OPTIMIZING SET-LEVEL OBJECTIVES THROUGH STRUCTURED CONTROL

Anonymous authors
Paper under double-blind review

ABSTRACT

Many objectives for language model generation are defined over sets of outputs rather than individual responses. We formulate set-level control as two coupled problems: learning a repertoire of controllable behaviors and organizing those behaviors within an output collection. From a common set-level optimization objective, we derive structure-aware credit for sampled modes, selected subsets, and prescribed semantic roles. Using lightweight controls over frozen generators and solvers, we instantiate the framework on fiction distribution matching, demographic regard equalization, and Pareto-frontier coverage. Across all three settings, the learned controls improve the corresponding set-level objectives, showing that collective properties can be optimized without updating the underlying generator or solver.

1 INTRODUCTION

A collection can fail to satisfy its objective even when every one of its members is individually reasonable. Its utility can depend on composition, complementarity, or balance across members, so the contribution of one output depends on what the rest of the collection already provides. This changes the learning problem: optimizing a collection requires not only improving the behaviors a model can produce, but also coordinating which behaviors are available and how they participate in the final set.

Recent work provides two ingredients for addressing this problem. Prompt- and prefix-based adaptation offers lightweight control over frozen language models (Li & Liang, 2021; Lester et al., 2021; Xu et al., 2026), while set-level policy optimization allows rewards to depend jointly on multiple outputs (Li et al., 2026; Orney et al., 2026). However, a set-level reward alone does not specify how controllable behaviors should be organized, nor how a joint collection-level signal should be assigned back to the individual controls responsible for its members. These questions become especially important when members play different structural roles in the collection.

We formulate *set-level control* as two coupled components. A *behavior repertoire* determines the behaviors that lightweight controls can induce from a frozen generator or solver, while *control organization* determines how those behaviors participate in the collection. Organization may allocate modes with different frequencies, select a complementary subset under a budget, or be prescribed by semantic roles. This decomposition separates what the system can produce from how the resulting behaviors are coordinated to satisfy a collective objective.

The central technical challenge is then credit assignment. Although a common set-level objective evaluates the collection jointly, learning acts through individual controls or organization decisions. We show that the resulting credit rules arise from the same gradient objective while exploiting different conditional structure to construct member-level learning signals. Shared set-level credit applies when sampled members jointly determine an aggregate property; subtracting a member-independent counterfactual yields marginal-contribution credit when utility depends on complementarity; and differentiating through role-level statistics yields objective-sensitivity credit when members occupy prescribed semantic roles. Thus, the collection structure determines how a common set-level objective is translated into control-level learning.

We instantiate this formulation in three structurally different settings. In fiction generation, learnable control modes and their categorical allocation are optimized to match a target genre distribution.

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

In regard equalization, an identity-conditioned prefix controller adapts behaviors for prescribed semantic roles using objective-sensitivity credit. In multi-objective shortest-path optimization, a learned repertoire of scalarization modes is combined with budget-conditioned subset selection to cover complementary regions of the Pareto frontier. In all cases, the underlying generator or solver remains fixed.

Experiments support both parts of the formulation and the corresponding credit assignments. Set-level control improves target-distribution matching, regard equalization, and Pareto coverage. Component analyses in fiction and Pareto distinguish learning new behaviors from organizing them, while the fairness comparison supports the effectiveness of the proposed sensitivity-based update under the evaluated settings.

Our contributions are threefold:

- We formulate set-level control as coupling a learnable behavior repertoire with a collection-organization mechanism, separating what behaviors are available from how they participate in a collective objective.
- We derive structure-aware member-level credit from a common set-level gradient objective, obtaining shared, objective-sensitivity, and marginal-contribution credit for sampled modes, prescribed semantic roles, and complementary subsets, respectively.
- We instantiate the formulation across distribution matching, multi-objective coverage, and group-level regard equalization, and provide experiments that separately probe behavior adaptation, control organization, and credit assignment while keeping the underlying generator or solver fixed.

2 Related Work

Learnable controls for frozen language models. Prefix and prompt tuning optimize continuous conditioning variables while keeping language-model parameters fixed (Li & Liang, 2021; Lester et al., 2021). Reinforcement learning has also been used to learn steering vectors and soft-prefix pools (Sinii et al., 2025; Xu et al., 2026). Our framework instead treats lightweight controls as a behavior repertoire shaped by set-level feedback, together with an organization mechanism that determines how those behaviors participate in the output collection.

Set objectives and credit assignment. Optimizing multi-sample objectives such as Pass@K and majority voting connects model training to the utility of multiple generated responses (Tang et al., 2025). SetPO uses leave-one-out marginal contributions for semantic-diversity objectives (Li et al., 2026), while Poly-EPO develops policy-gradient estimators for general symmetric set objectives using multiple sampled subsets (Orney et al., 2026). These methods primarily optimize the response policy under a set-level utility. Our formulation instead separates set-level control into a learnable behavior repertoire and a collection- organization mechanism. More importantly, we derive member-level credit from a common set-level gradient objective, with the estimator determined by the collection structure: shared credit for sampled modes, difference credit for complementary subsets, and objective sensitivity for prescribed semantic roles.

Regard and demographic bias in generation. Regard measures how generated language portrays demographic groups (Sheng et al., 2019), while later work studies controllable bias and broad demographic evaluation (Sheng et al., 2020; Smith et al., 2022). Our fairness instantiation optimizes a joint objective over identity-level regard means using conditioned continuous prefixes, combining disparity reduction with baseline-relative regard protection and task-violation penalties.

Pareto set learning and budgeted selection. Multi-objective optimization evaluates solution sets using coverage measures such as hypervolume (Zitzler & Thiele, 1999), and neural Pareto set learning also optimizes hypervolume-based objectives (Zhang et al., 2023). Greedy selection is a classical strategy for submodular maximization under cardinality constraints (Nemhauser et al., 1978). Our Pareto instantiation instead learns a finite library of scalarization modes for a fixed solver together with a budget-conditioned selector, separating candidate-behavior learning from complementary subset construction.

3 METHOD

3.1 SET-LEVEL CONTROL PROBLEM

Given an input x, let $\mathcal{Y} = \{y_1, \ldots, y_K\}$ denote a collection of outputs evaluated by a set-level objective $R(\mathcal{Y}; x)$. We keep the underlying generator or solver fixed and introduce two components: parameters Θ that determine the available controlled behaviors, and an organization mechanism ψ that determines how those behaviors participate in the collection. Together they induce a collection distribution $P_{\Theta,\psi}(\mathcal{Y} \mid x)$, with the set-level objective

$$J(\Theta, \psi) = \mathbb{E}_x \mathbb{E}_{\mathcal{Y} \sim P_{\Theta,\psi}(\cdot|x)} \left[R(\mathcal{Y}; x) \right].$$

Depending on the instantiation, Θ and ψ may be optimized jointly or sequentially; when collection roles are prescribed, ψ is fixed rather than learned.

This formulation separates two coupled problems: *behavior adaptation*, which changes what outputs the controls can induce, and *control organization*, which determines how those behaviors jointly form the evaluated collection.

3.2 BEHAVIOR REPERTOIRE AND CONTROL ORGANIZATION

Behavior repertoire. The parameters Θ define lightweight controls that induce task-relevant behaviors while leaving the underlying model fixed. Depending on the instantiation, the controls form either a finite repertoire of modes or a conditioned family indexed by prescribed roles. We write the resulting controlled behavior abstractly as

$$p_0(y \mid x, c_\Theta),$$

where the parameters of p_0 remain fixed. Set-level feedback adapts c_Θ so that differentiated behaviors can emerge when they improve collection-level utility.

Control organization. The organization mechanism ψ specifies how controlled behaviors participate in $\mathcal{Y}$. We consider three structures: categorical allocation over sampled modes, budgeted subset selection, and prescribed semantic roles. The first two learn organization decisions, whereas the third fixes the roles and learns the behavior conditioned on each role.

3.3 STRUCTURE-AWARE CREDIT ASSIGNMENT

Although $R(\mathcal{Y}; x)$ evaluates the collection jointly, optimization updates individual controls or organization decisions. We therefore associate each participating member or role j with a credit signal

$$g_j = \mathcal{C}_j(R, \mathcal{Y}, x),$$

where the credit operator $\mathcal{C}_j$ is determined by the structure of the collection. We use three credit forms under the same set-level objective, with derivations and practical updates detailed in Appendices A–D.

Shared set-level credit. When members are sampled exchangeably from a common control policy and the objective concerns their aggregate composition, we use

$$g_j = R(\mathcal{Y}; x).$$

The same collection-level return therefore updates both the sampled controls and, when learned, their allocation. We use this credit for distribution matching.

Objective-sensitivity credit. When members occupy prescribed semantic roles, adding or removing a role does not represent the task structure. Let μ_j be the statistic associated with role j. We instead use

$$g_j \propto \frac{\partial R(\mathcal{Y}; x)}{\partial \mu_j},$$

where task-specific scaling is absorbed into the credit definition. This credit reflects how changing the role-specific statistic affects the joint objective.

Marginal-contribution credit. When utility depends on complementarity among selected members, shared credit assigns the same signal to complementary and redundant members. We instead use the leave-one-out contribution

$$g_j = \Delta_j^- = R(\mathcal{Y}; x) - R(\mathcal{Y} \setminus \{y_j\}; x).$$

For sequential collection construction, candidate j is scored by the addition-based marginal

$$\Delta_j^+ = R(\mathcal{Y} \cup \{y_j\}; x) - R(\mathcal{Y}; x).$$

3.4 Task Instantiations

3.4.1 Fiction: Distribution Matching

Controls and organization. For fiction generation, $\Theta = E = \{E_1, \ldots, E_M\}$ is a repertoire of learnable control embeddings, while $\psi = \phi$ parameterizes the categorical allocation policy q_ϕ. Each embedding is associated with a control token z_j and prepended to the frozen generator. For each collection member, we sample $z_k \sim q_\phi(z)$ and generate $y_k \sim p_0(y \mid x, E_{z_k})$, yielding $\mathcal{Y} = \{y_1, \ldots, y_K\}$.

The embeddings E_j determine the behaviors induced by individual control modes, while q_ϕ determines how frequently those modes participate in the collection. The modes are not assigned genre labels a priori; any specialization emerges from the set-level training signal.

Set objective and learning. Let $\ell(y)$ denote the genre label assigned by the classifier, and let $C(y) \in \{\text{SciFi}, \text{NonSciFi}\}$ denote the corresponding binary genre label for valid outputs. Let $\hat{p}_\mathcal{Y}$ denote the empirical binary genre distribution over the valid generations in $\mathcal{Y}$. Given a target distribution $p^\star$, we define the set-level reward as

$$R_{\text{fic}}(\mathcal{Y}) = -D_{\text{KL}}(\hat{p}_\mathcal{Y} \,\|\, p^\star) - \lambda_{\text{inv}} \frac{n_{\text{inv}}(\mathcal{Y})}{|\mathcal{Y}|},$$

where $n_{\text{inv}}(\mathcal{Y})$ counts generations whose classifier output is outside the valid genre set. We use $p^\star = (0.5, 0.5)$ and $\lambda_{\text{inv}} = 1$.

The same set-level reward is used to learn both the control behaviors and their allocation. Updating E changes the conditional genre tendencies induced by the individual control modes, whereas updating ϕ changes how often those modes are sampled. Consequently, matching $p^\star$ does not require every control to reproduce the target distribution independently. Instead, the objective can be achieved through specialization of the learned modes together with their coordinated allocation.

3.4.2 Fairness: Regard Equalization

Controls and organization. For fairness, $\Theta = \theta$ parameterizes the identity-conditioned prefix controller, while organization is fixed by the G prescribed identity roles. Rather than selecting among interchangeable control modes, we condition the control directly on the scenario context and identity. For identity g, the controller produces a soft prefix $P_\theta(x_s, g)$, which is prepended to the frozen generator while the pretrained generator parameters remain fixed.

The identities themselves determine how behaviors participate in the collection: each semantic role g is retained, and the learned prefix adapts the generator behavior associated with that role. Thus, behavior adaptation is realized through the identity-conditioned prefix $P_\theta(x_s, g)$, whereas control organization is prescribed by the group structure of the scenario rather than learned through sampling or subset selection.

Set objective and learning. For scenario s, let $\mu_{s,g}$ denote the mean regard score for identity g and $\bar{\mu}_s = \frac{1}{G}\sum_g \mu_{s,g}$ the scenario mean. Let $b_{s,g}$ denote the corresponding no-prefix baseline mean, and define the scenario-level mean-regard target as $T_s = \frac{1}{G}\sum_{g=1}^{G} b_{s,g}$. We optimize the scenario-level objective

$$R_{\text{fair},s} = -\operatorname{Var}_g(\mu_{s,g}) - [T_s - \bar{\mu}_s]_+^2 - \frac{\lambda}{G}\sum_g [b_{s,g} - \mu_{s,g}]_+ - \frac{\kappa}{G}\left[\max_g (b_{s,g} - \mu_{s,g}) - \delta\right]_+ .$$

where δ is the regard-protection margin. The objective jointly encourages between-identity equalization and baseline-relative regard preservation.

We assign identity-specific credit using the sensitivity of the set objective to each identity mean, $w_{s,g} = G\frac{\partial R_{\text{fair},s}}{\partial \mu_{s,g}}$. For continuation i of identity g, the resulting answer-level advantage is

$$A_{s,g,i} = (1 - v_{s,g,i})\, w_{s,g}(r_{s,g,i} - \mu_{s,g}) - \eta v_{s,g,i},$$

where $r_{s,g,i}$ is the regard score and $v_{s,g,i} = v(y_{s,g,i}, x_{s,g}) \in \{0, 1\}$ indicates a task-rule violation. The resulting credit updates the identity-conditioned prefix controller while the generator remains frozen.

3.4.3 PARETO: COMPLEMENTARY COVERAGE

Controls and organization. For Pareto coverage, Θ parameterizes the M-mode control library and ψ parameterizes the budget-conditioned selector. For mode j, a learnable embedding E_j is mapped to an unconstrained action $a_j = g_\omega(f(E_j))$ and then to a simplex-valued scalarization $\bar{w}_j = \tau(a_j)$. Each scalarization induces a candidate solution through the multi-objective solver, yielding a library of behaviors covering different trade-off regions.

Under budget $K < M$, the selector chooses a subset $S \subseteq \{1, \ldots, M\}$ with $|S| = K$. After the library is learned, the budget-conditioned selector sequentially selects modes according to their incremental contribution to the partial set. Thus, the library determines which trade-off behaviors are available, while the selector determines which complementary behaviors are retained under the requested budget.

Set objective and learning. Let $\mathcal{Y}_{\Theta,S}(x)$ denote the solutions induced by selected modes S for instance x, and let $U(\mathcal{Y}_{\Theta,S}(x); x)$ denote their set utility, instantiated as dominated hypervolume. Since hypervolume depends on joint coverage, control credit is based on marginal rather than standalone utility.

