GEOMETRY-GUIDED EXPLORATION WITH ADAPTIVE TERMINATION FOR GFLOWNETS

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative Flow Networks (GFlowNets) learn to sample compositional objects in proportion to their rewards, requiring exploration that discovers diverse high-reward objects. Existing exploratory policies can repeatedly acquire experiences that remain useful for learning but are already available through replay. We propose *Geometry-guided Exploration with Adaptive Termination (GEAT)*, a simple yet effective algorithm for efficient exploration in GFlowNets. Its core mechanism, *novelty-aware termination* (NAT), generates terminal candidates through local backtracking and reconstruction, then selects the candidate with the highest novelty relative to reference states sampled from replay. Only the selected candidate requires a task-reward evaluation. To guide this selection, we derive *geometric novelty* from the derivative of geometry-aware Shannon entropy. The resulting score incorporates similarities between state representations and ranks candidates by their first-order effect on the entropy of the reference distribution. GEAT uses this score both as an intrinsic reward for the exploratory policy and as the terminal-selection criterion. Across extensive benchmarks, we show that GEAT consistently discovers high-reward modes faster and achieves broader mode coverage while maintaining competitive reward quality and distributional accuracy, demonstrating improved sample efficiency in mode discovery across diverse tasks.

1 INTRODUCTION

Generative Flow Networks (GFlowNets; Bengio et al., 2021) are amortized samplers that learn stochastic policies to sequentially construct objects and sample them with probabilities proportional to their rewards. Applications include scientific discovery (Jain et al., 2023), diffusion model alignment (Liu et al., 2025), language-model red-teaming (Kwon et al., 2026), and recommender systems (Yu et al., 2026). These applications often involve large combinatorial spaces with multiple high-reward modes. Learning the reward-proportional distribution therefore requires exploration that discovers diverse high-reward objects. To improve exploration, recent methods (Madan et al., 2025; Kim et al., 2025; Malek et al., 2026; Dall'Antonia et al., 2026) train a separate exploratory policy to collect training trajectories for the GFlowNet.

A key challenge is deciding which experiences warrant additional collection. Several exploratory-policy methods prioritize experiences according to discrepancies in the current GFlowNet. For example, Kim et al. (2025) and Malek et al. (2026) prioritize trajectories with high training loss, while Dall'Antonia et al. (2026) favor terminal states that are underrepresented relative to the reward-proportional target distribution. These signals identify where further learning is needed, but do not directly establish whether additional data collection is necessary. GFlowNets support off-policy training, allowing previously collected experiences to be reused through replay, which has been shown to improve mode discovery and sample efficiency (Vemgal et al., 2023; Shen et al., 2023; Kim et al., 2025). When relevant experiences are already available in replay, repeatedly acquiring them can consume exploration resources without expanding coverage. Thus, the need for further learning should be distinguished from the need for further exploration.

Figure 1 illustrates this issue on a two-dimensional HyperGrid, a discrete grid in which an agent moves by incrementing one coordinate at a time and receives a reward upon termination. The target distribution assigns high probability to four separated regions. We examine ACE (Dall'Antonia et al., 2026) as a representative prior method that trains a separate exploratory policy. Its exploratory

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

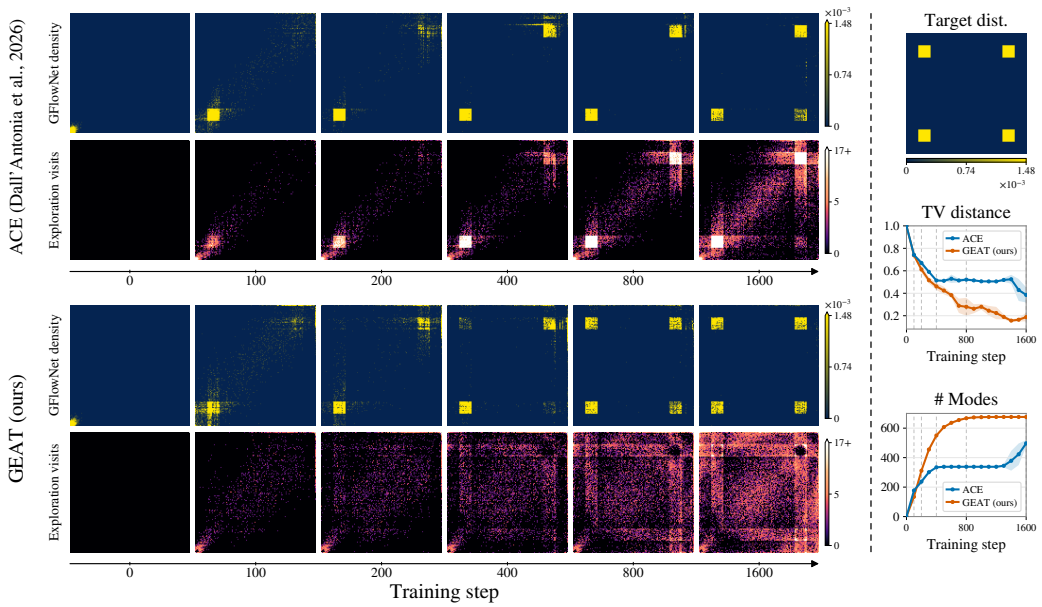

Figure 1: Empirical density of the GFlowNet and cumulative visits of the exploratory policy on the HyperGrid. Compared with a prior method, ACE (Dall'Antonia et al., 2026), GEAT distributes exploration visits more broadly across the high-reward regions, leading to lower total variation distance and broader mode coverage.

policy repeatedly visits already discovered high-reward regions, while other regions remain sparsely explored for much of training. The GFlowNet likewise concentrates its probability mass on only a subset of the high-reward regions. Hence, despite continued data collection, ACE fails to achieve broad high-reward coverage within the displayed training budget, and the learned distribution remains substantially different from the target distribution. This example highlights how repeated acquisition of previously collected experiences can limit the effectiveness of exploration, even when those experiences remain useful for further learning.

To address this issue, we propose *Geometry-guided Exploration with Adaptive Termination* (GEAT), which selects terminal states for acquisition according to their novelty relative to previously collected experience. The GFlowNet learns the reward-proportional target distribution, while a separate exploration GFlowNet generates candidates for data collection. Our approach distinguishes two decisions: where to generate candidates and which candidate to acquire. Task rewards guide candidate generation toward high-reward regions, while novelty determines the final selection among the generated candidates. Previously acquired experiences can continue to support learning through replay, allowing new reward evaluations to focus on candidates that extend coverage.

GEAT implements this principle through *novelty-aware termination* (NAT), which generates local alternatives to a sampled terminal state and selects the candidate with the highest novelty score. To evaluate novelty, we introduce *geometric novelty*, a score derived from geometry-aware Shannon entropy that accounts for similarities between candidate representations and reference states sampled from replay. This distinguishes candidates that are merely different from previously acquired objects from those that expand coverage of the representation space. The same score also serves as an intrinsic reward for training the exploration GFlowNet.

Our contributions are as follows:

- We propose GEAT (Section 4), an exploration algorithm for GFlowNets that combines reward-guided candidate generation with terminal-state selection based on geometric novelty relative to previously collected experience. Its core mechanism, NAT (Section 4.1), implements this selection by generating local alternatives to a sampled terminal state. Specifically, starting from a terminal state sampled by the exploration GFlowNet, NAT generates additional candidates through back-

tracking and reconstruction, then selects the candidate with the highest novelty score. Only the selected candidate requires a task-reward evaluation, enabling comparison among multiple local candidates without evaluating the rewards of discarded proposals.

- We derive geometric novelty (Section 4.2) by differentiating geometry-aware Shannon entropy at an empirical reference distribution formed from replay. Using a similarity kernel over state representations, the score accounts for overlap with previously collected experience, including similarity between distinct objects. It ranks candidates by their first-order effect on geometry-aware entropy when probability mass is shifted toward each candidate. GEAT uses this score both to train the exploratory policy and to select terminal states through NAT.
- We extensively evaluate GEAT on synthetic, sequence generation, molecular, and peptide design benchmarks. GEAT consistently discovers high-reward modes faster and achieves broader mode coverage while maintaining competitive reward quality and distributional accuracy, demonstrating improved sample efficiency in mode discovery across diverse tasks.

