Supplement to "Improving Generative Flow Networks with Path Regularization"

A RELATED WORK

GFlowNets The objective of GFlowNets is related to MCMC methods for sampling from a given unnormalized density function, especially in discrete spaces where exact sampling is intractable (Dai et al. (2020); Grathwohl et al. (2021)). However, GFlowNets amortize the complexity of iterative sampling by a training procedure that implies the data's compositional structure as its learning problem. Empirically, GFlowNets' performance is better than other earlier methods in a wide variety of tasks: small molecules generation (Bengio et al. (2021a)), discrete probabilistic modeling (Zhang et al. (2022)), Bayesian structure learning (Deleu et al. (2022)) and biological sequence design (Jain et al. (2022)). On the theoretical side, definitions and properties of GFlowNets are more investigated in Bengio et al. (2021b).

Optimal Transport The optimal transport theory (OT) (Villani (2003)) has established a natural and useful geometric tool for comparing measures supported on metric probability spaces. The development of OT theory has a long history, where it has been discovered in many settings and under different forms. And in recent years, another revolution in the spread of OT has been witnessed, thanks to the emergence of approximate solvers that can scale to the problem of large dimensions. As a consequence, OT is being widely used to solve various problems in computer graphics (Bonneel et al. (2011),Nguyen et al. (2021)), image processing (Xia et al. (2014)), and machine learning (Courty et al. (2014), Ho et al. (2017) Genevay et al. (2018), Bunne et al. (2019)).

Energy-based models EBMs, or energy functions parameterized by deep neural networks, have demonstrated effectiveness in generative modeling (Salakhutdinov & Hinton (2009); Hinton et al. (2006)). Contrastive divergence methods (Hinton (2002); Tieleman (2008); Du et al. (2021)) have been proposed to handle costly MCMC processes by approximating energy gradient. Recently, it has been shown that simultaneous learning of the proposal distribution can also be helpful (Dai et al. (2019); Arbel et al. (2021)). Then this finding has been extended to discrete spaces by using GFlowNets in Zhang et al. (2022).

Biological sequence design Various methods have been proposed to handle the biological sequence design tasks: deep model-based optimization (Trabucco et al. (2021)), Bayesian optimization (Belanger et al. (2019); Swersky et al. (2020)), reinforcement learning (Angermueller et al. (2020)), adaptive evolutionary methods (Hansen (2006); Sinai et al. (2020)), and so on. Recently, GFlowNets also have been proposed as a useful generator of diverse candidates for this problem in Jain et al. (2022).

B BACKGROUND OF GFLOWNETS

Generative Flow Networks (GFlowNets) are a recently proposed class of generative model, which aims to sample a structural object $\mathbf{x}$ with probability proportional to a given reward function $R(\mathbf{x})$. From the reinforcement learning viewpoint, GFlowNets learn a stochastic policy to generate object $\mathbf{x} \in \mathcal{X}$ by applying a sequence of discrete actions $a \in \mathcal{A}$ where $\mathcal{A}$ is the action space. The construction of an object $x \in \mathcal{X}$ defines a *complete trajectory* $\tau = (s_0, s_1, ..., s_n = x, s_f)$ where s_0 is the *initial state*, $s_n = x \in \mathcal{X}$ is the *terminal state* (indicating entirely constructed object), and s_f is the *final state*. Note that the same terminal state can be formed by different sequences of actions. These states and actions correspond to the vertices and edges of a directed acyclic graph $G = (\mathcal{S}, \mathcal{A})$. In addition, for each transition $s \to s' \in \mathcal{A}$, we call s a parent of s', and s' a child of s. $\mathcal{T}$ is defined as the set of all complete trajectories.

Following Bengio et al. (2021b), a *trajectory flow* is any nonnegative function defined on the set of complete trajectories, such as $F : \mathcal{T} \mapsto \mathbb{R}^+$. Correspondingly, the flow through a state (state flow) is defined as $F(s) = \sum_{\tau \in \mathcal{T}, s \in \tau} F(\tau)$ and the flow through a edge (edge flow) is defined as $F(s \to s') = \sum_{\tau \in \mathcal{T}, s \to s' \in \tau} F(\tau)$. Additionally, the forward transition probabilities P_F and the backward transition probabilities P_B are defined as follows:

$$P_F(s'|s) := \frac{F(s \to s')}{F(s)}, \tag{17}$$

$$P_B(s|s') := \frac{F(s \to s')}{F(s')}. \tag{18}$$

Then the training objective of the GFlowNet is to learn a *consistent flow* (Bengio et al. (2021b); Malkin et al. (2022)) that has the *terminal flow* $F(x \to s_f)$ approximately equal a given reward function $R(x)$ for any $x \in X$. In addition, when the flow is consistent, the forward transition probabilities P_F and the backward transition probabilities P_B correspondingly define a distribution over the children and parent of each state, which can be considered as the forward and backward policy of GFLowNets.