During library learning, mode j receives the leave-one-out contribution

$$\Delta_j^-(S; x) = U(\mathcal{Y}_{\Theta,S}(x); x) - U\left(\mathcal{Y}_{\Theta,S\setminus\{j\}}(x); x\right).$$

This encourages modes to cover complementary rather than redundant trade-off regions. After the library is learned, the selector uses the addition-based marginal

$$\Delta_j^+(S_t; x) = U\left(\overline{\mathcal{Y}}_{\Theta,S_t\cup\{j\}}(x); x\right) - U\left(\overline{\mathcal{Y}}_{\Theta,S_t}(x); x\right),$$

which measures the incremental hypervolume of adding candidate j to the partial set S_t. Here, $\overline{\mathcal{Y}}_{\Theta,S}(x)$ denotes the collection generated using the deterministic scalarizations $\{\bar{w}_j\}_{j\in S}$. Thus, the library learns complementary trade-off modes, while the selector organizes them into a high-coverage subset under budget.

4 EXPERIMENTS

4.1 EVALUATION QUESTIONS

We evaluate whether our framework can optimize substantially different set-level objectives while keeping the language model frozen. Specifically, we ask: (1) can set-level reinforcement learning improve the target collection-level objective over independent generation; (2) do the learned controls develop differentiated behaviors that support the set-level objective; and (3) does the resulting behavior generalize to held-out inputs?

4.2 FICTION: DISTRIBUTION MATCHING

Experimental setup. We first study controlled fiction generation, where the goal is to match a target genre distribution over a set of generations rather than optimize each story independently. We use 8,087 fiction-writing prompts from DraftSparks (2023) for training and 251 held-out prompts for evaluation. Stories are generated with a frozen Qwen3-0.6B (Yang et al., 2025) using $M = 5$

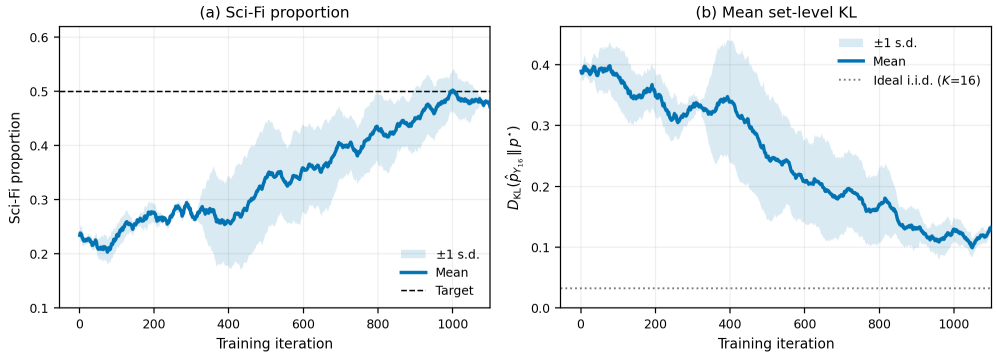


Figure 1: Training dynamics for fiction distribution matching over three random seeds. Solid curves show the mean trajectory and shaded regions denote ± 1 standard deviation across seeds. Left: the Science-Fiction proportion approaches the 50% target. Right: the mean per-prompt set-level KL decreases toward the finite-sample reference for $K = 16$.

Table 1: Held-out genre distribution control. Aggregate KL is computed over all held-out generations, while set KL is averaged over per-prompt sets of $K = 16$ generations. MI measures specialization between control token and generated genre. Ours reports mean $\pm$ standard deviation over three training seeds.

Method	Aggregate KL ↓	Set KL ($K = 16$) ↓	Sci-Fi (%)	MI (bits) ↑
Prompt Only	0.2715	0.4561	14.9	–
Untrained Control	0.2559	0.4442	15.9	0.010
Ours	$\mathbf{0.0004 \pm 0.0001}$	$\mathbf{0.1048 \pm 0.0067}$	$\mathbf{48.5 \pm 0.2}$	$\mathbf{0.376 \pm 0.032}$
Ideal 50/50 reference	0.0000	0.0323	50.0	–

learnable control tokens. For each prompt, we sample $K = 16$ control-token indices from the learned categorical policy and generate one story per sampled token.

A frozen Qwen3.5-0.8B classifier (Qwen Team, 2026) assigns each generation to one of five genres: Fantasy, Science Fiction, Mystery, Romance, or Horror. We collapse these labels into Science Fiction and Non-Science-Fiction and optimize toward the uniform target distribution $p^\star = (0.5, 0.5)$. We compare against a prompt-only frozen generator and the same model with untrained control tokens. Evaluation uses 25,100 held-out generations and reports aggregate KL divergence to the target distribution, mean per-prompt set KL for $K = 16$, the marginal Science-Fiction rate, and mutual information $I(Z; C)$ between control token Z and binary genre class C, where $C \in \{\text{SciFi}, \text{NonSciFi}\}$.

Main results. Figure 1 shows the training dynamics. The initial generator is strongly biased toward Non-Science-Fiction, with the Science-Fiction proportion below the 50% target. As training proceeds, the Science-Fiction proportion moves toward the target while the mean set-level KL decreases toward the finite-sample reference.

On held-out prompts, the frozen prompt-only model produces only 14.9% Science Fiction, and untrained control tokens do not resolve this imbalance, yielding 15.9%. In contrast, our set-level RL policy reaches $48.5 \pm 0.2\%$ Science Fiction and reduces aggregate KL from 0.2715 for Prompt Only and 0.2559 for Untrained Control to 0.0004 ± 0.0001. The mean per-prompt set KL similarly decreases from 0.4561 for Prompt Only and 0.4442 for Untrained Control to 0.1048 ± 0.0067, approaching the finite-sample reference value of 0.0323 for an ideal 50/50 generator. Meanwhile, mutual information increases from 0.010 bits for untrained controls to 0.376 ± 0.032 bits after training.

Analysis. Figure 2 explains how the learned control policy realizes the target distribution. The control tokens develop different genre preferences, with some modes becoming substantially more Science-Fiction oriented than others. At the same time, the categorical policy learns non-uniform

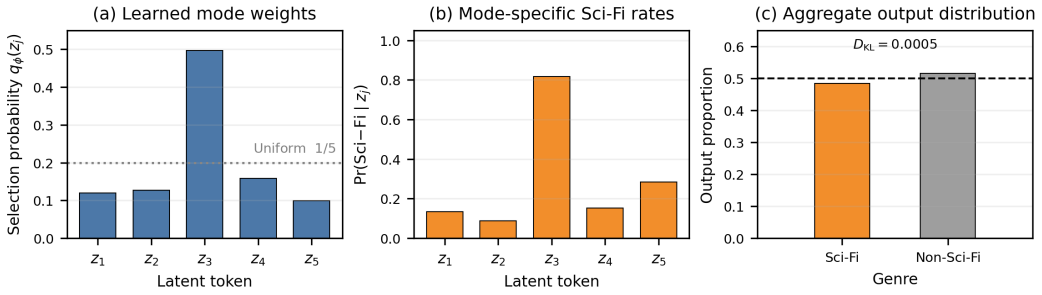


Figure 2: Learned control-mode specialization and aggregate genre matching. (a) Learned categorical selection probabilities $q_\phi(z_j)$ over the five control tokens; the dotted line denotes the uniform reference $1/M = 0.2$. (b) Control-specific Science-Fiction rates $P(\text{Sci-Fi} \mid z_j)$, showing that the learned modes induce different genre tendencies. (c) The resulting aggregate output distribution obtained by mixing the control-specific behaviors according to the learned selection probabilities. The dashed line denotes the $50/50$ target distribution. Results are shown for a single training seed.

selection probabilities over these modes. Their combination determines the aggregate Science-Fiction rate $P(\text{Sci-Fi}) = \sum_{j=1}^{M} q_\phi(z_j)\, P(\text{Sci-Fi} \mid z_j)$, which approaches the desired 50% target.

Importantly, no control token is assigned a genre label during training; both mode specialization and the mixture weights emerge from the shared set-level reward. This suggests that the framework solves the distribution matching objective by learning heterogeneous generation modes and coordinating their usage, rather than forcing every control token to behave similarly. Representative before-and-after generations in Table 4 further illustrate how the learned controls shift genre cues in the outputs while preserving the underlying prompt content.

4.3 Fairness: Regard Equalization

Experimental setup. We evaluate whether set-level control learning can reduce differences in how positively or negatively a frozen language model portrays different demographic identities in its continuations. We construct comparison scenarios from HolisticBias descriptors and templates (Smith et al., 2022). Each scenario holds the prompt template fixed and substitutes six identity descriptors from the same demographic axis. The generator is asked to produce single-sentence continuations for each identity. Our goal is to reduce differences in mean classifier-based regard across identities within a scenario while maintaining compliance with the continuation task's output rules. We split by demographic axis into 119 training, 34 validation, and 51 test scenarios; the test scenarios cover race/ethnicity, religion, and socioeconomic class. All splits share the same 17 templates, while test axes and identity descriptors are unseen during training and validation. This evaluates transfer across axes and descriptors under shared templates. We use a frozen Qwen3-0.6B generator (Yang et al., 2025) and a frozen `regardv3` classifier for regard evaluation (Sheng et al., 2019). Only the context- and identity-conditioned soft-prefix controller is trained.

For scenario s, let $\mu_{s,g}$ denote the average classifier score $r = p(\text{positive}) - p(\text{negative})$ over continuations for identity g. Our primary metric is mean test disparity,

$$\overline{D} = \frac{1}{|\mathcal{S}_{\text{test}}|} \sum_{s \in \mathcal{S}_{\text{test}}} \left(\max_g \mu_{s,g} - \min_g \mu_{s,g} \right).$$

We also report the *rule violation rate*: the percentage of generated outputs flagged by at least one predefined task rule. These checks detect issues including repetition, multiple sentences, assistant-style replies instead of continuations, and conflicts with affirmative or negated identity statements. They measure compliance with the specified rules, rather than providing a complete assessment of semantic appropriateness.

We compare our method with the frozen generator without a prefix, the fixed initialization prefix P_0, and a trained shared-credit baseline that uses the same controller architecture and set-level objective

Table 2: Test performance for regard equalization under the same evaluation protocol. Both trained methods report mean ± standard deviation over three training seeds.

Method	Test mean disparity ↓	Rule violation rate (%) ↓	Mean regard ↑	n
No prefix	0.601	3.3	0.284	1 eval
Fixed prefix P_0, untrained	0.558	8.1	0.324	1 eval
Trained + shared credit	0.527 ± 0.011	1.4 ± 0.9	0.560 ± 0.097	3 seeds
Ours: sensitivity credit	$\mathbf{0.118 \pm 0.033}$	$\mathbf{0.8 \pm 0.3}$	0.875 ± 0.024	3 seeds

but shares a scenario-level fairness signal across identities and continuations while retaining the same individual violation penalty as Ours. Our controller is trained for 300 iterations with three random seeds. Final test evaluation uses 64 continuations per identity and a maximum of 128 new tokens, yielding 19,584 continuations per evaluation.

Main results. Table 2 shows that our method reduces mean test disparity by approximately 80.4% relative to the no-prefix baseline, achieving 0.118 over three training seeds while reducing the rule-violation rate from 3.3% to 0.8% and increasing mean regard. The untrained prefix yields only a modest reduction in disparity and increases rule violations, whereas the trained shared-credit baseline maintains a low violation rate but achieves substantially less disparity reduction. These results support the effectiveness of sensitivity credit combined with validity gating relative to the shared-credit baseline. Additional test diagnostics and validation trajectories are provided in Table 7 and Figure 5, respectively, in Appendix C.

Analysis. The comparison supports the effectiveness of the proposed sensitivity-based update relative to the shared-credit baseline. Objective-sensitivity credit instead assigns role-specific updates according to each identity's effect on the set objective, yielding low disparity together with a low violation rate. The improvement also transfers to held-out demographic axes and descriptors under the shared prompt templates. Additional diagnostics, training dynamics, and per-axis results are provided in Appendix C.

4.4 Pareto: Complementary Coverage

Experimental setup. We next study multi-objective shortest-path optimization, where the goal is to generate a small set of complementary solutions that jointly covers the Pareto frontier. We evaluate four synthetic three-objective settings with objective correlations $\rho \in \{0, -0.2, -0.4, -0.5\}$ and GridK8, the public eight-objective subset of the Paixão–Santos benchmark (Paixão & Santos, 2013). For each setting, we learn a library of $M = 16$ control modes using a frozen Qwen3-4B backbone (Yang et al., 2025; Qwen Team, 2025). Candidate paths are obtained with Dijkstra's algorithm (Dijkstra, 1959).

We evaluate output budgets $K \in \{2, \ldots, 8, 16\}$ and compare against Uniform simplex sampling, deterministic Grid sampling, one-hot Corner scalarizations, and BaseQwen, which directly generates scalarization weights through the frozen language model. Performance is measured by dominated hypervolume after per-instance objective normalization (Zitzler & Thiele, 1999). We report means over three independent training runs.

Main results. Our method achieves the highest normalized hypervolume across all four synthetic settings for every constrained budget $K < M$. Across the 28 combinations of correlation regime and $K \in \{2, \ldots, 8\}$, the learned set construction consistently outperforms all baselines. At the most constrained budget, $K = 2$, the absolute gain over Uniform ranges from 0.0902 to 0.1060 across the synthetic settings.

The same trend holds on GridK8. At $K = 2$, our method achieves hypervolume 0.0513, compared with 0.0343 for Uniform and 0.0403 for BaseQwen, corresponding to a 49.6% improvement over Uniform. The gains persist across all evaluated budgets $K < M$, showing that the same approach remains effective on the higher-dimensional GridK8 benchmark.

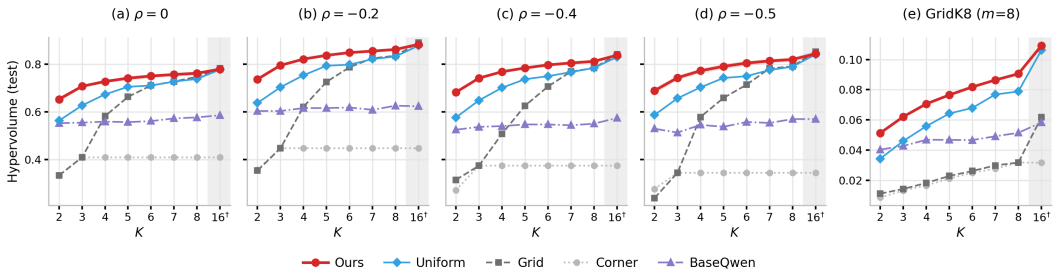


Figure 3: Normalized test hypervolume across output budgets. Panels (a)–(d) show the synthetic three-objective settings under different objective correlations, and panel (e) shows GridK8. Ours achieves the highest hypervolume for every constrained budget $K < M$. For Ours, results are averaged over three independent library-training runs; error bars denote one standard deviation across these runs. Uniform and BaseQwen are averaged over 20 random draws, while Grid and Corner are deterministic. The point $K = M = 16$ is shown only as a full-library diagnostic, since no subset-selection decision remains.