2 Related Work

Exploration in GFlowNets. Exploration has been studied to improve mode discovery in GFlowNets. Rector-Brooks et al. (2023) use an ensemble of forward policies to guide exploration through Thompson sampling. Kim et al. (2024) use local search to exploit high-reward regions through backtracking and reconstruction. Madan et al. (2025) train an exploratory GFlowNet using intrinsic rewards such as Random Network Distillation (RND), while Kim et al. (2025) and Malek et al. (2026) direct an auxiliary policy toward regions where the original GFlowNet exhibits high training loss. More recently, Dall'Antonia et al. (2026) focus exploration on high-reward regions that remain underrepresented by the original GFlowNet. Our work focuses on reducing repeated exploration of previously explored regions by adapting terminal-state selection according to novelty.

Novelty and similarity-aware exploration. Novelty is commonly estimated from visitation or prediction error. Count-based methods assign higher novelty to less frequently visited states (Bellemare et al., 2016), while RND (Burda et al., 2019) uses prediction error against a fixed random network as an intrinsic reward. These signals, however, do not explicitly account for similarity between distinct structured objects. Leinster & Cobbold (2012) introduced diversity measures that incorporate similarity between categories, and Posada et al. (2020) extended this idea to a geometry-aware information-theoretic framework. Guo et al. (2021) applied geometry-aware Shannon entropy to promote broad state-space coverage in RL through joint optimization of the policy and an auxiliary function. In contrast, we directly differentiate this entropy at an empirical reference distribution sampled from replay to obtain a pointwise novelty score for terminal-state selection.

3 Preliminaries

3.1 GFlowNets and Trajectory Balance

A GFlowNet is defined on a finite directed acyclic graph $\mathcal{G} = (\mathcal{S}, \mathcal{E})$, where s_0 denotes the initial state and $\mathcal{X} \subseteq \mathcal{S}$ the set of terminal states. For a complete trajectory $\tau = (s_0, \dots, s_T = x)$, define

$$P_F(\tau) = \prod_{t=0}^{T-1} P_F(s_{t+1} \mid s_t) \qquad \text{and} \qquad P_B(\tau \mid x) = \prod_{t=0}^{T-1} P_B(s_t \mid s_{t+1}),$$

absorbing any explicit `STOP` action into the final transition. A GFlowNet $G = (Z, P_F, P_B)$ aims to induce a terminal marginal $p^\top(x) := \sum_{\tau: s_0 \rightsquigarrow x} P_F(\tau)$ proportional to a positive reward $R : \mathcal{X} \to \mathbb{R}_{>0}$. Trajectory Balance (TB; Whittammer et al., 2022) enforces $ZP_F(\tau) = R(x)P_B(\tau \mid x)$ for every complete trajectory. When this condition holds, $p^\top(x) \propto R(x)$ with $Z = \sum_{x \in \mathcal{X}} R(x)$. In practice, TB minimizes the squared log discrepancy

$$\mathcal{L}_{\mathrm{TB}}(\tau; G, R) = \left[\log \frac{ZP_F(\tau)}{R(x)P_B(\tau \mid x)}\right]^2.$$

Zero TB loss for all complete trajectories therefore recovers the reward-proportional terminal distribution. Since this loss can be evaluated on trajectories drawn from any sufficiently supported behavior distribution, TB also supports off-policy training.

3.2 EXPLORATION GFLOWNETS

We consider a two-GFlowNet setup consisting of a *task GFlowNet* and an *exploration GFlowNet*, following prior work on exploratory policies for GFlowNets (Madan et al., 2025; Kim et al., 2025; Dall'Antonia et al., 2026). We denote the task GFlowNet by $G_\theta := (Z_\theta, P_F^\theta, P_B^\theta)$ and the exploration GFlowNet by $G_\phi := (Z_\phi, P_F^\phi, P_B^\phi)$. The task GFlowNet is trained with the task reward R^{task} and aims to recover the corresponding reward-proportional terminal distribution. The exploration GFlowNet is trained with a separate exploration reward

$$R_t^{\text{exp}}(x) = \big(R^{\text{task}}(x)\big)^\alpha + \beta N_t(x),$$

where $\alpha > 0$ controls task-reward tempering, $N_t(x)$ is a novelty-based intrinsic reward, and β controls its contribution. The task-reward term directs G_ϕ toward high-reward regions, while N_t encourages exploration of novel states. Importantly, R_t^{exp} determines only the sampling behavior of G_ϕ and does not alter the target distribution of G_θ. Trajectories generated by G_ϕ are evaluated using R^{task} and can be used to update G_θ off-policy.

3.3 REPLAY FOR GFLOWNET TRAINING

We maintain a replay buffer of previously acquired experiences and reuse them for additional updates of the task GFlowNet. At training step t, the replay buffer is denoted by $\mathcal{B}_t = \{(\tau_i, x_i, R^{\text{task}}(x_i))\}_{i=1}^{N_t}$, where τ_i is a trajectory terminating at x_i. Because GFlowNet objectives support off-policy learning, trajectories stored in $\mathcal{B}_t$ can be reused to update G_θ regardless of the policy that originally generated them. We use prioritized experience replay (PER; Schaul et al., 2016), which assigns each stored trajectory a priority $p_t(\tau_i)$ and samples it according to

$$q_t^{\text{replay}}(\tau_i) = \frac{p_t(\tau_i)^{\beta_{\text{replay}}}}{\sum_{\tau_j \in \mathcal{B}_t} p_t(\tau_j)^{\beta_{\text{replay}}}},$$

where $\beta_{\text{replay}} \geq 0$ controls the strength of prioritization. The priority $p_t(\tau)$ reflects the expected usefulness of an experience for further training. Throughout this work, acquired experiences are stored in $\mathcal{B}_t$ and selectively reused according to this prioritized replay distribution.

4 GEOMETRY-GUIDED EXPLORATION WITH ADAPTIVE TERMINATION

In this section, we present our main algorithm, *Geometry-guided Exploration with Adaptive Termination* (GEAT). It uses representation-space geometry to guide exploration and adaptively determines where to terminate. We first introduce its adaptive termination mechanism, *novelty-aware termination* (NAT), and then describe the *geometric novelty* criterion used to guide exploration and terminal selection. The complete training procedure is summarized in Algorithm 1.

4.1 NOVELTY-AWARE TERMINATION

To improve exploration efficiency by mitigating repeated concentration on previously discovered high-reward regions, we propose NAT. The key idea of NAT is to use the reward signal to guide trajectories toward high-reward regions, while avoiding the use of reward-proportional sampling to directly determine the final terminal state. NAT uses the novelty-based intrinsic reward $N_t(x)$ defined in Section 3.2 to adaptively select the final terminal state; the specific computation of N_t is introduced in the next section. In this way, NAT reduces repeated selection of previously explored high-reward states and promotes broader mode coverage through novelty-based terminal selection. NAT consists of the following three steps:

Step 1. Sampling a Terminal State. We first sample a complete trajectory using the forward policy of the exploration GFlowNet,

$$\tau^{(0)} = (s_0, \ldots, s_T = x^{(0)}) \sim P_F^\phi.$$

This procedure is applied whenever the construction process reaches a terminal state $x^{(0)}$, either through an explicit `STOP` action or a prescribed construction endpoint.

Algorithm 1 Geometry-guided Exploration with Adaptive Termination

Require: Task GFlowNet G_θ, Exploration GFlowNet G_ϕ, replay buffer $\mathcal{B}$, backtracking depth k, number of local proposals n, reference-set size M
  for each training round t **do**
    Sample $\mathcal{T}_t^{\text{task}} \sim P_F^\theta$ and $\mathcal{T}_t^{\text{exp},0} \sim P_F^\phi$
    Sample M previously acquired terminal states $\mathcal{D}_t$ from $\mathcal{B}$
    for each $\tau^{(0)} \in \mathcal{T}_t^{\text{exp},0}$ with terminal $x^{(0)}$ **do**
      Generate terminal candidates $\{x^{(i)}\}_{i=1}^n$ via k-step backtracking–reconstruction
      Select $j^\star \in \arg\max_{j\in\{0,\dots,n\}} N_t^{\text{Geo}}(x^{(j)}; \mathcal{D}_t)$
      Generate $\tau^\star$ by backward sampling from $x^{(j^\star)}$ using P_B^ϕ and add $\tau^\star$ to $\mathcal{T}_t^{\text{exp}}$
    end for
    Evaluate R^{task} for $\mathcal{T}_t^{\text{task}} \cup \mathcal{T}_t^{\text{exp}}$ and add them to $\mathcal{B}$
    Sample $\mathcal{T}_t^{\text{rep}}$ from $\mathcal{B}$ using prioritized experience replay
    Update θ by minimizing $\mathcal{L}_\theta$ on $\mathcal{T}_t^{\text{task}} \cup \mathcal{T}_t^{\text{exp}} \cup \mathcal{T}_t^{\text{rep}}$ using R^{task}
    Relabel $\mathcal{T}_t^{\text{exp}} \cup \mathcal{T}_t^{\text{rep}}$ with the current exploration reward R_t^{exp}
    Update ϕ by minimizing $\mathcal{L}_\phi$ on $\mathcal{T}_t^{\text{exp}} \cup \mathcal{T}_t^{\text{rep}}$
  end for

Step 2. Exploring the Local Neighborhood. Starting from $x^{(0)}$, NAT explores its local neighborhood by generating n additional terminal candidates through backtracking and reconstruction. Different mechanisms can be used to generate these candidates; in our implementation, we adopt the backtracking–reconstruction procedure of Kim et al. (2024). [1] Specifically, NAT backtracks part of the trajectory and reconstructs a new terminal state from the resulting intermediate state.

