KL DIVERGENCE OPTIMIZATION WITH ENTROPY-RATIO ESTIMATION FOR STOCHASTIC GFLOWNETS

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper introduces a novel approach for optimizing Generative Flow Networks (GFlowNets) in stochastic environments by incorporating KL divergence objectives with entropy-ratio estimation. We leverage the relationship between high and low entropy states, as defined in entropy-regularized Markov Decision Processes (MDPs), to dynamically adjust exploration and exploitation. Detailed proofs and analysis demonstrate the efficacy of this methodology in enhancing mode discovery, state coverage, and policy robustness in complex environments.

1 INTRODUCTION

Generative Flow Networks (GFlowNets) (Bengio et al., 2021a;b) have recently gained attention for their application in a variety of tasks, such as molecule discovery (Bengio et al., 2021a; Jain et al., 2022b), biological sequence design (Jain et al., 2022a), and robust scheduling (Zhang et al., 2023). GFlowNets learn policies that generate objects $x \in \mathcal{X}$ sequentially, where the generation process is similar to Monte-Carlo Markov chain (MCMC) methods (Metropolis et al., 1953; Hastings, 1970; Andrieu et al., 2003), generative models (Goodfellow et al., 2014; Ho et al., 2020), and amortized variational inference (Kingma & Welling, 2013). This sequential process of generating objects through a policy also closely resembles reinforcement learning (RL) (Sutton & Barto, 2018).

2 BACKGROUND

Generative Flow Networks (GFlowNets) are variational inference algorithms designed to treat sampling from a target probability distribution as a sequential decision-making process (Bengio et al. (2021a;b)). Below, we briefly summarize the formulation and primary training algorithms for GFlowNets. Consider a fully observed, deterministic Markov Decision Process (MDP) with a state space $\mathcal{S}$ and a set of actions $\mathcal{A} \subseteq \mathcal{S} \times \mathcal{S}$. The MDP has a designated *initial state* $\mathbf{s}_0$, and certain states, called *terminal states*, are designated as having no outgoing actions. Let $\mathcal{X}$ denote the set of terminal states. We assume that all states in $\mathcal{S}$ are reachable from $\mathbf{s}_0$ through a sequence of actions, though not necessarily by a unique sequence. A *complete trajectory* is a sequence of states $\tau = (\mathbf{s}_0 \to \mathbf{s}_1 \to \cdots \to \mathbf{s}_n)$, where $\mathbf{s}_n \in \mathcal{X}$, and each pair of consecutive states is connected by an action, i.e., $\forall i\, (\mathbf{s}_i, \mathbf{s}_{i+1}) \in \mathcal{A}$. A *policy* in this MDP defines a distribution $P_F(\mathbf{s}'|\mathbf{s})$ for each non-terminal state $\mathbf{s} \in \mathcal{S} \setminus \mathcal{X}$, specifying the probability of transitioning to the next state $\mathbf{s}'$ in a single action. The policy induces a distribution over complete trajectories as follows: $P_F(\mathbf{s}_0 \to \mathbf{s}_1 \to \cdots \to \mathbf{s}_n) = \prod_{i=0}^{n-1} P_F(\mathbf{s}_{i+1} \mid \mathbf{s}_i)$. The marginal distribution over terminal states, denoted $P_F^\top$, is the distribution on $\mathcal{X}$ induced by the policy over all complete trajectories. It may be computationally intractable to compute $P_F^\top$ directly, as $P_F^\top(\mathbf{x}) = \sum_{\tau \to \mathbf{x}} P_F(\tau)$, where the sum is taken over all complete trajectories that terminate at state $\mathbf{x}$.

3 RELATED WORK

Unlike Reinforcement Learning (RL), where the goal is typically to maximize the expected reward by learning a deterministic policy (Mnih et al., 2015; Lillicrap et al., 2015; Haarnoja et al., 2017; Fujimoto et al., 2018; Haarnoja et al., 2018), GFlowNets aim to learn a stochastic policy for generating composite objects x with probability proportional to the reward function $R(x)$. This is particularly

useful in real-world tasks where diversity in solutions is crucial, such as recommender systems (Kunaver & Požrl, 2017), drug discovery (Bengio et al., 2021a; Jain et al., 2022a), and sampling causal models from a Bayesian posterior (Deleu et al., 2022). However, existing GFlowNet approaches (Bengio et al., 2021a; Malkin et al., 2022; Madan et al., 2022) have primarily been developed for deterministic environments, where state transitions are fixed. In real-world applications, stochasticity in state transitions is common, presenting significant challenges for GFlowNets. Deterministic GFlowNet methods can fail to model the correct state visitation distribution under stochastic transitions. For instance, in the presence of stochastic dynamics, standard GFlowNets may learn incorrect probabilities for visiting states, which do not align with the ideal distribution. This mismatch occurs because existing methods do not properly account for randomness in state transitions. To address these limitations, we propose a novel approach called **KL Divergence Optimization with Entropy-Ratio Estimation for Stochastic GFlowNets**. Our method introduces a KL divergence objective that optimizes the policy distribution while incorporating an entropy-ratio estimation mechanism that dynamically balances exploration and exploitation. By adjusting the exploration-exploitation trade-off through entropy-ratio estimation, our method enables GFlowNets to capture the correct state visitation distribution, even in stochastic environments.