Specifically, followed by Malkin et al. (2022), the GflowNet models the forward policy, backward policy and total flow of a Markovian flow F by $P_F(.|.;\theta)$, $P_B(.|.;\theta)$ and Z_θ. The *trajectory balance objective* is then optimized for each complete trajectory τ sampled from the training policy π_θ:

$$\mathcal{L}_{TB}(\tau, \theta) = \left(\log(Z_\theta \prod_{t=1}^{n} P_F(s_t|s_{t-1};\theta)) - \log(R(x) \prod_{t=1}^{n} P_B(s_{t-1}|s_t;\theta)) \right)^2. \tag{19}$$

which is derived from the *trajectory balance constraint* (Malkin et al. (2022))

Moreover, as already proved by Bengio et al. (2021b), π_θ can be chosen as any distribution on the set of complete trajectories $\mathcal{T}$ with full supports, or the GflowNet can be trained with offline policy as well, such as a mixture between the GFlowNet's forward policy and an uniform distribution over allowed actions in each state:

$$\pi_\theta = (1-\alpha) P_F(.|.;\theta) + \alpha \text{ Uniform} \tag{20}$$

There also exist other objectives for learning a GFlowNet, which are based on *flow matching* constraint or *detail balance* constraint as in Bengio et al. (2021a;b). However, Malkin et al. (2022) empirically shows that the trajectory balance objective improves the training of a GFlowNet in terms of more efficient credit assignment and faster convergence, compared to the previously proposed objectives. These advantages make us choose it as the training objective in this paper.

C DROPOUT OPTIMAL TRANSPORT

A limitation of the current method is computing the optimal transport distances for all couples of nearest neighbor states, especially in high dimensional discrete data. Our proposed dropout OT might be a solution. This is because rather than sampling trajectories τ and using all edges from them, we can separately sample edges $s \to s'$ proportional to edge flows, allowing us to efficiently compute path regularization.

Theorem C.1 *For any complete trajectory $\tau = (s_0 \to s_1 \to \ldots \to s_n)$ sampled from the training policy π_θ*

$$\mathbb{E}_{\tau\sim\pi_\theta}(\mathcal{L}_{OT}(\tau)) \propto \mathbb{E}_{s\to s'\sim\pi_\theta}(OT(P_F(\cdot|s), P_F(\cdot|s'))). \tag{21}$$

The proof of Theorem C.1 is in Appendix D.3. Here we train GFlowNets with trajectory balance objective. Therefore, when sampling a trajectory τ, we get a set of edges from τ. We just sample uniformly a p percentage of edges to compute OT loss.

To sample p percentage of edges, let sample $r_s \sim Ber(p)$.

$$\mathbb{E}_{s\to s'\sim\pi_\theta}(\text{OT}(P_F(\cdot|s), P_F(\cdot|s'))) = \frac{1}{p}\mathbb{E}_{r_s\sim Ber(p)}\mathbb{E}_{s\to s'\sim\pi_\theta}(r_s.\text{OT}(P_F(\cdot|s), P_F(\cdot|s'))). \tag{22}$$

We approximate the path regularization loss via:

$$\mathcal{L}_{\text{OT}}(\tau) \simeq \frac{1}{p}\sum_{t=0}^{n-1} x_t \text{OT}(P_F(.|s_t), P_F(.|s_{t+1})) \tag{23}$$

with x_t drawn independently from Ber(p) for all $0 \le t \le n-1$. Intuitively, if $x_t = 0$ then we don't need to calculate the corresponding optimal transport cost anymore, which reduces a considerable amount of computing time and memory down to p percentage.

D PROOFS

D.1 PROOF OF THEOREM 3.1

For any trajectory $\tau = (s_0 \to s_1 \to \ldots \to s_n)$, we first prove that for any $t \in \overline{0, n-1}$

$$\text{OT}(P_F(\cdot|s_t), P_F(\cdot|s_{t+1})) \le \sum_{u\in\text{Child}(s_t)} P_F(u|s_t)\log(P_B(s_t|u)) - \log(P_F(s_{t+1}|s_t)) + \mathbf{H}(P_F(\cdot|s_{t+1})). \tag{24}$$

Consider two neigboor states s_t and s_{t+1} with the children sets: Child$(s_t) = \{u_1, ..., u_k\}$ and Child$(s_{t+1}) = \{v_1, ..., v_l\}$. By definition 5, the optimal transportation distance between two distributions $P_F(.|s_t)$ and $P_F(.|s_{t+1})$ is defined as:

$$\text{OT}_{\mathbf{C}}\left(P_F(\cdot|s_t), P_F(\cdot|s_{t+1})\right) := \min_{\pi \in \prod(P_F(\cdot|s_t), P_F(\cdot|s_{t+1}))} \langle \mathbf{C}, \pi \rangle, \tag{25}$$

where the admissible couplings set is defined as:

$$\Pi(P_F(.|s_t), P_F(.|s_{t+1})) = \left\{\pi \in \mathbb{R}_+^{k \times l} : \pi \mathbb{1}_l = P_F(\cdot|s_t), \pi^{\mathrm{T}} \mathbb{1}_k = P_F(\cdot|s_{t+1})\right\}. \tag{26}$$