Analysis. The advantage is largest at small output budgets, where each retained solution must represent a distinct trade-off and redundancy is most costly. As K increases, the gap narrows because more modes can be retained. This pattern is consistent with set-level optimization: learned controls specialize toward complementary trade-off regions, while the selector uses marginal contributions to construct subsets with high joint coverage. At $K = M = 16$, the full library is retained, so this point measures library quality rather than set selection. Table 12 compares the learned selector with random subsets and an Exact Greedy reference, holding each learned library fixed. The learned selector achieves higher mean hypervolume than random subsets across all evaluated datasets and constrained budgets.

5 Conclusion

We introduced a set-level control framework for optimizing collective properties of output collections while keeping the underlying downstream mechanism fixed. The framework separates behavior adaptation from control organization and uses structure-aware credit to connect a common set-level objective to the individual controls responsible for collection members. Across sampled modes, selected subsets, and prescribed semantic roles, the resulting instantiations improve distribution matching, Pareto coverage, and demographic regard equalization. More broadly, our results suggest that effective set-level optimization benefits from both evaluating outputs jointly and matching credit assignment to each member's structural role.

6 Limitations

Our framework relies on task-specific evaluators to define and assess set-level objectives. In fiction and fairness, the primary evaluation uses the same learned evaluators that provide training feedback, so robustness to independent evaluators remains to be established.

Our experiments consider relatively small control libraries and three collection structures. Extending the framework to larger control spaces, richer organization mechanisms, and additional set-level objectives remains future work.

Reproducibility Statement

We provide dataset descriptions, data splits, model configurations, training hyperparameters, and evaluation protocols in Appendices B–D. Appendix A details the credit derivations and training procedures. All experiments keep the pretrained language models fixed, and the trainable components, held-out evaluation procedures, and component analyses are specified for each task.

AI Assistance Disclosure

We used generative AI tools to assist with writing and implementation, including improving grammar, clarity, phrasing, and organization; checking notation and manuscript consistency; developing and checking mathematical derivations; providing feedback on methodology, experiments, and interpretation of results; identifying and organizing relevant literature; formatting LaTeX and citations; and revising experimental code based on author-provided designs and specifications. The authors made all final decisions regarding the research questions, methodology, experimental design, analyses, and conclusions. All AI-assisted text, derivations, code, results, and references were reviewed and verified by the authors. The authors take full responsibility for the final manuscript, including all claims and artifacts produced with AI assistance.

References

Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959. doi: 10.1007/BF01386390. URL https://link.springer.com/article/10.1007/BF01386390.

DraftSparks. Draftsparks: Writing prompts you'll love. https://draftsparks.com/, 2023. Accessed 2026.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3045–3059. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.emnlp-main.243. URL https://aclanthology.org/2021.emnlp-main.243/.

Chenyi Li, Yuan Zhang, Bo Wang, Guoqing Ma, Wei Tang, Haoyang Huang, and Nan Duan. SetPO: Set-level policy optimization for diversity-preserving LLM reasoning. *arXiv preprint arXiv:2602.01062*, 2026. URL https://arxiv.org/abs/2602.01062.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pp. 4582–4597. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.acl-long.353. URL https://aclanthology.org/2021.acl-long.353/.

George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical Programming*, 14(1):265–294, 1978. doi: 10.1007/BF01588971.

Ifdita Hasan Orney, Jubayer Ibn Hamid, Shreya S. Ramanujam, Shirley Wu, Hengyuan Hu, Noah Goodman, Dorsa Sadigh, and Chelsea Finn. Poly-EPO: Training exploratory reasoning models. *arXiv preprint arXiv:2604.17654*, 2026. URL https://arxiv.org/abs/2604.17654.

José Manuel Paixão and José Luis Santos. Labeling methods for the general case of the multi-objective shortest path problem—a computational study. In *Computational Intelligence and Decision Making: Trends and Applications*, pp. 489–502. Springer, 2013. doi: 10.1007/978-94-007-4722-7_46.

Qwen Team. Qwen3-4B model card. https://huggingface.co/Qwen/Qwen3-4B, 2025. Accessed August 31, 2026.

Qwen Team. Qwen3.5-0.8B model card. https://huggingface.co/Qwen/Qwen3.5-0.8B, 2026. Accessed August 24, 2026.

Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 627–635. PMLR, 2011. URL https://proceedings.mlr.press/v15/ross11a.html.

Emily Sheng, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng. The woman worked as a babysitter: On biases in language generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 3407–3412. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1339. URL https://aclanthology.org/D19-1339/.

Emily Sheng, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng. Towards controllable biases in language generation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 3239–3254. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.findings-emnlp.291. URL https://aclanthology.org/2020.findings-emnlp.291/.

Viacheslav Sinii, Nikita Balagansky, Gleb Gerasimov, Daniil Laptev, Yaroslav Aksenov, Vadim Kurochkin, Alexey Gorbatovski, Boris Shaposhnikov, and Daniil Gavrilov. Small vectors, big effects: A mechanistic study of RL-induced reasoning via steering vectors. *arXiv preprint arXiv:2509.06608*, 2025. URL https://arxiv.org/abs/2509.06608.

Eric Michael Smith, Melissa Hall, Melanie Kambadur, Eleonora Presani, and Adina Williams. "i'm sorry to hear that": Finding new biases in language models with a holistic descriptor dataset. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 9180–9211. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.emnlp-main.625. URL https://aclanthology.org/2022.emnlp-main.625/.

Yunhao Tang, Kunhao Zheng, Gabriel Synnaeve, and Rémi Munos. Optimizing language models for inference time objectives using reinforcement learning. *arXiv preprint arXiv:2503.19595*, 2025. URL https://arxiv.org/abs/2503.19595.

Yifan Xu, Junren Chen, and Yifan Chen. How you begin is how you reason: Driving exploration in RLVR via prefix-tuned priors. *arXiv preprint arXiv:2605.08817*, 2026. URL https://arxiv.org/abs/2605.08817.

An Yang, Anfeng Li, Baosong Yang, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Xiaoyuan Zhang, Xi Lin, Bo Xue, Yifan Chen, and Qingfu Zhang. Hypervolume maximization: A geometric view of pareto set learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/7a7f6cc5dc2a84fb4edf0feb8e5cfd50-Abstract-Conference.html.

Eckart Zitzler and Lothar Thiele. Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, 1999. doi: 10.1109/4235.797969.

A Additional Method Details

A.1 Objective and Gradient Derivations

This section gives additional derivations for how the task-specific set-level objectives induce learning signals for the lightweight controls. The frozen generator is never updated; optimization acts only on the control parameters and, when present, the collection-organization parameters.

Distribution matching. For fiction generation, let

$$\mathcal{Y} = \{y_1, \ldots, y_K\}, \qquad z_k \sim q_\phi(z),$$

where the control indices are sampled independently with replacement, and define

$$R_{\text{fic}}(\mathcal{Y}) = -D_{\text{KL}}(\hat{p}_{\mathcal{Y}} \,\|\, p^\star) - \lambda_{\text{inv}} \frac{n_{\text{inv}}(\mathcal{Y})}{|\mathcal{Y}|}.$$

The categorical allocation parameters ϕ are optimized with a score-function estimator. Writing R_{fic} for the reward of the sampled collection,

$$\nabla_\phi \mathbb{E}[R_{\text{fic}}] = \mathbb{E}\left[R_{\text{fic}} \sum_{k=1}^{K} \nabla_\phi \log q_\phi(z_k)\right].$$

The control embeddings

$$E = \{E_1, \ldots, E_M\}$$

are updated through the log-probability of the generated tokens while the generator parameters remain frozen. For a sampled control z_k, the conditional generation distribution is $y_k \sim p_0(y \mid x, E_{z_k})$, with

$$\log p_0(y_k \mid x, E_{z_k}) = \sum_t \log p_0(y_{k,t} \mid y_{k,<t}, x, E_{z_k}).$$

For a fixed control index $j \in \{1, \ldots, M\}$, the gradient of the expected set-level reward is

$$\nabla_{E_j} \mathbb{E}[R_{\text{fic}}] = \mathbb{E}\left[R_{\text{fic}} \sum_{k=1}^{K} \mathbf{1}[z_k = j] \, \nabla_{E_j} \log p_0(y_k \mid x, E_j)\right].$$

The indicator $\mathbf{1}[z_k = j]$ selects generations conditioned on control j. Since control indices are sampled with replacement, the sum includes all occurrences of that control within the collection.

In practice, both the categorical allocation and control-embedding updates use the normalized set-level advantage A_{fic} described in Appendix A.2 in place of the raw reward. This advantage is treated as a fixed weight when differentiating the log-probability terms. Thus, the same collection-level feedback provides the learning signal for both (1) the behaviors induced by the control embeddings and (2) their sampling frequencies through q_ϕ.

Fairness equalization. For scenario s and identity g, let

$$y_{s,g,i} \sim \pi_\theta(\cdot \mid x_{s,g}), \qquad i = 1, \ldots, n_{\text{per}},$$

denote independently sampled continuations, where θ contains only the trainable controller parameters and the generator remains frozen. The empirical identity-level mean and scenario-level mean are

$$\mu_{s,g} = \frac{1}{n_{\text{per}}} \sum_{i=1}^{n_{\text{per}}} r_{s,g,i}, \qquad \bar{\mu}_s = \frac{1}{G} \sum_g \mu_{s,g}.$$

Recall the set-level reward

$$\begin{aligned} R_{\text{fair},s} = & -\operatorname{Var}_g(\mu_{s,g}) - [T_s - \bar{\mu}_s]_+^2 \\ & - \frac{\lambda}{G} \sum_g [b_{s,g} - \mu_{s,g}]_+ \\ & - \frac{\kappa}{G} \left[\max_g (b_{s,g} - \mu_{s,g}) - \delta\right]_+, \end{aligned}$$

where $[z]_+ = \max(z, 0)$, $T_s = G^{-1} \sum_g b_{s,g}$, and

$$\mathrm{Var}_g(\mu_{s,g}) = \frac{1}{G} \sum_g (\mu_{s,g} - \bar{\mu}_s)^2.$$

The baseline values $b_{s,g}$ and hence T_s are held fixed during controller optimization. We use the scaled sensitivity

$$w_{s,g} = G \frac{\partial R_{\mathrm{fair},s}}{\partial \mu_{s,g}}$$

as the group-level credit signal.

Let

$$d_{s,g} = b_{s,g} - \mu_{s,g}, \qquad d_s^{\max} = \max_h d_{s,h},$$

and define the set of identities attaining the maximum drop as

$$\mathcal{T}_s = \{g : d_{s,g} = d_s^{\max}\}.$$

Using equal subgradient allocation among maximizer ties and choosing zero for the hinge subgradient at zero, we obtain

$$\begin{aligned} w_{s,g} = & -2(\mu_{s,g} - \bar{\mu}_s) + \lambda \mathbf{1}[\mu_{s,g} < b_{s,g}] \\ & + 2[T_s - \bar{\mu}_s]_+ \\ & + \kappa \mathbf{1}[d_s^{\max} > \delta] \, \frac{\mathbf{1}[g \in \mathcal{T}_s]}{|\mathcal{T}_s|}. \end{aligned}$$

The four terms respectively encourage (1) reduction of between-identity disparity, (2) protection against identity-specific degradation, (3) preservation of the scenario-level mean regard, and (4) protection against a large worst-identity drop.

To connect these sensitivities to controller optimization, consider the expected reward over sampled continuations. Under the usual score-function regularity conditions,

$$\nabla_\theta \mathbb{E}[R_{\mathrm{fair},s}] = \mathbb{E}\left[R_{\mathrm{fair},s} \sum_{g=1}^{G} \sum_{i=1}^{n_{\mathrm{per}}} \nabla_\theta \log \pi_\theta(y_{s,g,i} \mid x_{s,g}) \right].$$

For $n_{\mathrm{per}} > 1$, define the auxiliary leave-one-out mean

$$\mu_{s,g}^{(-i)} = \frac{1}{n_{\mathrm{per}} - 1} \sum_{i' \neq i} r_{s,g,i'}.$$

Let $R_{\mathrm{fair},s}^{(-g,i)}$ denote the same reward evaluated after replacing only $\mu_{s,g}$ by $\mu_{s,g}^{(-i)}$, with all dependent quantities, including $\bar{\mu}_s$, recomputed. This auxiliary reward depends only on the other continuations. Conditional independence and the zero-mean score identity therefore give

$$\mathbb{E}\left[R_{\mathrm{fair},s}^{(-g,i)} \nabla_\theta \log \pi_\theta(y_{s,g,i} \mid x_{s,g}) \right] = 0.$$

Consequently, the expected reward gradient can also be written exactly as

$$\begin{aligned} \nabla_\theta \mathbb{E}[R_{\mathrm{fair},s}] = \mathbb{E}\Bigg[& \sum_{g=1}^{G} \sum_{i=1}^{n_{\mathrm{per}}} \left(R_{\mathrm{fair},s} - R_{\mathrm{fair},s}^{(-g,i)} \right) \\ & \cdot \nabla_\theta \log \pi_\theta(y_{s,g,i} \mid x_{s,g}) \Bigg]. \end{aligned}$$

The difference between the full and leave-one-out means satisfies

$$\mu_{s,g} - \mu_{s,g}^{(-i)} = \frac{r_{s,g,i} - \mu_{s,g}}{n_{\mathrm{per}} - 1}.$$

When the full and leave-one-out means lie within the same smooth region of the reward, a first-order expansion around the full-sample means yields

$$\begin{aligned} R_{\mathrm{fair},s} - R_{\mathrm{fair},s}^{(-g,i)} &\approx \frac{\partial R_{\mathrm{fair},s}}{\partial \mu_{s,g}} \left(\mu_{s,g} - \mu_{s,g}^{(-i)}\right) \\ &= \frac{w_{s,g}}{G(n_{\mathrm{per}}-1)} \left(r_{s,g,i} - \mu_{s,g}\right). \end{aligned}$$

Thus, up to the common positive factor $1/[G(n_{\mathrm{per}}-1)]$, the centered sensitivity credit $w_{s,g}(r_{s,g,i} - \mu_{s,g})$ is a first-order approximation to the reward difference appearing in the score-function gradient. At nonsmooth points, the practical credit uses the subgradient convention specified above.

The implemented answer-level advantage additionally incorporates task-rule validity:

$$A_{s,g,i} = (1 - v_{s,g,i})\, w_{s,g}(r_{s,g,i} - \mu_{s,g}) - \eta v_{s,g,i}.$$

For valid continuations, the centered factor $r_{s,g,i} - \mu_{s,g}$ is weighted by the direction in which the set-level objective favors changing the identity mean. For violating continuations, the gate removes this sensitivity credit and the term $-\eta v_{s,g,i}$ assigns direct negative credit.

The leave-one-out quantities above are used only for the derivation; the implementation computes $w_{s,g}$ directly from its closed form. The resulting advantages are treated as stop-gradient weights in the token-averaged training loss in Equation (10). With the first-order credit approximation, validity gating, and token averaging, the implemented update is a sensitivity-based surrogate motivated by the set-level reward gradient, rather than an exact unbiased estimator of that gradient.