Our approach is general and can be applied to different GFlowNet learning objectives. It works by minimizing the divergence between forward and backward policies, ensuring flow consistency across stochastic transitions. The entropy-ratio estimation further enhances robustness by favoring high-entropy states in situations where the environment exhibits higher stochasticity. This approach allows for better mode discovery and improves state visitation coverage in stochastic tasks, such as molecule discovery, biological sequence generation, and other structured object generation tasks.

Our contributions of this paper are as follows:

- We propose **KL Divergence Optimization with Entropy-Ratio Estimation for Stochastic GFlowNets**, a novel approach that addresses the limitations of existing GFlowNet methods in stochastic environments.
- We provide a detailed analysis of how our method optimizes the flow consistency and dynamically adjusts exploration in stochastic transitions, making it suitable for a wide range of stochastic tasks.
- We conduct extensive experiments on benchmark tasks, demonstrating that our method significantly outperforms existing baselines, including Stochastic GFlowNets (SGFN), PPO, SAC and MCMC particularly in complex environments like biological sequence generation.

4 Detailed Balance in Stochastic GFlowNets

Detailed balance (DB) is a fundamental principle in GFlowNets, ensuring the alignment between forward and backward policies to maintain the desired state distribution. In stochastic GFlowNets, DB must accommodate the randomness inherent in state transitions, which is crucial for accurately representing the distribution over states under varying conditions.

4.1 Stochastic Environments

Stochastic GFlowNets (Pan et al. (2023)) extend the GFlowNet framework to environments where state transitions are stochastic. These models introduce a decomposition of state transitions into two steps: (1) a deterministic agent action and (2) a stochastic environment transition. This decomposition helps in managing stochastic dynamics but increases the complexity of learning due to the introduction of high variance in training, particularly when combined with trajectory balance objectives. Flow consistency is defined in the forward policy:

$$F(s_t)\pi(a_t|s_t) = \sum_{s_{t+1}} F(s_{t+1})\pi_B((s_t, a_t)|s_{t+1}). \tag{1}$$

This equation highlights the balance of flow at each state by equating the inflow (left-hand side) and outflow (right-hand side). It ensures that the total probability mass flowing out of state s_t

via policy π matches the backward flow from subsequent states s_{t+1}. The consistency is vital for GFlowNets as it stabilizes policy training, ensuring each decision balances the resulting flows in a manner proportional to the overall reward. The stochastic state transitions are then applied to the Detailed-Balance(DB) condition as follow

$$F(s_t)\pi(a_t|s_t)P(s_{t+1}|(s_t,a_t)) = F(s_{t+1})\pi_B((s_t,a_t)|s_{t+1}). \tag{2}$$

This equation explicitly introduces the transition probability $P(s_{t+1}|(s_t,a_t))$, capturing the stochastic nature of moving from state-action pairs (s_t,a_t) to the next state s_{t+1}. The need for this formulation arises because stochastic transitions introduce variability that must be accounted for in both forward and backward policies to ensure a robust and proportional sampling distribution. This representation ensures that the detailed balance condition holds, preserving proportional flows between forward and backward states, critical for maintaining the integrity of GFlowNets in stochastic environments.

4.2 From Detailed Balance to KL Divergence with Entropy-Ratio Estimation

To transform the detailed balance equation into a practical training objective, we express it as a KL divergence minimization problem by incorporating entropy ratio density estimation:

Given:

$$P(s_{t+1}|(s_t,a_t)) = \frac{H_{high}(s_{t+1})}{\gamma H_{high}(s_{t+1}) + (1-\gamma)H_{low}(s_{t+1})}, \tag{3}$$

where $H_{high}(s_{t+1})$ and $H_{low}(s_{t+1})$ represent the densities related to high and low entropy states, respectively and $0 < \gamma < 1$. We can rewrite the detailed balance in terms of this density ratio. GFlowNet detailed balance is an off-policy algorithm that leverages training data from a variety of distributions. Specifically, we can reframe the detailed balance objective from (Eq. 2) given the environemnt dynamic (Eq. 3) into a KL divergence formulation

$$\min_{\theta} D_{KL}\left(\pi_B((s_t,a_t)|s_{t+1}) \middle\| \frac{F((s_t,a_t))\frac{H_{high}(s_{t+1})}{\gamma H_{high}(s_{t+1})+(1-\gamma)H_{low}(s_{t+1})}}{F(s_{t+1})}\right). \tag{4}$$

The KL divergence can also be expressed as a summation over state-action pairs for policy π_B:

$$D_{KL} = \sum_{s,a,s'} \pi_B((s,a)|s')\left(\log \pi_B((s,a)|s') - \log F(s,a) - \log \frac{H_{high}(s')}{\gamma H_{high}(s') + (1-\gamma)H_{low}(s')} + \log F(s')\right). \tag{5}$$

This summation highlights the direct contribution of state-action pairs, incorporating entropy ratio estimations in the policy optimization process.

