statistics

All methods are evaluated on the same 480 test requests and the same perturbed and re-weighted variants, under the decision rules of Section 3. We report accuracy (over nine actions, with manual review counting as correct only for OOS), macro-F1 over the nine classes, expected cost (Eq. 4), the critical-error rate (fraction of critical requests routed automatically to a wrong category), critical recall, the false-escalation rate (fraction of routine requests routed to a critical category), manual-review rate, coverage, and the OOS-to-review rate.

Training-seed variability is summarised by the mean, standard deviation, and 95% t-interval across the five seeds. For paired comparisons on the shared test set we compute per-example differences between two configurations with matched seeds and report two 95% bootstrap intervals (10,000 resamples): one resampling test examples of the seed-averaged difference (test-sampling uncertainty only) and one resampling both seeds and test examples (training randomness plus test sampling), with bootstrap p-values Holm-adjusted within each statistic and decision rule (Koehn 2004; Berg-Kirkpatrick et al. 2012; Dror et al. 2018). We also report a Wilcoxon signed-rank test over the five per-seed differences; with five pairs its smallest attainable two-sided p is 0.0625, so it cannot reach the 0.05 level and is shown only as a descriptive check.

5 Results

All numbers are means ± standard deviations over five training seeds on the 480-request test set, unless stated otherwise. TF-IDF/LR trained on the full pool is deterministic, so its standard deviation is zero; with fewer labels its variance comes only from which examples are drawn. “Labels” counts distinct labelled examples, including the 64 warm-start examples for GRPO and RFT. “CPU s” is single-thread training CPU time, including the warm start. In total the 255 main runs consumed 7.3 CPU-hours (123.7 minutes wall-clock with five concurrent workers), plus 0.33 CPU-hours for the validation pilot.

5.1 Accuracy, macro-F1, and expected cost (RQ1)

Table 1 reports test results under the argmax rule. The linear TF-IDF baseline is the strongest method on every headline metric: 0.860 accuracy, 0.877 macro-F1, and expected cost 0.613, trained in under one CPU second. Full-data SFT of Pythia-14M reaches 0.787 ± 0.006 accuracy and expected cost 0.909 ± 0.064 after 332 CPU seconds. The paired difference against TF-IDF is +0.297 in expected cost (95% CI with seed and test resampling [+0.148, +0.449]; Holm-adjusted $p = 0.004$).

GRPO improves on its warm start, but it stays far below SFT trained on the same labels. With 512 distinct labels, binary-reward GRPO ($G = 4$, $V = 2048$) reaches 0.594 ± 0.036 accuracy and expected cost 1.706 ± 0.241. SFT on exactly the same 512 examples reaches 0.745 ± 0.027 and 1.097 ± 0.073, a cost difference of +0.609 [+0.346, +0.892]. With access to every training label ($V = 8192$), binary GRPO reaches 0.608 accuracy and cost 1.711, against 0.787 and 0.909 for SFT. Quadrupling the verifier budget to $V = 32,768$ (560 CPU seconds, more than full SFT) raises cost-reward GRPO only to 0.645 accuracy and 1.357 cost,

which is still worse than SFT with 512 labels. Fig. 1 shows the same pattern against labelled-set size: every GRPO configuration lies below the SFT curve, and every SFT point lies below the TF-IDF curve.

Table 1. Test results, argmax decision rule (mean ± SD over 5 seeds). Labels include the 64 warm-start examples for GRPO and RFT; CPU s includes the warm start.

Method	Labels	Verifier calls	CPU s	Accuracy	Macro-F1	Expected cost	Critical-error rate	False-escalation rate	Manual-review rate
TFIDF-LR n=1106	1106	0	1	0.860 ± 0.000	0.877 ± 0.000	0.613 ± 0.000	0.050 ± 0.000	0.019 ± 0.000	0.242 ± 0.000
TFIDF-LR-bal n=1106	1106	0	1	0.854 ± 0.000	0.871 ± 0.000	0.648 ± 0.000	0.050 ± 0.000	0.025 ± 0.000	0.217 ± 0.000
SFT n=64	64	0	25	0.558 ± 0.042	0.574 ± 0.044	1.855 ± 0.331	0.305 ± 0.075	0.081 ± 0.028	0.180 ± 0.092
SFT n=512	512	0	145	0.745 ± 0.027	0.766 ± 0.028	1.097 ± 0.073	0.146 ± 0.010	0.041 ± 0.016	0.205 ± 0.042
SFT n=1106	1106	0	332	0.787 ± 0.006	0.807 ± 0.005	0.909 ± 0.064	0.113 ± 0.010	0.021 ± 0.008	0.232 ± 0.029
WSFT-cost n=512	512	0	138	0.730 ± 0.022	0.758 ± 0.017	1.088 ± 0.078	0.126 ± 0.016	0.051 ± 0.017	0.182 ± 0.038
WSFT-cost n=1106	1106	0	326	0.779 ± 0.021	0.799 ± 0.018	0.956 ± 0.110	0.121 ± 0.030	0.034 ± 0.013	0.224 ± 0.041
WSFT-balanced n=1106	1106	0	327	0.789 ± 0.023	0.811 ± 0.016	0.870 ± 0.128	0.098 ± 0.016	0.031 ± 0.006	0.227 ± 0.059
GRPO-bin G=4 R=2048	512	2048	75	0.594 ± 0.036	0.615 ± 0.045	1.706 ± 0.241	0.269 ± 0.053	0.079 ± 0.013	0.174 ± 0.079
GRPO-cost G=4 R=2048	512	2048	66	0.602 ± 0.036	0.616 ± 0.046	1.557 ± 0.304	0.240 ± 0.062	0.062 ± 0.031	0.227 ± 0.105
GRPO-bin G=4 R=8192	1106	8192	170	0.608 ± 0.036	0.632 ± 0.041	1.711 ± 0.283	0.265 ± 0.072	0.062 ± 0.032	0.137 ± 0.072
GRPO-cost G=4 R=8192	1106	8192	167	0.637 ± 0.030	0.656 ± 0.041	1.405 ± 0.341	0.204 ± 0.084	0.040 ± 0.011	0.220 ± 0.080
GRPO-bin G=8 R=8192	1024	8192	104	0.638 ± 0.028	0.657 ± 0.035	1.528 ± 0.228	0.240 ± 0.057	0.040 ± 0.017	0.200 ± 0.075
GRPO-cost G=8 R=8192	1024	8192	104	0.644 ± 0.031	0.658 ± 0.046	1.156 ± 0.248	0.144 ± 0.044	0.040 ± 0.016	0.290 ± 0.125
GRPO-bin G=4 R=32768	1106	32768	557	0.637 ± 0.041	0.672 ± 0.035	1.541 ± 0.253	0.220 ± 0.055	0.044 ± 0.022	0.132 ± 0.064
GRPO-cost G=4 R=32768	1106	32768	560	0.645 ± 0.040	0.674 ± 0.038	1.357 ± 0.300	0.181 ± 0.066	0.042 ± 0.016	0.193 ± 0.071
RFT G=4 R=8192	1106	8192	165	0.671 ± 0.031	0.698 ± 0.037	1.319 ± 0.307	0.182 ± 0.078	0.029 ± 0.022	0.193 ± 0.070

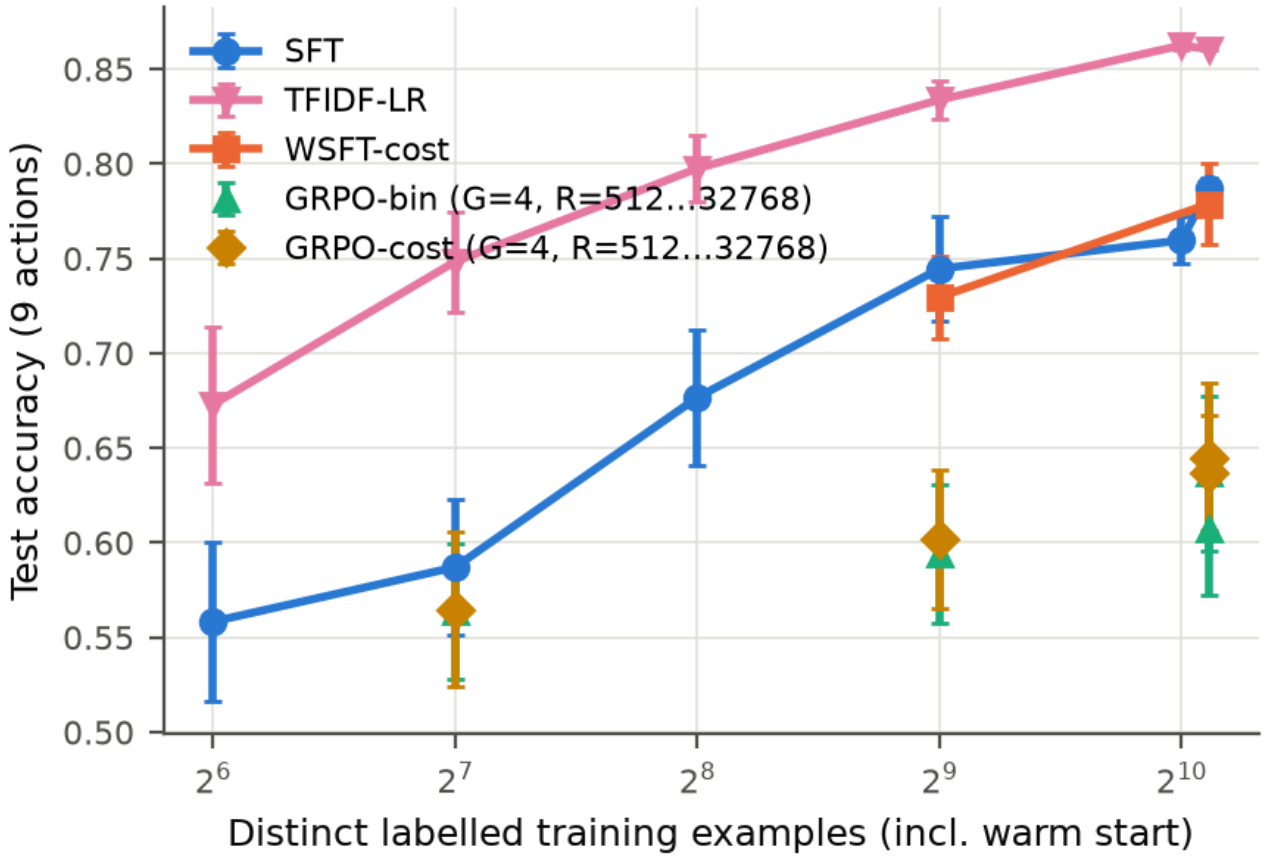


Figure 1. Test accuracy versus distinct labelled examples. GRPO points use $G = 4$ and $V \in \{512, 2048, 8192, 32768\}$.

The label-efficiency gap is also visible within the supervised methods (Table 2). TF-IDF/LR with 256 labels (0.797 accuracy, cost 0.847) already matches full-data SFT with 1,106 labels (0.787, 0.909).

Table 2. Label-efficiency curves of SFT and TF-IDF/LR on the nested labelled subsets shared by all methods (argmax rule).

Labelled examples n	SFT accuracy	SFT expected cost	TF-IDF/LR accuracy	TF-IDF/LR expected cost
64	0.558 ± 0.042	1.855 ± 0.331	0.673 ± 0.041	1.256 ± 0.217
128	0.587 ± 0.036	1.768 ± 0.276	0.748 ± 0.026	1.064 ± 0.166
256	0.677 ± 0.036	1.271 ± 0.345	0.797 ± 0.017	0.847 ± 0.064
512	0.745 ± 0.027	1.097 ± 0.073	0.834 ± 0.010	0.700 ± 0.023
1024	0.760 ± 0.012	1.030 ± 0.107	0.863 ± 0.003	0.604 ± 0.026
1106	0.787 ± 0.006	0.909 ± 0.064	0.860 ± 0.000	0.613 ± 0.000

5.2 Compute and verifier budgets (RQ1, RQ3)

Table 3 and Fig. 2 itemise the budget of representative runs. GRPO is cheap per update: a $V = 8192$, $G = 4$ run took 167 CPU seconds, half the cost of full SFT, because each prompt needs only one sampling forward pass, one reference forward pass, and $\mu = 2$ backward passes. Cheap updates did not translate into performance. Every SFT run with at least 512 labels reached validation accuracy 0.70, on average after 36–41 CPU seconds. Cost-reward GRPO at $V = 8192$ took 76 seconds on average to reach it, including the warm

start. Only two of five cost-reward runs at $V = 32,768$ ever reached 0.80, against all five full-data SFT runs. Peak resident memory was below 1.04 GB for every run.

Table 3. Budget accounting (seed means). Forward/backward sequences count single-sequence passes of the language model. The last column is the CPU time at which periodic validation accuracy first reached 0.70 (mean over the runs that reached it).