For each $j = 1, \dots, n$, the backward policy P_B^ϕ backtracks k steps from $x^{(0)}$ to an intermediate state $u^{(j)}$, and the forward policy P_F^ϕ reconstructs a terminal state $x^{(j)}$ from $u^{(j)}$:

$$x^{(0)} \xrightarrow[k \text{ steps}]{P_B^\phi} u^{(j)} \xrightarrow{P_F^\phi} x^{(j)}, \qquad j = 1, \dots, n.$$

Here, k controls the backtracking depth, and n is the number of additional candidates. Together with $x^{(0)}$, these states form the candidate set for terminal selection.

Step 3. Novelty-Based Terminal Selection. NAT adaptively selects the candidate with the highest novelty score:

$$j^\star \in \arg\max_{j\in\{0,\dots,n\}} N_t\left(x^{(j)}\right), \qquad x^\star = x^{(j^\star)}.$$

Only the selected candidate $x^\star$ requires a task-reward evaluation, allowing NAT to compare multiple local candidates without evaluating the rewards of discarded proposals. After selection, a training trajectory $\tau^\star$ is generated by backward sampling from $x^\star$ using the current exploration backward policy P_B^ϕ. We use this max-novelty rule throughout our experiments, with alternative selection rules examined in Appendix D.

We next introduce geometric novelty, which evaluates terminal candidates based on representation-space similarity to reference states sampled from replay. This score is used both for terminal selection in NAT and as an intrinsic reward for training the exploration GFlowNet.

4.2 Geometric Novelty

Both NAT and the reward of the exploration GFlowNet require a criterion for evaluating the novelty of terminal candidates. Our key principle in defining novelty is to measure *how much new information a candidate provides relative to previously explored experience.* In structured discrete spaces, however, distinct states can still be highly similar in the representation space, so simply treating different states as equally novel may fail to reflect how much new information they actually provide. To address this, we introduce geometric novelty, which accounts for representation-space similarity and measures how much a candidate expands the coverage of previously explored experience.

[1] A detailed comparison between GEAT and Kim et al. (2024) is provided in Appendix D.1.

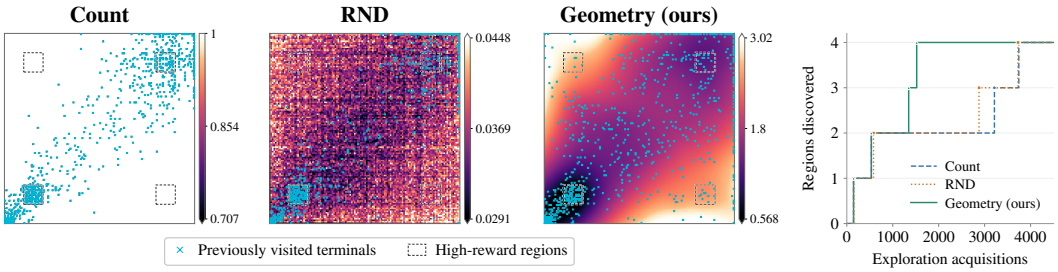

Figure 2: Comparison of novelty criteria within NAT on the HyperGrid with four high-reward regions. The heatmaps show the novelty-based intrinsic reward fields of count, RND, and geometric novelty, with 1,024 reference states sampled from the replay buffer overlaid. With NAT otherwise fixed, Geometric Novelty leads to faster discovery of all four high-reward regions.

Figure 2 illustrates the effect of incorporating representation-space similarity into the novelty-based intrinsic reward on the HyperGrid. We keep the NAT procedure fixed and vary only the novelty criterion among count, RND, and geometric novelty, visualizing the resulting novelty-based intrinsic reward fields. Geometric novelty accounts for similarity to previously visited terminal states, assigning higher novelty to less-covered regions of the representation space. As a result, NAT is guided toward less-covered regions and discovers all four high-reward regions faster than the others.

To formalize this similarity-aware notion of coverage, we use geometry-aware Shannon entropy (Posada et al., 2020; Guo et al., 2021). Let $z : \mathcal{X} \to \mathcal{Z}$ map each terminal state to a representation space $\mathcal{Z}$ equipped with a distance function $d_{\mathcal{Z}}$, and let $p \in \Delta(\mathcal{X})$ denote a probability distribution over terminal states. We define a symmetric kernel $\kappa : \mathcal{Z} \times \mathcal{Z} \to [0, 1]$ that reflects distances in the representation space, assigning larger values to representations that are closer under $d_{\mathcal{Z}}$, with $\kappa(z(x), z(x)) = 1$. The corresponding geometry-aware Shannon entropy is

$$H_\kappa(p) := -\mathbb{E}_{X\sim p}\left[\log \mathbb{E}_{Y\sim p}\left[\kappa(z(X), z(Y))\right]\right].$$

Unlike categorical entropy, H_κ accounts for similarity between distinct terminal states through their representations. At training round t, let $\mathcal{D}_t = \{y_1, \ldots, y_M\}$ denote a set of previously visited terminal states, with empirical distribution $\hat{p}_t := \frac{1}{M}\sum_{i=1}^{M} \delta_{y_i}$. For a terminal state x, define

$$m_t(x) := \mathbb{E}_{Y\sim \hat{p}_t}\left[\kappa(z(x), z(Y))\right].$$

A small $m_t(x)$ indicates that x is weakly represented by previously explored terminal states in the representation space. Assuming $m_t(x) > 0$ for the terminal states being compared by NAT, we define Geometric Novelty as

$$N_t^{\mathrm{Geo}}(x) := \left.\frac{\delta H_\kappa(p)}{\delta p(x)}\right|_{p=\hat{p}_t} = -\log m_t(x) - \mathbb{E}_{Y\sim\hat{p}_t}\left[\frac{\kappa(z(Y), z(x))}{m_t(Y)}\right].$$

The first term, $-\log m_t(x)$, assigns higher novelty to terminal states with low similarity to previously explored states. The second term accounts for how x overlaps with the representation-space coverage provided by the previously explored states. Together, the two terms quantify how adding x changes the coverage of the representation space. The derivation and further interpretation are provided in Appendix A.1 and Appendix A.2.

4.3 Training with GEAT

GEAT integrates NAT and geometric novelty into the overall GFlowNet training procedure.

Trajectory sampling. At each training round t, the task GFlowNet G_θ and the exploration GFlowNet G_ϕ generate a task batch $\mathcal{T}_t^{\mathrm{task}}$ and an initial exploration batch $\mathcal{T}_t^{\mathrm{exp},0}$, respectively. Both models share the replay buffer $\mathcal{B}_t$ defined in Section 3.3. To assess novelty relative to previously acquired experience, we sample a reference set $\mathcal{D}_t$ of terminal states from $\mathcal{B}_t$ to compute geometric novelty.

Novelty-aware termination. We apply NAT as described in Section 4.1 to each trajectory in $\mathcal{T}_t^{\mathrm{exp},0}$, using geometric novelty N_t^{Geo} as the selection criterion. The selected trajectories form the final

exploration batch $\mathcal{T}_t^{\text{exp}}$. Only the selected exploration states are evaluated with the task reward R^{task}. The task batch and the final exploration batch are then added to the shared replay buffer.

GFlowNet training. We use the exploration reward defined in Section 3.2, with geometric novelty as the intrinsic reward. We clip negative novelty scores to zero to keep the exploration reward positive for GFlowNet training.

Following the prioritized replay framework in Section 3.3, we sample a replay batch $\mathcal{T}_t^{\text{rep}}$ from the shared buffer. The task GFlowNet is trained on the task batch, the final exploration batch, and the replay batch using R^{task}, allowing trajectories acquired through NAT to provide off-policy training data. The exploration GFlowNet is trained on the final exploration batch and the replay batch using R_t^{exp}. Before updating the exploration GFlowNet, replay trajectories are relabeled with the current exploration reward to reflect their novelty relative to the current reference set.