5 Dynamics Loss Using Cross Entropy with Entropy-Ratio Estimation

5.1 Dynamics Loss

The dynamics loss is a crucial component that aligns the model's predictions with the empirical state transitions observed in stochastic environments. By integrating entropy-ratio estimations, this loss function effectively adjusts the weight given to transitions based on their uncertainty, captured through entropy measures. High-entropy transitions correspond to exploratory actions that increase state visitation diversity, while low-entropy transitions focus on consolidating high-reward paths, aiding exploitation.

Deriving the Dynamics Loss (Mohammadpour et al. (2024)) defines the flow entropy which is strictly concave function as

$$H(\pi) = E\left[\sum_{t=0}^{T-1} H(\pi(\cdot|s_t))\right] = \sum_{s\in S} \mu_\pi(s) H(\pi(\cdot|s)), \tag{6}$$

where $H(\pi(\cdot|s)) = -\sum_{a\in A(s)} \pi(a|s)\log\pi(a|s)$ measures the randomness of actions at state s. The term $\mu_\pi(s)$ represents the state visitation frequency under the policy π, denoting how often the state s is encountered when following π. The flow entropy is calculated as a weighted sum of the entropy of the policy at each state, where $\mu_\pi(s)$ acts as the weighting factor. This approach ensures that states visited more frequently have a greater influence on the overall entropy, which is essential for analyzing the exploration behavior of the policy over time. We express the density ratio entropy for a given γ as:

$$r_\gamma(s) = \frac{H_{high}(s)}{\gamma H_{high}(s) + (1-\gamma)H_{low}(s)}, \tag{7}$$

which adjusts the probability of each state-action pair based on the weighted contributions of high and low entropy states. The dynamics loss, incorporating this entropy ratio, is derived as:

$$\mathcal{L}_{Dynamics} = -\sum_{s\in S, a\in A(s)} \mu_\pi(s) H(\pi(\cdot|s)) \left(\log r_\gamma(s) + (1-\gamma)(1-H(\pi(\cdot|s)))\log(1-r_\gamma(s))\right). \tag{8}$$

This loss penalizes deviations from expected entropy-weighted transitions, pushing the policy to optimize flows that balance exploration with exploitation.

γ in Exploration vs. Exploitation Trade-off: The parameter γ plays a pivotal role in managing the trade-off between exploration and exploitation by modulating the influence of high and low entropy states in the transition dynamics. High values of γ emphasize high-entropy transitions, favoring exploration by allowing the policy to sample diverse actions and visit more states. This promotes the discovery of new modes, avoiding local optima by spreading the probability mass across a wider set of states. Conversely, lower values of γ increase the influence of low-entropy transitions, focusing on exploitation by reinforcing actions that lead to predictable and high-reward states. This controlled trade-off ensures that the policy can balance between exploring new opportunities and exploiting known profitable actions, directly impacting state visitation patterns and the robustness of the learned policy. We provide the algorithm of our proposed method in algorithm 1.

Algorithm 1 KL Divergence Optimization with Entropy-Ratio Estimation for Stochastic GFlowNets

1: Initialize policy parameters θ, environment dynamics, and γ (exploration-exploitation trade-off).
2: **for** each episode **do**
3: Initialize state s_0.
4: **while** not in terminal state s_T **do**
5: Sample action $a_t \sim \pi_\theta(a|s_t)$.
6: Transition to next state $s_{t+1} \sim P(s_{t+1}|s_t, a_t)$.
7: Compute reward $R(s_{t+1})$.
8: Compute entropy ratio for state s_{t+1}:

$$r_\gamma(s_{t+1}) = \frac{H_{\text{high}}(s_{t+1})}{\gamma H_{\text{high}}(s_{t+1}) + (1-\gamma)H_{\text{low}}(s_{t+1})}$$

9: **end while**
10: Minimize the KL divergence:

$$D_{KL} = \sum_{s,a,s'} \pi_\theta(s,a|s')\left[\log\pi_\theta(s,a|s') - \log F(s,a) - \log r_\gamma(s')\right]$$

11: Update dynamics loss using entropy ratio estimation.
12: Adjust γ to balance exploration and exploitation.
13: **end for**

Practical Approximation of Dynamics Loss To make this loss computationally tractable, we approximate it by discretizing the state-action pairs:

$$\mathcal{L}(\theta) = -\frac{1}{N}\sum_{n=1}^{N}\left(z_n H(\pi_\theta(a_n|s_n))\log r_\gamma(s_n) + (1-z_n)H(\pi_\theta(a_n|s_n))\log(1-r_\gamma(s_n))\right), \quad (9)$$

where z_n denotes a binary classification that captures the occurrence of state-action pairs. This practical form illustrates how high entropy (exploration) encourages broader state sampling, while low entropy (exploitation) consolidates high-value trajectories, effectively guiding the policy.