We have,

$$\begin{aligned}
&\text{OT}(P_F(.|s_t), P_F(.|s_{t+1})) \\
&\leq \sum_i \sum_j \pi_{ij} \mathbf{C}_{ij} \\
&\leq -\sum_i \sum_j \pi_{ij} \log\left(P_B(s_t|u_i) P_F(s_{t+1}|s_t) P_F(v_j|s_{t+1})\right) \\
&= -\sum_i \sum_j \pi_{ij} \log\left(P_B(s_t|u_i)\right) - \sum_i \sum_j \pi_{ij} \log\left(P_F(s_{t+1}|s_t)\right) - \sum_i \sum_j \pi_{ij} \log\left(P_F(v_j|s_{t+1})\right)) \\
&= -\sum_i \log\left(P_B(s_t|u_i)\right) \sum_j \pi_{ij} - \log\left(P_F(s_{t+1}|s_t)\right) \sum_i \sum_j \pi_{ij} - \sum_i \log\left(P_F(v_j|s_{t+1})\right)) \sum_j \pi_{ij} \\
&= -\sum_i \log\left(P_B(s_t|u_i)\right) P_F(u_i|s_t) - \log\left(P_F(s_{t+1}|s_t)\right) - \sum_j \log\left(P_F(v_j|s_{t+1})\right)) P_F(v_j|s_{t+1}) \\
&= \sum_{u \in \text{Child}(s_t)} P_F(u|s_t) \log(P_B(s_t|u)) - \log(P_F(s_{t+1}|s_t)) + \mathbf{H}(P_F(.|s_{t+1}).
\end{aligned} \tag{27}$$

The first inequality obtained by the definition of optimal transport distance in Eq. 25, the second inequality comes from Eq. 8, the fifth equality is due to the constraints of admissible couplings in Eq. 26.

As a consequence, the upper bound loss is obtained by summing up all inequality 24 for all t.

D.2 PROOF OF THEOREM 3.2

Recall from definition 5 the optimal transportation distance between two distributions $P_F(.|s)$ and $P_F(.|s')$ is defined as:

$$\text{OT}_{\mathbf{C}}\left(P_F(\cdot|s), P_F(\cdot|s')\right) := \min_{\pi \in \Pi(P_F(\cdot|s), P_F(\cdot|s'))} \langle \mathbf{C}, \pi \rangle. \tag{28}$$

Let decompose the total cost $\langle \mathbf{C}, \pi \rangle$

$$\begin{aligned}
\langle \mathbf{C}, \pi \rangle &= \sum_{i,j} \pi_{ij} \mathbf{C}_{ij} \\
&= \sum_{i,j} \pi_{ij} (-\log(P_B(s|u_i)) - \log(P_F(s'|s)) - \log(P_F(v_j|s'))) \\
&+ \sum_{u_i = s', j} \pi_{ij} (\log(P_B(s|s')) + \log(P_F(s'|s))) \\
&+ \sum_{u_i \neq s', v_j \in Child(u_i), a_i \neq a^{\top}} \pi_{ij} (\log(P_B(s|u_i)) + \log(P_F(s'|s)) + \log(P_F(v_j|s') + \mathbf{C}_{ij}) \\
&+ \sum_{u_i = v_j} \pi_{ij} (\log(P_B(s|u_i)) + \log(P_F(s'|s)) + \log(P_F(v_j|s'))).
\end{aligned} \tag{29}$$

We will prove that $u_i \neq v_j \quad \forall i, j$, i.e, Child$(s) \cap$ Child$(s') = \emptyset$, indeed,

$$a_i \neq a_k + a_h \quad \forall a_i, a_k, a_h \in \mathcal{A} \Longrightarrow a_i \neq a_s^* + a_j \Longrightarrow s + a_i \neq s + a_s^* + a_j \Longrightarrow u_i \neq v_j. \quad \forall i, j \tag{30}$$

We have:

$$\begin{aligned} & u_i \neq s', \quad v_j \in Child(u_i), \quad a_i \neq a^\top \\ \Longrightarrow \quad & a_i \neq a_s^*, \quad s + a_i + a_{u_i}^* = s + a_s^* + a_j, \quad a_i \neq a^\top \\ \Longrightarrow \quad & a_i \neq a_s^*, \quad a_i + a_{u_i}^* = a_s^* + a_j, \quad a_i \neq a^\top \\ \Longrightarrow \quad & a_i \neq a_s^*, \quad a_i = a_j \neq a^\top, \quad a_{u_i}^* = a_s^*. \end{aligned} \tag{31}$$

As a result, we can rewrite Eq. 29 as:

$$\begin{aligned} \langle \mathbf{C}, \pi \rangle = & \sum_{i,j} \pi_{ij}(-\log(P_B(s|u_i)) - \log(P_F(s'|s)) - \log(P_F(v_j|s'))) \\ & + \sum_{u_i = s', j} \pi_{ij}(\log(P_B(s|s')) + \log(P_F(s'|s))) \\ & + \sum_{u_i \neq s', a_i = a_j \neq a^\top} \pi_{ij}(\log(P_B(s|u_i)) + \log(P_F(s'|s)) + \log(P_F(v_j|s') + \mathbf{C}_{ii}). \end{aligned} \tag{32}$$