Method	Labels (warm start)	Verifier calls	Sampled actions	Optimiser steps	Forward seqs	Backward seqs	CPU s	Peak RSS (MB)	CPU s to val. acc. ≥ 0.70 (runs reaching it)
TFIDF-LR n=1106	1106 (0)	0	0	0	0	0	0.9	351	n/a
SFT n=64	64 (0)	0	0	60	960	960	25.4	835	— (0/5)
SFT n=512	512 (0)	0	0	320	5120	5120	145.0	832	38 (5/5)
SFT n=1106	1106 (0)	0	0	700	11060	11060	332.2	837	36 (5/5)
WSFT-cost n=1106	1106 (0)	0	0	700	11060	11060	325.7	858	41 (5/5)
GRPO-cost G=2 R=8192	1106 (64)	8192	8192	572	17344	9152	296.3	990	139 (4/5)
GRPO-cost G=4 R=2048	512 (64)	2048	2048	124	3008	1984	65.8	966	63 (2/5)
GRPO-cost G=4 R=8192	1106 (64)	8192	8192	316	9152	5056	167.2	978	76 (5/5)
GRPO-cost G=8 R=8192	1024 (64)	8192	8192	188	5056	3008	104.4	976	70 (5/5)
GRPO-cost G=4 R=32768	1106 (64)	32768	32768	1084	33728	17344	559.8	982	74 (5/5)
RFT G=4 R=8192	1106 (64)	8192	8192	316	9152	5056	164.9	985	70 (5/5)

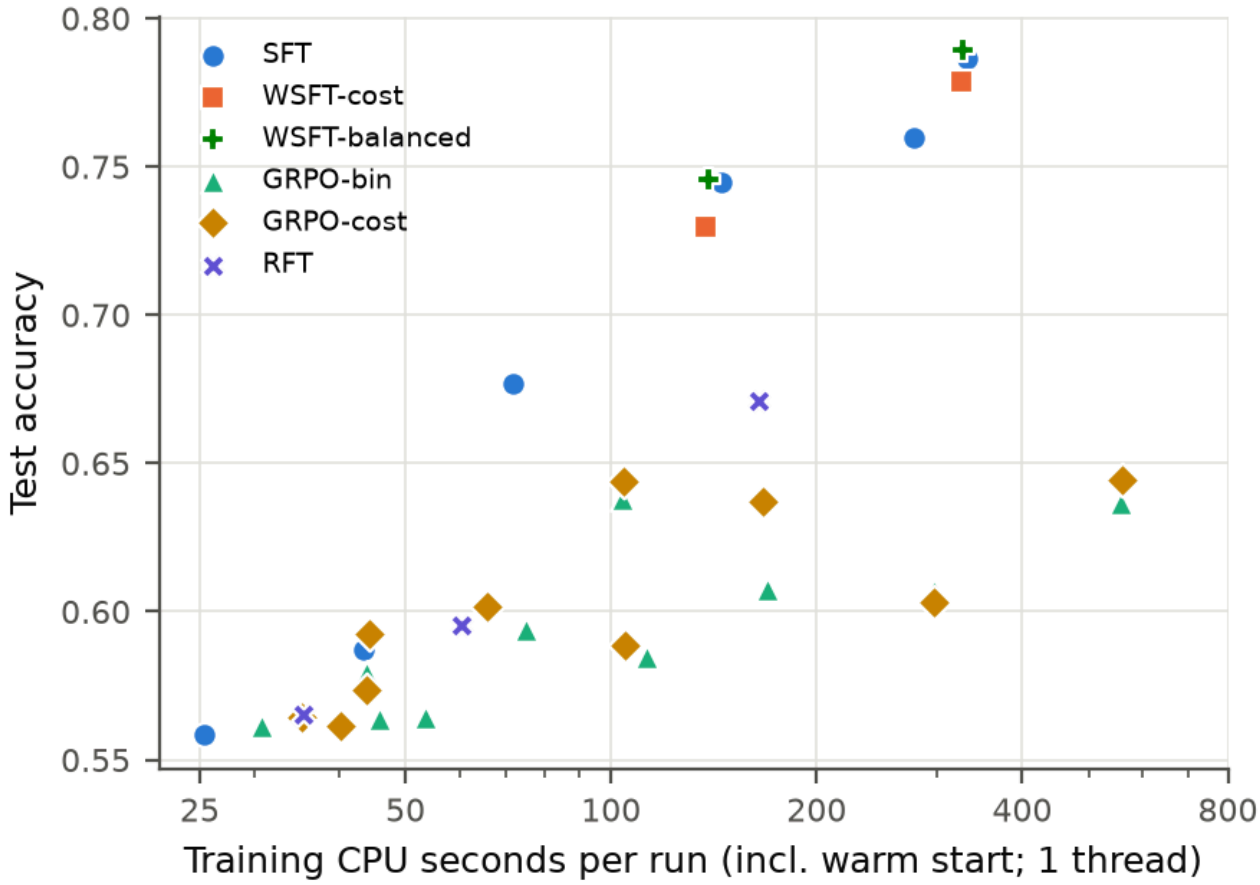


Figure 2. Single-thread training CPU time versus test accuracy (seed means); TF-IDF/LR (under 1 CPU second) is omitted.

Fig. 3 and Table 4 show performance against the verifier budget. Expected cost falls slowly with V. For cost-reward GRPO with $G = 4$ it moves from 1.756 ($V = 512$) to 1.557 (2,048), 1.405 (8,192), and 1.357 (32,768), against 1.855 for the warm start. Per thousand verifier calls, the largest gain was about 0.2 cost units at the smallest budgets, falling to about 0.015 at $V = 32,768$.

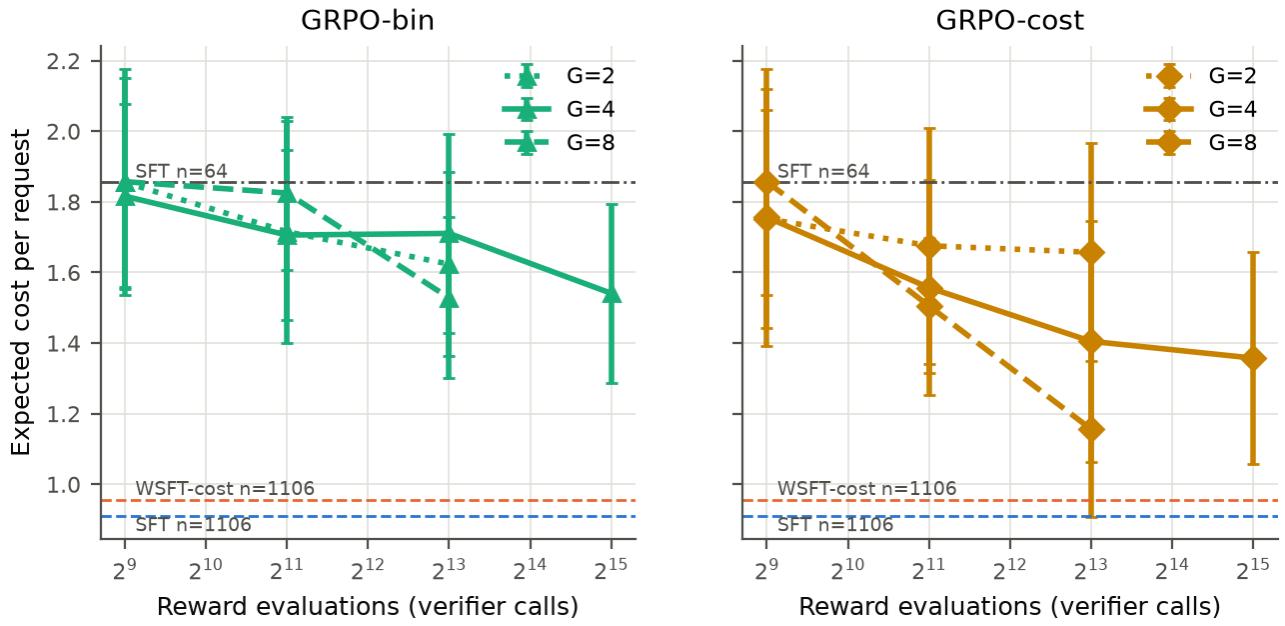


Figure 3. Expected cost versus reward-evaluation budget for each group size; horizontal lines mark the warm start, SFT, and cost-weighted SFT.

5.3 Group size and homogeneous-reward groups (RQ3)

Table 4. GRPO grid, argmax rule. Homogeneous groups is the fraction of sampled groups whose rewards were all identical and which therefore contributed no policy gradient.

Reward	G	V	Distinct labels	Homogeneous groups	CPU s	Accuracy	Expected cost	Critical-error rate
bin	2	512	256	0.939 ± 0.014	54	0.564 ± 0.042	1.855 ± 0.297	0.300 ± 0.069
bin	4	512	128	0.927 ± 0.025	46	0.564 ± 0.036	1.816 ± 0.262	0.294 ± 0.056
bin	8	512	64	0.997 ± 0.007	31	0.561 ± 0.042	1.857 ± 0.320	0.307 ± 0.073
bin	2	2048	1024	0.931 ± 0.015	113	0.585 ± 0.046	1.715 ± 0.315	0.264 ± 0.064
bin	4	2048	512	0.883 ± 0.018	75	0.594 ± 0.036	1.706 ± 0.241	0.269 ± 0.053
bin	8	2048	256	0.841 ± 0.032	44	0.579 ± 0.038	1.825 ± 0.217	0.300 ± 0.045
bin	2	8192	1106	0.946 ± 0.015	297	0.607 ± 0.037	1.624 ± 0.261	0.249 ± 0.052
bin	4	8192	1106	0.894 ± 0.021	170	0.608 ± 0.036	1.711 ± 0.283	0.265 ± 0.072
bin	8	8192	1024	0.827 ± 0.034	104	0.638 ± 0.028	1.528 ± 0.228	0.240 ± 0.057
bin	4	32768	1106	0.922 ± 0.015	557	0.637 ± 0.041	1.541 ± 0.253	0.220 ± 0.055
cost	2	512	256	0.931 ± 0.019	44	0.573 ± 0.039	1.752 ± 0.308	0.281 ± 0.064

Reward	G	V	Distinct labels	Homogeneous groups	CPU s	Accuracy	Expected cost	Critical-error rate
cost	4	512	128	0.917 ± 0.026	35	0.565 ± 0.041	1.756 ± 0.364	0.284 ± 0.075
cost	8	512	64	0.997 ± 0.007	40	0.561 ± 0.042	1.857 ± 0.320	0.307 ± 0.073
cost	2	2048	1024	0.912 ± 0.022	105	0.589 ± 0.056	1.676 ± 0.334	0.258 ± 0.070
cost	4	2048	512	0.850 ± 0.041	66	0.602 ± 0.036	1.557 ± 0.304	0.240 ± 0.062
cost	8	2048	256	0.794 ± 0.020	44	0.593 ± 0.031	1.505 ± 0.191	0.231 ± 0.047
cost	2	8192	1106	0.932 ± 0.022	296	0.603 ± 0.048	1.657 ± 0.308	0.256 ± 0.058
cost	4	8192	1106	0.858 ± 0.025	167	0.637 ± 0.030	1.405 ± 0.341	0.204 ± 0.084
cost	8	8192	1024	0.779 ± 0.018	104	0.644 ± 0.031	1.156 ± 0.248	0.144 ± 0.044
cost	4	32768	1106	0.906 ± 0.021	560	0.645 ± 0.040	1.357 ± 0.300	0.181 ± 0.066

Most sampled groups carried no learning signal (Table 4, Fig. 4). The homogeneous fraction ranged from 0.779 to 0.997. It was lower for the cost reward than for the binary reward, because two different wrong actions can receive different costs, and it fell as G grew: 0.946, 0.894, and 0.827 for binary GRPO at $G = 2, 4, 8$ with $V = 8192$. The extreme case is $G = 8$, $V = 512$. Its 64 prompts are exactly the warm-start examples, which the warm-start policy had already fitted, and 99.7% of its groups were homogeneous, so the policy did not change measurably. During training the homogeneous fraction rose sharply when the prompt stream wrapped around to these memorised examples (Fig. 4, right).

At a fixed verifier budget (Fig. 5), larger groups mean fewer distinct prompts but more informative groups. With the cost reward and $V = 8192$, expected cost improved from 1.657 ($G = 2$) to 1.405 ($G = 4$) and 1.156 ($G = 8$, with 1,024 distinct prompts instead of 1,106). For the binary reward the ordering was not monotone (1.624, 1.711, 1.528). After Holm correction none of the group-size contrasts was significant: cost-reward $G = 4$ vs. $G = 2$ gave −0.252 [−0.430, −0.074], adjusted $p = 0.091$. The direction suggests that more samples per prompt help when most groups are otherwise homogeneous. This is the advantage-collapse mechanism of He et al. (2026). The evidence for this direction is not conclusive at five seeds.

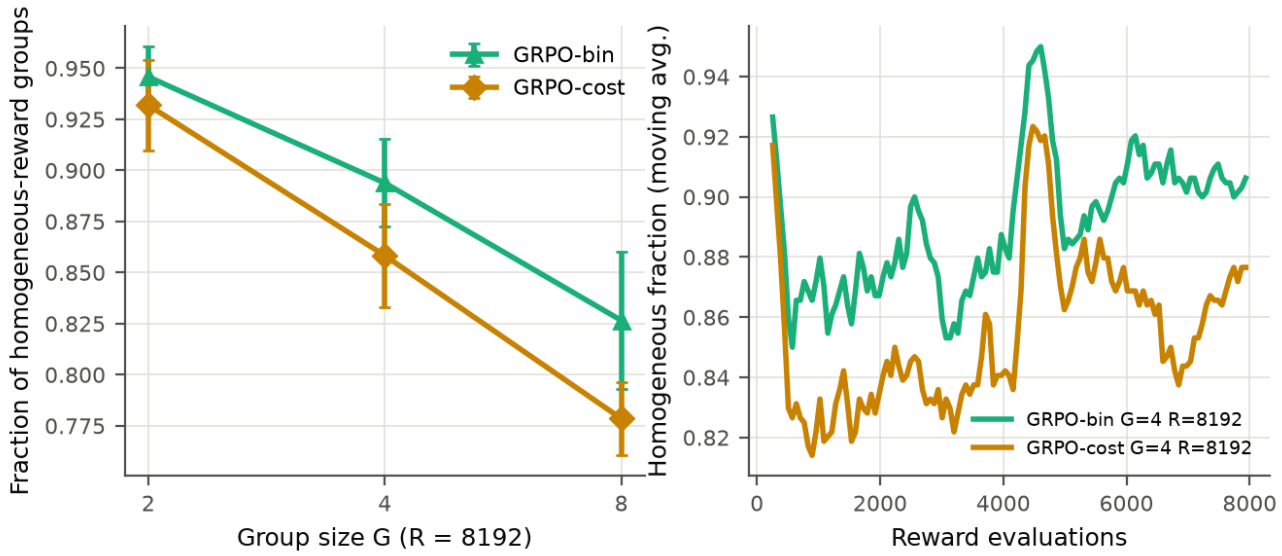


Figure 4. Fraction of homogeneous-reward groups by group size (left) and during training (right).