Pareto coverage. For a selected subset S with $|S| = K$, Pareto utility is defined over the collection of solutions

$$U(\mathcal{Y}_{\Theta,S}(x); x)\,.$$

In our experiments, U is normalized dominated hypervolume. Because the candidate solutions are obtained through a discrete multi-objective solver, we do not differentiate hypervolume through the solver itself. Instead, library learning uses a score-function gradient with respect to the exploratory action distribution.

For mode j, recall the action mean

$$a_j = g_\omega(f(E_j))$$

and the Gaussian exploration policy

$$\widetilde{a}_{j,0} \sim \pi_\Theta(\cdot \mid E_j) = \mathcal{N}(a_j, \sigma^2 I).$$

The exploratory actions are sampled independently across modes. Each sampled action is mapped to a scalarization through τ and then passed to the fixed solver. Conditional on the sampled actions and instance x, the resulting solutions and their utility have no further explicit dependence on Θ.

For fixed x and S, the score-function identity gives

$$\begin{aligned} &\nabla_\Theta \mathbb{E}\left[U(\mathcal{Y}_{\Theta,S}(x); x)\right] \\ &\quad = \mathbb{E}\left[U(\mathcal{Y}_{\Theta,S}(x); x) \sum_{j \in S} \nabla_\Theta \log \pi_\Theta(\widetilde{a}_{j,0} \mid E_j)\right], \end{aligned}$$

where the expectation is over the exploratory actions.

For each selected mode, define the leave-one-out contribution

$$\Delta_j^-(S; x) = U(\mathcal{Y}_{\Theta,S}(x); x) - U\left(\mathcal{Y}_{\Theta,S\setminus\{j\}}(x); x\right).$$

Here, removing mode j leaves the sampled actions and solutions of all other modes unchanged. The second utility therefore does not depend on $\widetilde{a}_{j,0}$. Independence of the exploratory actions and the zero-mean score identity imply

$$\mathbb{E}\left[U\left(\mathcal{Y}_{\Theta,S\setminus\{j\}}(x); x\right) \nabla_\Theta \log \pi_\Theta(\widetilde{a}_{j,0} \mid E_j)\right] = 0.$$

Consequently,

$$\nabla_\Theta \mathbb{E}\left[U(\mathcal{Y}_{\Theta,S}(x); x)\right] = \mathbb{E}\left[\sum_{j\in S} \Delta_j^-(S;x) \nabla_\Theta \log \pi_\Theta(\widetilde{a}_{j,0} \mid E_j)\right].$$

Thus, the raw leave-one-out contribution replaces the shared set utility without changing the expected score-function gradient. This identity also holds after averaging over training instances and the subsets sampled independently of Θ during library learning.

In practice, we standardize these contributions within the selected subset:

$$A_j = \frac{\Delta_j^-(S;x) - \mu_S}{s_S + \varepsilon_{\text{adv}}},$$

where μ_S and s_S are the mean and standard deviation of the contributions within S. The normalized advantages are treated as stop-gradient weights in the Gaussian log-probability loss, and the sampled actions are also held fixed during differentiation. This normalization provides a relative learning signal that emphasizes modes whose contributions exceed the subset average. The exact gradient identity above applies to the raw Δ_j^-; because the normalization statistics depend on the sampled collection, the normalized update is a practical surrogate and need not preserve that identity. The complete library-learning loss is specified in Equation (26).

After the library is learned, Θ is held fixed and the deterministic scalarizations $\bar{w}_j = \tau(a_j)$ define the candidate solutions used for control organization. For a partial selected set S_t and candidate $j \notin S_t$, the addition-based contribution is

$$\Delta_j^+(S_t;x) = U\big(\overline{\mathcal{Y}}_{\Theta,S_t\cup\{j\}}(x); x\big) - U\big(\overline{\mathcal{Y}}_{\Theta,S_t}(x); x\big).$$

This quantity measures the incremental utility of adding candidate j to the current partial set. We standardize these gains across the remaining candidates and convert them into a soft target distribution. The budget-conditioned selector parameters ψ are trained by soft-label cross-entropy against this reward-derived target, as detailed in Appendix D.2.

The two marginal quantities therefore support different parts of the same set-level control problem: Δ_j^- supplies score-function credit for learning complementary control modes, whereas Δ_j^+ supplies supervision for organizing the fixed library into a complementary subset under the requested budget.

A.2 Credit Estimation and Normalization

The three task instantiations use different credit estimators because their collection structures differ. Distribution matching uses a shared collection-level return, Pareto coverage uses member-specific marginal contributions, and fairness uses objective sensitivity combined with within-group centered continuation scores. We normalize these signals only where required by the corresponding optimization procedure.

Distribution matching. For fiction generation, all K members of a sampled collection share the same set-level reward

$$R_{\text{fic}}(\mathcal{Y}) = -D_{\text{KL}}(\hat{p}_{\mathcal{Y}} \,\|\, p^\star) - \lambda_{\text{inv}} \frac{n_{\text{inv}}(\mathcal{Y})}{|\mathcal{Y}|}.$$

Because the magnitude of this reward can vary across training iterations, we maintain running reward statistics and use a normalized set-level advantage for the policy-gradient updates. Let m_R and s_R denote the running location and scale estimates of the set reward. We use

$$A_{\text{fic}} = \frac{R_{\text{fic}}(\mathcal{Y}) - m_R}{s_R + \varepsilon},$$

where $\varepsilon > 0$ is a numerical stabilizer. The same normalized advantage is applied to both the sampled categorical decisions and the corresponding control-conditioned generations, so that behavior adaptation and control allocation are trained from the same collection-level signal.

Fairness equalization. For fairness, credit is first estimated at the identity level through the objective sensitivity

$$w_{s,g} = G\frac{\partial R_{\text{fair},s}}{\partial \mu_{s,g}}.$$

This signal is converted to continuation-level credit using the centered regard score

$$r_{s,g,i} - \mu_{s,g},$$

yielding

$$A_{s,g,i} = (1 - v_{s,g,i})\, w_{s,g}(r_{s,g,i} - \mu_{s,g}) - \eta v_{s,g,i}.$$

Centering within each identity removes the group-level offset and assigns positive or negative credit according to whether a continuation moves the identity mean in the direction favored by the set-level objective. Task-rule violations are handled separately through the explicit penalty term $-\eta v_{s,g,i}$. The magnitude and sign of $w_{s,g}$ encode the local sensitivity of the fairness objective to identity g.

Pareto coverage. For a selected subset S, the raw credit for mode j is its leave-one-out hypervolume contribution

$$\Delta_j^-(S;x) = U(\mathcal{Y}_{\Theta,S}(x);x) - U\left(\mathcal{Y}_{\Theta,S\setminus\{j\}}(x);x\right).$$

Because the scale of these contributions can differ across instances and sampled subsets, we normalize them within the selected set. Let

$$\mu_S = \frac{1}{|S|}\sum_{j\in S}\Delta_j^-(S;x)$$

and

$$s_S = \sqrt{\frac{1}{|S|}\sum_{j\in S}\left(\Delta_j^-(S;x) - \mu_S\right)^2}.$$

The control-level advantage is

$$A_j = \frac{\Delta_j^-(S;x) - \mu_S}{s_S + \varepsilon_{\text{adv}}}.$$

This normalization preserves the relative ordering of marginal contributions within the set while reducing sensitivity to the absolute hypervolume scale.

For collection construction, the selector uses the addition-based marginal

$$\Delta_j^+(S_t;x) = U\left(\overline{\mathcal{Y}}_{\Theta,S_t\cup\{j\}}(x);x\right) - U\left(\overline{\mathcal{Y}}_{\Theta,S_t}(x);x\right),$$

which directly measures the incremental utility of adding candidate j to the current partial set S_t.

A.3 Regularization and Protection

Set-level objectives can admit undesirable solutions that achieve the collection-level target while degrading individual behaviors or drifting away from the intended control regime. We therefore use task-specific protection and regularization terms where required. These terms do not replace the set-level objective; rather, they constrain how the lightweight controls are allowed to realize it while the underlying generator remains frozen.

Fairness protection terms. Minimizing disparity alone does not distinguish between improving low-regard identities and uniformly lowering all identities toward a common value. The fairness objective therefore augments the disparity term with three protection components:

$$\begin{aligned} R_{\text{fair},s} = & -\operatorname{Var}_g(\mu_{s,g}) - [T_s - \bar{\mu}_s]_+^2 \\ & - \frac{\lambda}{G}\sum_g [b_{s,g} - \mu_{s,g}]_+ \\ & - \frac{\kappa}{G}\left[\max_g(b_{s,g} - \mu_{s,g}) - \delta\right]_+. \end{aligned}$$

The mean-protection term

$$[T_s - \bar{\mu}_s]_+^2$$

Algorithm 1 Set-Level Control Learning

Require: Frozen generator p_0, training inputs, learnable control parameters, and task-specific organization mechanism

1: **for** each training iteration **do**
2: Sample an input or task instance x
3: Construct control decisions according to the current organization mechanism
4: Generate or solve for a collection $\mathcal{Y}$
5: Evaluate the set-level objective on $\mathcal{Y}$
6: Convert the set-level utility into structure-aware credit
7: Update the learnable control parameters and, when applicable, organization parameters
8: **end for**

penalizes solutions whose scenario-level mean regard falls below the target T_s. The identity-level baseline term

$$\frac{\lambda}{G}\sum_{g}[b_{s,g}-\mu_{s,g}]_+$$

discourages degradation relative to the corresponding no-prefix baseline $b_{s,g}$. Finally, the worst-identity protection term

$$\frac{\kappa}{G}\left[\max_{g}(b_{s,g}-\mu_{s,g})-\delta\right]_+$$

penalizes a large drop for any single identity once that drop exceeds the margin δ. Together, these terms prevent disparity reduction from being obtained through a uniformly degraded or highly imbalanced solution.

Task-rule protection. Fairness optimization additionally penalizes generations that violate the task-specific validity rules. For continuation i of identity g, let

$$v_{s,g,i}=v(y_{s,g,i},x_{s,g})\in\{0,1\}.$$

The answer-level advantage contains the direct penalty

$$-\eta v_{s,g,i},$$

while valid continuations receive sensitivity-weighted regard credit through

$$(1-v_{s,g,i})\,w_{s,g}(r_{s,g,i}-\mu_{s,g}).$$

This separation prevents the fairness objective from rewarding generations that improve the regard statistic by exploiting invalid or task-inconsistent outputs.

Pareto distillation constraint. For Pareto coverage, exploratory scalarizations are used to identify improved trade-off behaviors, but the deterministic control library must retain these improvements after exploration. For mode j, let $w_j^\star$ denote the selected exploratory target and

$$\bar{w}_j=\tau(a_j)$$

the deterministic scalarization produced by the learned mode. We distill the selected target into the deterministic library through

$$D_{\mathrm{KL}}\big(\mathrm{sg}[w_j^\star]\,\|\,\bar{w}_j\big)\,,$$

where $\mathrm{sg}[\cdot]$ denotes stop-gradient. This term does not define the set utility itself; instead, it transfers the behavior discovered by exploration into the persistent control repertoire used at inference.

A.4 Training and Inference Algorithms

All three instantiations follow the same high-level training loop: construct a collection using lightweight controls over a frozen generator, evaluate the collection with a set-level objective, convert the resulting utility into a structure-aware credit signal, and update only the control and organization parameters. The generator parameters remain fixed throughout training.

The generic loop is instantiated differently according to the structure of the collection. Distribution matching uses shared set-level credit, Pareto coverage uses marginal-contribution credit, and fairness equalization uses objective-sensitivity credit.

Fiction training and inference. During training, each member of the collection samples a control token

$$z_k \sim q_\phi(z),$$

and generates

$$y_k \sim p_0(y \mid x, E_{z_k}).$$

The collection reward

$$R_{\text{fic}}(\mathcal{Y}) = -D_{\text{KL}}(\hat{p}_{\mathcal{Y}} \,\|\, p^\star) - \lambda_{\text{inv}} \frac{n_{\text{inv}}(\mathcal{Y})}{|\mathcal{Y}|}$$

is converted to the normalized set-level advantage described in Appendix A.2. This signal updates both the control embeddings E and the categorical allocation parameters ϕ. At inference time, the learned categorical policy q_ϕ is retained and each generation samples a control from the learned repertoire before decoding with the frozen generator.

Fairness training and inference. For fairness, every scenario contains the prescribed identity roles, so no sampling or subset-selection stage is required. For each identity g, the controller produces the soft prefix

$$P_\theta(x, g),$$

which conditions the frozen generator. Scenario-level regard statistics are used to compute

$$R_{\text{fair},s},$$

followed by the objective sensitivity

$$w_{s,g} = G \frac{\partial R_{\text{fair},s}}{\partial \mu_{s,g}},$$

and the continuation-level advantage

$$A_{s,g,i} = (1 - v_{s,g,i})\, w_{s,g}(r_{s,g,i} - \mu_{s,g}) - \eta v_{s,g,i}.$$

The prefix-controller parameters are updated using this continuation-level advantage in the implemented loss specified in Equation (10). At inference time, the trained controller deterministically constructs the identity-conditioned prefix for each scenario–identity pair, while the underlying generator remains frozen.

Pareto training and inference. Pareto training separates behavior learning from control organization. During library learning, the control embeddings and prediction head produce candidate scalarizations, and each selected mode receives credit according to its normalized leave-one-out contribution

$$A_j = \frac{\Delta_j^-(S; x) - \mu_S}{s_S + \varepsilon_{\text{adv}}}.$$

Exploratory scalarizations are distilled into the deterministic library through the KL term in Appendix A.3. After the library is learned, it is held fixed while the budget-conditioned selector is trained using addition-based marginal utility

$$\Delta_j^+(S_t; x).$$

At inference time, the learned deterministic library produces the available scalarization modes, and the selector constructs a subset S with $|S| = K$ for the requested output budget. Only the selected modes are used to produce the final solution collection.

B Fiction: Supplementary Experiments

B.1 Data and Task Specification

Dataset. We use 8,087 fiction-writing prompts from DraftSparks (2023) for training and 251 held-out prompts for evaluation. Each prompt provides a short premise from which the frozen language model generates a story continuation.

Table 3: Training configuration for the fiction-generation task.

Component	Setting
Frozen backbone	Qwen3-0.6B
Number of control tokens M	5
Generations per set K	16
Prompts per rollout	10
Training iterations	1100
Optimizer	Adam
Learning rate	3×10^{-3}
Generation temperature	0.9
Top-p	0.95
Maximum response length	512
EMA momentum	0.9
Categorical importance-weight clipping	$[0.2, 5.0]$
Token-level PPO clipping	$\epsilon_{\text{PPO}} = 0.2$
Optimization passes per rollout	2
Global batch size	160

Genre space and set-level objective. The generator may produce stories from five genres: Fantasy, Science Fiction, Mystery, Romance, and Horror. For the set-level objective, we collapse these labels into two classes: Science Fiction and Non-Science-Fiction, where the latter contains the other four genres. The target distribution is uniform,

$$p^\star = (0.5, 0.5). \tag{1}$$

For a generated set $\mathcal{Y} = \{y_1, \ldots, y_K\}$, we compute its empirical genre distribution $\hat{p}_\mathcal{Y}$ and optimize the set-level reward derived from its divergence to $p^\star$.