Effect of Entropy on Loss Dynamics High entropy in state-action pairs incentivizes the policy to discover new modes by exploring diverse states, preventing overfitting to high-reward areas and maintaining broad state coverage. Low entropy, conversely, focuses on refining the policy toward known high-reward paths, ensuring stability in high-reward regions. The balance between these influences is controlled by γ, enabling adaptive policy adjustments that enhance overall robustness.

6 Optimum Analysis with Density-Ratio Entropy

Detailed Explanation of Optimal Dynamics Loss The optimal dynamics loss balances the entropy-driven exploration and exploitation by incorporating density-ratio terms:

$$\mathcal{L}^*(\pi) = -\sum_{s\in S, a\in A(s)} \mu_\pi(s) H(\pi(\cdot|s))\left(\gamma\log\pi(s,a) + (1-\gamma)\log(1-\pi(s,a))\right). \quad (10)$$

The loss gradient is given by:

$$\frac{\partial\mathcal{L}^*}{\partial\pi_\theta} = -\sum_{s,a}\left(\frac{\gamma H_{high}(s)}{\pi_\theta(s,a)} - \frac{(1-\gamma)H_{low}(s)}{1-\pi_\theta(s,a)}\right). \quad (11)$$

Solving this leads to:

$$\pi^*(s,a) = \frac{\gamma H_{high}(s)}{\gamma H_{high}(s) + (1-\gamma)H_{low}(s)}. \quad (12)$$

This result integrates entropy-driven state-action dynamics, demonstrating how policy tuning with γ guides the exploration-exploitation trade-off, directly impacting the policy's adaptive behavior.

7 Impact of γ and α: Mathematical Analysis with Entropy

7.1 Sensitivity of γ

The parameter γ serves as a critical modulator in balancing exploration and exploitation. High values of γ amplify high-entropy states, promoting mode discovery by encouraging the policy to explore less frequently visited areas. This expands the state visitation landscape, allowing the GFlowNet to sample a broader set of states, which is crucial for uncovering new, potentially optimal paths.

In addition, the sensitivity of the loss to γ is given by:

$$\frac{\partial\mathcal{L}_{KL}}{\partial\gamma} = -\sum_{s,a}\mu_\pi(s)H(\pi(a|s))\left[\frac{\partial}{\partial\gamma}\log\left(\gamma H_{high}(\pi(\cdot|s)) + (1-\gamma)H_{low}(\pi(\cdot|s))\right)\right].$$

This formulation explicitly connects γ to the entropy-weighted adjustments, demonstrating how increasing γ enhances exploration, while lowering γ concentrates on exploitation.

Influence of α on State-Action Transitions The parameter α controls the variability in state-action transitions by modulating the transition probabilities through entropy adjustments:

$$\frac{\partial \mathcal{L}_{KL}}{\partial \alpha} = -\sum_{s,a} \mu_\pi(s) H(\pi(a|s)) \left[\frac{\partial \log P(s_{t+1}|(s_t, a_t))}{\partial \alpha} \right].$$

This influence helps manage the exploration stability by adjusting how stochastic or deterministic the transitions are, directly impacting the policy's ability to explore while maintaining consistent high-reward pathways.

7.2 Optimal Policy with γ and α

Optimal policy performance emerges when γ and α are tuned to balance the exploration-exploitation trade-off effectively. High γ fosters the discovery of novel modes by weighting high-entropy paths, while a carefully adjusted α ensures transitions remain robust yet adaptive, leading to stable policy convergence.

8 Experiments

In this section, we conduct extensive experiments to investigate the following key questions: i) How much can KL Divergence Optimization with Entropy-Ratio Estimation improve the performance of Stochastic GFlowNets over standard GFlowNets in the presence of stochastic transition dynamics? ii) Can our method scale to more complex and challenging tasks, such as generating biological sequences and what is the effect of stochasticity level on its performance?