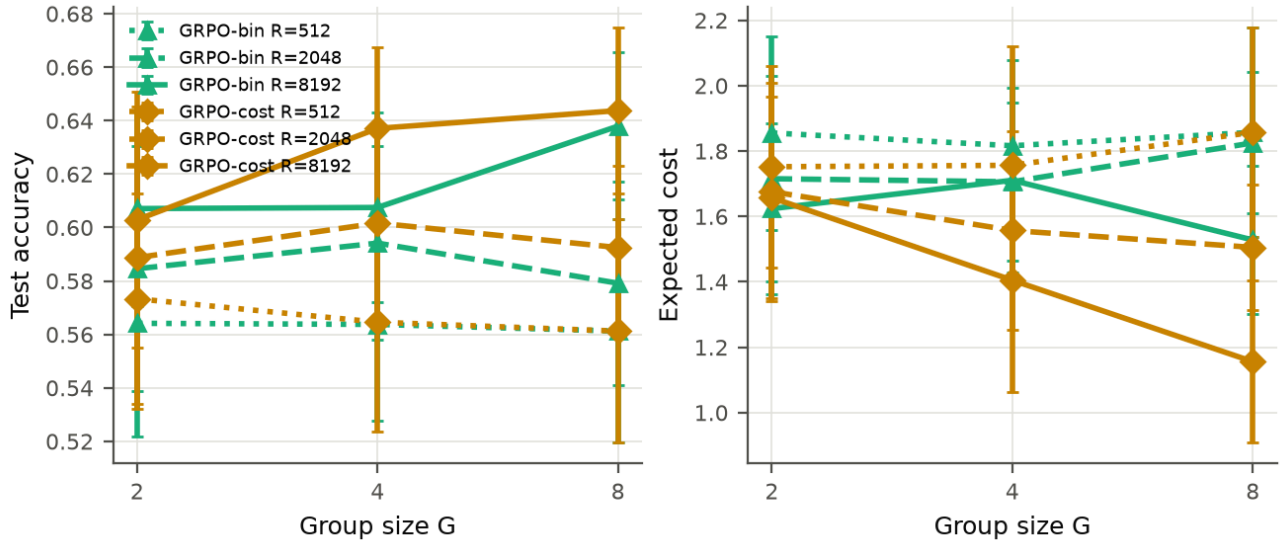


Figure 5. Effect of group size at fixed verifier budget.

5.4 Critical errors and the risk-sensitive reward (RQ2)

Replacing the binary reward by the negative decision cost lowered expected cost and critical errors compared with binary GRPO at the same budget (Tables 1 and 5). At $G = 4$, $V = 8192$, cost-reward GRPO had expected cost 1.405 vs. 1.711, a difference of −0.306 [−0.436, −0.180] with Holm-adjusted $p = 0.004$. Its critical-error rate was 0.204 vs. 0.265, a difference of −0.061 [−0.102, −0.022] with $p = 0.038$. Accuracy did not change significantly (error-rate difference −0.030, adjusted $p = 0.153$). At $G = 8$ the reductions were larger in magnitude (cost −0.372, critical errors −0.096) but did not survive correction ($p = 0.128$ and 0.051). At $G = 2$ there was no difference (+0.034 [−0.122, +0.158]). The cost reward also sent more out-of-scope requests to review (0.417 vs. 0.287 at $G = 4$, $V = 8192$). The critical-error reductions therefore come partly from more abstention, not only from better discrimination between intents.

Cost-sensitive training did not close the gap to supervised learning. Cost-reward GRPO at $V = 8192$ made more critical errors than plain SFT on the same labels (0.204 vs. 0.113; +0.091 [+0.014, +0.172], adjusted $p = 0.31$). Its expected cost was higher by +0.496 [+0.186, +0.811], adjusted $p = 0.033$. Cost-weighted SFT did not reduce critical errors relative to SFT either (0.121 vs. 0.113; cost difference +0.047 [−0.090, +0.186]). Among the supervised variants, inverse-frequency weighting gave the lowest critical-error rate (0.098) and argmax cost (0.870), but the difference is within seed variability.

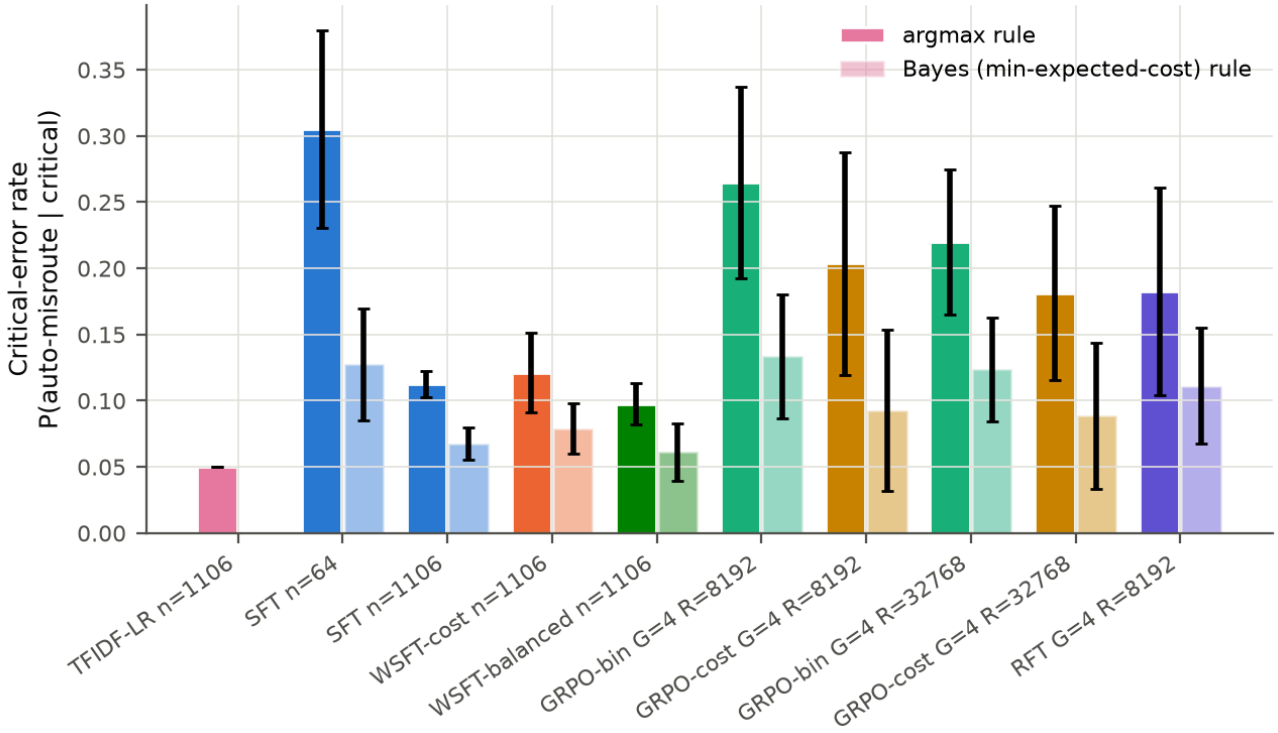


Figure 6. Critical-error rate by method under the argmax rule (solid) and the Bayes minimum-expected-cost rule (faded).

The largest reduction in critical errors came from the decision rule, not the training objective (Fig. 6; confusion matrices in Fig. 7). Applying the Bayes rule (Eq. 5) to the same SFT probabilities lowered the critical-error rate from 0.113 to 0.068 and expected cost from 0.909 to 0.689 (Table 5). Applied to TF-IDF/LR it gave zero critical errors and expected cost 0.332, at the price of sending 61.5% of requests to review.

Table 5. Test results under the Bayes minimum-expected-cost rule applied to each model's own probabilities.

Method	Labels	Verifier calls	CPU s	Accuracy	Macro-F1	Expected cost	Critical-error rate	False-escalation rate	Manual-review rate
TFIDF-LR n=1106	1106	0	1	0.702 ± 0.000	0.695 ± 0.000	0.332 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.615 ± 0.000
TFIDF-LR-bal n=1106	1106	0	1	0.723 ± 0.000	0.727 ± 0.000	0.322 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.594 ± 0.000
SFT n=64	64	0	25	0.592 ± 0.032	0.592 ± 0.049	1.071 ± 0.193	0.128 ± 0.042	0.054 ± 0.021	0.410 ± 0.101
SFT n=512	512	0	145	0.744 ± 0.021	0.766 ± 0.031	0.729 ± 0.068	0.062 ± 0.017	0.022 ± 0.010	0.335 ± 0.050

Method	Labels	Verifier calls	CPU s	Accuracy	Macro-F1	Expected cost	Critical-error rate	False-escalation rate	Manual-review rate
SFT n=1106	1106	0	332	0.798 ± 0.010	0.818 ± 0.011	0.689 ± 0.060	0.068 ± 0.012	0.013 ± 0.004	0.306 ± 0.028
WSFT-cost n=512	512	0	138	0.755 ± 0.014	0.782 ± 0.011	0.747 ± 0.032	0.059 ± 0.007	0.032 ± 0.012	0.292 ± 0.025
WSFT-cost n=1106	1106	0	326	0.795 ± 0.015	0.814 ± 0.014	0.732 ± 0.081	0.079 ± 0.019	0.021 ± 0.016	0.302 ± 0.033
WSFT-balanced n=1106	1106	0	327	0.793 ± 0.015	0.816 ± 0.010	0.690 ± 0.133	0.061 ± 0.022	0.024 ± 0.007	0.297 ± 0.064
GRPO-bin G=4 R=2048	512	2048	75	0.623 ± 0.030	0.630 ± 0.053	1.066 ± 0.225	0.128 ± 0.048	0.046 ± 0.021	0.365 ± 0.108
GRPO-cost G=4 R=2048	512	2048	66	0.607 ± 0.032	0.612 ± 0.053	1.015 ± 0.178	0.113 ± 0.036	0.040 ± 0.032	0.397 ± 0.096
GRPO-bin G=4 R=8192	1106	8192	170	0.639 ± 0.026	0.655 ± 0.039	1.117 ± 0.194	0.134 ± 0.047	0.044 ± 0.025	0.310 ± 0.084
GRPO-cost G=4 R=8192	1106	8192	167	0.645 ± 0.034	0.659 ± 0.042	0.937 ± 0.257	0.092 ± 0.061	0.030 ± 0.014	0.360 ± 0.094
GRPO-bin G=8 R=8192	1024	8192	104	0.646 ± 0.024	0.660 ± 0.039	1.040 ± 0.198	0.124 ± 0.043	0.024 ± 0.021	0.344 ± 0.091
GRPO-cost G=8 R=8192	1024	8192	104	0.645 ± 0.037	0.654 ± 0.053	0.857 ± 0.183	0.077 ± 0.040	0.026 ± 0.018	0.398 ± 0.120
GRPO-bin G=4 R=32768	1106	32768	557	0.653 ± 0.037	0.680 ± 0.032	1.090 ± 0.215	0.124 ± 0.039	0.029 ± 0.021	0.284 ± 0.063
GRPO-cost G=4 R=32768	1106	32768	560	0.660 ± 0.031	0.684 ± 0.022	0.941 ± 0.235	0.089 ± 0.055	0.028 ± 0.014	0.324 ± 0.072
RFT G=4 R=8192	1106	8192	165	0.675 ± 0.029	0.701 ± 0.040	1.012 ± 0.184	0.111 ± 0.044	0.021 ± 0.021	0.291 ± 0.068

GRPO's critical errors were concentrated in the two smallest training classes (Fig. 7): `compromised_card` (71 training examples) and `lost_or_stolen_card` (64) were routed correctly in 45.5% and 58.5% of cases by binary GRPO ($G = 4$, $V = 8192$, pooled over seeds), against 78.5% and 87.5% for SFT. Cost-reward GRPO sent 22.5% and 24.5% of these two classes to review instead (binary GRPO: 11.5% and 8.5%).

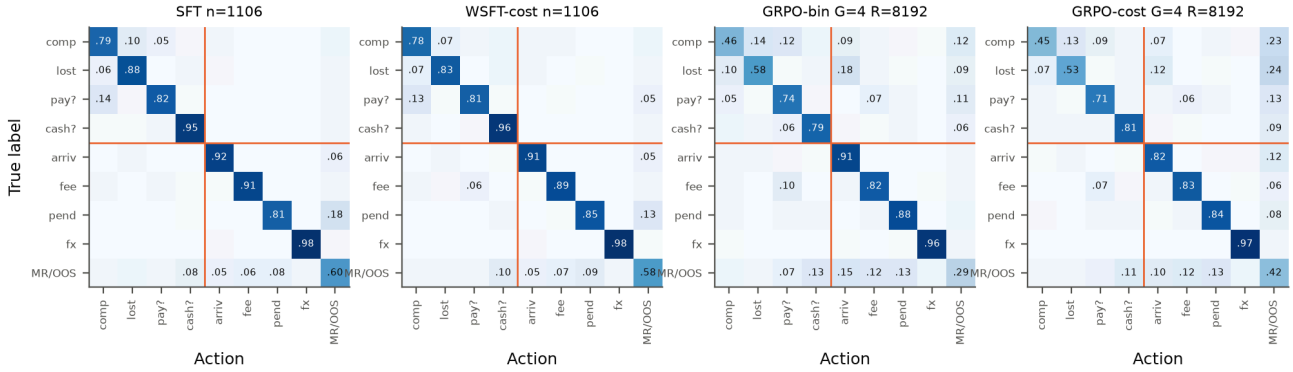


Figure 7. Pooled row-normalised confusion matrices (argmax rule).

5.5 Coverage, cost, and the always-review baseline

Under the hypothetical cost matrix, sending every request to manual review costs exactly $c_{\text{rev}} = 0.5$ per request. Every method's argmax rule is *worse* than this trivial policy: TF-IDF/LR at 0.613, SFT at 0.909, and the best GRPO configuration at 1.156. A model is useful under these assumptions only when combined with abstention. Table 6 and Fig. 8 compare methods at validation-calibrated coverage targets. At a target of 0.5, TF-IDF/LR costs 0.308 with no critical errors, SFT costs 0.417 (critical-error rate 0.021), and the best GRPO configurations cost 0.679–0.685 (0.046–0.049), which is still above the always-review cost. No GRPO configuration in this study beat always-review under the default costs.

Table 6. Results at validation-calibrated automation-coverage targets (fractions of all traffic; an oracle router would automate 2/3 of the test requests).