The first term of above equation actually is the upper bound of the optimal transport distance. Therefore, we can rewrite the total transportation cost as:

$$\begin{aligned} \langle \mathbf{C}, \pi \rangle = & \sum_{u \in \text{Child}(s)} P_F(u|s) \log(P_B(s|u)) - \log(P_F(s'|s)) + \mathbf{H}(P_F(\cdot|s')) \\ & + P_F(s'|s).(\log(P_B(s'|s)) + \log(P_F(s'|s))) \\ & + \sum_{u_i \neq s', a_i = a_j \neq a^\top} \pi_{ij}(\log(P_B(s|u_i)) + \log(P_F(s'|s)) + \log(P_F(v_j|s') + \mathbf{C}_{ii}). \end{aligned} \tag{33}$$

From the definition of c'_i in Eq. 15, we have

$$c'_i = \begin{cases} \log(P_B(s|u_i)) + \log(P_F(s'|s)) + \log(P_F(v_j|s') + \mathbf{C}_{ii}, & \text{if } u_i \neq s', a_i = a_j \neq a^\top \\ 0 & \text{if } u_i = s' \text{ or } a_i = a^\top. \end{cases} \tag{34}$$

From Eq. 33 and Eq. 34, we have

$$\sum_{u_i \neq s', a_i = a_j \neq a^\top} \pi_{ij}(\log(P_B(s|u_i)) + \log(P_F(s'|s)) + \log(P_F(v_j|s') + \mathbf{C}_{ii}) = \sum_i \pi_{ij}.c'_i. \tag{35}$$

Thus, we find that

$$\underset{\pi \in \prod(P_F(\cdot|s), P_F(\cdot|s'))}{\arg\min} \langle \mathbf{C}, \pi \rangle = \underset{\pi \in \prod(P_F(\cdot|s), P_F(\cdot|s'))}{\arg\min} \langle \mathbf{C}', \pi \rangle \tag{36}$$

where, $\mathbf{C}'$ is a diagonal matrix with the diagonal $c'_i \leq 0$. For convenience, if action a_i is invalid at state s, we assign $P_F(u_i \mid s) := 0$, so the cost matrix of the optimal transport distance still is a square matrix with the zero cost a invalid actions, then applying the Lemma 1, we have:

$$\min_{\pi \in \prod(P_F(\cdot|s), P_F(\cdot|s'))} \langle \mathbf{C}', \pi \rangle = \sum_i \min(P_F(u_i|s), P_F(v_i|s))\mathbf{C}^{'}_{ii}. \tag{37}$$

We obtain the closed-form formulation for optimal transport distance

$$\begin{aligned} \text{OT}\,(P_F(\cdot|s), P_F(\cdot|s')) = & \sum_{u \in \text{Child}(s)} P_F(u|s) \log(P_B(s|u)) + \mathbf{H}(P_F(\cdot|s')) \\ & + P_F(s'|s).(\log(P_B(s'|s)) + \log(P_F(s'|s))) \\ & + \sum_{i \in A_s^* \bigcap A_{s'}^*} \min(P_F(u_i|s), P_F(v_i|s'))c'_i. \end{aligned} \tag{38}$$

Lemma 1 *Given a squared diagonal cost matrix $\mathbf{C}'$ with non-positive entities in the diagonal, the solution of optimal transport problem between two distribution $P_F(\cdot|s)$ and $P_F(\cdot|s')$, which has the same number of support points, given cost matrix $\mathbf{C}'$ is given by:*

$$\min_{\pi \in \Pi(P_F(\cdot|s), P_F(\cdot|s'))} \langle \mathbf{C}', \pi \rangle = \sum_i \min(P_F(u_i|s), P_F(v_i|s))\mathbf{C}^{'}_{ii}. \tag{39}$$

Proof of Lemma 1: Let define

$$F(\pi) = \langle \mathbf{C}', \pi \rangle,$$

$$\overline{p}_{ij} = \begin{cases} \min(p_s^i, p_{s'}^i), & \text{if } i = j \\ \frac{\left(p_s^i - \min(p_s^i, p_{s'}^i)\right)\left(p_{s'}^j - \min(p_s^j, p_{s'}^j)\right)}{1 - \sum_k min(p_s^k, p_{s'}^k)} & \text{if } i \neq j. \end{cases}$$

where $p_s^i := P_F(u_i|s)$ and $p_{s'}^j := P_F(v_j|s')$.

We will prove that $\overline{\pi} \in \Pi\left(P_F(\cdot|s), P_F(\cdot|s')\right)$ and $F(\pi) \geq F(\overline{\pi}) \quad \forall \pi \in \Pi\left(P_F(\cdot|s), P_F(\cdot|s')\right)$.