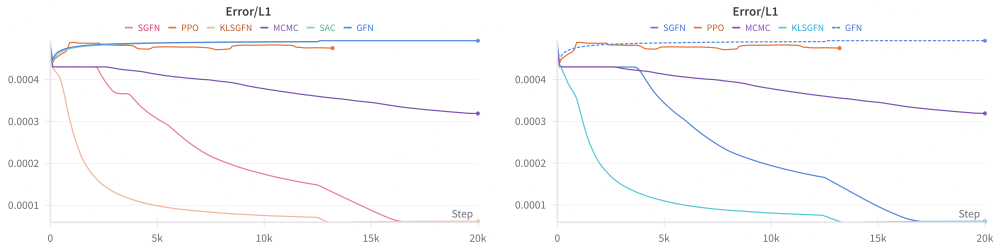
8.1 GridWorld

8.1.1 Experimental Setup

We begin by conducting a series of experiments in the GridWorld task, originally introduced in Bengio et al. (2021a), to evaluate the effectiveness of GFlowNets optimization scheme. An illustration of the task, with a grid size of $H \times H$, is shown in Figure 1. At each time step, the agent selects an action to navigate the grid. The available actions include increasing a coordinate, and a stop operation, which terminates the episode and ensures the underlying Markov decision process (MDP) forms a directed acyclic graph (DAG). The agent receives a reward $R(x)$, as defined in Bengio et al. (2021a), when a trajectory reaches a terminal state x. The reward function $R(x)$ has four distinct modes, located in the corners of the map (Figure 1). The agent's objective is to model the target reward distribution and capture all reward modes. The shade of color reflects the magnitude of the rewards, with darker colors indicating higher rewards. We introduce stochasticity into the environment by adopting the transition dynamics from Machado et al. (2017) and Yang et al. (2022). Specifically, with probability $1 - \alpha$, the environment follows the selected action, but with probability α, a uniformly chosen random action is executed (leading to slips or missteps to neighboring regions, as shown in Figure 1).

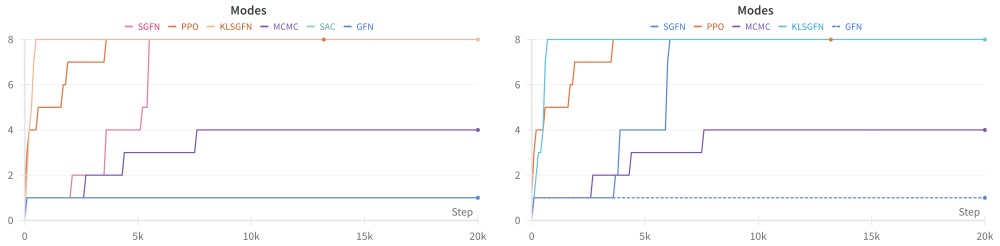
Figure 1: The GridWorld environment. The agent starts at the top-left corner and receives the highest reward at the four dark blue positions near the corners (with keys), lower rewards at the 2×2 squares near the corners, and even lower rewards at the lighter blue positions. Different grid sizes H and noise levels α can be explored.

We compare the performance of KL Divergence Optimization with Entropy-Ratio Estimation against vanilla GFlowNets, which are trained using trajectory balance (TB) Malkin et al. (2022),



(a) Medium. (b) Large.

Figure 2: Comparison results of L_1 error in GridWorld for varying map sizes.



(a) Medium. (b) Large.

Figure 3: Comparison results of the number of modes captured during training in GridWorld with varying map sizes.

Stochastic GFlowNets (SGFN) Pan et al. (2023), as well as other methods like Metropolis-Hastings MCMC Xie et al. (2021) and PPO Schulman et al. (2017). The evaluation is based on the empirical L_1 error, which measures the difference between the true reward distribution $p(x) = \frac{R(x)}{Z}$ and the estimated distribution $\pi(x)$, derived by repeated sampling and frequency counting of visits to all possible states x. Additionally, we compare the methods by counting the number of modes captured during training. Each algorithm is run with five different random seeds, and we report the mean of performance. The implementation details for each baseline are based on open-source code[1].

8.1.2 Performance Comparison

We now evaluate the effectiveness of KL Divergence Optimization with Entropy-Ratio Estimation across different map sizes and stochasticity levels in the GridWorld environment.

Varying map sizes. Figure 2 shows the empirical L_1 error for each method in GridWorld (with stochasticity level $\alpha = 0.25$) as the grid size increases. The results demonstrate that MCMC struggles with larger grids, and GFN fails to converge. Additionally, the performance of TB degrades significantly as the grid size grows, likely due to higher gradient variance, as suggested by Madan et al. (2022). In contrast, our proposed method (KL Divergence Optimization with Entropy-Ratio Estimation) consistently achieves the lowest L_1 error and converges faster than all baselines, including stochastic GFlowNets with DB-objective and Vanilla GFlowNets with TB-objective.

8.2 Biological Sequence Generation

In biological sequence generation, the objective is to discover sequences with optimal properties by maximizing a reward function corresponding to specific biological traits. This task is particularly challenging due to the inherent complexity and stochasticity present in biological environments. For

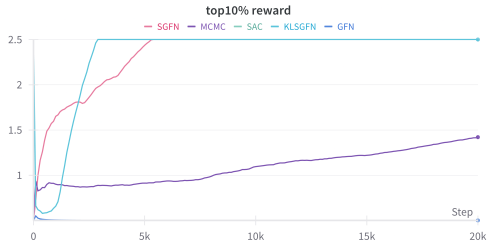
[1] https://github.com/GFNOrg/gflownet

example, in the task of generating DNA sequences, the objective might be to find sequences that exhibit high binding affinity to a particular transcription factor.