Method	Target 0.6: coverage	cost	critical-error	Target 0.5: coverage	cost	critical-error
TFIDF-LR n=1106	0.59 ± 0.00	0.349 ± 0.000	0.013 ± 0.000	0.46 ± 0.00	0.308 ± 0.000	0.000 ± 0.000
SFT n=1106	0.62 ± 0.03	0.551 ± 0.096	0.045 ± 0.023	0.48 ± 0.05	0.417 ± 0.036	0.021 ± 0.014
WSFT-cost n=1106	0.61 ± 0.03	0.600 ± 0.144	0.059 ± 0.034	0.49 ± 0.01	0.466 ± 0.079	0.029 ± 0.021
WSFT-balanced n=1106	0.59 ± 0.03	0.496 ± 0.019	0.033 ± 0.010	0.46 ± 0.02	0.421 ± 0.017	0.019 ± 0.008
GRPO-bin G=4 R=8192	0.62 ± 0.02	0.986 ± 0.098	0.114 ± 0.024	0.50 ± 0.03	0.779 ± 0.047	0.074 ± 0.012
GRPO-cost G=4 R=8192	0.60 ± 0.01	0.872 ± 0.097	0.082 ± 0.024	0.50 ± 0.03	0.685 ± 0.085	0.046 ± 0.025
GRPO-cost G=8 R=8192	0.60 ± 0.04	0.852 ± 0.031	0.084 ± 0.016	0.47 ± 0.02	0.679 ± 0.052	0.049 ± 0.016
RFT G=4 R=8192	0.59 ± 0.01	0.780 ± 0.152	0.067 ± 0.044	0.47 ± 0.02	0.628 ± 0.083	0.037 ± 0.030

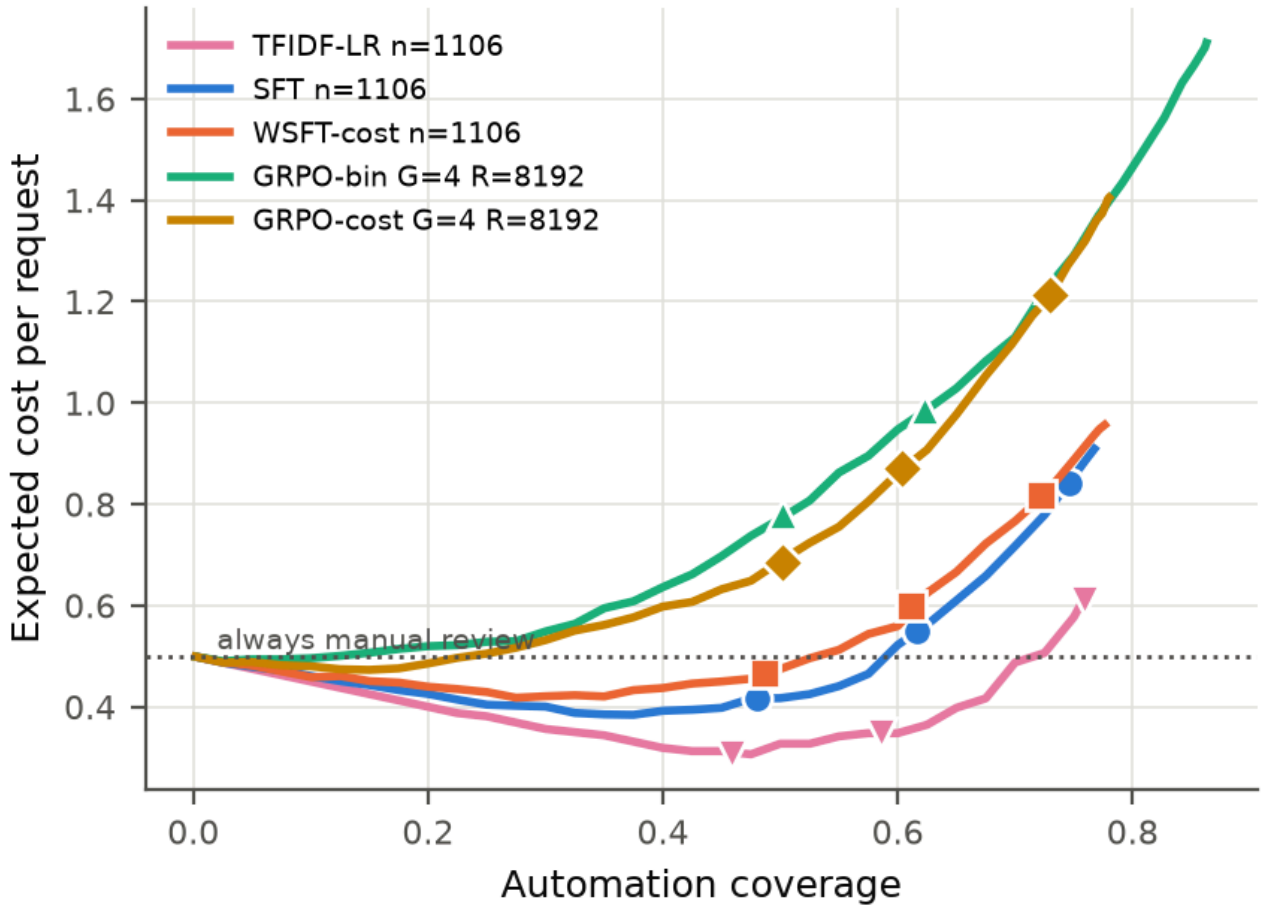


Figure 8. Cost–coverage curves (lines) and validation-calibrated operating points (markers); dotted line: always manual review.

5.6 Simulated prevalence shift, wording perturbations, and out-of-scope requests (RQ4)

The method ranking held under every simulated critical-class prevalence (Table 7, Fig. 9). Expected cost grows with the critical share for all methods, and the gap between cost-reward and binary GRPO widens as critical requests become more common. At a critical share of 0.75, the costs are 1.736 vs. 2.134 under argmax. This follows mechanically from the lower critical-error rate of the cost-reward policy and involves no new data.

Table 7. Expected cost under simulated critical-class prevalence (share of in-scope traffic; the unweighted test set corresponds to 0.50). Each cell gives argmax / Bayes rule.

Method	0.05	0.10	0.25	0.50	0.75
TFIDF-LR n=1106	0.472 / 0.286	0.487 / 0.291	0.534 / 0.307	0.613 / 0.332	0.691 / 0.358
SFT n=1106	0.592 / 0.494	0.627 / 0.515	0.733 / 0.580	0.909 / 0.689	1.085 / 0.798
WSFT-cost n=1106	0.617 / 0.510	0.655 / 0.534	0.768 / 0.609	0.956 / 0.732	1.145 / 0.856
WSFT-balanced n=1106	0.599 / 0.516	0.629 / 0.536	0.719 / 0.594	0.870 / 0.690	1.020 / 0.787
GRPO-bin G=4 R=8192	0.948 / 0.714	1.033 / 0.758	1.287 / 0.893	1.711 / 1.117	2.134 / 1.342
GRPO-cost G=4 R=8192	0.808 / 0.645	0.875 / 0.677	1.074 / 0.775	1.405 / 0.937	1.736 / 1.100

Method	0.05	0.10	0.25	0.50	0.75
RFT G=4 R=8192	0.783 / 0.670	0.842 / 0.708	1.021 / 0.822	1.319 / 1.012	1.616 / 1.202

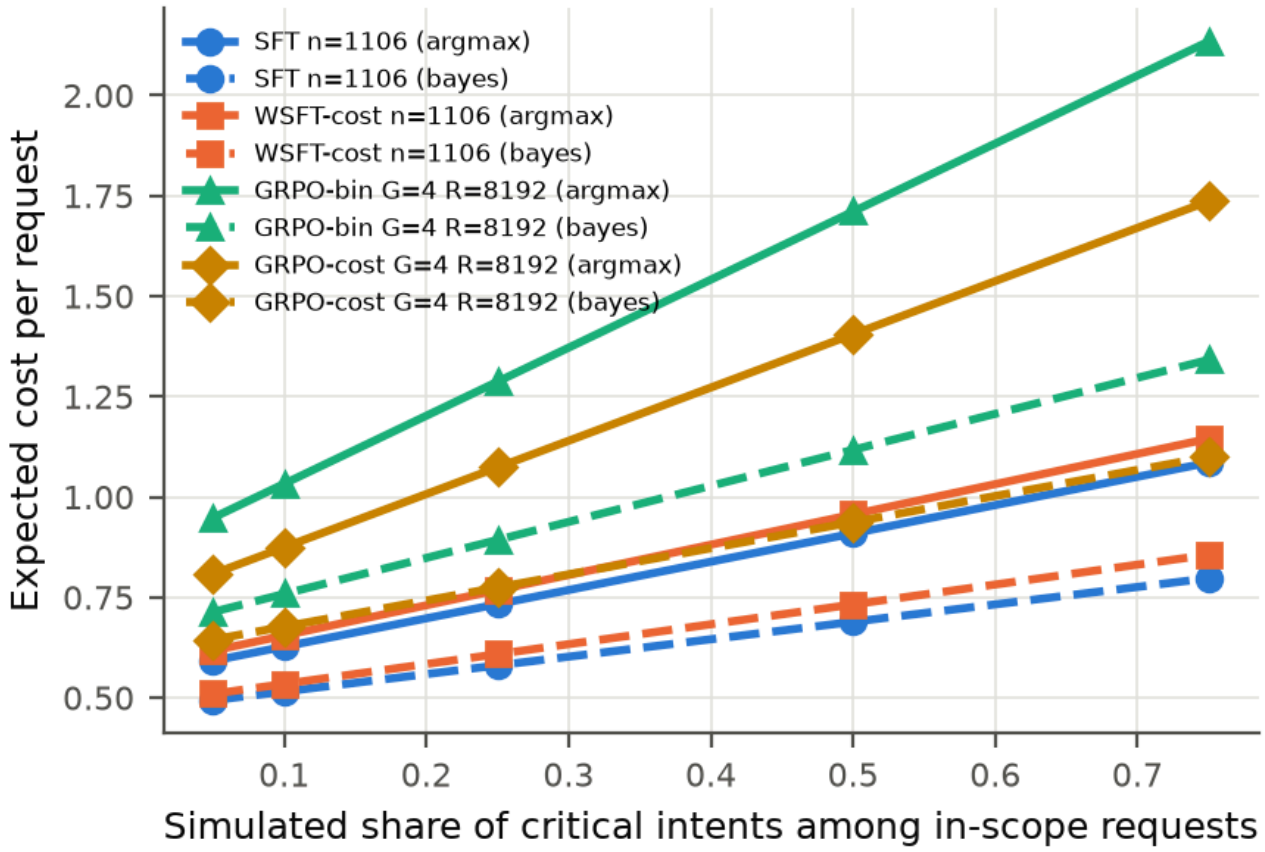


Figure 9. Expected cost under simulated prevalence shift.

Synthetic typos were the most damaging perturbation for every Pythia-based model (Table 8, Fig. 10). SFT lost 9.6 accuracy points (0.787 → 0.691) and GRPO configurations lost 7–9 points, while TF-IDF/LR, whose character n-grams are robust to character swaps, lost 1.2 points. Punctuation removal and clause reordering changed accuracy by at most about two points for any method. GRPO's smaller absolute typo drop starts from a much lower base and does not make it more robust in any useful sense.

Table 8. Accuracy under deterministic synthetic perturbations of the test set (argmax rule), and the rate at which out-of-scope requests are sent to review for OOS intents seen during training (validation) and never seen (test).

Method	Clean	Punctuation	Typos	Clause order	Typos: expected cost	OOS → review (val, seen OOS intents)	OOS → review (test, unseen OOS intents)
TFIDF-LR n=1106	0.860	0.852	0.848	0.860	0.719	0.863	0.688
SFT n=1106	0.787	0.772	0.691	0.793	1.294	0.792	0.597
WSFT-cost n=1106	0.779	0.761	0.667	0.783	1.494	0.753	0.584
GRPO-bin G=4 R=8192	0.608	0.618	0.531	0.605	1.985	0.430	0.287

Method	Clean	Punctuation	Typos	Clause order	Typos: expected cost	OOS → review (val, seen OOS intents)	OOS → review (test, unseen OOS intents)
GRPO-cost G=4 R=8192	0.637	0.634	0.547	0.635	1.638	0.568	0.417
GRPO-cost G=8 R=8192	0.644	0.647	0.574	0.644	1.253	0.625	0.499
RFT G=4 R=8192	0.671	0.671	0.590	0.668	1.654	0.622	0.412

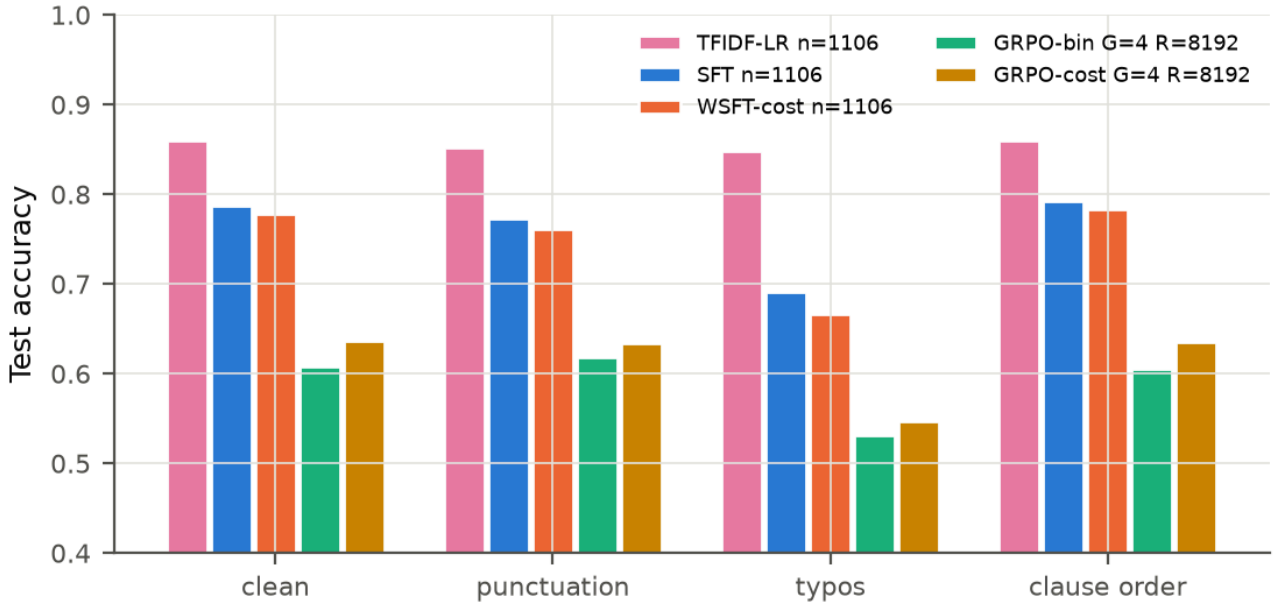


Figure 10. Accuracy under synthetic wording perturbations.