Practical details on reference-set sampling, kernel construction, and novelty computation are provided in Appendix B. Hyperparameter sensitivity and analyses isolating the contributions of NAT and geometric novelty are provided in Appendix D, with additional ablations in Appendix D.1.

5 Experiments

In this section, we describe our experimental procedures and report the corresponding results to address two key questions:

- **(i)** Does GEAT improve the coverage and speed of high-reward mode discovery?
- **(ii)** How consistent are these mode-discovery gains across tasks?

We compare GEAT with existing GFlowNet methods on synthetic and biological discovery tasks. To address question **(i)**, we examine cumulative mode-discovery curves and final counts to assess discovery speed and coverage. Cross-task comparisons address question **(ii)** by examining the consistency of these discovery gains. Detailed experimental settings are provided in Appendix C.

5.1 Results

5.1.1 HyperGrid

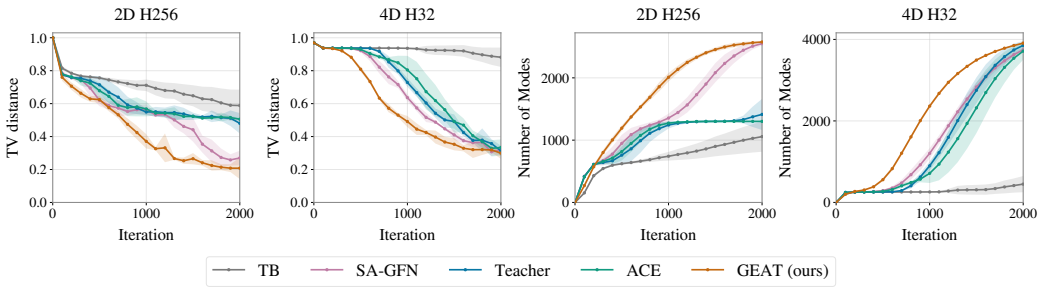

Figure 3: HyperGrid results. We report total variation (TV) distance to the reward-proportional target distribution and cumulative high-reward states discovered during training for the two grid configurations. Lower TV distance and higher cumulative discovery are better.

Figure 3 shows that GEAT improves both distributional learning and high-reward state discovery on HyperGrid. In the 2D setting, GEAT reduces TV distance substantially faster than the baselines and discovers a larger set of high-reward states, while Teacher and ACE plateau after repeatedly concentrating on a smaller portion of the high-reward space. In the 4D setting, GEAT again provides the fastest early reduction in TV distance and the earliest growth in cumulative high-reward discovery, while eventually reaching broad coverage comparable to or better than the strongest baselines. These results support the intended role of GEAT: preserving guidance toward high-reward regions while using geometry-aware novelty at acquisition time to reduce redundant exploration.

5.1.2 BIT SEQUENCE GENERATION

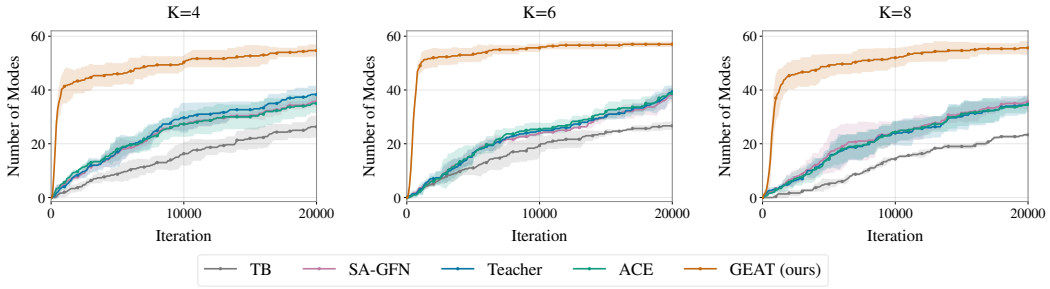

Figure 4: Cumulative mode discovery on Bit with word sizes $K = 4, 6, 8$. Curves show the mean across three random seeds, with shaded bands representing one standard deviation. Higher counts indicate broader mode coverage.

Table 1: Final Bit results for word sizes $K = 4, 6, 8$, where K denotes the number of bits appended at each construction step. Results are averaged over three seeds with sample standard deviations. Top-100 denotes the Top-100-Reward metric. Best mean values are shown in bold.

Word size →	$K = 4$			$K = 6$			$K = 8$		
Algorithm ↓	**Modes ↑**	**Top-100 ↑**	**FCS ↓**	**Modes ↑**	**Top-100 ↑**	**FCS ↓**	**Modes ↑**	**Top-100 ↑**	**FCS ↓**
TB	26.3 ±4.0	8.485 ±0.056	0.028 ±0.007	26.7 ±1.5	8.472 ±0.025	0.023 ±0.000	23.3 ±0.6	8.459 ±0.012	0.047 ±0.004
Teacher	38.3 ±3.2	8.470 ±0.021	**0.020** ±0.000	39.3 ±2.5	8.461 ±0.041	0.024 ±0.000	34.7 ±3.1	8.459 ±0.015	0.048 ±0.007
SA-GFN	35.7 ±2.1	8.460 ±0.041	0.026 ±0.005	38.3 ±2.5	8.450 ±0.032	0.023 ±0.001	35.7 ±2.5	**8.510** ±0.035	0.056 ±0.005
ACE	35.0 ±3.6	8.470 ±0.009	0.022 ±0.002	38.7 ±2.1	8.485 ±0.012	0.025 ±0.001	34.7 ±2.3	8.461 ±0.045	0.045 ±0.003
GEAT (ours)	**54.7** ±2.3	**8.521** ±0.016	0.021 ±0.001	**57.0** ±1.0	**8.503** ±0.054	**0.021** ±0.001	**55.7** ±2.5	8.475 ±0.039	**0.044** ±0.003

The Bit results in Table 1 show that GEAT achieves the highest mean mode coverage across all word sizes $K = 4, 6, 8$, where K denotes the number of bits appended at each construction step. GEAT discovers an average of 54.7, 57.0, and 55.7 modes, respectively. These correspond to improvements of approximately 43%–56% over the strongest baseline in each setting, demonstrating a consistent advantage in exploration. The cumulative discovery curves further show that GEAT rapidly discovers a broad set of modes early in training, establishing a substantial coverage advantage that persists throughout training. This broader and faster discovery is accompanied by competitive reward quality: GEAT achieves the highest mean Top-100-Reward at $K = 4$ and $K = 6$, while remaining close to the baselines at $K = 8$. GEAT also maintains competitive FCS values, achieving the lowest mean FCS at $K = 6$ and $K = 8$, with its mean FCS remaining within the baseline range at $K = 4$.

5.1.3 BIOLOGICAL DISCOVERY

Table 2: Final performance on QM9, sEH, and AMP, averaged over three seeds with sample standard deviations. Top-100 denotes the Top-100-Reward metric. Best mean values are shown in bold.

Bio task →	QM9			sEH			AMP		
Algorithm ↓	**Modes ↑**	**Top-100 ↑**	**FCS ↓**	**Modes ↑**	**Top-100 ↑**	**FCS ↓**	**Modes ↑**	**Top-100 ↑**	**FCS ↓**
TB	601.0 ±25.5	9.239 ±0.130	0.119 ±0.016	3875.3 ±227.2	6.560 ±0.028	0.669 ±0.088	207.3 ±11.6	0.761 ±0.001	0.097 ±0.003
Teacher	724.7 ±9.1	9.216 ±0.143	0.122 ±0.006	4764.3 ±113.9	6.569 ±0.033	0.662 ±0.072	247.0 ±18.5	0.769 ±0.003	0.099 ±0.002
SA-GFN	706.7 ±7.8	9.168 ±0.052	0.114 ±0.008	4563.7 ±190.9	6.538 ±0.063	0.620 ±0.009	300.3 ±16.0	0.767 ±0.001	0.097 ±0.002
ACE	706.3 ±23.9	9.156 ±0.018	0.126 ±0.011	4662.0 ±259.9	6.638 ±0.062	0.662 ±0.074	307.7 ±21.7	**0.770** ±0.001	0.096 ±0.002
GEAT (ours)	**731.3** ±13.6	**9.286** ±0.098	**0.106** ±0.001	**18442.0** ±807.9	**6.767** ±0.058	**0.589** ±0.069	**442.0** ±11.1	0.763 ±0.003	**0.096** ±0.001

GEAT's coverage advantage extends to molecular and peptide discovery, with the highest mean mode counts on QM9, sEH, and AMP (Table 2). On QM9, GEAT provides a modest coverage improvement over the strongest baseline while achieving the highest mean Top-100-Reward. The gains are particularly pronounced on sEH, where GEAT discovers approximately $3.9\times$ as many modes as

the strongest baseline and simultaneously achieves the highest mean Top-100-Reward and lowest mean FCS. The qualitative comparison further shows that GEAT extends exploration into regions of molecular space that remain largely unexplored by the baselines (Figure 6). On AMP, GEAT increases mode coverage by approximately 44% over the strongest baseline, with reward quality remaining within the baseline range. GEAT achieves the lowest mean FCS across all three tasks, although its value on AMP is closely matched by ACE. These results suggest that GEAT maintains competitive distributional fidelity while expanding exploration. The findings support the intended role of geometry-aware novelty at acquisition time: broadening the discovery of distinct high-reward modes across molecular and peptide spaces while maintaining competitive reward quality.