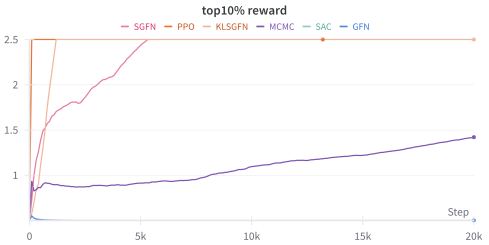
To demonstrate the efficacy of our proposed method, we evaluate it on the TFBind8 task, where the goal is to generate strings of nucleotides (e.g., DNA sequences of length 8). Conventionally, such tasks are modeled using an autoregressive Markov Decision Process (MDP). However, we utilize a prepend-append MDP (PA-MDP), where actions involve adding tokens (e.g., nucleotides) either to the beginning or the end of a partial sequence. The reward function, in this case, measures the DNA binding affinity to a human transcription factor, providing feedback on the sequence's fitness.

The complexity of the biological sequence generation problem lies in handling the vast combinatorial search space and the stochastic nature of sequence interactions with biological targets. By introducing stochasticity into the environment through transition dynamics, our method dynamically balances exploration and exploitation using KL divergence optimization and entropy-ratio estimation, which improves the robustness of the generated sequences and ensures better mode discovery.

TFBind8. Our goal is to generate a string of length 8 of nucleotides. Though an autoregressive MDP is conventionally used for strings, we use a prepend-append MDP (PA-MDP) Shen et al. (2023), in which the action involves either adding one token to the beginning or the end of a partial sequence. The reward is a DNA binding affinity to a human transcription factor Trabucco et al. (2022).



(a) $\alpha = 0.25$.



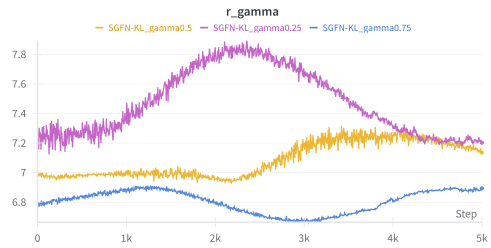
(b) $\alpha = 0.50$.

(c) $\alpha = 0.75$.

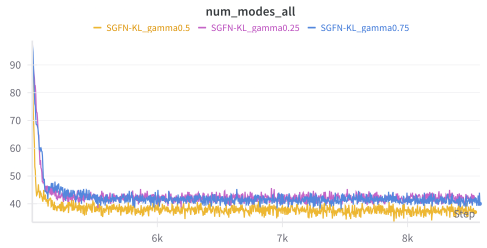
Figure 4: Comparison results of the quality of rewards captured during training for TFBIND experiment for different levels of stochasticity.

Quality of Rewards. In Figure 4, highlights the quality of rewards achieved by the models during training, for various levels of stochasticity level $\alpha = (0.25, 0.75)$. With increasing stochasticity, both SGFN and GFN show a significant decline in reward quality, reflecting their inability to handle uncertain transitions. In contrast, our method, equipped with entropy-ratio estimation, maintains high reward quality across all stochasticity levels. This result implies that our method not only captures more modes but also generates sequences with superior biological properties (e.g., higher binding affinities), thus ensuring better performance in biological sequence generation tasks.

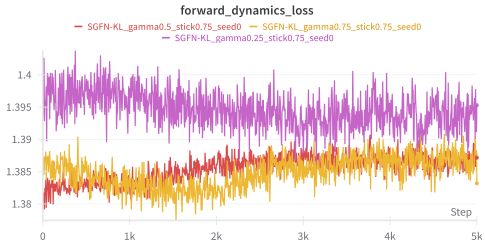
Effect of Gamma on Entropy Search, Number of Modes, and Dynamic Loss. Figure 5, illustrates the impact of varying γ values on entropy search, mode discovery, and dynamic loss for a stochasticity level $\alpha = (0.25, 0.75)$. The parameter γ controls the exploration-exploitation trade-off by adjusting the balance between high-entropy (exploratory) and low-entropy (exploitative) states.



(a) Entropy Search.

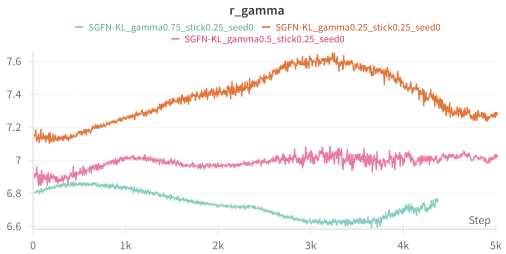


(b) Number of modes.