Out-of-scope detection degraded for intents not seen in training. SFT sent 79.2% of validation OOS requests to review but only 59.7% of test OOS requests from unseen intents; TF-IDF/LR fell from 86.3% to 68.8%. GRPO policies routed fewer OOS requests to review overall (binary: 28.7% on test). The cost reward raised this to 41.7% ($G = 4$) and 49.9% ($G = 8$).

5.7 Variability across seeds

Training randomness was substantial for GRPO (Fig. 11, Appendix Table A3). Across seeds, the expected cost of cost-reward GRPO ($G = 4, V = 8192$) ranged from 0.971 to 1.854; full-data SFT ranged from 0.823 to 0.978. For most GRPO comparisons the interval that also resamples seeds is two to three times wider than the test-resampling interval (Table 9). Test-set sampling alone therefore understates the uncertainty of these conclusions.

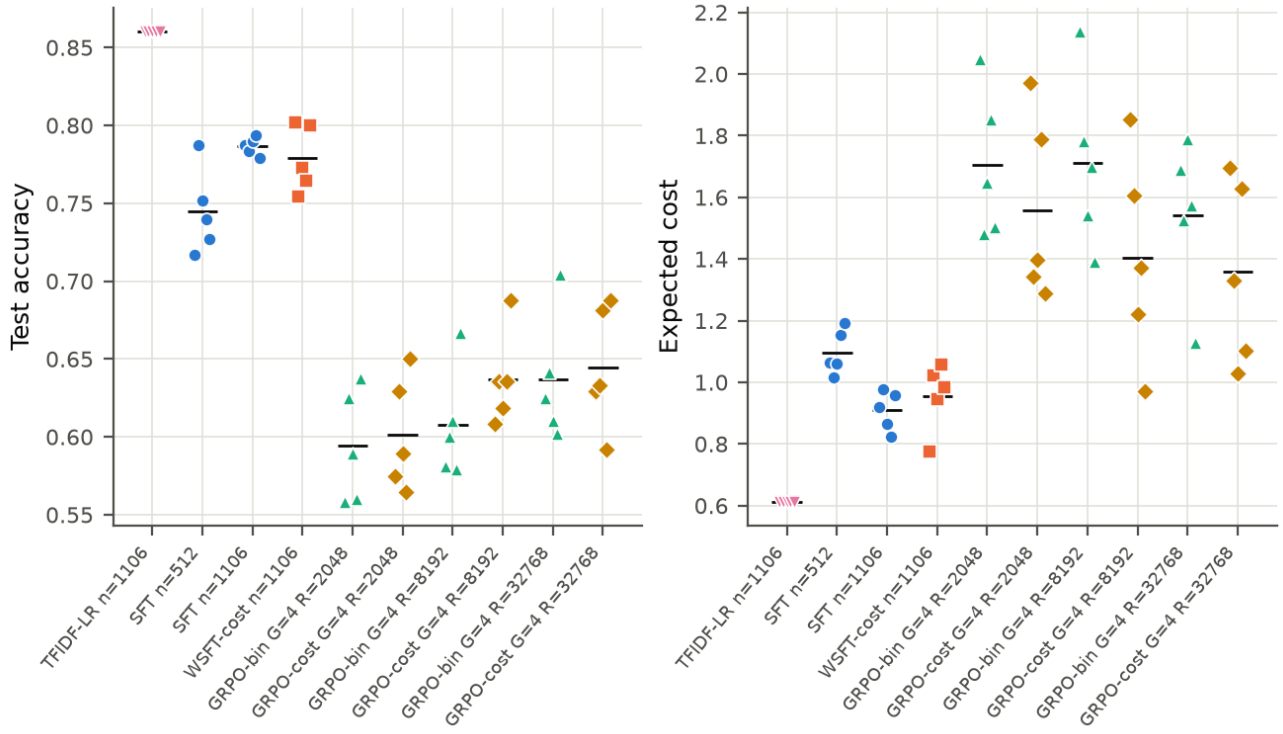


Figure 11. Per-seed test accuracy and expected cost.

Table 9. Paired comparisons on the shared test set (argmax rule; seeds matched). Negative cost differences favour A. Holm adjustment is within each statistic across all 22 predeclared comparisons (full list in `results/tables/comparisons.csv`).

A	B	Statistic	A − B	95% CI, test resampling	95% CI, seeds + test	Holm-adj. p
GRPO-bin G=4 R=2048	SFT n=512	expected cost	+0.609	[+0.450, +0.771]	[+0.346, +0.892]	<0.001
GRPO-bin G=4 R=2048	SFT n=512	critical-error rate	+0.122	[+0.076, +0.171]	[+0.055, +0.194]	0.013
GRPO-cost G=4 R=2048	WSFT-cost n=512	expected cost	+0.469	[+0.316, +0.628]	[+0.146, +0.819]	0.029
GRPO-cost G=4 R=2048	WSFT-cost n=512	critical-error rate	+0.114	[+0.067, +0.160]	[+0.040, +0.198]	0.036
GRPO-bin G=4 R=8192	SFT n=1106	expected cost	+0.801	[+0.645, +0.960]	[+0.519, +1.104]	<0.001
GRPO-bin G=4 R=8192	SFT n=1106	critical-error rate	+0.152	[+0.106, +0.201]	[+0.079, +0.236]	<0.001
GRPO-cost G=4 R=8192	SFT n=1106	expected cost	+0.496	[+0.357, +0.640]	[+0.186, +0.811]	0.033
GRPO-cost G=4 R=8192	SFT n=1106	critical-error rate	+0.091	[+0.050, +0.132]	[+0.014, +0.172]	0.308
GRPO-cost G=4 R=8192	WSFT-cost n=1106	expected cost	+0.449	[+0.308, +0.592]	[+0.107, +0.826]	0.091
GRPO-cost G=4 R=8192	WSFT-cost n=1106	critical-error rate	+0.083	[+0.042, +0.122]	[-0.013, +0.185]	0.908
GRPO-cost G=4 R=32768	SFT n=1106	expected cost	+0.448	[+0.313, +0.582]	[+0.157, +0.755]	0.042

A	B	Statistic	A − B	95% CI, test resampling	95% CI, seeds + test	Holm-adj. p
GRPO-cost G=4 R=32768	SFT n=1106	critical-error rate	+0.069	[+0.030, +0.109]	[+0.001, +0.141]	0.595
SFT n=1106	TFIDF-LR n=1106	expected cost	+0.297	[+0.168, +0.432]	[+0.147, +0.449]	0.004
SFT n=1106	TFIDF-LR n=1106	critical-error rate	+0.062	[+0.026, +0.101]	[+0.024, +0.104]	0.043
WSFT-cost n=1106	SFT n=1106	expected cost	+0.047	[-0.016, +0.114]	[-0.090, +0.186]	0.999
WSFT-cost n=1106	SFT n=1106	critical-error rate	+0.009	[-0.007, +0.026]	[-0.028, +0.046]	1.000
GRPO-cost G=4 R=2048	GRPO-bin G=4 R=2048	expected cost	-0.149	[-0.225, -0.075]	[-0.271, -0.034]	0.128
GRPO-cost G=4 R=2048	GRPO-bin G=4 R=2048	critical-error rate	-0.029	[-0.051, -0.007]	[-0.064, +0.003]	0.796
GRPO-cost G=4 R=8192	GRPO-bin G=4 R=8192	expected cost	-0.306	[-0.390, -0.223]	[-0.436, -0.180]	0.004
GRPO-cost G=4 R=8192	GRPO-bin G=4 R=8192	critical-error rate	-0.061	[-0.087, -0.037]	[-0.102, -0.022]	0.038
GRPO-cost G=2 R=8192	GRPO-bin G=2 R=8192	expected cost	+0.034	[-0.039, +0.106]	[-0.122, +0.158]	0.999
GRPO-cost G=2 R=8192	GRPO-bin G=2 R=8192	critical-error rate	+0.007	[-0.013, +0.028]	[-0.034, +0.041]	1.000
GRPO-cost G=8 R=8192	GRPO-bin G=8 R=8192	expected cost	-0.372	[-0.470, -0.280]	[-0.695, -0.080]	0.128
GRPO-cost G=8 R=8192	GRPO-bin G=8 R=8192	critical-error rate	-0.096	[-0.126, -0.069]	[-0.174, -0.028]	0.051
GRPO-cost G=4 R=32768	GRPO-bin G=4 R=32768	expected cost	-0.184	[-0.288, -0.079]	[-0.515, +0.066]	0.826
GRPO-cost G=4 R=32768	GRPO-bin G=4 R=32768	critical-error rate	-0.039	[-0.070, -0.007]	[-0.128, +0.034]	1.000
GRPO-cost-meanadv G=4 R=8192	GRPO-cost G=4 R=8192	expected cost	+0.169	[+0.084, +0.255]	[+0.013, +0.314]	0.275
GRPO-cost-meanadv G=4 R=8192	GRPO-cost G=4 R=8192	critical-error rate	+0.029	[+0.004, +0.055]	[-0.019, +0.074]	1.000
RFT G=4 R=8192	GRPO-bin G=4 R=8192	expected cost	-0.392	[-0.494, -0.294]	[-0.910, +0.028]	0.424
RFT G=4 R=8192	GRPO-bin G=4 R=8192	critical-error rate	-0.083	[-0.113, -0.054]	[-0.212, +0.026]	1.000
GRPO-cost G=4 R=8192	SFT n=64	expected cost	-0.450	[-0.550, -0.357]	[-0.674, -0.236]	<0.001
GRPO-cost G=4 R=8192	SFT n=64	critical-error rate	-0.101	[-0.131, -0.072]	[-0.163, -0.036]	0.080

6 Ablation Studies

Table 10 collects the ablations (all $G = 4$; the warm start is shown for reference).

Table 10. Ablations (mean ± SD over 5 seeds; argmax rule unless labelled Bayes).

Configuration	Homogeneous groups	Accuracy	Macro-F1	Expected cost (argmax)	Expected cost (Bayes)	Critical-error rate	Coverage
SFT n=64	—	0.558 ± 0.042	0.574 ± 0.044	1.855 ± 0.331	1.071 ± 0.193	0.305 ± 0.075	0.820 ± 0.092
GRPO-bin G=4 R=8192	0.894 ± 0.021	0.608 ± 0.036	0.632 ± 0.041	1.711 ± 0.283	1.117 ± 0.194	0.265 ± 0.072	0.863 ± 0.072
GRPO-cost G=4 R=8192	0.858 ± 0.025	0.637 ± 0.030	0.656 ± 0.041	1.405 ± 0.341	0.937 ± 0.257	0.204 ± 0.084	0.780 ± 0.080
GRPO-cost-meanadv G=4 R=2048	0.854 ± 0.034	0.594 ± 0.036	0.612 ± 0.041	1.615 ± 0.296	1.076 ± 0.142	0.250 ± 0.065	0.792 ± 0.075
GRPO-cost-meanadv G=4 R=8192	0.855 ± 0.030	0.590 ± 0.036	0.610 ± 0.041	1.574 ± 0.305	0.956 ± 0.226	0.233 ± 0.068	0.795 ± 0.105
GRPO-cost-crit5 G=4 R=8192	0.864 ± 0.031	0.627 ± 0.042	0.646 ± 0.048	1.360 ± 0.197	0.914 ± 0.202	0.181 ± 0.038	0.780 ± 0.089
GRPO-cost-crit20 G=4 R=8192	0.868 ± 0.032	0.626 ± 0.030	0.652 ± 0.034	1.395 ± 0.289	0.925 ± 0.218	0.191 ± 0.060	0.788 ± 0.107
RFT G=4 R=512	0.938 ± 0.041	0.565 ± 0.044	0.586 ± 0.047	1.867 ± 0.242	1.109 ± 0.153	0.304 ± 0.051	0.846 ± 0.063
RFT G=4 R=2048	0.900 ± 0.015	0.595 ± 0.039	0.620 ± 0.038	1.774 ± 0.239	1.145 ± 0.036	0.274 ± 0.053	0.886 ± 0.068
RFT G=4 R=8192	0.917 ± 0.018	0.671 ± 0.031	0.698 ± 0.037	1.319 ± 0.307	1.012 ± 0.184	0.182 ± 0.078	0.807 ± 0.070
GRPO-bin-nowarm G=4 R=8192	0.859 ± 0.013	0.430 ± 0.037	0.247 ± 0.063	1.171 ± 0.411	0.593 ± 0.106	0.175 ± 0.127	0.307 ± 0.125
GRPO-cost-nowarm G=4 R=8192	0.961 ± 0.009	0.333 ± 0.000	0.056 ± 0.000	0.500 ± 0.000	0.500 ± 0.000	0.000 ± 0.000	0.000 ± 0.000

Reward: binary vs. cost. This is the one design choice that changed GRPO's cost metrics consistently (Section 5.4). Its effect depended on group size. It was clear at $G = 4$ and $G = 8$ and absent at $G = 2$, where 94–95% of groups were homogeneous and the reward rarely had a chance to discriminate between actions.

Advantage normalisation. Mean-centring the cost reward without dividing by the group standard deviation, as proposed by Zuo et al. (2026) to preserve cost magnitudes, made results worse in our setting. At $V = 8192$, expected cost was 1.574 vs. 1.405 (+0.169 [+0.012, +0.314], adjusted $p = 0.28$) and the error rate was higher by 0.047 (adjusted $p = 0.004$). One plausible explanation is the reward scale. Our cost rewards lie in $[-1, 0]$, so un-normalised advantages are small for routine-intent groups whose costs differ by 0.1–0.2, and those groups contribute little gradient at a learning rate tuned for normalised advantages. We did not retune the learning rate for this variant, so the ablation shows that standard-deviation scaling is not harmful here. It does not show that mean-centring is inferior in general.