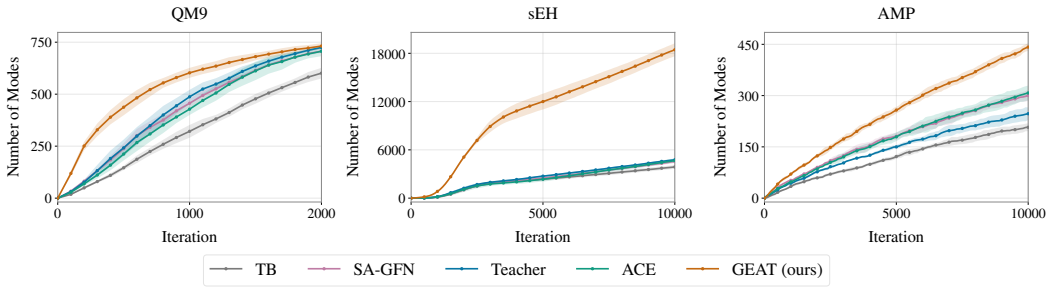

Figure 5: Number of modes on QM9, sEH, and AMP. We track the cumulative number of modes discovered during training according to each task's mode definition, combining discoveries from canonical and auxiliary samples. Results are averaged over three random seeds, and shading denotes one standard deviation. Discovering more modes corresponds to better coverage.

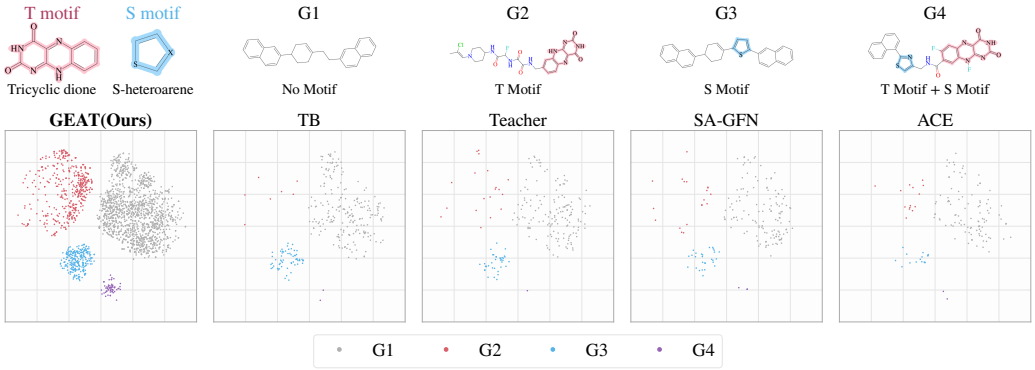

Figure 6: Structural coverage of high-reward sEH molecules. The top row shows the T and S motifs and example molecules for groups G1–G4. The bottom row compares the molecules discovered by each method in a shared motif-guided t-SNE embedding. GEAT achieves broader coverage of regions that remain sparsely explored by the baselines.

6 Conclusion

GEAT improves exploration in GFlowNets by using entropy-derived geometric novelty both as an intrinsic reward for the exploratory policy and as the terminal-selection criterion in NAT. By accounting for similarities between state representations, this score ranks terminal candidates by their first-order effect on the entropy of a replay-sampled reference distribution, guiding exploration toward underrepresented regions. As a result, GEAT outperforms competing algorithms in high-reward mode coverage across diverse benchmarks. One limitation of GEAT is that selecting an appropriate similarity kernel requires an understanding of the environment's geometric structure. Future work could explore adaptive or learned kernels to reduce reliance on prior geometric knowledge.

AI USE STATEMENT

Large Language Models (LLMs) were used to improve the clarity, grammar, and style of the manuscript. LLMs were not used for research ideation, algorithm design, theoretical analysis, or interpretation of experimental results. All AI-assisted work was reviewed by the authors. The authors take full responsibility for the final content of the paper.

REFERENCES

Marc G. Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Rémi Munos. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*, volume 29, pp. 1471–1479, 2016.

Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. Flow network based generative models for non-iterative diverse candidate generation. In *Advances in Neural Information Processing Systems*, volume 34, pp. 27381–27394, 2021.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. In *International Conference on Learning Representations*, 2019.

Tiago da Silva, Rodrigo Barreto Alves, Eliezer de Souza da Silva, Amauri Souza, Vikas Garg, Samuel Kaski, and Diego Mesquita. When do GFlowNets learn the right distribution? In *International Conference on Learning Representations*, 2025.

Pedro Dall'Antonia, Tiago da Silva, Daniel Csillag, Salem Lahlou, and Diego Mesquita. Avoid what you know: Divergent trajectory balance for GFlowNets. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*. PMLR, 2026.

Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Alaa Saade, Shantanu Thakoor, Bilal Piot, Bernardo Avila Pires, Michal Valko, Thomas Mesnard, Tor Lattimore, and Rémi Munos. Geometric entropic exploration. *arXiv preprint arXiv:2101.02055*, 2021.

Moksh Jain, Emmanuel Bengio, Alex Hernández-García, Jarrid Rector-Brooks, Bonaventure F. P. Dossou, Chanakya Ajit Ekbote, Jie Fu, Tianyu Zhang, Michael Kilgour, Dinghuai Zhang, Lena Simine, Payel Das, and Yoshua Bengio. Biological sequence design with GFlowNets. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 9786–9801. PMLR, 2022.

Moksh Jain, Tristan Deleu, Jason Hartford, Cheng-Hao Liu, Alex Hernández-García, and Yoshua Bengio. GFlowNets for AI-driven scientific discovery. *Digital Discovery*, 2(3):557–577, 2023.

Minsu Kim, Taeyoung Yun, Emmanuel Bengio, Dinghuai Zhang, Yoshua Bengio, Sungsoo Ahn, and Jinkyoo Park. Local search GFlowNets. In *International Conference on Learning Representations*, 2024.

Minsu Kim, Sanghyeok Choi, Taeyoung Yun, Emmanuel Bengio, Leo Feng, Jarrid Rector-Brooks, Sungsoo Ahn, Jinkyoo Park, Esmeralda S. Whitammer, and Yoshua Bengio. Adaptive teachers for amortized samplers. In *International Conference on Learning Representations*, 2025.

Minchan Kwon, Sunghyun Baek, Minseo Kim, Jaemyung Yu, Dongyoon Han, and Junmo Kim. Stable-GFlowNet: Toward diverse and robust LLM red-teaming via contrastive trajectory balance. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*. PMLR, 2026.

Tom Leinster and Christina A. Cobbold. Measuring diversity: The importance of species similarity. *Ecology*, 93(3):477–489, 2012.

Zhen Liu, Tim Z. Xiao, Weiyang Liu, Yoshua Bengio, and Dinghuai Zhang. Efficient diversity-preserving diffusion alignment via gradient-informed GFlowNets. In *International Conference on Learning Representations*, 2025.

Kanika Madan, Alex Lamb, Emmanuel Bengio, Glen Berseth, and Yoshua Bengio. Towards improving exploration through sibling augmented GFlowNets. In *International Conference on Learning Representations*, 2025.

Idriss Malek, Aya Laajil, Abhijith Sharma, Eric Moulines, and Salem Lahlou. Loss-guided auxiliary agents for overcoming mode collapse in GFlowNets. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(29):24326–24334, 2026.

Jose Gallego Posada, Ankit Vani, Max Schwarzer, and Simon Lacoste-Julien. Gait: A geometric approach to information theory. In *International Conference on Artificial Intelligence and Statistics*, pp. 2601–2611. PMLR, 2020.