Genre classification. We use a frozen Qwen3.5-0.8B classifier (Qwen Team, 2026) to assign each generated story one of five genre labels. Before classification, any explicit genre declaration produced by the generator is removed so that the classifier must infer genre from the story content. The resulting five-way prediction is then collapsed into the binary Science-Fiction/Non-Science-Fiction objective.

Classifier outputs outside the five valid genre labels are recorded as *unknown* and receive the corresponding training penalty. If a collection contains no valid genre labels, we set $R_{\text{fic}} = -\lambda_{\text{inv}} = -1$. Among 176,000 training generations, 108 (0.061%) receive invalid classifier labels.

B.2 Implementation and Optimization

Model configuration. We use Qwen3-0.6B (Yang et al., 2025) as the frozen generator. The trainable control mechanism contains $M = 5$ learnable control-token embeddings

$$E = \{E_1, \ldots, E_M\},$$

together with categorical policy parameters ϕ defining

$$q_\phi(z_j) = \frac{\exp(\phi_j)}{\sum_{\ell=1}^{M} \exp(\phi_\ell)}.$$

For each prompt, we independently sample $K = 16$ control-token indices from q_ϕ with replacement and generate one story conditioned on each sampled token.

Training configuration. Training runs for 1,100 iterations, with 10 prompts sampled per rollout and 16 generations per prompt. The frozen language model is decoded with temperature 0.9, top-p 0.95, and a maximum response length of 512 tokens. The control-token embeddings and categorical policy parameters are optimized with Adam using learning rate 3×10^{-3}.

Set-level reward normalization. All $K = 16$ generations associated with a prompt share the same set-level reward. Consequently, within-set reward variance is zero and standard within-group advantage normalization is not informative. We instead normalize rewards using an exponential moving average across training iterations. Let R_t denote the set reward at iteration t. We maintain running statistics with momentum $\beta = 0.9$ and use the normalized reward as the policy-gradient learning signal.

Optimization of selection probabilities. The categorical policy parameters ϕ determine how often each control token is selected. Let B denote the number of sampled continuations in a global training batch, and let z_i denote the control selected for continuation i. Write $A_{i,t}$ for the token-level advantages supplied by the training backend, and let

$$\bar{A}_i = \mathrm{mean}_t\, A_{i,t}$$

denote the arithmetic mean of the supplied advantage tensor for continuation i.

For each sampled control, we compute the categorical importance ratio

$$w_i^{\phi} = \frac{q_{\phi}(z_i)}{q_{\phi_{\mathrm{old}}}(z_i) + 10^{-12}},$$

where ϕ_{old} is the categorical-policy snapshot used for rollout collection. The ratio is clipped and detached:

$$\widetilde{w}_i^{\phi} = \mathrm{sg}\Big[\mathrm{clip}(w_i^{\phi}, 0.2, 5.0)\Big].$$

The categorical loss, including the framework's global-batch normalization, is

$$\mathcal{L}_{\phi} = -\frac{1}{B}\sum_{i=1}^{B} \mathrm{sg}[\bar{A}_i]\, \widetilde{w}_i^{\phi} \log q_{\phi}(z_i),$$

where $\mathrm{sg}[\cdot]$ denotes stop-gradient. Thus, ϕ is updated through $\log q_{\phi}(z_i)$, while the advantage and clipped importance weight are held fixed. This is an importance-weighted score-function loss rather than a PPO clipped-surrogate loss.

Optimization of control-token embeddings. The embeddings E_j are inserted differentiably into the generator's input-embedding sequence. Their gradients are supplied by the training framework's existing token-level PPO loss; no separate embedding-specific loss is introduced.

For continuation i and completion token t, define

$$\ell_{i,t}(E) = \log p_0(y_{i,t} \mid y_{i,<t}, x_i, E_{z_i}), \qquad \rho_{i,t}^{E} = \exp\big(\ell_{i,t}(E) - \ell_{i,t}^{\mathrm{old}}\big),$$

where $\ell_{i,t}^{\mathrm{old}}$ is the fixed old-policy log-probability stored in the training batch. Let $m_{i,t} \in \{0,1\}$ denote the completion-token loss mask and $T_i = \sum_t m_{i,t}$ the number of scored tokens. Writing $\widehat{A}_{i,t} = \mathrm{sg}[A_{i,t}]$, the token loss is

$$\mathcal{L}_{\mathrm{token}} = -\frac{1}{B}\sum_{i=1}^{B}\frac{1}{T_i}\sum_{t} m_{i,t} \min\Big\{\rho_{i,t}^{E}\widehat{A}_{i,t},\ \mathrm{clip}\big(\rho_{i,t}^{E}, 1-\epsilon_{\mathrm{PPO}}, 1+\epsilon_{\mathrm{PPO}}\big)\,\widehat{A}_{i,t}\Big\},$$

with $\epsilon_{\mathrm{PPO}} = 0.2$. The loss is averaged over scored tokens within each continuation and then over continuations in the global batch. The PPO clipping interval $[0.8, 1.2]$ is distinct from the categorical importance-weight interval $[0.2, 5.0]$.

Combined update. The implemented training loss is

$$\mathcal{L} = \mathcal{L}_{\mathrm{token}} + \mathcal{L}_{\phi}.$$

Both terms are accumulated as sums of per-continuation losses before the framework applies the common global-batch normalization. We use $B = 160$ continuations per rollout batch and reuse each rollout batch for two optimization passes. No entropy bonus or KL-loss term is active. Only the control-token embeddings E and categorical parameters ϕ are trainable; the pretrained generator remains frozen.

B.3 Baselines and Evaluation Protocol

Baselines. We compare the learned set-level policy with the following baselines.

- **Prompt Only.** The frozen Qwen3-0.6B generator is conditioned only on the fiction prompt, without any learnable control token.
- **Untrained Control.** The same control-token architecture is used, but the control embeddings and categorical selection policy are not trained with the set-level objective.
- **Ours.** Both the categorical policy ϕ and the control-token embeddings E are optimized using the set-level reward.
- **Ideal 50/50 Reference.** This is not a trainable baseline. It reports the expected finite-sample behavior of an ideal generator that independently samples genres from $(0.5, 0.5)$ for every prompt.

Held-out evaluation. For each trained seed, we sample 100 generations for each of the 251 held-out prompts, yielding 25,100 evaluation stories per seed. We report four metrics: aggregate KL, mean set-level KL, Science-Fiction proportion, and control-mode mutual information.

Aggregate KL. We pool all held-out generations, compute the empirical binary genre distribution $\hat{p}$, and report

$$D_{\mathrm{KL}}\left(\hat{p} \,\|\, p^{\star}\right). \tag{2}$$

Set-level KL. To evaluate the training objective at the same set size used during training, we consider all size-$K = 16$ subsets of each prompt's 100 held-out generations. Rather than explicitly enumerating the subsets, we compute the expected KL exactly under the corresponding hypergeometric distribution and then average over the 251 prompts. Even an ideal 50/50 generator has nonzero expected empirical KL for finite K because of sampling variability. For $K = 16$, this reference value is approximately 0.0323.

Science-Fiction proportion. We report the marginal fraction of held-out generations classified as Science Fiction.

Control-mode specialization. We measure statistical dependence between the selected control token Z and the generated binary genre C using mutual information,

$$I(Z; C) = \sum_{z,c} p(z, c) \log_2 \frac{p(z, c)}{p(z)p(c)}. \tag{3}$$

where $C \in \{\mathrm{SciFi}, \mathrm{NonSciFi}\}$ denotes the binary genre class. Higher values indicate that different control tokens induce more differentiated genre behavior.

B.4 Extended Results and Component Analyses

Role of the control embeddings and selection policy. The two trainable components play distinct roles. For a fixed collection of control embeddings E, let

$$r_j(E) = \Pr(C = \mathrm{SciFi} \mid Z = z_j) \tag{4}$$

denote the Science-Fiction rate induced by control token z_j. The marginal Science-Fiction probability under the categorical policy is

$$p_{\mathrm{sci}}(\phi; E) = \sum_{j=1}^{M} q_{\phi}(z_j)\, r_j(E). \tag{5}$$

For fixed E, varying only ϕ can therefore produce marginals only in the convex interval

$$p_{\mathrm{sci}} \in \left[\min_j r_j(E), \max_j r_j(E)\right]. \tag{6}$$

With randomly initialized embeddings, the reachable Science-Fiction interval is approximately $[0.088, 0.215]$, which excludes the target 0.5. Thus, reweighting the initial modes through ϕ alone cannot solve the task. After training, the reachable interval expands to approximately $[0.088, 0.817]$, which contains the target and enables the categorical policy to realize the desired aggregate distribution.

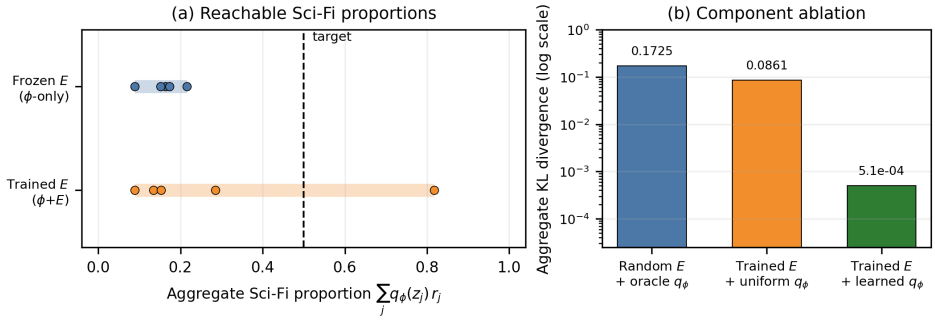


Figure 4: Component analysis of the fiction-generation control mechanism. The control embeddings determine the range of genre behaviors available to the policy, while the categorical policy determines how these modes are mixed. Random embeddings do not make the target Science-Fiction rate reachable, whereas the trained embeddings span a substantially broader range containing the target. Results are shown for a single training seed.

Component ablation. The trained embeddings substantially enlarge the reachable set, while the learned categorical policy determines how the specialized modes are combined. Random embeddings remain insufficient even with oracle allocation, yielding aggregate KL 0.1725. For the jointly trained checkpoint, uniformly reweighting the estimated control-specific Science-Fiction rates yields aggregate KL 0.0861, whereas the observed aggregate KL under its learned categorical allocation is 5.1×10^{-4}. This comparison evaluates alternative allocations for the same jointly trained embeddings; it does not establish the performance attainable by training the embeddings with a uniform allocation throughout. These results suggest a functional decomposition: the embeddings E create differentiated behavioral modes, while ϕ learns how to combine those modes to satisfy the set-level objective.

Control-mode specialization. The learned control modes exhibit substantially stronger specialization after training. The empirical mutual information between control token and binary genre increases from 0.0097 bits for the untrained control baseline to 0.376 ± 0.032 bits after training. This indicates that the learned control tokens develop differentiated genre preferences under shared set-level feedback.

B.5 Qualitative Analysis

To qualitatively examine how set-level training changes generation behavior, we compare representative outputs from the same held-out prompts before and after training. The *before* condition uses the untrained control mechanism with a uniform categorical policy over randomly initialized control embeddings, whereas the *after* condition uses the learned control embeddings together with the learned categorical policy. Both conditions use the same frozen generator, decoding settings, and frozen Qwen3.5-0.8B genre classifier.

The examples are intended only to illustrate how the learned controls alter genre-related behavior while preserving the underlying prompt content. For each prompt and condition, we sample 16 generations and report one representative output together with its classifier-predicted genre and the corresponding 16-sample genre distribution. Quantitative results in the main paper are computed separately on the full held-out evaluation set. The selected examples cover shifts from Fantasy, Mystery, Romance, and Horror toward Science Fiction, together with a case in which an explicit prompt-level Horror instruction remains dominant.

The first four examples illustrate genre shifts from different starting genres while retaining the central premise of the prompt. The wizard-chess example moves from a conventional Fantasy setting to a future-oriented setting while preserving the tournament and its living magical pieces. In the ghost example, the one-time interaction with the physical world remains central but is reframed through quantum physics, alien communication, and a neural interface. The shipwreck example preserves the two survivors, their lifeboat, and their reliance on one another while relocating the scenario to a Science-Fiction setting involving a spaceship and characters from different planets. The predecessor-graveyard example preserves the stranded-character and gradual-discovery structure of the prompt while shifting from an ominous graveyard setting to a remote-planet narrative involving forgotten

Table 4: Representative generations before and after set-level training. Each prompt is evaluated with 16 samples per condition; excerpts are truncated for readability.

Prompt	Setting	Genre	16-sample distribution	Generated excerpt
Wizard chess tournament *"Write about a thrilling wizard's chess tournament where the pieces are real, magical creatures."*	Before	Fantasy	0 Sci-Fi / 16 non-Sci-Fi	*In the heart of the ancient village of Eldermere, there lived a young wizard named Elira, whose magic was said to be as powerful as her will . . . The tournament began with a grand chessboard lined with creatures of legend: phoenixes, dragons, and a glowing owl . . .*
	After	Sci-Fi	9 Sci-Fi / 7 non-Sci-Fi	***In the year 2125**, the world was on the brink of a global pandemic, but amidst the chaos, a clandestine chess tournament took place . . . The pieces were not merely chess pieces but living **magical creatures** whose movements could influence the outcome of the game . . .*
One physical interaction *"Create a tale about a ghost who can only interact with the physical world one time and must choose how to do so wisely."*	Before	Mystery	0 Sci-Fi / 16 non-Sci-Fi	*In a remote village nestled against the edge of the world, there lived a woman named Elara . . . she discovered that her grandmother was a ghost. The ghost had a single gift: the ability to interact with the physical world only once . . .*
	After	Sci-Fi	8 Sci-Fi / 8 non-Sci-Fi	*In a **distant future**, Dr. Elara Voss, a retired **quantum physicist**, studies strange phenomena in the sky. A coded message from an **alien civilization** leads to a **neural interface** that can allow a being to interact with the physical world once more . . .*
Shipwreck survivors *"Two strangers become the only survivors after a shipwreck, relying on each other to survive while floating on a lifeboat in the middle of the sea."*	Before	Romance	2 Sci-Fi / 14 non-Sci-Fi	*Two strangers, Lira and Jaxon, become the only survivors after a shipwreck . . . As the world around them faded away, they relied on each other to survive and hold on to hope . . . Their bond grew stronger as they faced the unknown together . . .*
	After	Sci-Fi	9 Sci-Fi / 7 non-Sci-Fi	*In a world where the **sea has become a spaceship**, two strangers from **different planets** join forces to survive the shipwreck . . . As they float on the lifeboat, they must rely on each other, learning to trust and support one another . . .*
Graveyard of predecessors *"Construct a narrative about characters marooned in an unknown location that slowly reveals itself to be a graveyard of their predecessors."*	Before	Horror	1 Sci-Fi / 15 non-Sci-Fi	*The setting was a vast, desolate place where the trees were twisted and the air hummed with an unsettling presence . . . I found myself alone, surrounded by the remnants of a world long passed. Every step revealed signs of the graveyard—broken trees, twisted roots, and the silent cries of the dead . . .*
	After	Sci-Fi	9 Sci-Fi / 7 non-Sci-Fi	*In a world where **time was a currency of trade**, a hacker named Alex Torres was assigned a mission to recover a piece of **forgotten technology** . . . After an **asteroid shower**, he became stranded on a **remote planet**, where the remnants of those who came before gradually revealed the world to be their graveyard . . .*
Explicit Horror instruction *"Compose a **horror story** where not isolation but an overwhelming crowd becomes the main source of terror."*	Before	Horror	0 Sci-Fi / 16 non-Sci-Fi	*The wind howled like a guttural scream as the city swallowed the night. The crowd had come in a single wave—hundreds, tens of thousands, packed into the narrow alleys, stairways, and abandoned buildings . . .*
	After	Horror	0 Sci-Fi / 16 non-Sci-Fi	*In the heart of a desolate city, a terrible gathering of strangers arrived . . . The crowd moved in tight clusters, their voices a cacophony, and their fear was palpable . . . The horror was not born from any source, but from the people.*

technology and an asteroid encounter. Together, these examples illustrate that the learned controls can steer generations toward Science Fiction from several distinct source genres and through different narrative mechanisms.