Critical-error cost used in training. Training the cost reward with $c_{\text{crit}} = 5$ or 20 instead of 10 changed little. Expected cost (evaluated with $c_{\text{crit}} = 10$) was 1.360 and 1.395 vs. 1.405, and critical-error rates were 0.181 and 0.191 vs. 0.204, all within one seed standard deviation. With group-normalised advantages, scaling the penalty for critical misses changes only the relative ordering of rewards within groups that mix critical misses with other outcomes. Because most groups are homogeneous, the policy rarely sees such groups.

Rejection-sampling fine-tuning. Online RFT with the same verifier budget matched or exceeded binary GRPO. At $V = 8192$ it reached 0.671 accuracy vs. 0.608, cost 1.319 vs. 1.711, and critical-error rate 0.182 vs. 0.265. After Holm correction these differences were not significant (cost −0.392 [−0.910, +0.028]), because RFT varied widely across seeds. This agrees with Xiong et al. (2025): GRPO's group-relative machinery provided no measurable benefit over simply imitating successful samples.

Warm start. Without the supervised warm start, GRPO from the pretrained model failed. The pretrained model places on average 0.70 of the constrained probability mass on " I", the token mapped to MANUAL_REVIEW (probably because "I" is a very frequent English word), and it is the argmax action for 1,105 of the 1,106 training prompts (`results/tables/pretrained_action_prior.json`). Most groups were therefore homogeneous (86% for the binary reward, 96% for the cost reward), and both variants stayed near MANUAL_REVIEW. With the binary reward, 69% of requests went to review and accuracy was 0.430. With the cost reward, all five seeds converged to sending *every* request to review: coverage 0, accuracy 0.333 (the OOS share), and expected cost exactly 0.500. Under the cost reward, review is always the "safe" action with reward -0.05, while a wrong guess costs up to -1. A policy that has not yet learned the intents therefore minimises expected cost by abstaining. This degenerate policy has a *lower* argmax expected cost than every trained router in Table 1. This is why we report coverage and compare at fixed coverage targets throughout.

Coverage targets and cost matrices. Changing the decision rule had a much larger effect than any training ablation. Table 11 re-scores all methods under alternative cost matrices. The method ordering by expected cost (TF-IDF/LR < SFT < WSFT-cost < RFT/GRPO-cost < GRPO-bin under argmax) holds for every combination of $c_{\text{crit}} \in \{5, 10, 20\}$ and $c_{\text{rev}} \in \{0.25, 0.5, 1\}$ we evaluated. The Bayes rule improves every method under every matrix, most when review is cheap.

Table 11. Cost-matrix sensitivity: expected cost re-scored under alternative critical-miss and review costs (argmax / Bayes, the latter re-derived for each matrix). All nine combinations are in `results/tables/tables.md`.

Method	c_{crit}=5, c_{rev}=0.5	c_{crit}=10, c_{rev}=0.25	c_{crit}=10, c_{rev}=0.5	c_{crit}=10, c_{rev}=1.0	c_{crit}=20, c_{rev}=0.5
TFIDF-LR n=1106	0.53 / 0.31	0.55 / 0.19	0.61 / 0.33	0.73 / 0.58	0.78 / 0.35
SFT n=1106	0.72 / 0.60	0.85 / 0.54	0.91 / 0.69	1.03 / 0.88	1.28 / 0.84
WSFT-cost n=1106	0.75 / 0.64	0.90 / 0.59	0.96 / 0.73	1.07 / 0.96	1.36 / 0.93
GRPO-bin G=4 R=8192	1.27 / 0.94	1.68 / 0.89	1.71 / 1.12	1.78 / 1.39	2.59 / 1.42
GRPO-cost G=4 R=8192	1.07 / 0.80	1.35 / 0.73	1.41 / 0.94	1.51 / 1.19	2.08 / 1.13
GRPO-cost G=8 R=8192	0.92 / 0.76	1.08 / 0.66	1.16 / 0.86	1.30 / 1.15	1.64 / 1.02
RFT G=4 R=8192	1.01 / 0.86	1.27 / 0.85	1.32 / 1.01	1.42 / 1.23	1.93 / 1.28

7 Scaling Check: Pythia-31M

The main study uses a very small policy. To test whether its conclusions were an artefact of that scale, we repeated a reduced grid with `EleutherAI/pythia-31m` (Hub revision `e556ace21b489575e94e9d50b6dad2fcc7419679`; 6 layers, width 256, 30,494,720 parameters; same tokenizer, prompt, and action tokens). The experiment was added after the 14M study was complete and used the same splits, seeds (0–4), budgets, and evaluation code. The grid covered:

- the 64-label warm start;
- SFT with $n \in \{512, 1106\}$;
- cost-weighted SFT with $n = 1106$;
- GRPO with $G = 4$, reward $\in$ {binary, cost}, and $V \in \{2048, 8192\}$;
- RFT with $G = 4$ and $V = 8192$.

That is 45 runs, which used 4.6 single-thread CPU-hours in total.

Learning rates were re-selected for 31M with the same validation-only pilot protocol (seed 100):

- **SFT.** 10^{-4} was best at both $n = 64$ and $n = 1106$. Validation cost was **0.611**, 0.637 and 0.771 at $n = 1106$, for 10^{-4}, 3×10^{-4} and 10^{-3}.
- **GRPO** (binary reward, $G = 4$, $V = 2048$). 3×10^{-6} was best. Validation cost was 1.692, **1.658**, 1.983, 1.801 and 1.765 for 10^{-6}, 3×10^{-6}, 10^{-5}, 3×10^{-5} and 10^{-4}. The grid was extended to 10^{-6} after 3×10^{-6} was best at its edge.

The validation costs above 3×10^{-6} are not monotone, so the GRPO choice is only weakly determined by a single pilot seed.

Peak resident memory rose from about 0.95 GB to 1.4 GB per GRPO run. One cost-weighted SFT run initially failed with an out-of-memory error, and the five warm-start runs initially failed to write their checkpoints. Both failures were caused by contention for memory and disk from unrelated processes on the same machine. The affected runs were repeated with identical specifications, and all 45 runs completed.

Table 12. Pythia-14M vs. Pythia-31M on the test set (mean ± s.d. over five seeds). Accuracy, expected cost, and critical-error rate use the argmax rule; "Expected cost (Bayes)" uses the minimum-expected-cost rule. CPU s is single-thread training time per run, including the warm start for GRPO/RFT.

Method	Model	n seeds	CPU s	Accuracy	Expected cost	Critical-error rate	Expected cost (Bayes)	Homogeneous groups
SFT n=64	14M	5	25	0.558 ± 0.042	1.855 ± 0.331	0.305 ± 0.075	1.071 ± 0.193	—
SFT n=64	31M	5	64	0.530 ± 0.043	2.046 ± 0.297	0.354 ± 0.070	0.954 ± 0.177	—
SFT n=512	14M	5	145	0.745 ± 0.027	1.097 ± 0.073	0.146 ± 0.010	0.729 ± 0.068	—
SFT n=512	31M	5	381	0.777 ± 0.032	0.878 ± 0.115	0.092 ± 0.018	0.687 ± 0.088	—
SFT n=1106	14M	5	332	0.787 ± 0.006	0.909 ± 0.064	0.113 ± 0.010	0.689 ± 0.060	—
SFT n=1106	31M	5	891	0.804 ± 0.015	0.887 ± 0.048	0.115 ± 0.017	0.686 ± 0.046	—
WSFT-cost n=1106	14M	5	326	0.779 ± 0.021	0.956 ± 0.110	0.121 ± 0.030	0.732 ± 0.081	—
WSFT-cost n=1106	31M	5	850	0.786 ± 0.030	0.911 ± 0.079	0.100 ± 0.013	0.768 ± 0.084	—
GRPO-bin G=4 R=2048	14M	5	75	0.594 ± 0.036	1.706 ± 0.241	0.269 ± 0.053	1.066 ± 0.225	0.883 ± 0.018
GRPO-bin G=4 R=2048	31M	5	145	0.575 ± 0.023	1.869 ± 0.162	0.323 ± 0.047	0.920 ± 0.129	0.836 ± 0.050
GRPO-cost G=4 R=2048	14M	5	66	0.602 ± 0.036	1.557 ± 0.304	0.240 ± 0.062	1.015 ± 0.178	0.850 ± 0.041
GRPO-cost G=4 R=2048	31M	5	141	0.584 ± 0.020	1.604 ± 0.226	0.267 ± 0.066	0.839 ± 0.115	0.805 ± 0.049
GRPO-bin G=4 R=8192	14M	5	170	0.608 ± 0.036	1.711 ± 0.283	0.265 ± 0.072	1.117 ± 0.194	0.894 ± 0.021

Method	Model	n seeds	CPU s	Accuracy	Expected cost	Critical-error rate	Expected cost (Bayes)	Homogeneous groups
GRPO-bin G=4 R=8192	31M	5	394	0.613 ± 0.033	1.688 ± 0.061	0.270 ± 0.009	0.892 ± 0.146	0.851 ± 0.028
GRPO-cost G=4 R=8192	14M	5	167	0.637 ± 0.030	1.405 ± 0.341	0.204 ± 0.084	0.937 ± 0.257	0.858 ± 0.025
GRPO-cost G=4 R=8192	31M	5	387	0.642 ± 0.028	1.278 ± 0.130	0.191 ± 0.039	0.733 ± 0.091	0.816 ± 0.028
RFT G=4 R=8192	14M	5	165	0.671 ± 0.031	1.319 ± 0.307	0.182 ± 0.078	1.012 ± 0.184	0.917 ± 0.018
RFT G=4 R=8192	31M	5	392	0.639 ± 0.034	1.620 ± 0.140	0.261 ± 0.030	0.927 ± 0.099	0.870 ± 0.030

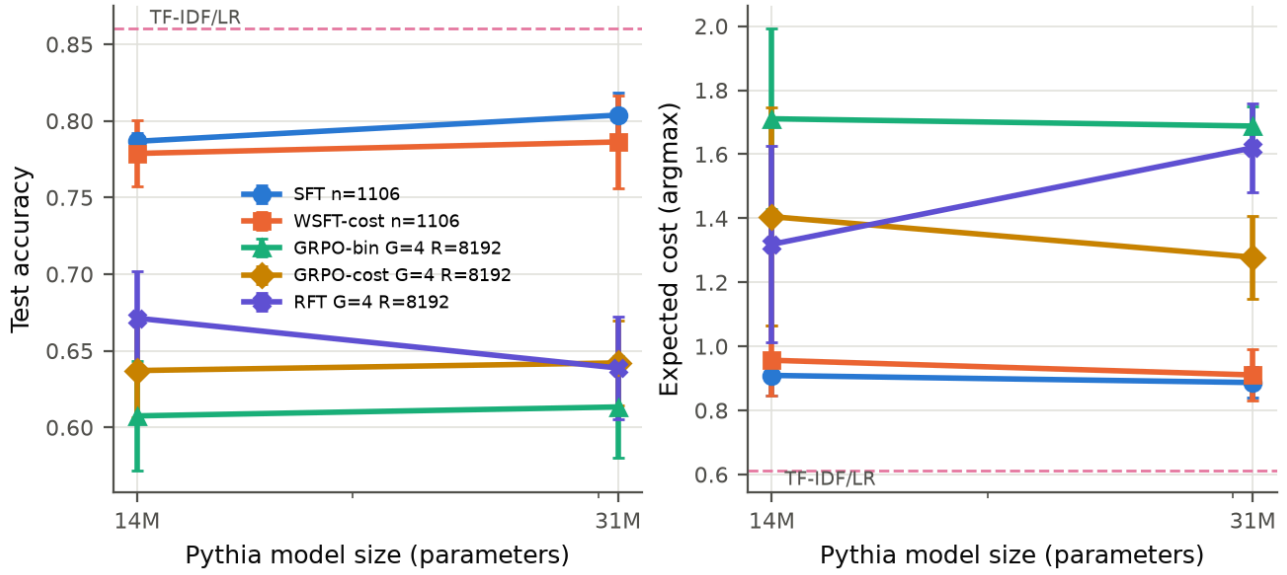


Figure 12. Test accuracy (left) and argmax expected cost (right) for 14M and 31M policies (mean ± s.d. over five seeds); dashed line: full-data TF-IDF/LR.

SFT barely changed. With all 1,106 labels, 31M SFT reached 0.804 accuracy and 0.887 expected cost, against 0.787 and 0.909 for 14M. Neither difference is significant (Table 13). The larger model helped more at 512 labels (0.777 vs. 0.745 accuracy; 0.878 vs. 1.097 expected cost). TF-IDF/LR remained better than 31M SFT in accuracy and in expected cost under both decision rules.

The GRPO gap did not close. Under the argmax rule, every 31M GRPO and RFT configuration remained far behind 31M SFT. With all labels ($V = 8192$), binary-reward GRPO had 0.613 accuracy and 1.688 expected cost, and cost-reward GRPO had 0.642 and 1.278, compared with 0.804 and 0.887 for 31M SFT. Both cost differences survive Holm correction (+0.801 and +0.391). Cost-reward GRPO also made more critical errors than 31M SFT (0.191 vs. 0.115), although that difference did not survive correction. RFT did not benefit from scale (expected cost 1.620 at 31M vs. 1.319 at 14M).

The cost reward still helped within GRPO. At 31M, the cost reward lowered argmax expected cost relative to the binary reward by 0.410 (95% CI [−0.575, −0.257]; Holm-adjusted $p < 0.001$). It also lowered the

critical-error rate by 0.079 (Holm-adjusted $p = 0.004$). This replicates the main study's RQ2 finding at the larger scale.

What did change. Three things moved with scale:

- **Seed variability.** GRPO varied much less across seeds at 31M. The standard deviation of expected cost for cost-reward GRPO fell from 0.341 to 0.130, and for binary-reward GRPO from 0.283 to 0.061.
- **Homogeneous groups.** The homogeneous-group fraction fell modestly, from 0.85–0.92 for the same configurations at 14M to 0.81–0.87.
- **Bayes-rule cost.** Under the Bayes rule, 31M GRPO had lower expected cost than 14M GRPO, and cost-reward GRPO at $V = 8192$ was statistically indistinguishable from 31M SFT (0.733 vs. 0.686; difference +0.047, 95% CI [−0.109, +0.200]).