Jarrid Rector-Brooks, Kanika Madan, Moksh Jain, Maksym Korablyov, Cheng-Hao Liu, Sarath Chandar, Nikolay Malkin, and Yoshua Bengio. Thompson sampling for improved exploration in GFlowNets. In *ICML 2023 Workshop on Structured Probabilistic Inference and Generative Modeling*, 2023.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *International Conference on Learning Representations*, 2016.

Max W. Shen, Emmanuel Bengio, Ehsan Hajiramezanali, Andreas Loukas, Kyunghyun Cho, and Tommaso Biancalani. Towards understanding and improving GFlowNet training. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 30956–30975. PMLR, 2023.

Nikhil Murali Vemgal, Elaine Lau, and Doina Precup. An empirical study of the effectiveness of using a replay buffer on mode discovery in GFlowNets. In *ICML 2023 Workshop on Structured Probabilistic Inference and Generative Modeling*, 2023.

Esmeralda S. Whitammer, Moksh Jain, Emmanuel Bengio, Chen Sun, and Yoshua Bengio. Trajectory balance: Improved credit assignment in GFlowNets. In *Advances in Neural Information Processing Systems*, volume 35, pp. 5955–5967, 2022.

Xuan Yu, Feng Niu, Rui Zhu, Yudong Zhang, Xu Wang, and Yang Wang. Improving LLM-based recommenders with conservative generative flow networks. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*. PMLR, 2026.

A PROOFS

A.1 GEOMETRIC NOVELTY DERIVATION

We derive the pointwise geometric novelty score used in Section 4.2. Since $\mathcal{X}$ is finite, write $p_x = p(x)$, $K_{xy} := \kappa(z(x), z(y))$, and

$$m_x = (Kp)(x) = \sum_{y \in \mathcal{X}} K_{xy} p_y.$$

The geometry-aware Shannon entropy is

$$H_\kappa(p) = -\sum_{x \in \mathcal{X}} p_x \log m_x.$$

For a candidate $u \in \mathcal{X}$, applying the product rule gives

$$\frac{\partial H_\kappa(p)}{\partial p_u} = -\log m_u - \sum_{x \in \mathcal{X}} p_x \frac{1}{m_x} \frac{\partial m_x}{\partial p_u}.$$

Since $\partial m_x / \partial p_u = K_{xu}$,

$$\frac{\partial H_\kappa(p)}{\partial p_u} = -\log m_u - \sum_{x \in \mathcal{X}} p_x \frac{K_{xu}}{m_x}.$$

Using the symmetry of κ and evaluating at $p = \hat{p}_t$ yields

$$\left.\frac{\delta H_\kappa(p)}{\delta p(u)}\right|_{p=\hat{p}_t} = -\log m_t(u) - \mathbb{E}_{X \sim \hat{p}_t}\left[\frac{\kappa(z(X), z(u))}{m_t(X)}\right] = N_t^{\mathrm{Geo}}(u).$$

We define *geometric novelty* from the derivative of geometry-aware Shannon entropy with respect to the probability mass assigned to a candidate state. Intuitively, $N_t^{\mathrm{Geo}}(x)$ captures the first-order change in geometry-aware entropy when additional probability mass is assigned to x relative to the current reference distribution.

A.2 DIRECTIONAL INTERPRETATION OF GEOMETRIC NOVELTY

The pointwise score can also be interpreted as the first-order effect of acquiring candidates from a new distribution. Let $q \in \Delta(\mathcal{X})$ denote a candidate distribution and define

$$p_\eta = (1 - \eta)\hat{p}_t + \eta q,$$

where $\eta \in [0, 1]$ controls the amount of new probability mass added to the current reference distribution. For example, if A_t previously acquired candidates are augmented with a batch of B_t new candidates, then $\eta = B_t/(A_t + B_t)$.

Applying the chain rule at $\eta = 0$ gives

$$\begin{aligned}\left.\frac{d}{d\eta} H_\kappa(p_\eta)\right|_{\eta=0} &= \sum_{x \in \mathcal{X}} (q(x) - \hat{p}_t(x)) \left.\frac{\delta H_\kappa(p)}{\delta p(x)}\right|_{p=\hat{p}_t} \\ &= \mathbb{E}_{X \sim q}[N_t^{\mathrm{Geo}}(X)] - \mathbb{E}_{X \sim \hat{p}_t}[N_t^{\mathrm{Geo}}(X)].\end{aligned}$$

Since the second term is independent of q, distributions with larger expected Geometric Novelty produce a larger first-order increase in geometry-aware entropy. Equivalently, whenever the relevant similarity masses remain positive,

$$H_\kappa(p_\eta) = H_\kappa(\hat{p}_t) + \eta \left(\mathbb{E}_{X \sim q}[N_t^{\mathrm{Geo}}(X)] - \mathbb{E}_{X \sim \hat{p}_t}[N_t^{\mathrm{Geo}}(X)]\right) + O(\eta^2).$$

This provides a distribution-level interpretation of the pointwise score used by NAT. The training algorithm itself does not optimize q jointly over a batch; candidate sets are evaluated independently using the same reference distribution.

B Practical Computation and Kernel Choices

In practice, geometric novelty does not need to be computed against every candidate stored in the replay buffer. At each training round, we uniformly sample M reference entries without replacement from the buffer and use their terminal candidates to form $\mathcal{D}_t$ and the empirical distribution $\hat{p}_t$. The same reference set is reused for all candidate comparisons within that round. With reference candidates $\mathcal{D}_t = \{y_1, \ldots, y_M\}$, the local similarity mass can be computed as

$$m_t(x) = \frac{1}{M} \sum_{i=1}^{M} \kappa(z(x), z(y_i)).$$

The representation and kernel are chosen according to the natural notion of similarity in each domain. Table 3 specifies the distance $d(x, y)$, kernel function, and bandwidth h used in each task. We use an exponential distance kernel for Grid, Bit, QM9, and AMP, and a Gaussian kernel for sEH. For sEH and AMP, bandwidths are calibrated using the median positive pairwise distance among 512 calibration embeddings.

Table 3: Domain-specific representations, distances, kernels, and bandwidths used in GEAT. Here, $d = d(x, y)$. Embedding bandwidths are rounded for display.

Domain	Representation $z(x)$	Distance $d(x, y)$	Kernel $\kappa(z(x), z(y))$	Bandwidth h
Grid	$x/(H-1)$	$\|z(x) - z(y)\|_2/\sqrt{D}$	$\exp(-d/h)$	0.1
Bit	Bit string	$d_{\text{Lev}}(x, y)/n$	$\exp(-d/h)$	0.1
QM9	Morgan fingerprint	$1 - \text{Tan}(z(x), z(y))$	$\exp(-d/h)$	0.3
sEH	Frozen MPNN embedding	$\|z(x) - z(y)\|_2$	$\exp(-d^2/(2h^2))$	6.07
AMP	Frozen ProtAlbert embedding	$\|z(x) - z(y)\|_2$	$\exp(-d/h)$	2.63

Because the replay buffer and the reference set are finite, previously acquired candidates may be absent from the current reference set and can therefore be selected again. GEAT is intended to reduce redundant acquisition by measuring coverage relative to retained reference experience rather than to prohibit reacquisition of every previously observed candidate.

C Experimental Details

In this section, we describe the experimental setup, including the benchmark tasks, reward functions, mode-discovery criteria, and baseline methods. We also provide the training configurations for each task and algorithm.

C.1 Experimental Setup

Hypergrid. We consider grids $\{0, \ldots, H-1\}^D$ with $(D, H) \in \{(2, 256), (4, 32)\}$. The agent starts from $\mathbf{0}$, and each non-STOP action increments one coordinate by one. A trajectory terminates when STOP is selected or a coordinate reaches $H-1$. The terminal reward is $R(x) = \epsilon + 2 \prod_{i=1}^{D} \mathbf{1}\{0.6 + (D-2)/40 < |2x_i/(H-1) - 1| < 0.8 + (D-2)/20\}$, where $\epsilon > 0$ is a small environment-specific background reward. In the 2D implementation, float64 rounding additionally includes the boundary coordinate value $x_i = 204$ in each dimension, yielding $51^2 = 2{,}601$ maximal-reward states. We define the set of maximal-reward states as $\mathcal{M}_{\text{Grid}} = \{x \in \mathcal{X} : R(x) = \max_{y \in \mathcal{X}} R(y)\}$. We measure high-reward state coverage by counting the distinct states in $\mathcal{M}_{\text{Grid}}$ generated during training, accumulated over iterations.