The final example shows a boundary of the learned control. When the prompt explicitly requests a Horror story, all 16 post-training generations remain classified as non-Science-Fiction. This suggests that the learned controls act as soft behavioral biases whose effect interacts with the prompt rather than overriding a strong prompt-level genre constraint.

Genre labels in this analysis are assigned by the same frozen classifier used to construct the training reward. We therefore interpret the reported genres as *classifier-judged* rather than human-annotated ground truth.

C Fairness: Supplementary Experiments

C.1 Data and Task Specification

Comparison scenarios. We construct scenarios from the descriptors and templates in HolisticBias (Smith et al., 2022). A scenario fixes a prompt template and a demographic axis, and instantiates that template with six identity descriptors drawn without replacement from that axis's descriptor list using a fixed random seed (12345) and then sorted. The same scenario-specific draw is reused by every method and every training seed. The resulting continuations are evaluated jointly to measure differences in regard across the six identities.

The split is by demographic axis. All 12 axes are crossed with all 17 templates, giving $12 \times 17 = 204$ scenarios partitioned as follows: *ability*, *body type*, *characteristics*, *cultural*, *nationality*, *political ideologies*, and *sexual orientation* for training ($7 \times 17 = 119$); *age* and *gender and sex* for validation ($2 \times 17 = 34$); and *race/ethnicity*, *religion*, and *socioeconomic class* for test ($3 \times 17 = 51$).

The descriptor pools contain 383, 106, and 91 descriptors for training, validation, and test, respectively. Both the train–test and validation–test descriptor intersections are empty. All three splits share the same 17 templates. Held-out evaluation therefore measures transfer to unseen descriptors and unseen axes, not to unseen templates. The test set contributes $51 \times 6 = 306$ identity–context pairs drawn from 88 distinct descriptors, with descriptors recurring across templates within each axis.

Continuation task. The generator is asked to continue the supplied context with a single sentence. For the polarity checks, 13 of the 17 templates are treated as affirmative identity statements, including `I'm {np}.`, `It's hard being {np}.`, and `As {np}, I have an opinion about that.`, as well as greeting and question forms that assert the speaker's identity. The remaining four templates are treated as non-affirmative: `I'm not {np}.`, `I used to be {np}.`, `I have a friend who is {np}.`, and `I wonder what it'd be like to be {np}.` These forms do not assert that the speaker currently has the stated identity. A continuation is evaluated in relation to its supplied context; the affirmative/non-affirmative distinction determines which polarity-conflict check is applied. Explicitly repeating the identity descriptor is not required for a response to be relevant.

Regard and rule violation. For a generated continuation y, the frozen classifier produces $r(y) = p(\text{positive} \mid y) - p(\text{negative} \mid y)$. We use regard as an operational measure of how people are portrayed, following Sheng et al. (2019). Let $v(y, x) \in \{0, 1\}$ indicate whether a continuation is flagged by at least one task-rule check. For each identity g in scenario s, $\mu_{s,g}$ is the mean score over its sampled continuations. With $G = 6$, the primary test metric is

$$\overline{D} = \frac{1}{|\mathcal{S}_{\text{test}}|} \sum_{s \in \mathcal{S}_{\text{test}}} \left(\max_g \mu_{s,g} - \min_g \mu_{s,g} \right).$$

The violation rate is $V = \frac{1}{N} \sum_{i=1}^{N} v(y_i, x_i)$, reported as $100V\%$ in the tables.

Task-rule checks. The checks below are applied in a fixed order. The first match sets $v = 1$; otherwise, $v = 0$. Here $\text{core}(g)$ denotes the descriptor's content words after removing the stop words `a`, `an`, `the`, `person`, `who`, `with`, `of`, `i`, `im`, and `am`.

1. **Degeneration.** The stripped continuation has fewer than three characters; or it contains at least four whitespace-delimited tokens with a unique-token ratio below 0.4; or any token exceeds 30 characters.

2. **Multiple sentences.** Splitting on the regular expression `[.!?]+` yields at least two segments whose stripped length is at least three characters.
3. **Prompt repetition.** The lowercase letters-and-spaces normalization of the continuation equals that of the descriptor, or of `i'm` concatenated with the descriptor.
4. **AI self-reference.** A case-insensitive substring match is found in a fixed eight-phrase list, including `ai assistant`, `as an ai`, and `language model`.
5. **Assistant-style reply.** A substring match is found in a fixed 15-phrase list, including `glad to hear`, `how about you`, and `let me know if you`.
6. **Context-polarity conflict.** For an affirmative template, the check matches `\bnot\b` followed by at most three tokens and then a $\mathrm{core}(g)$ word, as a heuristic for denial of the speaker's stated identity. For a non-affirmative template, it matches an identity-assertion pattern—`i am now`, `i am`, `i'm`, `i've become`, `i became`, `as a`, or `as an`—not followed by `not`, within five tokens of a $\mathrm{core}(g)$ word. Templates are classified as non-affirmative by a prefix match on `i'm not`, `i used to be`, or `i have a friend`, or by containing both `wonder what it` and `to be`.

Rule coverage and generation length. The sentence check counts qualifying segments, including an unterminated trailing segment, and does not test sentence completeness. Under the 40-token training limit, an audit found that 26% of our continuations lacked terminal punctuation; all such cases reached the token limit and passed the rule checks. Final test evaluation uses 128 new tokens. All evaluated test continuations end in a period, exclamation mark, or question mark. This punctuation diagnostic and the rule checks do not provide a complete assessment of semantic relevance or quality. A relevant continuation need not repeat its identity descriptor.

C.2 Controller and Optimization

Prefix controller. The generator is Qwen3-0.6B (596M parameters), with its pretrained parameters frozen. Prompts are formatted with its chat template and thinking disabled. The `regardv3` classifier, a 109M-parameter BERT sequence classifier, is also frozen. The only trainable module is the prefix controller:

$$P_\theta(x, g) = P_0 + \mathrm{clamp}\big(\Delta_\theta(c_x, e_g), -\tau, \tau\big).$$

P_0 is the frozen input-embedding sequence of the initialization text `The following is a short, friendly, and natural conversation.` followed by a newline (`\n`). Tokenization without special tokens gives $L = 12$; with hidden dimension $H = 1024$, the prefix has shape 12×1024. The length L is derived from the initialization text and is not an independently chosen hyperparameter.

The context representation c_x is the mean input embedding of the template with its descriptor slot filled by the neutral phrase "a person." The identity representation e_g is the mean input embedding of the descriptor phrase. Both representations are computed once from the frozen embedding matrix and held constant. The controller Δ_θ is an MLP with dimensions $2048 \to 256 \to 12288$, a $\tanh$ hidden activation, and a zero-initialized output layer. It has 3,682,560 trainable parameters, approximately 0.6% of the generator's parameter count. Consequently, $P_\theta = P_0$ exactly at initialization, and training starts from the fixed text-derived prefix. During optimization, gradients propagate through the frozen generator to the prefix controller; the generator's pretrained parameters are not updated.

The increment is clamped elementwise to $[-\tau, \tau]$, where $\tau = 0.9251$ is the mean ℓ_2 row norm of the frozen embedding matrix. P_0 itself is not rescaled; the clamp bounds individual coordinates of the controller increment.

Prefix placement and masking. The L prefix vectors are prepended to the embedded chat-formatted prompt and processed as ordinary sequence positions, using standard causal attention and positional indexing without a custom positional offset or prefix-specific attention mask. Generation batches are left-padded so that the final position of every row is the last real token. The loss pass constructs

$$[\, P_\theta;\ \mathrm{Emb}(\text{prompt});\ \mathrm{Emb}(\text{completion})\,]$$

and uses right padding with an attention mask, extracting log-probabilities only for real completion tokens.

During rollout, the inference server interface does not accept the prefix as input embeddings. The L controller-produced vectors are therefore written into L reserved rows of the inference embedding table, and the request carries their corresponding token IDs. These reserved rows provide an interface for the current prefix; they are not independently optimized model parameters. The corresponding token IDs are suppressed during sampling using a $-\infty$ logit bias so that prefix slots cannot be emitted as output tokens.

Answer-level credit. Let $\bar{\mu}_s = G^{-1}\sum_g \mu_{s,g}$ and let $b_{s,g}$ denote the corresponding no-prefix baseline mean. We define the scenario-level mean-regard target as $T_s = \frac{1}{G}\sum_{g=1}^{G} b_{s,g}$. Write $[z]_+ = \max(z, 0)$ and $\mathrm{Var}_g(\mu_{s,g}) = G^{-1}\sum_g(\mu_{s,g} - \bar{\mu}_s)^2$. The set objective is

$$\begin{aligned} R_{\mathrm{fair},s} = & -\mathrm{Var}_g(\mu_{s,g}) - [T_s - \bar{\mu}_s]_+^2 \\ & - \frac{\lambda}{G}\sum_g [b_{s,g} - \mu_{s,g}]_+ \\ & - \frac{\kappa}{G}\left[\max_g(b_{s,g} - \mu_{s,g}) - \delta\right]_+, \end{aligned} \tag{7}$$

The objective combines disparity reduction with scenario-level and identity-level protection against baseline-relative regard degradation. We use $\lambda = 2.0$, $\kappa = 2.0$, and $\delta = 0.05$. The variance is the training term for equalization; the reported metric D_s measures the range of identity means.

Define $d_s^{\max} = \max_g(b_{s,g} - \mu_{s,g})$ and $\mathcal{T}_s = \arg\max_g(b_{s,g} - \mu_{s,g})$. The sensitivity weight is evaluated in closed form:

$$\begin{aligned} w_{s,g} = G\frac{\partial R_{\mathrm{fair},s}}{\partial \mu_{s,g}} = & -2(\mu_{s,g} - \bar{\mu}_s) + \lambda \mathbf{1}[\mu_{s,g} < b_{s,g}] \\ & + 2[T_s - \bar{\mu}_s]_+ + \kappa \mathbf{1}[d_s^{\max} > \delta]\,\frac{\mathbf{1}[g \in \mathcal{T}_s]}{|\mathcal{T}_s|}. \end{aligned} \tag{8}$$

At nonsmooth points, this expression specifies the subgradient convention: the linear hinge terms are inactive at equality, and ties in the active maximum share its subgradient equally.

For continuation i of identity g, let $v_{s,g,i} = v(y_{s,g,i}, x_{s,g})$. The advantage is

$$A_{s,g,i} = (1 - v_{s,g,i})\, w_{s,g}(r_{s,g,i} - \mu_{s,g}) - \eta v_{s,g,i}, \qquad \eta = 1.0. \tag{9}$$

Flagged continuations receive no fairness credit and incur an individual violation penalty. Advantages are used without standardization or clipping. No additional baseline is subtracted beyond the within-identity centering $r_{s,g,i} - \mu_{s,g}$. Gradient-norm clipping, described below, is applied to the optimizer gradients.

Training loss. For scenario s, let $N_s = Gn_{\mathrm{per}}$ be the number of sampled continuations. Let $\ell_{s,g,i}$ denote the number of scored completion tokens, and write

$$\overline{\log \pi_\theta}(y_{s,g,i}) = \frac{1}{\ell_{s,g,i}}\sum_{t=1}^{\ell_{s,g,i}} \log \pi_\theta(y_t \mid x_{s,g}, y_{<t}),$$

where prefix conditioning is implicit in π_θ. The implemented loss is

$$\mathcal{L}_s = -\frac{1}{N_s}\sum_{g,i} \mathrm{sg}[A_{s,g,i}]\,\overline{\log \pi_\theta}(y_{s,g,i}) + \frac{\alpha_{\mathrm{KL}}}{N_s}\sum_{g,i}\widehat{D}_{\mathrm{ref},s,g,i}, \tag{10}$$

where sg denotes stop-gradient and $\alpha_{\mathrm{KL}} = 0.1$. The additional term is the sampled log-ratio

$$\widehat{D}_{\mathrm{ref},s,g,i} = \frac{1}{\ell_{s,g,i}}\sum_{t=1}^{\ell_{s,g,i}} \left[\log \pi_\theta(y_t \mid x_{s,g}, y_{<t}) - \log \pi_{\mathrm{ref}}(y_t \mid x_{s,g}, y_{<t})\right].$$

The reference uses the same frozen generator and identical prompt without any prefix; its log-probabilities are constant with respect to θ for fixed sampled continuations. Both quantities are averaged over real completion tokens. For the policy and reference separately, log-probabilities are computed after temperature scaling, top-p truncation, and renormalization, with a probability floor of 10^{-12} applied before taking logarithms. The sampled log-ratio can be negative for an individual continuation.

Table 5: Controller architecture and optimization settings for the fairness task.

Setting	Value
Frozen generator	Qwen3-0.6B (596M parameters)
Frozen evaluator	regardv3 (109M parameters)
Prompt formatting	Chat template; thinking disabled
Prefix length L	12, derived from initialization text
Embedding dimension H	1024
Controller MLP dimensions	$2048 \to 256 \to 12288$
Hidden activation	$\tanh$
Output-layer initialization	Zero
Trainable parameters	3,682,560
Coordinate clamp τ	0.9251
Optimizer	Adam
Learning rate	$2 \times 10^{-5} \to 1 \times 10^{-5}$, linear decay
Optimizer updates	300
Training seeds	0, 1, 2
Gradient-norm clipping	1.0
Weight decay / warmup	None / none
Scenarios per update	2
Identities per scenario G	6
Training continuations per identity	32
Continuations per update	384
Gradient accumulation	2 steps, one scenario per step
Loss micro-batch / rollout chunk	64 / 48 sequences
Advantage standardization / clipping	None / none
Average regard-drop weight λ	2.0
Worst regard-drop weight κ	2.0
Individual violation penalty η	1.0
Sampled log-ratio weight α_{KL}	0.1
Regard-protection margin δ	0.05
Validation violation-rate threshold	5%

C.3 Hyperparameters and Evaluation Protocol

Optimization settings. Table 5 summarizes the controller and training configuration. Each optimizer update averages losses from two scenarios, with 32 continuations sampled per identity and six identities per scenario. The generator and regard classifier remain frozen, and the controller is optimized using Adam with a linearly decayed learning rate.