These 31M-vs-14M Bayes-rule differences did not survive correction. As at 14M, no GRPO or RFT configuration beat the always-review cost of 0.5 under any decision rule; the lowest was 0.668. The supervised models did beat it with a selective rule: 31M SFT at the 50% coverage target cost 0.410 (critical-error rate 0.014), close to 14M SFT's 0.417.

Table 13. Predeclared paired comparisons for the scaling experiment (seeds matched; two-level bootstrap over seeds and test examples). Configurations without "[31M]" are the 14M runs from the main study. Negative cost differences favour A. Holm adjustment is within each statistic and decision rule across the eight predeclared scaling comparisons, separately from the main study's 22 (full list in `results/tables/comparisons.csv`).

A	B	Rule	Statistic	A − B	95% CI, seeds + test	Holm-adj. *p*
GRPO-bin G=4 R=8192 [31M]	SFT n=1106 [31M]	argmax	expected cost	+0.801	[+0.605, +1.008]	<0.001
GRPO-bin G=4 R=8192 [31M]	SFT n=1106 [31M]	argmax	error rate	+0.190	[+0.146, +0.236]	<0.001
GRPO-bin G=4 R=8192 [31M]	SFT n=1106 [31M]	argmax	critical-error rate	+0.155	[+0.098, +0.212]	<0.001
GRPO-bin G=4 R=8192 [31M]	SFT n=1106 [31M]	bayes	expected cost	+0.206	[+0.026, +0.383]	0.138
GRPO-cost G=4 R=8192 [31M]	SFT n=1106 [31M]	argmax	expected cost	+0.391	[+0.177, +0.602]	0.002
GRPO-cost G=4 R=8192 [31M]	SFT n=1106 [31M]	argmax	error rate	+0.162	[+0.120, +0.203]	<0.001
GRPO-cost G=4 R=8192 [31M]	SFT n=1106 [31M]	argmax	critical-error rate	+0.076	[+0.016, +0.135]	0.059
GRPO-cost G=4 R=8192 [31M]	SFT n=1106 [31M]	bayes	expected cost	+0.047	[-0.109, +0.200]	1.000
GRPO-cost G=4 R=8192 [31M]	GRPO-bin G=4 R=8192 [31M]	argmax	expected cost	-0.410	[-0.575, -0.257]	<0.001
GRPO-cost G=4 R=8192 [31M]	GRPO-bin G=4 R=8192 [31M]	argmax	error rate	-0.029	[-0.056, -0.004]	0.106
GRPO-cost G=4 R=8192 [31M]	GRPO-bin G=4 R=8192 [31M]	argmax	critical-error rate	-0.079	[-0.124, -0.036]	0.004
GRPO-cost G=4 R=8192 [31M]	GRPO-bin G=4 R=8192 [31M]	bayes	expected cost	-0.159	[-0.279, -0.044]	0.069
GRPO-cost G=4 R=8192 [31M]	WSFT-cost n=1106 [31M]	argmax	expected cost	+0.368	[+0.133, +0.602]	0.011

A	B	Rule	Statistic	A − B	95% CI, seeds + test	Holm-adj. p
GRPO-cost G=4 R=8192 [31M]	WSFT-cost n=1106 [31M]	argmax	error rate	+0.144	[+0.106, +0.183]	<0.001
GRPO-cost G=4 R=8192 [31M]	WSFT-cost n=1106 [31M]	argmax	critical-error rate	+0.091	[+0.031, +0.152]	0.022
GRPO-cost G=4 R=8192 [31M]	WSFT-cost n=1106 [31M]	bayes	expected cost	-0.035	[-0.205, +0.133]	1.000
SFT n=1106 [31M]	TFIDF-LR n=1106	argmax	expected cost	+0.274	[+0.126, +0.427]	<0.001
SFT n=1106 [31M]	TFIDF-LR n=1106	argmax	error rate	+0.057	[+0.028, +0.087]	<0.001
SFT n=1106 [31M]	TFIDF-LR n=1106	argmax	critical-error rate	+0.065	[+0.022, +0.110]	0.008
SFT n=1106 [31M]	TFIDF-LR n=1106	bayes	expected cost	+0.354	[+0.239, +0.480]	<0.001
SFT n=1106 [31M]	SFT n=1106	argmax	expected cost	-0.022	[-0.152, +0.119]	1.000
SFT n=1106 [31M]	SFT n=1106	argmax	error rate	-0.017	[-0.044, +0.010]	0.575
SFT n=1106 [31M]	SFT n=1106	argmax	critical-error rate	+0.003	[-0.033, +0.040]	1.000
SFT n=1106 [31M]	SFT n=1106	bayes	expected cost	-0.003	[-0.114, +0.112]	1.000
GRPO-bin G=4 R=8192 [31M]	GRPO-bin G=4 R=8192	argmax	expected cost	-0.023	[-0.287, +0.221]	1.000
GRPO-bin G=4 R=8192 [31M]	GRPO-bin G=4 R=8192	argmax	error rate	-0.006	[-0.048, +0.035]	1.000
GRPO-bin G=4 R=8192 [31M]	GRPO-bin G=4 R=8192	argmax	critical-error rate	+0.005	[-0.065, +0.072]	1.000
GRPO-bin G=4 R=8192 [31M]	GRPO-bin G=4 R=8192	bayes	expected cost	-0.225	[-0.408, -0.041]	0.102
GRPO-cost G=4 R=8192 [31M]	GRPO-cost G=4 R=8192	argmax	expected cost	-0.127	[-0.418, +0.199]	1.000
GRPO-cost G=4 R=8192 [31M]	GRPO-cost G=4 R=8192	argmax	error rate	-0.005	[-0.038, +0.028]	1.000
GRPO-cost G=4 R=8192 [31M]	GRPO-cost G=4 R=8192	argmax	critical-error rate	-0.013	[-0.096, +0.081]	1.000
GRPO-cost G=4 R=8192 [31M]	GRPO-cost G=4 R=8192	bayes	expected cost	-0.204	[-0.394, -0.017]	0.138

In short, more than doubling the policy size left the main conclusions unchanged. SFT stayed ahead of GRPO at equal labels, TF-IDF/LR stayed ahead of both, and the cost reward still beat the binary reward within GRPO. A 2× change at this end of the scale is, of course, far from the model sizes at which GRPO is usually applied.

8 Discussion

RQ1: GRPO vs. SFT at equal labels and measured compute. In this setting GRPO was not worth it. With the same labelled examples, SFT was more accurate than GRPO at every label budget we tested (128 to 1,106), and at 512 and 1,106 labels it was also better on macro-F1, expected cost, and critical errors (Table 9). The exception is the smallest budget: with 128 labels ($G = 4, V = 512$), cost-reward GRPO and SFT were tied on expected cost (1.756 vs. 1.768) and critical errors (0.284 vs. 0.282), both barely better than the warm start. TF-IDF/LR was better than both at a tiny fraction of the compute. Hypothesis H1, that SFT would be competitive with or better than GRPO when reliable labels are available, is supported in its stronger form. The mechanism is visible in the budget accounting. When the verifier is an exact-match check against a gold label in a nine-way action space, a single cross-entropy step uses the full label, while a GRPO group uses it only through G sampled actions, most of which receive the same reward and produce no gradient. The cheaper per-update cost of GRPO (one forward pass to sample, one for the reference, $\mu = 2$ backward passes per prompt) did not make up for this: at matched or larger CPU time, GRPO reached lower accuracy than SFT. We do not claim this holds for generative tasks, where a verifier may check properties that no single label specifies.

RQ2: does the risk-sensitive reward reduce critical errors? Relative to binary-reward GRPO, yes, partly. At $G = 4$ and $V = 8192$ the cost reward lowered expected cost by about 18% and the critical-error rate from 0.265 to 0.204, without a significant accuracy change. Part of the reduction came from sending more requests to review. Relative to supervised baselines, no: cost-reward GRPO still made more critical errors than plain SFT, and cost-weighted SFT gave no benefit over SFT. H2 is therefore only partly supported. The cost reward reduced high-severity mistakes within GRPO, mainly through more abstention, but it did not produce a method competitive with supervised learning. In this study the most effective and cheapest risk-reduction tool was post-hoc: the Bayes decision rule, or a validation-calibrated abstention threshold, applied to an ordinary supervised model.

RQ3: group size and homogeneous groups. Under a fixed verifier budget, larger groups tended to perform better despite seeing fewer distinct prompts, and the homogeneous fraction fell with G. More than three-quarters of groups (77.9%–99.7%) were homogeneous in every configuration, so most verifier calls were wasted. H3, that conclusions about group size depend on what is held fixed, is consistent with what we observed. Holding verifier calls fixed favours larger G in direction, and holding prompts fixed would trivially favour larger G because it buys more samples. The group-size differences were not statistically significant after correction, however.

RQ4: robustness. No method's advantage depended on the evaluation condition. Rankings were stable under simulated prevalence shifts, synthetic perturbations, and unseen OOS intents. Every Pythia-based model was sensitive to character-level typos, and OOS recall dropped sharply for intents not seen during training. These are properties of the tiny model and the OOS construction more than of the training algorithm.

H4: the best method under accuracy vs. under cost. H4 was not supported among trained methods: TF-IDF/LR was best under both accuracy and expected cost. The decision rule changed the picture much more than the training method. The best-accuracy rule (argmax) was never the best-cost rule, and the lowest expected cost overall (0.307) came from class-balanced TF-IDF/LR operated at about 46% automation coverage.

When might GRPO be justified? Our results narrow, rather than close, the conditions under which GRPO is attractive for routing: (i) when only a verifier is available and labels cannot be written down, which is not the

case for intent classification; (ii) when the policy is already strong enough that groups are mixed, rather than overwhelmingly homogeneous; (iii) when outputs are structured or multi-step, so a verifier can reward properties that cross-entropy on a single label cannot express. None of these held here. A practitioner with labelled routing data and a CPU budget should start with a linear model and a calibrated abstention rule, then move to a supervised neural model, and consider reinforcement learning only if a concrete gap remains that a verifier, but not a label, can address.

Competing explanations. Three alternatives deserve mention. First, the GRPO learning rate was tuned at one budget on one pilot seed; a schedule tuned per budget might narrow the gap, although the size of the gap (0.18 accuracy at full labels) makes reversal unlikely. Second, the warm start was weak (64 labels), which inflates homogeneity. A stronger warm start would, however, consume more labels and compute, which the label-matched comparison would charge to GRPO. Third, the action-token prior of the pretrained model (Section 6) biases every Pythia-based method toward abstention at initialisation. All methods share it, so it should not favour one training objective over another, but it may depress OOS behaviour in ways that a different token mapping would not. Fourth, the 14M model may simply be too small for reinforcement learning to help. The 31M replication (Section 7) argues against this at the scales we could test. Doubling the parameters improved SFT slightly and made GRPO more stable across seeds, but the argmax gap between GRPO and SFT stayed large and significant.

9 Limitations and Threats to Validity

- **Model scale.** Pythia-14M has 6 layers and width 128, and most of its parameters are the token embedding. A reduced replication with Pythia-31M (Section 7) did not change the conclusions, but that is only a twofold increase. Neither model's behaviour need predict that of the much larger models for which GRPO benefits have been reported.
- **Simplified GRPO.** Our GRPO acts on a single constrained token per episode. There is no multi-token generation, no reasoning trace, and no length effects. The exact KL over nine actions is a convenience unavailable for sequence-level policies.
- **Restricted task.** Eight intents of one dataset, one prompt template, and one action-token mapping were studied. The mapping assigns " I" to MANUAL_REVIEW, which the pretrained model strongly prefers; a different mapping could change initial behaviour.
- **Synthetic costs and severity.** The cost matrix and the critical/routine designation are assumptions made for this study. BANKING77 supplies neither, and no value was validated against real operational losses. Under the default matrix, review is cheap enough that always-review beats every argmax router; conclusions about absolute cost levels are therefore matrix-dependent, although method rankings were stable across the nine matrices evaluated.
- **Simulated shifts.** Prevalence shifts are importance re-weightings of one test set, and wording perturbations are deterministic synthetic rewrites. Neither is a sample of real distribution shift.
- **Out-of-scope construction.** OOS requests are other BANKING77 intents, which are topically close to the in-scope intents and do not represent arbitrary off-topic traffic.
- **Hyperparameters and tuning.** Learning rates came from a small single-seed validation pilot. GRPO settings (β, ε, μ, batch size, temperature) were fixed a priori and not tuned. Better settings may exist for every method.

- **CPU-specific measurements.** CPU times are single-thread process times on one laptop CPU with five concurrent runs, and they will differ on other hardware. Relative costs between methods are more transferable than absolute seconds.
- **Statistical power.** Five seeds per configuration give wide intervals for GRPO, whose seed-to-seed variability is large. Several directional effects (group size, $G = 8$ cost reward, RFT vs. GRPO) did not survive multiple-comparison correction. The Wilcoxon test over five seeds cannot reach $p < 0.05$.
- **Class imbalance.** Training data are imbalanced (64–150 examples per in-scope intent, 238 OOS), while the test set is balanced across intents. Inverse-frequency weighting helped slightly, which suggests imbalance affects all neural methods.
- **Verifier.** The reward is an exact comparison with a gold label, so it is noise-free and as informative as the label. Our conclusions do not cover noisy, partial, or learned verifiers.
- **Evaluation protocol change.** The selective-classification coverage targets were changed from 0.9/0.8/0.7 to 0.7/0.6/0.5 during analysis, after a bug fix in the selective rule and after inspecting partial test results of SFT runs. Targets above 2/3 are infeasible without auto-routing OOS traffic. The change affects only the coverage-target tables, not any training or model selection.
- **No deployment validation.** Nothing here was tested with real customers, real queues, or human reviewers.

10 Practical Implications

For a team choosing how to adapt a small model to a routing task with labelled data, our measurements suggest a decision procedure rather than a universal answer:

1. **Write down the cost assumptions first, including the cost of human review.** Under our hypothetical matrix, sending everything to review was a strong baseline that most automatic routers failed to beat. Whether automation pays off at all depends on these values, and evaluating against the always-review policy guards against mistaking abstention for skill.
2. **Establish a linear baseline.** TF-IDF with logistic regression trained in under a second and was the best method on every metric in this study.
3. **Add a cost-aware decision layer before changing the training objective.** The Bayes rule and validation-calibrated thresholds cut critical errors far more than cost-weighted losses or cost-shaped rewards did, at negligible cost.
4. **If a neural model is needed, use supervised fine-tuning when labels exist.** Every verifier call that checks a label consumes that label. Count verifier-scored prompts as labelled data when comparing methods.
5. **Before investing in GRPO, measure the homogeneous-group fraction on a pilot.** If most groups have identical rewards, most of the verifier budget is buying no gradient.