It is not difficult to show that $\overline{\pi}_{ij} \geq 0$. From the definition of $\overline{\pi}$, we have

$$\sum_j^n \overline{\pi}_{ij} = \sum_{j \neq i} \overline{\pi}_{ij} + \overline{\pi}_{ii} = \sum_{j \neq i} \frac{\left(p_s^i - \min(p_s^i, p_{s'}^i)\right)\left(p_{s'}^j - \min(p_s^j, p_{s'}^j)\right)}{1 - \sum_k min(p_s^k, p_{s'}^k)} + \min(p_s^i, p_{s'}^i). \tag{40}$$

If $\min(p_s^i, p_{s'}^i) = p_s^i$ then

$$\sum_j^n \overline{\pi}_{ij} = 0 + \min(p_s^i, p_{s'}^i) = p_s^i. \tag{41}$$

else $\min(p_s^i, p_{s'}^i) = p_{s'}^i$ then

$$\begin{aligned} &\sum_{j \neq i} \left(p_{s'}^j - \min(p_s^j, p_{s'}^j)\right) = \sum_j \left(p_{s'}^j - \min(p_s^j, p_{s'}^j)\right) = 1 - \sum_k \min(p_s^k, p_{s'}^k) \\ &\Longrightarrow \sum_j^n \overline{\pi}_{ij} = \left(p_s^i - \min(p_s^i, p_{s'}^i)\right) \frac{\sum_{j \neq i}\left(p_{s'}^j - \min(p_s^j, p_{s'}^j)\right)}{1 - \sum_k \min(p_s^k, p_{s'}^k)} + \min(p_s^i, p_{s'}^i) = p_s^i. \end{aligned} \tag{42}$$

Therefore $\sum_j^n \overline{\pi}_{ij} = p_s^i = P_F(u_i|s)$. Similarly, $\sum_i^n \overline{\pi}_{ij} = p_s^j = P_F(v_j|s')$, combining with $\overline{\pi}_{ij} \geq 0$, we have

$$\overline{\pi} \in \Pi\left(P_F(\cdot|s), P_F(\cdot|s')\right). \tag{43}$$

Moreover

$$F(\pi) = \langle \mathbf{C}', \pi \rangle = \sum_i \pi_{ii} \mathbf{C}'_{\mathbf{ii}} \geq \sum_i \min(p_s^i, p_{s'}^i) \mathbf{C}'_{\mathbf{ii}} = \langle \mathbf{C}', \overline{\pi} \rangle = F(\overline{\pi}) \quad \forall \pi \in \Pi\left(P_F(\cdot|s), P_F(\cdot|s')\right). \tag{44}$$

As a consequence, we obtained the solution of optimal transport problem.

closed-form solution for optimal transport distance at terminal state. We will derive the closed-form solution for optimal transport distance in case of two neighbor states $s < s'$, in which s' is a terminal state. In the case of Hyper-grid environment, EB-GFN experiments, and Biological Sequence Design, all terminal state x have only one child that is the final state s_f, and $P_F(s_f|x) = 1 \quad \forall x$. Thus, the admissible couplings set $\prod(P_F(\cdot|s), P_F(\cdot|s'))$ has only one element. That is $\pi^* = P_F(\cdot|s)$. As a result, the optimal transportation distance between $P_F(.|s)$ and $P_F(.|s')$ is:

$$\text{OT}\left(P_F(\cdot|s), P_F(\cdot|s')\right) = \min_{\pi \in \prod(P_F(\cdot|s), P_F(\cdot|s'))} \langle \mathbf{C}, \pi \rangle = \langle \mathbf{C}, \pi^* \rangle. \tag{45}$$

Specially, in EB-GFN experiments, all children u_i of s is a terminal state so $d(u_i, s_f) = -\log(1) = 0$. This makes $\mathbf{C} = 0$ and $\text{OT}\left(P_F(\cdot|s), P_F(\cdot|s')\right) = 0$. In Hyper-grid environment experiment, for terminal sate s' because $c_i' = 0$, we have:

$$\begin{aligned} \text{OT}\left(P_F(\cdot|s), P_F(\cdot|s')\right) = &\sum_{u \in \text{Child}(s)} P_F(u|s) \log(P_B(s|u)) \\ &+ P_F(s'|s).(\log(P_B(s'|s)) + \log(P_F(s'|s))). \end{aligned} \tag{46}$$

The Hyper-grid environment (Bengio et al., 2021a) (in section 4.1) and EB-GFN experiments (Zhang et al., 2022) (in section 4.3) satisfy two condition in Theorem 3.2. In Biological Sequence Design (Jain et al., 2022) (in section 4.2) such as protein and DNA sequences, the action space consists of actions adding a nucleic acid in $\{A, T, G, U\}$ and a amino acid respectively. Such settings satisfy former condition $a_i \neq a_k + a_h \quad \forall a_i, a_k, a_h \in \mathcal{A}$. However, the later condition $a_i + a_h = a_m + a_n, a_i \neq a_m \Longleftrightarrow a_i = a_n, a_h = a_m, a_i \neq a_m$ is no longer true because the order property of action space, i.e, $a_i + a_j \neq a_j + a_i$. In this situation, the third terms in Eq. 14 is zero and we can still using the formulation in Eq. 14. Generally, the action space is independence and unique factorization.