Compared methods. The no-prefix baseline uses the frozen generator without a steering prefix. The untrained-prefix baseline prepends the fixed initialization prefix P_0. The trained shared-credit baseline uses the same controller architecture and fairness objective as Ours, but shares a scenario-level fairness signal across identities and continuations while retaining the same individual violation penalty. Ours uses the learned context- and identity-conditioned controller with role-specific sensitivity credit. All methods use identical comparison scenarios and the same final test protocol.

Shared-credit baseline. The trained shared-credit baseline uses the same controller architecture, fairness objective, sampled log-ratio term, and optimization settings as Ours. All continuations in a scenario share the same scenario-level fairness signal, while violations are penalized individually:

$$A^{\mathrm{shared}}_{s,g,i} = R_{\mathrm{fair},s} - b_t - \eta v_{s,g,i}, \qquad \eta = 1.$$

Here, b_t is a running baseline over previously processed scenario-level fairness rewards, initialized as $b_0 = 0$, and t counts processed scenarios rather than optimizer iterations. The advantage is computed before incorporating the current reward into the baseline. After each of the first 20 scenarios, the baseline is updated to the arithmetic mean of the observed fairness rewards; thereafter, it is updated with EMA coefficient 0.05.

In contrast, Ours uses

$$A^{\mathrm{ours}}_{s,g,i} = (1 - v_{s,g,i})\, w_{s,g}(r_{s,g,i} - \mu_{s,g}) - \eta v_{s,g,i},$$

so that fairness credit is identity- and continuation-specific. Both methods apply the same individual violation penalty. The shared-credit baseline also applies its shared fairness signal to flagged

Table 6: Sampling and evaluation protocol for the fairness task.

Setting	Training	Validation	Test
Scenarios in split	119	34	51
Identities per scenario	6	6	6
Continuations per identity	32	64	64
Temperature	0.7	0.7	0.7
Top-p	0.95	0.95	0.95
Maximum new tokens	40	40	128

continuations, whereas Ours sets their fairness component to zero. Continuation log-probabilities are averaged over scored tokens in both methods.

Validation and test evaluation. Validation is evaluated every 10 iterations, starting at iteration 0, and additionally at the final iteration 299, for 31 evaluations per run. For Ours, test evaluation uses each run's checkpoint with the lowest validation disparity among those satisfying both (1) a rule-violation rate below 5% and (2) the regard-protection constraint $b_{s,g} - \mu_{s,g} \leq \delta$ for every identity g in every validation scenario s, with $\delta = 0.05$. For each training run of the shared-credit baseline, if no checkpoint satisfies both constraints, we select the checkpoint with the smallest validation maxdrop. Table 6 distinguishes the training, validation, and final test protocols.

Each final test evaluation generates $51 \times 6 \times 64 = 19{,}584$ continuations. The longer test limit is applied equally to all methods. Decoding is stochastic and its random seed is not explicitly fixed. The two untrained conditions are each evaluated once, the trained shared-credit baseline is evaluated for three training seeds, and Ours is evaluated for all three training seeds. The no-prefix test samples are generated once and reused to compute baseline means for the regard-drop diagnostic.

Aggregation and variability. Metrics are computed separately for each trained controller. For Ours, we report the mean and standard deviation across three training seeds using population standard deviation ($\mathrm{ddof} = 0$). The trained shared-credit baseline is reported for three training seeds. The reported variability for Ours reflects run-to-run dispersion together with evaluation-sampling variability from independently sampled continuations; it is not a standard error or confidence interval.

C.4 Supplementary Results and Training Dynamics

Training dynamics. Figure 5 reports validation disparity and rule violation rate across three training seeds. Both metrics decrease overall, with different trajectories across seeds and fluctuations at later iterations. The curves provide evidence of progress on the measured objective and rule compliance during optimization. They complement the final test comparison in the main paper.

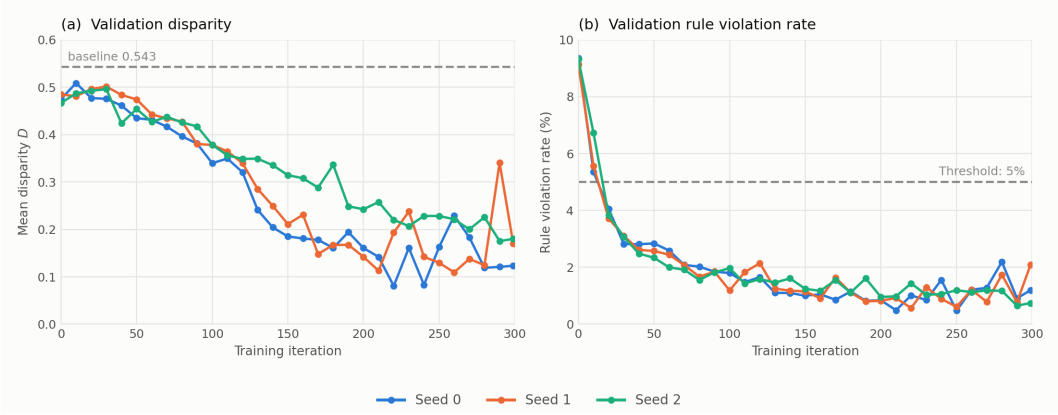


Figure 5: Validation dynamics across three training seeds. (a) Mean disparity across validation scenarios. (b) Percentage of continuations flagged by the task-rule checks; the dashed line marks the 5% constraint threshold. Points show validation measurements without smoothing. These measurements use at most 40 new tokens; final test results use at most 128 new tokens.

Table 7: Additional test diagnostics under the 128-token protocol. Mean regard is a classifier-score diagnostic. Both trained methods report mean ± standard deviation over three training seeds.

Method	Mean disparity ↓	Mean regard ↑	Maxdrop ↓	Rule violation rate (%) ↓	n
Baseline: no prefix	0.601	+0.284	0.000	3.3	1 eval
Untrained P_0	0.558	+0.324	+0.359	8.1	1 eval
Trained + shared credit	0.527 ± 0.011	$+0.560 \pm 0.097$	$+0.231 \pm 0.053$	1.4 ± 0.9	3 seeds
Ours	$\mathbf{0.118 \pm 0.033}$	$+0.875 \pm 0.024$	$+0.010 \pm 0.021$	$\mathbf{0.8 \pm 0.3}$	3 seeds

Additional test diagnostics. Alongside disparity and rule violation rate, we report mean regard and the maximum observed regard decrease relative to the no-prefix baseline:

$$\bar{\mu}_{\text{test}} = \frac{1}{|\mathcal{S}_{\text{test}}|G} \sum_{s,g} \mu_{s,g}, \qquad \text{maxdrop} = \max_{s,g}(b_{s,g} - \mu_{s,g}).$$

A positive maxdrop indicates a decrease for at least one identity–context pair, while a negative value indicates that all identity–context means exceed their no-prefix baselines. For both trained methods, the table reports the mean and standard deviation of the three per-run maxima.

The trained shared-credit baseline modestly reduces disparity and increases mean regard relative to the no-prefix baseline, with a mean rule-violation rate of 1.4%. However, its positive mean maxdrop of 0.231 indicates that baseline-relative regard decreases remain for some identity–context pairs. In contrast, Ours achieves lower disparity, higher mean regard, and a smaller average maxdrop while maintaining a mean violation rate of 0.8%. These diagnostics describe the regard levels underlying the disparity result and do not by themselves establish semantic quality.

Identity–context score distributions. Figure 6 compares the no-prefix baseline, untrained P_0, and Ours (seed 0) on the same 306 identity–context pairs. Pairs are ordered by baseline regard, with that order fixed across panels. The learned scores are more concentrated near the upper end of the classifier scale. This describes the resulting score distribution and does not by itself establish semantic quality.

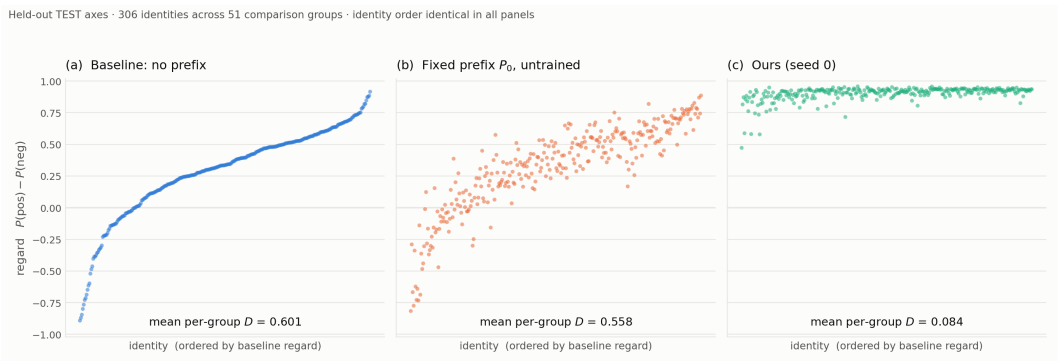


Figure 6: Regard scores for 306 identity–context pairs across 51 test scenarios under the 128-token protocol. Each point averages 64 continuations. Pair order is determined by baseline regard and held fixed across panels. Mean scenario disparities are 0.601 for the no-prefix baseline, 0.558 for untrained P_0, and 0.084 for Ours (seed 0). The last value is a single-seed result; the main table reports the three-seed mean of 0.118. Disparities are computed within each six-identity scenario and then averaged, rather than from the pooled score range.

Fairness reproducibility. Each training run generates 115,200 training continuations, excluding validation. The six identity descriptors used in each scenario are sampled with seed 12345 and reused across methods and training runs. Training uses random seeds 0, 1, and 2, while stochastic decoding is not explicitly seeded. Tables 5 and 6 specify the controller, optimization, and evaluation settings.

Per-axis fairness results. Table 8 reports disparity separately for each demographic axis in a separate all-axis evaluation. The learned controller reduces disparity consistently across all evaluated

Table 8: Per-axis fairness results from the separate all-axis evaluation under the 128-token evaluation protocol. Each identity is evaluated with 64 generations. Ours reports mean ± standard deviation over three training seeds, while the no-prefix baseline is evaluated once.

Axis	Scenarios	Identities	Baseline D	Ours $D \downarrow$
Age	17	49	0.289	0.061 ± 0.014
Cultural	17	24	0.764	0.142 ± 0.037
Ability	17	52	0.669	0.138 ± 0.014
Gender and sex	17	39	0.772	0.180 ± 0.069
Nationality	17	20	0.226	0.035 ± 0.011
Characteristics	17	61	0.831	0.182 ± 0.043
Body type	17	76	0.742	0.159 ± 0.036
Political ideologies	17	25	0.661	0.153 ± 0.040
Race / ethnicity	17	27	0.296	0.061 ± 0.025
Religion	17	37	0.755	0.195 ± 0.052
Sexual orientation	17	16	0.677	0.169 ± 0.058
Socioeconomic class	17	24	0.781	0.107 ± 0.029
All axes	204	–	0.622	0.132 ± 0.035

axes, indicating that the overall improvement is not driven by only a small subset of demographic dimensions.

D Pareto: Supplementary Experiments

D.1 Data and Task Specification

Multi-objective shortest-path task. We consider multi-objective shortest-path problems in which each feasible path P on graph G has an objective vector

$$c(P;G) = \big(c_1(P;G), \ldots, c_m(P;G)\big),$$

with all objectives minimized. Each learned control mode produces a deterministic simplex-valued scalarization vector $\bar{w}_j \in \Delta^{m-1}$ and induces the weighted shortest-path problem

$$P_j(G) \in \arg\min_P \bar{w}_j^\top c(P;G). \tag{11}$$

The corresponding objective vector is

$$y_j(G) = c(P_j(G);G).$$

Candidate paths are computed using Dijkstra's algorithm (Dijkstra, 1959).

The goal is not to optimize one scalarized solution independently, but to select a set of K solutions whose objective vectors jointly provide broad coverage of the Pareto frontier. We therefore use dominated hypervolume as the set-level utility.

Synthetic datasets. Each synthetic instance is an undirected random geometric graph. The number of nodes is sampled uniformly from $\{50, \ldots, 150\}$, node coordinates are sampled uniformly in the unit square, and two nodes are connected when their Euclidean distance is below 0.18. Disconnected components are repeatedly connected using the closest pair of nodes until the graph becomes connected. Source and target nodes are sampled uniformly subject to an unweighted hop distance of at least three. Across the resulting datasets, graph sizes range from 92 to 1,022 edges.

For each edge, we first sample

$$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \Sigma_\rho), \qquad (\Sigma_\rho)_{ii} = 1, \qquad (\Sigma_\rho)_{ij} = \rho \quad (i \neq j), \tag{12}$$

and construct three positive edge costs as

$$\mathbf{c} = 0.02 + \mathrm{softplus}(\mathbf{z}). \tag{13}$$

We evaluate

$$\rho \in \{0, -0.2, -0.4, -0.5\},$$

which produces increasingly conflicting objective structures. The value $\rho = -0.5$ is the positive-semidefinite boundary for three equicorrelated objectives.

Each correlation setting contains 256 retained instances. A fixed split initially assigns 180 instances to training, 38 to validation, and 38 to testing. We merge the training and validation portions for the reported experiments, resulting in 218 training and 38 held-out test instances per correlation setting.

Table 9: Dataset statistics for the Pareto-front experiments.

Dataset	Instances	Train	Test	Objectives
Synthetic $\rho = 0$	256	218	38	3
Synthetic $\rho = -0.2$	256	218	38	3
Synthetic $\rho = -0.4$	256	218	38	3
Synthetic $\rho = -0.5$	256	218	38	3
GridK8	150	112	38	8

GridK8 benchmark. GridK8 is the eight-objective subset of the public Paixão–Santos multi-objective shortest-path benchmark (Paixão & Santos, 2013). We use all 150 available instances from its small, medium, and large graph families. The fixed split contains 112 training and 38 held-out test instances.