These recommendations are derived from one dataset, one tiny model, and hypothetical costs. They are not measured business savings, and they say nothing about deployment safety.

11 Conclusion

We compared GRPO with supervised and classical baselines for nine-action banking intent routing with a 14M-parameter language model on a CPU. Budgets were matched by labelled examples and by verifier calls, and every method was scored on accuracy, expected cost, critical errors, coverage, and measured compute. In this setting GRPO was not worth it. At 512 and 1,106 shared labels it trailed SFT by 0.14–0.18 accuracy, and SFT in turn trailed a TF-IDF/logistic-regression baseline. More than three-quarters of GRPO's sampled groups produced no gradient. A cost-shaped reward lowered GRPO's expected cost and critical-error rate relative to a binary reward, partly through more abstention, but it did not reach supervised performance. The most effective risk control was a cost-aware decision rule on top of an ordinary supervised model. A reduced replication with Pythia-31M left these conclusions unchanged. It reduced GRPO's seed-to-seed variability but did not close its gap to SFT. Code, data splits, per-run records, and scripts that regenerate every table and figure are provided with the paper.

Berg-Kirkpatrick, Taylor, David Burkett, and Dan Klein. 2012. "An Empirical Investigation of Statistical Significance in NLP." *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 995–1005. https://aclanthology.org/D12-1091/.

Biderman, Stella, Hailey Schoelkopf, Quentin Gregory Anthony, et al. 2023. "Pythia: A Suite for Analyzing Large Language Models Across Training and Scaling." *Proceedings of the 40th International Conference on Machine Learning*, Proceedings of machine learning research, vol. 202: 2397–430. https://proceedings.mlr.press/v202/biderman23a.html.

Buda, Mateusz, Atsuto Maki, and Maciej A. Mazurowski. 2018. "A Systematic Study of the Class Imbalance Problem in Convolutional Neural Networks." *Neural Networks*, ahead of print. https://doi.org/10.1016/j.neunet.2018.07.011.

Casanueva, Iñigo, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. "Efficient Intent Detection with Dual Sentence Encoders." *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, 38–45. https://doi.org/10.18653/v1/2020.nlp4convai-1.5.

Chow, C. K. 1970. "On Optimum Recognition Error and Reject Tradeoff." *IEEE Transactions on Information Theory* 16 (1): 41–46. https://doi.org/10.1109/TIT.1970.1054406.

Dang, Quy-Anh, and Chris Ngo. 2025. "Reinforcement Learning for Reasoning in Small LLMs: What Works and What Doesn't." *arXiv Preprint arXiv:2503.16219*. https://arxiv.org/abs/2503.16219.

Dong, Hanze, Wei Xiong, Deepanshu Goyal, et al. 2023. "RAFT: Reward rAnked FineTuning for Generative Foundation Model Alignment." *Transactions on Machine Learning Research*. https://arxiv.org/abs/2304.06767.

Dror, Rotem, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. "The Hitchhiker's Guide to Testing Statistical Significance in Natural Language Processing." *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1383–92. https://doi.org/10.18653/v1/P18-1128.

Elkan, Charles. 2001. "The Foundations of Cost-Sensitive Learning." *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI)*, 973–77. https://www.ijcai.org/proceedings/2001-2.

El-Yaniv, Ran, and Yair Wiener. 2010. "On the Foundations of Noise-Free Selective Classification." *Journal of Machine Learning Research* 11 (53): 1605–41. https://jmlr.org/papers/v11/el-yaniv10a.html.

Feng, Zihao, Xiaoxue Wang, Ziwei Bai, et al. 2025. "Improving Generalization in Intent Detection: GRPO with Reward-Based Curriculum Sampling." *arXiv Preprint arXiv:2504.13592*. https://arxiv.org/abs/2504.13592.

Geifman, Yonatan, and Ran El-Yaniv. 2017. "Selective Classification for Deep Neural Networks." *Advances in Neural Information Processing Systems 30*. https://proceedings.neurips.cc/paper/2017/hash/4a8423d5e91fda00bb7e46540e2b0cf1-Abstract.html.

Guo, Daya, Dejian Yang, Haowei Zhang, Junxiao Song, et al. 2025. "DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning." *Nature* 645: 633–38. https://doi.org/10.1038/s41586-025-09422-z.

He, Haibo, and Edwardo A. Garcia. 2009. "Learning from Imbalanced Data." *IEEE Transactions on Knowledge and Data Engineering* 21 (9): 1263–84. https://doi.org/10.1109/TKDE.2008.239.

He, Xixiang, Qiyao Sun, Ao Cheng, et al. 2026. "Advantage Collapse in Group Relative Policy Optimization: Diagnosis and Mitigation." *Proceedings of the 43rd International Conference on Machine Learning*, Proceedings of machine learning research, vol. 306: 41800–41825. https://proceedings.mlr.press/v306/he26ae.html.

Hendrycks, Dan, and Kevin Gimpel. 2017. "A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks." *International Conference on Learning Representations*. https://arxiv.org/abs/1610.02136.

Hu, Edward J., Yelong Shen, Phillip Wallis, et al. 2021. "LoRA: Low-Rank Adaptation of Large Language Models." *arXiv Preprint arXiv:2106.09685*. https://arxiv.org/abs/2106.09685.

Joulin, Armand, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. “Bag of Tricks for Efficient Text Classification.” *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, 427–31. https://aclanthology.org/E17-2068/.

Koehn, Philipp. 2004. “Statistical Significance Tests for Machine Translation Evaluation.” *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, 388–95. https://aclanthology.org/W04-3250/.

Larson, Stefan, Anish Mahendran, Joseph J. Peper, et al. 2019. “An Evaluation Dataset for Intent Classification and Out-of-Scope Prediction.” *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 1311–16. https://doi.org/10.18653/v1/D19-1131.

Liu, Shih-Yang, Xin Dong, Ximing Lu, et al. 2026. “GDPO: Group Reward-Decoupled Normalization Policy Optimization for Multi-Reward RL Optimization.” *Proceedings of the 43rd International Conference on Machine Learning*, Proceedings of machine learning research, vol. 306: 75978–6000. https://proceedings.mlr.press/v306/liu26aa.html.

Liu, Zichen, Changyu Chen, Wenjun Li, et al. 2025. “Understanding R1-Zero-Like Training: A Critical Perspective.” *arXiv Preprint arXiv:2503.20783*. https://arxiv.org/abs/2503.20783.

Mohamadi, Mohamad Amin, Tianhao Wang, and Zhiyuan Li. 2025. “Honesty over Accuracy: Trustworthy Language Models Through Reinforced Hesitation.” *arXiv Preprint arXiv:2511.11500*. https://arxiv.org/abs/2511.11500.

Pan, Muyu, Shu Zhao, Nan Zhang, et al. 2026. “TIAR: Trajectory-Informed Advantage Reweighting for LLM Abstention Learning.” *arXiv Preprint arXiv:2605.25850*. https://arxiv.org/abs/2605.25850.

Schulman, John, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. “Proximal Policy Optimization Algorithms.” *arXiv Preprint arXiv:1707.06347*. https://arxiv.org/abs/1707.06347.

Schwartz, Roy, Jesse Dodge, Noah A. Smith, and Oren Etzioni. 2019. “Green AI.” *arXiv Preprint arXiv:1907.10597*. https://arxiv.org/abs/1907.10597.

Shao, Zhihong, Peiyi Wang, Qihao Zhu, et al. 2024. “DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models.” *arXiv Preprint arXiv:2402.03300*. https://arxiv.org/abs/2402.03300.

Tamar, Aviv, Yonatan Glassner, and Shie Mannor. 2015. “Optimizing the CVaR via Sampling.” *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. https://arxiv.org/abs/1404.3862.

Wang, Sida, and Christopher Manning. 2012. “Baselines and Bigrams: Simple, Good Sentiment and Topic Classification.” *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 90–94. https://aclanthology.org/P12-2018/.

Wei, Zhepei, Xiao Yang, Kai Sun, et al. 2026. “TruthRL: Incentivizing Truthful LLMs via Reinforcement Learning.” *Proceedings of the 43rd International Conference on Machine Learning*, Proceedings of machine learning research, vol. 306: 134156–75. https://proceedings.mlr.press/v306/wei26ab.html.

Williams, Ronald J. 1992. “Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning.” *Machine Learning* 8 (3–4): 229–56. https://doi.org/10.1007/BF00992696.

Xiong, Wei, Jiarui Yao, Yuhui Xu, et al. 2025. “A Minimalist Approach to LLM Reasoning: From Rejection Sampling to Reinforce.” *arXiv Preprint arXiv:2504.11343*. https://arxiv.org/abs/2504.11343.

Yu, Qiying, Zheng Zhang, Ruofei Zhu, et al. 2025. “DAPO: An Open-Source LLM Reinforcement Learning System at Scale.” *arXiv Preprint arXiv:2503.14476*. https://arxiv.org/abs/2503.14476.

Yuan, Zheng, Hongyi Yuan, Chengpeng Li, et al. 2023. “Scaling Relationship on Learning Mathematical Reasoning with Large Language Models.” *arXiv Preprint arXiv:2308.01825*. https://arxiv.org/abs/2308.01825.

Zuo, Simiao, Chenhui Xu, Yimeng Jia, Qiang Lou, Jian Jiao, and Denis Charles. 2026. “AdsWorldEngine: A Self-Evolving Conversational Advertising Agent Through Orchestrator and Tool Coevolution.” *arXiv Preprint arXiv:2608.13833*. https://arxiv.org/abs/2608.13833.

Appendix

A1. Cost matrix (default)

Rows: true label; columns: action (C = critical intent, R = routine intent, MR = manual review). Entries are $C(y, a)$ from Eq. (3) with $c_{\text{crit}} = 10$, $c_{\text{esc}} = 2$, $c_{\text{rr}} = 1$, $c_{\text{oos}} = 3$, $c_{\text{rev}} = 0.5$.

True \ action	same intent	other critical	routine	MANUAL_REVIEW
critical intent	0	10	10	0.5
routine intent	0	2	1	0.5
out-of-scope	—	3	3	0.5

Sensitivity analyses use $c_{\text{crit}} \in \{5, 10, 20\} \times c_{\text{rev}} \in \{0.25, 0.5, 1.0\}$, with the other entries unchanged.

A2. Hyperparameters

Setting	Value
Model	`EleutherAI/pythia-14m`, revision `cf967c0a9a04383db6f7b1108d86b2962634b4ac`, 14,067,712 parameters, all trained, float32
Prompt	`Banking customer message: {text}\nRouting code:`; message truncated to 64 tokens
Action tokens	" A" … " H" → intents 0–7, " I" → MANUAL_REVIEW (token ids 329, 378, 330, 399, 444, 401, 443, 388, 309)
Optimiser	AdamW, $\beta = (0.9, 0.999)$, weight decay 0, gradient-norm clip 1.0
SFT / WSFT	lr 10^{-4}, batch 16, 10 epochs, min. 60 steps, final checkpoint
Warm start	SFT on the first 64 stream examples, 60 steps
GRPO	lr 10^{-5}, 16 prompts per batch, $\mu = 2$, clip $\varepsilon = 0.2$, KL $\beta = 0.04$ (exact, to frozen warm start), temperature 1, $\epsilon = 10^{-6}$, homogeneity tolerance 10^{-8}
Cost reward	$R = -C(y, a)/c_{\text{crit}}$ (range $[-1, 0]$)
RFT	same sampling as GRPO; NLL on samples with reward 1, normalised by PG
TF-IDF/LR	word 1–2-grams + char_wb 2–5-grams (min df 2), sublinear TF; L2 logistic regression, $C = 10$, lbfgs, 5,000 iterations
Seeds	main study 0–4; pilot 100; data split 1234
Bootstrap	10,000 resamples; Holm correction per statistic and rule

A3. Per-seed results

Accuracy / expected cost per seed (argmax rule). All per-seed metrics for all 300 runs (255 with 14M, 45 with 31M) are in `results/tables/per_run_metrics.csv`.

Method	seed 0	seed 1	seed 2	seed 3	seed 4
SFT n=1106	0.787 / 0.919	0.783 / 0.978	0.790 / 0.867	0.794 / 0.823	0.779 / 0.959
WSFT-cost n=1106	0.802 / 0.774	0.754 / 1.022	0.773 / 0.946	0.765 / 1.056	0.800 / 0.984
GRPO-bin G=4 R=8192	0.581 / 2.139	0.600 / 1.782	0.610 / 1.543	0.579 / 1.700	0.667 / 1.390
GRPO-cost G=4 R=8192	0.608 / 1.854	0.635 / 1.605	0.619 / 1.222	0.635 / 1.373	0.688 / 0.971
GRPO-cost G=8 R=8192	0.627 / 1.401	0.683 / 0.908	0.606 / 0.882	0.635 / 1.369	0.667 / 1.220
RFT G=4 R=8192	0.721 / 0.833	0.679 / 1.191	0.660 / 1.546	0.654 / 1.506	0.642 / 1.517

A4. Environment

Python 3.12.10; PyTorch 2.13.0+cpu; Transformers 5.14.1; Hugging Face Hub 1.33.0; Datasets 5.2.0; Tokenizers 0.22.2; NumPy 2.5.1; pandas 3.0.5; scikit-learn 1.9.0; SciPy 1.18.0; Matplotlib 3.11.1. Hardware: AMD64 Family 25 Model 116 (6 physical / 12 logical cores), 32 GB RAM, Windows 11 Pro; no GPU was used. The full environment record is `environment.txt` in the repository.

A5. Data counts

Split	Critical	Routine	Out-of-scope	Total	OOS intents
Train	404	464	238	1,106	35 train/val-only intents
Validation	71	83	80	234	same 35 intents
Test	160	160	160	480	34 test-only intents

Per-intent counts, intent lists, and file hashes: `data/splits/manifest.json`.