D.3 PROOF OF THEOREM C.1

By definition of the edge flow we have

$$\sum_{\tau : s \to s' \in \tau} P(\tau) = \sum_{\tau : s \to s' \in \tau} \frac{F(\tau)}{Z} = \frac{F(s \to s')}{Z} = P(s \to s'). \tag{47}$$

From that equation, we find that

$$\begin{aligned}
\mathbb{E}_{\tau \sim \pi_{\boldsymbol{\theta}}}(\mathcal{L}_{\text{OT}}(\tau)) &= \mathbb{E}_{\tau \sim \pi_{\boldsymbol{\theta}}} \left(\sum_{s \to s' \in \tau} \text{OT}\left(P_F(\cdot|s), P_F(\cdot|s')\right) \right) \\
&= \sum_{\tau} \sum_{s \to s' \in \tau} \text{OT}\left(P_F(\cdot|s), P_F(\cdot|s')\right).P(\tau) \\
&= \sum_{s \to s'} \sum_{\tau : s \to s' \in \tau} \text{OT}\left(P_F(\cdot|s), P_F(\cdot|s')\right).P(\tau) \\
&= \sum_{s \to s'} \text{OT}\left(P_F(\cdot|s), P_F(\cdot|s')\right) \sum_{\tau : s \to s' \in \tau} P(\tau) \\
&= \sum_{s \to s'} \text{OT}\left(P_F(\cdot|s), P_F(\cdot|s')\right).P(s \to s') \\
&\propto \mathbb{E}_{s \to s' \sim \pi_{\boldsymbol{\theta}}}(\text{OT}\left(P_F(\cdot|s), P_F(\cdot|s')\right)).
\end{aligned} \tag{48}$$

□

E EXPERIMENT SETTINGS

In this part, we report experiment settings, including evaluation metrics for comparing the methods, hyper-parameter choices, and neural network architectures for all experiments. For biological sequence design tasks, we also give more details about the task description and datasets used for training. Note that the regularization coefficients provided in this part are task-specific. Specifically, λ is chosen from a predefined set of values with different scales in each task. However, because all tasks in our experiment parts do not change the target distribution between the training and test time, the reported λ is chosen to have the best model's performance.

E.1 HYPER-GRID ENVIRONMENT

E.1.1 EVALUATION CRITERIA

To evaluate the performance, we measure the KL divergence between the actual and empirical distribution of the last 2×10^5 visited states. The number of modes found during the training progress is also used to measure the learned models' performance.

E.1.2 IMPLEMENTATION DETAILS

GFlowNet: For the implementation of the GFlowNet model, we also follow the framework of Malkin et al. (2022): an MLP with two hidden layers of 256 dimensions each. The GFlowNet policy model, which includes both P_F and P_B, is trained with a learning rate of 0.001 while the learning rate for total flow Z_θ is 0.1. We use a mini-batch size of 16 and 62500 training steps with the trajectory balance objective.

Proposed OT regularization The regularization coefficient is 0.02 for both Min OT, UB-OT, and Max OT in $4 - D$ hypergrid environment and 0.1 for both Min OT, UB-OT, and Max OT in $8 - D$ hypergrid environment.

E.2 BIOLOGICAL SEQUENCE DESIGN

E.2.1 TASK DESCRIPTION & DATASETS

These experiments simulate the process of designing biological sequences, such as anti-microbial peptides, DNA, and protein sequences..., in drug discovery applications. This process often consists of an active loop with several rounds of ideating molecules and multiple-stage evaluations for filtering candidates, with rising levels of precision and cost. This characteristic makes the diversity of proposed candidates a considerable concern in the ideation phase because many similar candidates can all fail in the later phases.

Specifically, we consider the problem of finding objects x in the space of discrete objects $\mathcal{X}$, that maximize a given oracle $f : \mathcal{X} \mapsto \mathbb{R}^+$. Here, we can only query this oracle N times, each with an input batch of fixed size b. This can form N rounds of evaluation in the active learning setting, where the generative policy is initially given a dataset $D_0 = \left\{\left(x_1^0, y_1^0\right), \ldots, \left(x_n^0, y_n^0\right)\right\}$ collected from the oracle, where $y_i^0 = f(x_i^0)$ for $1 \leq i \leq n$.

Because the oracle can only be called limited, we also train a supervised proxy model M that predicts y from x to approximate the oracle f. Specifically, in i-th round, given the current dataset $\mathcal{D}_i$, this proxy model can be used as a reward function R to collect additional observations to train our generative policy to propose a batch of candidates $\mathcal{B}_i = \left\{x_1^i, \ldots, x_b^i\right\}$. Then the current dataset $\mathcal{D}_i$ is updated for the next round of evaluation as $\mathcal{D}_{i+1} = \mathcal{D}_i \cup \left\{\left(x_1^i, y_1^i\right), \ldots, \left(x_b^i, y_b^i\right)\right\}$ where $y_j^i = f\left(x_j^i\right)$.

Following the framework of Jain et al. (2022), we will conduct experiments on the biological sequence design tasks:

Anti-Microbial Peptide Design: This task aims to generate short amino-acid sequences of length lower than 51, which have anti-microbial properties. The vocabulary has 20 amino-acids $[A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y]$. The active learning algorithm is evaluated for $N = 10$ rounds, with the number of candidates generated each round $b = 1000$. The initial dataset $\mathcal{D}_0$ contains 3219 AMPs and 4611 non-AMP sequences, which is collected from the DBAASP database Pirtskhalava et al. (2021).