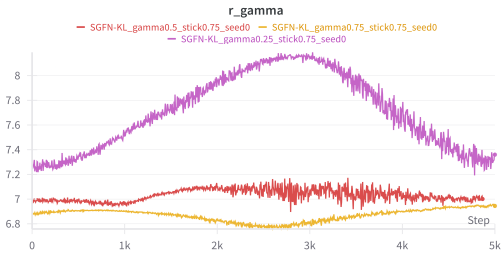


(c) Dynamic Loss.

Figure 5: Comparison results of the Entropy Search, number of modes, and dynamic loss captured during training for TFBIND experiment for different levels of gamma with stochasticity level $\alpha = 0.25$.



(a) Entropy Search with $\alpha = 0.25$.



(b) Entropy Search with $\alpha = 0.75$.

Figure 6: Entropy Search on handling dynamic stochasticity, for TFBIND experiment for different level of γ with low stochasticity (left) and low stochasticity (right).

As γ increases, the policy becomes more exploratory, leading to greater mode discovery (i.e., capturing a larger variety of solutions). This high entropy search encourages the model to explore less-visited regions of the sequence space, avoiding premature convergence to suboptimal sequences. The dynamic loss also decreases as γ increases, indicating more effective learning. This dynamic adjustment is crucial in biological sequence generation, where exploration can uncover novel, high-reward sequences.

Entropy Search on Dynamic Stochasticity In Figure 6, we explore the interaction between γ and the stochasticity level α during entropy search. A higher γ value leads to a broader exploration of the sequence space, especially under lower stochasticity ($\alpha = 0.25$), where the environment is more predictable. However, as α increases (making the environment more random), the search becomes more erratic. Our method, by adjusting γ dynamically, manages to stabilize the exploration process, maintaining a balance between discovering new high-quality sequences and exploiting the already learned ones. This adaptive behavior is key in environments with varying degrees of uncertainty, ensuring that the model can handle dynamic stochasticity effectively.

Impact on Performance and Search The introduction of γ and αin the KL divergence optimization framework plays a pivotal role in improving the search process for biological sequences. γ, by modulating the entropy ratio, allows the model to dynamically adjust its behavior based on the task's stochastic nature, promoting exploration when necessary and ensuring reliable exploitation when high-quality sequences are found. α, on the other hand, controls the environment's stochasticity, influencing how the model handles uncertain transitions. Together, these parameters ensure that the model strikes an optimal balance between exploring diverse sequences and refining high-reward ones, leading to improved performance in biological sequence generation tasks.

9 Conclusion

In this paper, we introduced a novel methodology, *KL Divergence Optimization with Entropy-Ratio Estimation for Stochastic GFlowNets*, which effectively extends GFlowNets to more complex and realistic stochastic environments, where existing GFlowNet approaches tend to underperform. Our method not only learns the GFlowNet policy but also incorporates entropy-ratio estimation to dynamically balance exploration and exploitation, making it more robust to stochastic transitions.

We conducted extensive experiments on standard GFlowNet benchmark tasks augmented with stochastic transition dynamics, demonstrating that our method significantly outperforms previous methods in terms of both mode discovery and state visitation coverage. The results show that by leveraging KL divergence optimization and entropy-ratio estimation, our approach can better handle the stochasticity in environments, leading to more efficient and accurate policy learning.

Future research could explore advanced model-based approaches for approximating transition dynamics in stochastic environments. Additionally, our method opens new possibilities for applying GFlowNets to other challenging real-world tasks, such as biological sequence generation and molecule discovery, where stochasticity plays a key role.

10 Acknowledgements

The authors would like to express their gratitude to colleagues and collaborators for their valuable insights and feedback on this work. We also appreciate the support from the research community for providing open-source resources and discussions that contributed to the development of our method. Finally, we would like to thank our respective institutions and funding agencies for their continued support and encouragement in advancing this research.

References

Christophe Andrieu et al. An introduction to mcmc for machine learning. *Machine learning*, 50(1): 5–43, 2003.

Yoshua Bengio et al. Flow network based generative models for non-iterative diverse candidate generation. *arXiv preprint arXiv:2106.04399*, 2021a.

Yoshua Bengio et al. Foundations of flow networks. *arXiv preprint arXiv:2107.00610*, 2021b.

Tristan Deleu et al. Bayesian structure learning with generative flow networks. *arXiv preprint arXiv:2202.13903*, 2022.

Scott Fujimoto, Herke van Hoof, et al. Addressing function approximation error in actor-critic methods. *arXiv preprint arXiv:1802.09477*, 2018.

Ian Goodfellow et al. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Tuomas Haarnoja et al. Reinforcement learning with deep energy-based policies. *arXiv preprint arXiv:1702.08165*, 2017.

Tuomas Haarnoja et al. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *International conference on machine learning*, pp. 1861–1870, 2018.