Bit Sequence. We study a variant of the bit sequence benchmark introduced by Whitammer et al. (2022), where the goal is to generate binary sequences $x \in \{0, 1\}^n$ with $n = 120$. Sequences are constructed from the empty sequence by appending a K-bit word at each step, where $K \in \{4, 6, 8\}$, until length n is reached. Let $\mathcal{M}_{\text{Bit}}$ denote a fixed set of 60 reference sequences. The terminal reward is $R(x) = \exp\{3(1 - \min_{m \in \mathcal{M}_{\text{Bit}}} d_{\text{Lev}}(x, m)/n)\}$, where d_{Lev} denotes the Levenshtein distance between binary sequences. This reward favors sequences close to the reference modes. A reference mode m is considered discovered when a generated sequence x satisfies $d_{\text{Lev}}(x, m) \leq 28$. We report the cumulative number of distinct reference modes discovered during training.

QM9. Molecules are represented as sequences of five building blocks drawn from a library $\mathcal{B}$ of 11 fragments, following the molecular construction framework of Kim et al. (2025). Construction begins with the empty sequence and proceeds by prepending or appending one block at a time, terminating after five blocks have been added. Let $\mathcal{X} = \mathcal{B}^5$ denote the terminal sequence space, and let $q(x)$ denote the precomputed proxy prediction of the HOMO–LUMO gap. The terminal reward is $R(x) = c \max\{q(x), 10^{-3}\}^5$ where c is chosen such that $\max_{x\in\mathcal{X}} R(x) = 100$. We define $\mathcal{M}_{\mathrm{QM9}}$ as the 805 terminal sequences with the highest rewards, corresponding to the top 0.5% of $\mathcal{X}$. High-reward state coverage is determined by exact sequence matching, with each sequence in $\mathcal{M}_{\mathrm{QM9}}$ counted once upon its first generation during training.

sEH. This task seeks molecules with high predicted binding scores for soluble epoxide hydrolase (Bengio et al., 2021). Molecular graphs are constructed using a library of 105 fragment attachment variants. Starting from an empty graph, the agent selects an initial fragment and grows the molecule by attaching additional fragments at available attachment sites. STOP becomes available after two blocks have been added. A trajectory terminates when STOP is selected, the molecule reaches eight blocks, or no attachment sites remain. Let $q(x)$ denote the binding score predicted by a pretrained proxy. The terminal reward is $R(x) = (\max\{q(x), 0.1\}/8)^{10}$. For mode discovery, generated molecules are processed in chronological order. A molecule is accepted as a new mode representative if $q(x) > 7$ and its Tanimoto similarity to every previously accepted representative is below 0.7. Similarities are computed using Morgan fingerprints with radius 2 and 2,048 bits. We record the cumulative number of accepted representatives throughout training.

AMP. We study a fixed-proxy variant of the antimicrobial peptide design benchmark of Jain et al. (2022). Peptides are represented as sequences $x \in \mathcal{A}^{50}$, where $\mathcal{A}$ is the alphabet of 20 standard amino acids. Construction begins with the empty sequence and proceeds by prepending or appending one amino acid at each step, terminating at length 50. Let $q(x)$ denote the predicted antimicrobial probability from a frozen pretrained classifier operating on ProtAlbert embeddings, averaged over 32 fixed dropout subnetworks. The terminal reward is $R(x) = \max\{q(x), 10^{-6}\}^3$. For mode discovery, we use a fixed score threshold τ calibrated to approximately the top 0.1% by reward in a reference set of 4,096 uniformly sampled length-50 peptides. A peptide is accepted as a new mode representative if $q(x) \geq \tau$ and $\|\phi(x) - \phi(m)\|_2 \geq 2$ for every previously accepted representative m, where $\phi(x) \in \mathbb{R}^{4096}$ is the unnormalized, mean-pooled ProtAlbert embedding used by the classifier. We report the cumulative number of accepted representatives discovered during training.

C.2 Evaluations

Baselines. We compare GEAT with four GFlowNet baselines: (i) Trajectory Balance (TB) (Whitammer et al., 2022), which learns a reward-proportional sampling policy by optimizing a balance objective over complete trajectories; (ii) Adaptive Complementary Exploration (ACE) (Dall'Antonia et al., 2026), which trains an exploration GFlowNet using divergent trajectory balance to target high-reward regions underrepresented by the current sampling policy; (iii) Adaptive Teacher (Teacher) (Kim et al., 2025), which trains a teacher GFlowNet to generate training trajectories from regions with large student trajectory balance errors; and (iv) Sibling Augmented GFlowNets (SA-GFN) (Madan et al., 2025), which uses a separate sibling network guided by both task rewards and intrinsic rewards from random network distillation to encourage exploration. For these baselines, we follow the experimental settings described in the respective papers, with adaptations to our benchmark tasks and shared training protocol. All methods use the same loss-based prioritized experience replay protocol.

Metrics. Mode discovery is evaluated using the task-specific criteria described above. As supplementary metrics, we report exact Total Variation (TV) distance on HyperGrid and Flow Consistency in Subgraphs (FCS) (da Silva et al., 2025) on Bit Sequence, QM9, sEH, and AMP. TV measures the discrepancy between the learned terminal distribution and the reward-proportional target distribution over all reachable terminal states. For FCS, we draw 2,048 valid terminal samples from the learned task policy, separately for each method and training seed. We divide these samples into 32 groups of 64, remove duplicate states within each group, and average the TV distances between the learned terminal probabilities and training rewards, each normalized within the group. For the non-grid tasks, we also report Top-100-Reward: the mean of the highest 100 evaluation rewards among unique candidates obtained from 1,000 samples generated by each trained model.

Table 4: Policy architectures. Depth denotes hidden layers for MLPs, Transformer layers for Bit Sequence, and message-passing steps for sEH. For QM9, depth is listed as encoder + scorer.

Domain	Network	Depth	Hidden dimension	Backward policy
Hypergrid	MLP	2	256	Fixed uniform
Bit Sequence	Transformer	3	64	Deterministic
QM9	MLP	$1+1$	1,024	Learned
sEH	GNN	10	256	Learned
AMP	MLP	2	2,048	Learned

Table 5: Training settings shared across all methods within each task. The listed learning rates apply to both task and exploration GFlowNets.

Setting	Hypergrid	Bit Sequence	QM9	sEH	AMP
Policy learning rate η_θ	10^{-3}	10^{-4}	10^{-4}	10^{-4}	10^{-4}
Log-partition learning rate η_Z	10^{-1}	5×10^{-2}	10^{-2}	10^{-2}	10^{-2}
Training iterations	2,000	20,000	2,000	10,000	10,000
Total fresh batch size	32	64	64	64	64
Random-action probability ϵ	0	5×10^{-4}	0.1	0.05	0.05
Gradient norm clipping	1	None	10	10	10

C.3 Implementation Details

Within each task, all methods use the policy architecture summarized in Table 4. The Task GFN is trained with the trajectory balance objective using the benchmark reward $R(x)$. Exploration GFNs use the same architecture with separate parameters and their own learnable log-normalizing constants.

The MLP policies use one-hot inputs and LeakyReLU activations. For QM9, the encoder and transition scorer each have one hidden layer, and the state embedding dimension is 1,024. The Bit Sequence Transformer uses eight attention heads and a feedforward dimension of 64.

We use Adam with learning rates η_θ for policy parameters and η_Z for log-normalizing constants, as reported in Table 5. Within each task, these learning rates are shared across algorithms and apply to both Task and Exploration GFNs. All methods use the same total fresh batch size and number of training iterations. TB samples the entire batch from its Task GFN, whereas Teacher, SA-GFN, ACE, and GEAT split the batch equally between their Task and Exploration GFNs. All methods use loss-based prioritized replay with a batch size of 16, equal weighting of fresh-data and replay losses, and priority refreshes every 100 iterations. The replay buffer stores terminal states and their task rewards, with the corresponding trajectories retained when trajectory reuse is enabled. For each entry selected for replay, we either reuse its stored trajectory or generate a new complete trajectory by backward sampling from its terminal state using the current task backward policy, depending on the replay configuration. Following Kim et al. (2025), we use prioritized experience replay with priority $p(x) = R(x)^\alpha \log\big(1 + \epsilon + (1 + C\mathbb{I}_{\delta(\tau)>0})\delta(\tau)^2\big)$, where $\delta(\tau)$ is the TB residual computed by adding a small positive constant to the reward before taking the logarithm. We set $C = 19$, and use $\alpha = 0.5$ for HyperGrid and $\alpha = 0.35$ for the remaining tasks.