TFBind 8: The goal of this task is to generate DNA sequences of length 8, which have high binding activity with human transcription factors. The vocabulary has 4 nucleobases $[A, C, T, G]$. The active learning algorithm is evaluated for $N = 10$ round, with the number of candidates generated each round $b = 128$. The initial dataset $\mathcal{D}_0$ contains $32,898$ samples, which is half of all possible DNA sequences of length 8 having lower scores. The data and the oracle used are from Barrera et al. (2016).

GFP: The objective of this task is to generate protein sequences of length 237 that have high fluorescence. The vocabulary is similar to the one of the AMP task (size 20). The active learning algorithm is evaluated for $N = 10$ round, with the number of candidates generated each round $b = 128$. The initial dataset $\mathcal{D}_0$ contains $5,000$ samples, which is from Rao et al. (2019); Sarkisyan et al. (2016) together with the oracle.

E.2.2 EVALUATION CRITERIA

To evaluate the performance, we also use the metrics as in Jain et al. (2022). Specifically, considering a set of candidates $\mathcal{D}$, we have the following metrics:

Performance score: mean score of the candidates in the set

$$\operatorname{Mean}(\mathcal{D}) = \frac{\sum_{(x_i, y_i) \in \mathcal{D}} y_i}{|\mathcal{D}|}, \tag{49}$$

Diversity: a measurement of how well the generated candidates can capture the modes of the distribution implied by the oracle

$$\operatorname{Diversity}(\mathcal{D}) = \frac{\sum_{(x_i, y_i) \in \mathcal{D}} \sum_{(x_j, y_j) \in \mathcal{D} \setminus \{(x_i, y_i)\}} d\left(x_i, x_j\right)}{|\mathcal{D}|(|\mathcal{D}| - 1)}, \tag{50}$$

where d is a distance defined over $\mathcal{X}$, such as Levenshtein distance Miller et al. (2009).

Novelty: a measure of the difference between the candidates in $\mathcal{D}$ and $\mathcal{D}_0$

$$\operatorname{Novelty}(\mathcal{D}) = \frac{\sum_{(x_i, y_i) \in \mathcal{D}} \min_{s_j \in \mathcal{D}_0} d\left(x_i, s_j\right)}{|\mathcal{D}|}. \tag{51}$$

These metrics will be evaluated on the set of candidates that have top K scores $\mathcal{D} = \operatorname{TopK}\left(\mathcal{D}_N \setminus \mathcal{D}_0\right)$.

E.2.3 IMPLEMENTATION DETAILS

For the implementation of the GFlowNet-AL baseline model, we use the previously published implementation with slight changes, which follows the training setups of Jain et al. (2022):

Proxy model: We parameterize it as an MLP with two hidden layers, each having 2048 hidden units, and use ReLU activation. We also use ensembles of 5 models with same architecture for uncertainty estimation. For the acquisition function, we use UCB $(\mu + \kappa\sigma)$ with $\kappa = 0.1$. The proxy is trained with MSE loss using mini-batch of size 256 and Adam optimizer with $(\beta_0, \beta_1) = (0.9, 0.999)$ and learning rate 10^{-4}. During training, early stopping is also used by evaluating the validation set containing 10% of the data.

GFlowNet generator: We use an MLP with 2 hidden layers of 2048 hidden units each. The model is trained with trajectory balance objective as the main loss function, by using Adam optimizer with $(\beta_0, \beta_1) = (0.9, 0.999)$. Additionally, $\log Z$ is trained with a learning rate of 10^{-3} for AMP, TF Bind 8 task, and 5×10^{-3} for GFP task. Other hyper-parameters are shown in the following table:

Hyper-parameter	**AMP**	**TF Bind 8**	**GFP**
δ : Uniform Policy Coefficient	0.001	0.001	0.05
Learning rate	5×10^{-4}	10^{-5}	10^{-3}
m : Minibatch size	32	32	32
β : Reward Exponent $R(x)^{\beta}$	3	3	3
T : Training steps	10,000	5,000	50,000

Table 5: Hyper-parameters for the GFlowNet.

There are some changes in hyper-parameter choices and the number of active learning rounds in the TF Bind 8 task and the GFP task compared to the original training setups of Jain et al. (2022). However, during the experiment, we observed that these settings helped us get the closest results to the reported one in Jain et al. (2022).

Proposed OT regularization The regularization coefficients for Min OT, UB OT, and Max OT are the same for each biological sequence design task. Specifically, the coefficients for the AMP, TF Bind 8, and GFP task are 0.025, 0.1, and 0.02 correspondingly.

E.3 SYNTHETIC DISCRETE PROBABILISTIC MODELING TASKS

E.3.1 EVALUATION CRITERIA

To evaluate the performance, we keep the same evaluation criteria in Zhang et al. (2022), where they use the NLL of a large independent sample of ground truth data and the exponential Hamming MMD (Gretton et al. (2012)) between ground truth data and generated samples as performance metrics. To measure NNL and MMD, we use 10 fixed sets, and each set consists of 4000 ground truth data samples.

E.3.2 IMPLEMENTATION DETAILS

GFlowNet: For the implementation of the GFlowNet model, we use an MLP with 2 hidden layers of 512 dimensions each. The GFlowNet policy model, which includes both P_F and P_B, is trained with a learning rate of 0.001. We use a mini-batch size of 128 and $1e5$ training steps with the trajectory balance objective.