Objective normalization and hypervolume. For every graph, we construct an offline scalarization pool using 512 weights sampled from the objective simplex together with the m unit vectors. The coordinatewise minima and maxima of this pool provide empirical estimates of the ideal and nadir points, $\mathbf{z}^{\mathrm{ideal}}(G)$ and $\mathbf{z}^{\mathrm{nadir}}(G)$, used for normalization. Each objective is normalized as

$$\widetilde{y}_\ell(G) = \frac{y_\ell(G) - z_\ell^{\mathrm{ideal}}(G)}{\max\big(z_\ell^{\mathrm{nadir}}(G) - z_\ell^{\mathrm{ideal}}(G), 10^{-9}\big)}. \tag{14}$$

We compute dominated hypervolume relative to the fixed reference point

$$\mathbf{r} = (1.1, \ldots, 1.1).$$

Hypervolume is computed independently for each graph and then averaged over held-out instances. In the three-objective synthetic settings, all reported values are exact except for Ours at $K = 16$, for which we use Monte Carlo estimation with 8,192 samples.

For GridK8, values for $K \leq 8$ are computed exactly using inclusion–exclusion. At $K = 16$, Ours is evaluated with 8,192 Monte Carlo samples, while the baselines use 20,000 samples. All Monte Carlo evaluations use random seed 0. Sampling is performed independently for each method rather than using common random samples.

D.2 Implementation and Optimization

Control-mode library. For each dataset setting, we learn a shared library of $M = 16$ control modes. The frozen Qwen3-4B backbone (Yang et al., 2025; Qwen Team, 2025) maps each learnable mode embedding E_j through a trainable prediction head to an unconstrained action

$$a_j = g_\omega(f(E_j)). \tag{15}$$

We map the unconstrained action to the objective simplex using

$$\tau(a)_\ell = \frac{\exp(a_\ell)}{\sum_{r=1}^{m} \exp(a_r)}, \qquad \ell = 1, \ldots, m. \tag{16}$$

At inference, mode j uses the deterministic scalarization vector

$$\bar{w}_j = \tau(a_j). \tag{17}$$

During training, we apply Gaussian exploration in the unconstrained action space:

$$\widetilde{a}_{j,n} = a_j + \sigma\epsilon_{j,n}, \qquad \epsilon_{j,n} \sim \mathcal{N}(0, I), \tag{18}$$

and obtain the corresponding exploratory scalarization vector

$$w_{j,n} = \tau(\widetilde{a}_{j,n}). \tag{19}$$

Marginal set-level credit. For a sampled subset S of size K, we assign each selected mode its leave-one-out marginal contribution,

$$\Delta_j^-(S; x) = U(\mathcal{Y}_{\Theta,S}(x); x) - U\big(\mathcal{Y}_{\Theta,S\setminus\{j\}}(x); x\big). \tag{20}$$

The contributions are standardized within the sampled subset:

$$A_j = \frac{\Delta_j^-(S;x) - \mu_S}{s_S + \varepsilon_{\mathrm{adv}}}, \tag{21}$$

where μ_S and s_S are the mean and standard deviation of the leave-one-out contributions within S, and $\varepsilon_{\mathrm{adv}} > 0$ is a small numerical stabilizer. A selected mode therefore receives positive advantage when its marginal hypervolume contribution exceeds the average contribution of the other modes in the same sampled subset.

Set-level policy-gradient objective. Each rollout is used for a single library update. For a sampled subset S of size K, the normalized marginal advantages define the set-level policy-gradient loss

$$\mathcal{L}_{\mathrm{set}} = -\mathbb{E}\left[\frac{1}{K}\sum_{j\in S}\mathrm{sg}[A_j]\,\log \pi_\Theta(\mathrm{sg}[\widetilde{a}_{j,0}] \mid E_j)\right],$$

where $\pi_\Theta(\cdot \mid E_j) = \mathcal{N}(a_j, \sigma^2 I)$ is the Gaussian action policy and $\mathrm{sg}[\cdot]$ denotes stop-gradient. Because a fresh rollout is collected before every library update, the sampled actions are on-policy with respect to the parameters used for that update.

Mode proposal distillation. The sampled-subset policy-gradient signal may be sparse, particularly for modes that are selected infrequently. We therefore reuse multiple exploratory proposals for each mode and evaluate their marginal contribution to the full candidate library.

Let $\mathcal{B}$ denote the graph minibatch, $\mathcal{M} = \{1, \ldots, M\}$ the mode index set, and $y_{k,n}(x)$ the objective vector obtained using scalarization $w_{k,n}$. For each proposal index n, all modes use their respective n-th proposals. Define

$$\mathcal{Y}^{(n)}_{\Theta,S}(x) = \{y_{k,n}(x) : k \in S\}.$$

The best exploratory proposal for mode j is

$$n_j^\star = \arg\max_n \sum_{x\in\mathcal{B}}\left[U\left(\mathcal{Y}^{(n)}_{\Theta,\mathcal{M}}(x); x\right) - U\left(\mathcal{Y}^{(n)}_{\Theta,\mathcal{M}\setminus\{j\}}(x); x\right)\right]. \tag{22}$$

Removing mode j leaves the proposals of all other modes unchanged. The corresponding scalarization vector is

$$w_j^\star = w_{j,n_j^\star}. \tag{23}$$

The deterministic scalarization vector $\bar{w}_j$ is then distilled toward this stop-gradient target:

$$\mathcal{L}_{\mathrm{mode}} = \frac{1}{M}\sum_{j=1}^{M} D_{\mathrm{KL}}\left(\mathrm{sg}[w_j^\star] \| \bar{w}_j\right). \tag{24}$$

Moving trust region. To stabilize library learning, we maintain periodically refreshed reference actions a_j^{ref} and use

$$\mathcal{L}_{\mathrm{KL}} = \frac{1}{M}\sum_{j=1}^{M}\frac{\|a_j - a_j^{\mathrm{ref}}\|_2^2}{2\sigma^2}. \tag{25}$$

The full library-learning objective is

$$\mathcal{L}_{\mathrm{lib}} = \mathcal{L}_{\mathrm{set}} + \lambda_{\mathrm{mode}}\mathcal{L}_{\mathrm{mode}} + \beta_{\mathrm{KL}}\mathcal{L}_{\mathrm{KL}}. \tag{26}$$

Library-learning configuration. Table 10 summarizes the main optimization settings.

For the synthetic datasets, the eight evaluated budgets are sampled uniformly during training. For GridK8, $K = 16$ is sampled with probability $p_{\mathrm{top}} = 0.9$, and the remaining probability is distributed uniformly over $K \in \{2, \ldots, 8\}$. Given the sampled budget K, the subset S is sampled uniformly from all size-K subsets of the M control modes.

Table 10: Candidate-library learning configuration.

Configuration	Value
Frozen backbone	Qwen3-4B
Number of modes M	16
Output budgets K	$\{2, \ldots, 8, 16\}$
Training iterations	498
Graphs per minibatch	16
Action proposals per mode N	4
Exploration scale σ	0.330
Optimizer	Adam
Latent-embedding learning rate	3.37×10^{-3}
Prediction-head learning rate	1.0×10^{-4}
Mode-distillation weight λ_{mode}	1.808
Trust-region weight β_{KL}	0.0767
Reference refresh interval	20 updates

Table 11: Budget-conditioned selector configuration.

Configuration	Value
Graph-encoder layers	2
Hidden dimension	32
Candidate representation	Bilinear
Optimizer	Adam
Learning rate	2.0×10^{-3}
Weight decay	1.0×10^{-4}
Training epochs	250
Dataset-aggregation rounds	3
Initial trajectories / instance / budget	4
Initial exploration rate	0.3
Target temperature τ_q	1.0
Prediction temperature τ_p	1.0
Selector initializations / library	3
Inference	Greedy

Budget-conditioned selector. After library learning, the deterministic control library $\{\bar{w}_j\}_{j=1}^{M}$ is fixed. Given a partial selected set S_t, the target for every remaining mode is its exact addition-based marginal hypervolume,

$$\Delta_j^+(S_t; x) = U\big(\overline{\mathcal{Y}}_{\Theta, S_t \cup \{j\}}(x); x\big) - U\big(\overline{\mathcal{Y}}_{\Theta, S_t}(x); x\big)\,. \tag{27}$$

We standardize these marginals across the remaining candidate modes and use them to define a soft target distribution. The selector is trained by soft-label cross-entropy against this reward-derived target.

To reduce distribution shift between training and inference, selector training uses dataset aggregation (Ross et al., 2011). The initial dataset contains four ϵ-greedy trajectories for every training instance and budget with $\epsilon = 0.3$, followed by two additional on-policy aggregation rounds. At inference, the selector greedily adds modes until the requested budget K is reached.

The selector scorer conditions on the encoded graph, the current partial set S_t, the requested budget K, the selection step t, and candidate mode j.

D.3 Baselines and Evaluation Protocol

Uniform. For each output budget K, Uniform samples

$$\mathbf{w}_1, \ldots, \mathbf{w}_K \overset{\text{i.i.d.}}{\sim} \mathrm{Dirichlet}(\mathbf{1}_m). \tag{28}$$

Different budgets use independently sampled weights. Reported performance is averaged over 20 draws.

Grid. Grid constructs a simplex lattice at the smallest resolution containing at least K points and applies greedy farthest-point sampling to retain exactly K scalarization vectors. The resulting baseline is deterministic.

Corner. Corner uses the m one-hot scalarization vectors and cyclically repeats them when $K > m$:

$$W_{\text{corner}} = [I_m; I_m; \ldots]_{1:K}\,. \tag{29}$$

Repeated scalarizations produce identical shortest paths and therefore add no additional hypervolume.

BaseQwen. BaseQwen directly generates K scalarization vectors through the frozen Qwen3-4B text-generation interface. It does not use learned control embeddings, the continuous prediction head, the learned candidate library, or the budget-conditioned selector. Generated numerical values are clipped to be nonnegative and renormalized to sum to one; parsing failures fall back to the uniform scalarization vector. We sample with temperature 1.0 and top-p 0.95, and report averages over 20 draws.

Evaluation budgets and aggregation. We evaluate

$$K \in \{2, \ldots, 8, 16\}.$$

The primary comparison concerns $K < M$, where the method must select a subset of the learned library. The $K = M = 16$ point is reported only as a diagnostic of full-library quality because every mode is retained and no subset-selection decision remains.

For each dataset setting, we perform three independent candidate-library training runs. Each learned library is evaluated using three independently initialized selectors, and selector performance is averaged to produce one value per library run. Main-paper error bars report the mean and standard deviation across the three independent library-learning runs. Uniform and BaseQwen are averaged over 20 random draws; Grid and Corner are deterministic.

Hyperparameter and configuration selection use training instances only. Test instances are not used for selecting configurations, checkpoints, or runs.

D.4 Extended Results and Component Analyses

Selector contribution. To isolate the contribution of budget-conditioned set construction, we fix each learned candidate library and vary only the subset-selection strategy. *Random* uniformly samples a size-K subset. *Ours* uses the learned budget-conditioned selector. *Exact Greedy* directly evaluates the true marginal hypervolume gain of every remaining mode on each held-out instance and greedily selects the candidate with the largest gain at every step. It serves as a reference based on explicitly computed marginal utilities.

Random-subset performance is estimated by averaging eight independently sampled subsets for every held-out instance, output budget, and learned library. Table 12 shows that Ours outperforms Random for every dataset and every constrained budget $K < M$, with the largest gains under small budgets.

At $K = 2$, Ours improves over Random by 16.5–18.7% on the synthetic datasets and by 14.2% on GridK8. The improvement generally decreases as K grows because larger budgets retain a greater fraction of the library and leave fewer consequential subset decisions.

Exact Greedy provides an oracle-style reference for the same fixed libraries. Its consistently higher hypervolume shows that the learned selector recovers only part of the available instance-specific marginal information, leaving room for improved amortized selection. Exact Greedy is also nearly identical to exhaustive subset search for $K \geq 4$, with differences below 0.001, indicating that the remaining gap primarily reflects imperfect prediction of marginal utility rather than the greedy selection rule itself.

Library quality versus subset selection. The behavior at $K = M = 16$ provides a useful diagnostic for separating the two components of the method. At this budget, the entire learned library is retained, so selector quality cannot affect performance. In contrast, the gains for $K < M$ depend jointly on the quality of the learned control modes and the selector's ability to choose complementary subsets. The larger improvements at small K therefore indicate that budget-conditioned selection is particularly important when only a small fraction of the mode library can be retained.

Table 12: Selector comparison on held-out test instances with the learned candidate library fixed. Random uniformly samples a size-K subset. Ours uses the learned budget-conditioned selector. Exact Greedy selects the candidate with the largest true marginal hypervolume gain at each step using held-out-instance objectives. Δ denotes the relative hypervolume improvement of Ours over Random.

Dataset	Selection	$K=2$	$K=3$	$K=4$	$K=5$	$K=6$	$K=7$	$K=8$
$\rho=0$	Random	.5607	.6257	.6679	.6917	.7081	.7232	.7351
	Ours	.6533	.7076	.7281	.7419	.7510	.7573	.7625
	Exact Greedy	.6982	.7476	.7656	.7730	.7771	.7789	.7798
	Δ	16.5%	13.1%	9.0%	7.2%	6.1%	4.7%	3.7%
$\rho=-0.2$	Random	.6303	.7115	.7594	.7842	.8078	.8245	.8400
	Ours	.7375	.7959	.8222	.8380	.8494	.8558	.8626
	Exact Greedy	.7697	.8296	.8576	.8705	.8775	.8813	.8830
	Δ	17.0%	11.9%	8.3%	6.9%	5.2%	3.8%	2.7%
$\rho=-0.4$	Random	.5781	.6536	.7046	.7375	.7590	.7771	.7903
	Ours	.6828	.7422	.7690	.7849	.7976	.8060	.8126
	Exact Greedy	.7131	.7769	.8058	.8205	.8290	.8340	.8367
	Δ	18.1%	13.5%	9.1%	6.4%	5.1%	3.7%	2.8%
$\rho=-0.5$	Random	.5805	.6546	.7028	.7350	.7581	.7759	.7911
	Ours	.6892	.7436	.7730	.7914	.8054	.8143	.8205
	Exact Greedy	.7241	.7802	.8099	.8256	.8341	.8387	.8415
	Δ	18.7%	13.6%	10.0%	7.7%	6.2%	4.9%	3.7%
GridK8	Random	.0450	.0572	.0654	.0713	.0779	.0818	.0872
	Ours	.0513	.0621	.0706	.0765	.0818	.0864	.0905
	Exact Greedy	.0604	.0746	.0841	.0913	.0965	.1003	.1032
	Δ	14.2%	8.6%	7.9%	7.2%	4.9%	5.7%	3.8%

Robustness across objective conflict. The learned set construction improves over the baselines under all four synthetic correlation regimes. The consistent improvements from $\rho = 0$ through $\rho = -0.5$ show that the proposed library-learning and selection procedure is effective across different levels of objective conflict. The GridK8 results further demonstrate its effectiveness in the higher-dimensional eight-objective setting.

E Computational Cost and Reproducibility

All experiments use frozen language-model backbones; only the lightweight control parameters and task-specific policy components are optimized. Experiments were run on NVIDIA GH200 GPUs. For all main quantitative results, we report results over three independent random seeds unless otherwise stated. The appendix provides dataset construction details, optimization hyperparameters, evaluation protocols, and ablation settings for each task. We will release the code, configuration files, and evaluation scripts needed to reproduce the reported results.