For GEAT, the geometry coefficient β controls the contribution of the geometry-based reward component and is set to 10^{-2} across all tasks. The backtracking depth k specifies the maximum number of backward construction steps taken before resampling a suffix. We use $k = 64$ for both Hypergrid configurations; 4, 3, and 2 for Bit Sequence with $K = 4, 6, 8$, respectively; 2 for QM9 and sEH; and 6 for AMP. In Bit Sequence, each backward step removes one K-bit word. The suffix-proposal count n specifies the number of new suffixes sampled from the backtracked state, excluding the original terminal state from the proposal count. We use $n = 16$ for Hypergrid, 8 for Bit Sequence, QM9, and sEH, and 4 for AMP.

D Ablation Study and Sensitivity Analysis

D.1 Ablation Study

We analyze GEAT on the two-dimensional HyperGrid with $H = 256$ and 2,601 modes. Unless otherwise specified, we use $k = 32$ and $n = 8$ and train each configuration for 2,000 iterations. Figure 7 reports the mean and sample standard deviation across three random seeds. We evaluate both the exact total variation (TV) distance between the task GFlowNet and the target distribution and the cumulative number of distinct target modes discovered during training. Mode discovery counts fresh task-policy samples and retained explorer samples. All configurations share the same two-policy training and replay protocol. Baseline (TB) disables both NAT and the geometric novelty learning bonus within this common training framework.

Effect of the terminal selection rule. We first compare three terminal selection rules while keeping candidate generation and the geometric novelty learning bonus fixed: selecting the candidate with the maximum novelty score (Max), sampling according to a softmax over novelty scores with temperature $T = 0.3$, and sampling uniformly from the candidate set. As shown in Figure 7, GEAT with Max selection reaches an average of 2,298 modes, compared with 1,515 for Softmax and 1,301 for Uniform. Its final TV distance is 0.292, compared with 0.435 for Softmax and 0.525 for Uniform. These results support selecting the highest-novelty candidate under the evaluated configuration, yielding broader mode coverage and a closer approximation to the target distribution than the two tested stochastic selection rules.

Contributions of NAT and geometric novelty. We next examine the contributions of NAT and the geometric novelty learning bonus in the middle column of Figure 7. The NAT-only variant retains novelty-based candidate selection but removes the geometric novelty bonus from the explorer's learning reward. The Geometric-only variant retains this learning bonus but disables NAT. Starting from Baseline (TB), which achieves a final TV distance of 0.557 and discovers 1,171 modes on average, NAT only improves these values to 0.502 and 1,301, while Geometric only achieves 0.477 and 1,603. Combining both components in full GEAT further improves the final TV distance to 0.292 and increases mode discovery to 2,298. The stronger performance of the combined method supports complementary contributions from novelty-based terminal selection and geometric novelty in the explorer's learning reward.

Comparison with LSGFN. We compare GEAT with the reward-based and Metropolis–Hastings (MH) variants of Local Search GFlowNets (LSGFN; Kim et al., 2024), using $k = 32$ and $n = 8$ in each local-search implementation. The LSGFN variants do not use the geometric novelty learning bonus, whereas full GEAT does; the comparison fixes the number of training iterations. As shown in the right column of Figure 7, both LSGFN variants discover modes rapidly during early training, but their subsequent gains in coverage are limited. GEAT continues to discover additional modes later in training, reaching an average of 2,298 modes, compared with 1,302 for LSGFN (Reward) and 1,397 for LSGFN (MH). This broader coverage is accompanied by a lower final TV distance: GEAT achieves 0.292, compared with 0.503 and 0.488 for the two LSGFN variants, respectively. Under this training protocol, full GEAT exhibits more sustained mode discovery and a lower final TV distance than the two LSGFN variants.

D.2 Sensitivity Analysis

We study the sensitivity of full GEAT to the backtracking depth k and the number of local variants n, using the same environment, training duration, and evaluation protocol as in Section D.1. We vary one parameter at a time while fixing the other at its default value. Figure 8 reports the results across three random seeds.

Effect of backtracking depth. We vary $k \in \{16, 32, 64, 128\}$ while fixing $n = 8$. Increasing k from 16 to 64 reduces the final TV distance from 0.323 to 0.201 and increases the average number of discovered modes from 2,066 to 2,578, corresponding to approximately 99.1% of all target modes. Increasing k further to 128 yields a TV distance of 0.196 and 2,572 discovered modes on average. These results show diminishing gains beyond $k = 64$ in this environment.

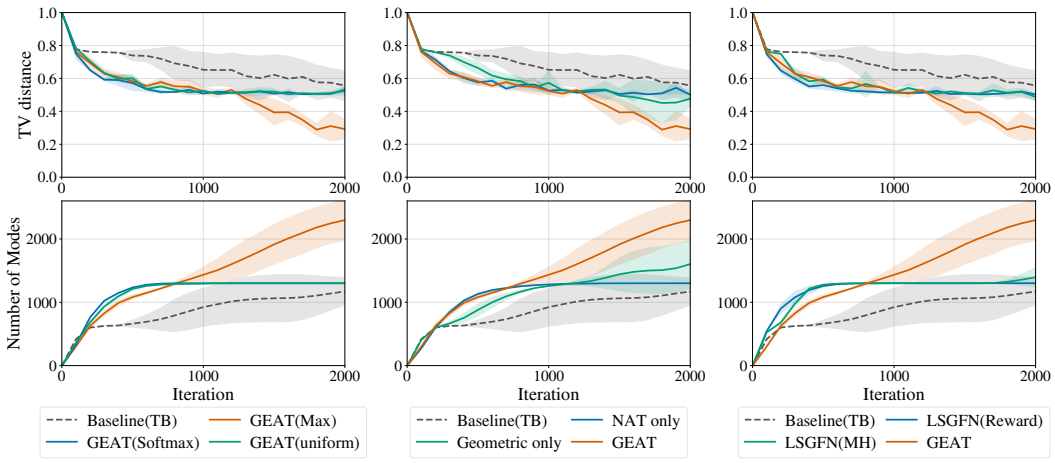

Figure 7: Ablation study and comparison with LSGFN on the two-dimensional HyperGrid with $H = 256$ and 2,601 modes. Left: Max, Softmax ($T = 1$), and Uniform terminal selection within GEAT. Middle: NAT only, Geometric only, and full GEAT. Right: LSGFN (Reward), LSGFN (MH), and GEAT. Baseline (TB) is included in every column. Local-search configurations use $k = 32$ and $n = 8$. The top row reports the exact TV distance of the task GFlowNet from the target distribution, and the bottom row reports the cumulative number of distinct target modes discovered during training. Lines and shaded regions denote the mean and $\pm$ one sample standard deviation across three random seeds.

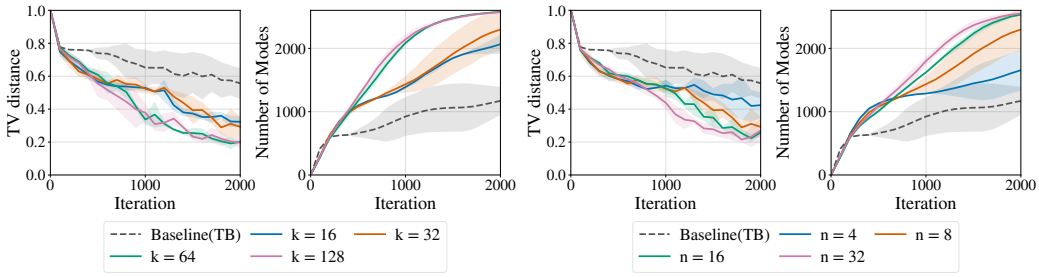

Figure 8: Sensitivity analysis of GEAT on the two-dimensional HyperGrid with $H = 256$ and 2,601 modes. The left two panels vary the backtracking depth $k \in \{16, 32, 64, 128\}$ with $n = 8$ and report exact TV distance followed by the cumulative number of distinct target modes discovered during training. The right two panels vary the number of additional local variants $n \in \{4, 8, 16, 32\}$ with $k = 32$ and report the same two metrics in the same order. Baseline (TB) is included in every panel. All configurations are trained for 2,000 iterations. Lines and shaded regions denote the mean and $\pm$ one sample standard deviation across three random seeds.

Effect of the number of local variants. We vary $n \in \{4, 8, 16, 32\}$ while fixing $k = 32$, where n denotes the number of local variants generated by GEAT in addition to the original terminal candidate. Increasing n from 4 to 16 reduces the final TV distance from 0.424 to 0.258 and increases the average number of discovered modes from 1,655 to 2,534. With $n = 32$, mode discovery increases slightly further to 2,551 modes, while the mean final TV distance is 0.271 with greater variation across seeds. Thus, generating more local variants improves mode discovery over the tested range, although larger candidate sets do not necessarily translate into further improvements in the learned terminal distribution.