W Keith Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

Jonathan Ho, Tim Salimans, et al. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Moksh Jain, Yoshua Bengio, et al. Biological sequence design with generative flow networks. *arXiv preprint arXiv:2205.10477*, 2022a.

Moksh Jain, Yoshua Bengio, et al. Multi-objective molecule generation using generative flow networks. *arXiv preprint arXiv:2203.09964*, 2022b.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Matevž Kunaver and Tomaž Požrl. Diversity in recommender systems–a survey. *Knowledge-based systems*, 123:154–162, 2017.

Timothy P Lillicrap et al. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Marlos C. Machado, Marc G. Bellemare, Erik Talvitie, Joel Veness, Matthew Hausknecht, and Michael Bowling. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents, 2017. URL `https://arxiv.org/abs/1709.06009`.

Ritika Madan et al. Learning gflownets from partial episodes for improved convergence and stability. *arXiv preprint arXiv:2201.13279*, 2022.

Nikolay Malkin et al. Trajectory balance: Improved credit assignment in gflownets. *arXiv preprint arXiv:2201.13259*, 2022.

Nicholas Metropolis et al. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

Volodymyr Mnih et al. Human-level control through deep reinforcement learning. *Nature*, 518 (7540):529–533, 2015.

Sobhan Mohammadpour, Emmanuel Bengio, Emma Frejinger, and Pierre-Luc Bacon. Maximum entropy gflownets with soft q-learning, 2024. URL `https://arxiv.org/abs/2312.14331`.

Ling Pan, Dinghuai Zhang, Moksh Jain, Longbo Huang, and Yoshua Bengio. Stochastic generative flow networks, 2023. URL `https://arxiv.org/abs/2302.09465`.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL `https://arxiv.org/abs/1707.06347`.

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

Richard S Sutton and Andrew G Barto. Reinforcement learning: An introduction. 2018.

Brandon Trabucco, Xinyang Geng, Aviral Kumar, and Sergey Levine. Design-bench: Benchmarks for data-driven offline model-based optimization. In *International Conference on Machine Learning*, pp. 21658–21676. PMLR, 2022.

Shunshi Xie, Wei Zheng, et al. Mars: Markovian score climbing for efficient structure learning of markov random fields. In *International Conference on Learning Representations (ICLR)*, 2021. URL `https://openreview.net/forum?id=rsUpk06ZFPu`.

Yang Yang et al. A dichotomy of solutions to high-dimensional linear inverse problems: One proximal operator and one gradient descent step. *arXiv preprint arXiv:2201.09460*, 2022.

David W Zhang, Corrado Rainone, Markus Peschl, and Roberto Bondesan. Robust scheduling with GFlownets. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=ZBUthI6wK9h`.

ARCHITECTURAL DETAILS FOR EXPERIMENTS

GRIDWORLD EXPERIMENTS

To reproduce the GridWorld experiments, we used the following **network architecture**:

- **Policy Network**:
  - **Input Layer**: The state representation is a 2D grid encoded as a flattened vector of size $H \times H$, where H is the grid size.
  - **Hidden Layers**:
    - **Layer 1**: Fully connected layer with 128 neurons, ReLU activation.
    - **Layer 2**: Fully connected layer with 64 neurons, ReLU activation.
  - **Output Layer**: Outputs the probabilities over all possible actions, implemented as a softmax layer to ensure a valid probability distribution.
- **Training Details**:
  - **Optimizer**: Adam optimizer with a learning rate of 0.001.
  - **Batch Size**: 32.
  - **Exploration Parameter** (γ): Initially set to 0.5 and adjusted adaptively during training based on the observed variance in the state-action values.
  - **Entropy Regularization**: An additional entropy regularization term is added to the loss function to encourage exploration, with a weight of 0.01.

BIOLOGICAL SEQUENCE GENERATION (TFBIND8)

For the TFBind8 biological sequence generation experiment, we used the following **architecture**:

- **Policy Network**:
  - **Input Layer**: The input is a partial sequence of nucleotides represented as a one-hot encoded vector. For sequences of length 8, the input size is 8×4 (since each nucleotide can be one of 4 bases).
  - **Embedding Layer**: An embedding layer maps the one-hot encoded representation to a continuous vector space of dimension 16.
  - **LSTM Layer**: A single LSTM layer with 128 hidden units is used to capture dependencies between different positions in the sequence.
  - **Fully Connected Layer**: The LSTM output is passed through a fully connected layer with 64 neurons and ReLU activation.
  - **Output Layer**: Outputs the probability distribution over the four nucleotides for the next position, implemented as a softmax layer.
- **Training Details**:
  - **Optimizer**: Adam optimizer with a learning rate of 0.0005.
  - **Batch Size**: 64.
  - **Sequence Augmentation**: During training, random noise is added to the nucleotide embeddings to simulate stochasticity in biological environments.
  - **Entropy Regularization**: To ensure mode discovery, we add an entropy regularization term with a weight of 0.05.