EBMs: For the implementation of the Energy-Based Model, we use an MLP with 3 hidden layers of 256 dimensions each. The learning rate is 0.001.

Proposed OT regularization: The regularization coefficient is 0.001 for both Min OT and UB OT and is the same for all tasks.

F ADDITIONAL EXPERIMENT RESULTS

F.1 ABLATION STUDY ABOUT VARYING λ

Specifically, we will further investigate the proposed path regularization via OT with different values of the regularization coefficient λ in the 8-D hyper-grid environment in Section 4.1. In addition, the regularization coefficient is selected from the set $(0.001, 0.01, 0.1, 0.4)$. We plot the mean results over 10 runs for each configuration in Fig. 3.

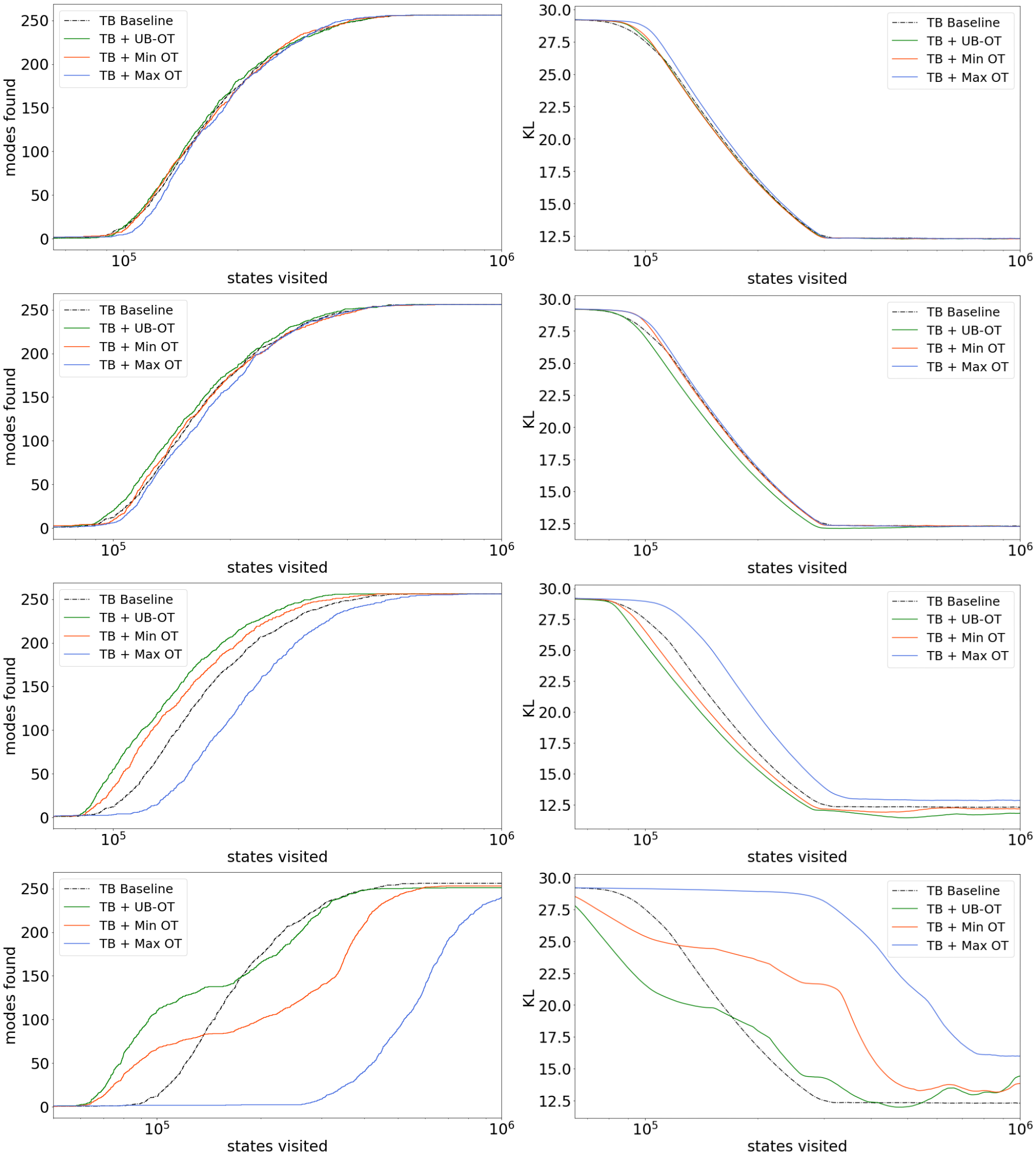

Figure 3: Results on the $8-D$ hyper-grid environment with $\lambda \in (0.001, 0.01, 0.1, 0.4)$ (from top to bottom). Left: Number of modes found during training. Right: KL divergence between the true and empirical distribution.

Note that the good range of values for the regularization coefficient is observed to highly depend on the specific setting of the experiment task. Here, we can see that when λ is relatively small, such as $\lambda \in (0.001, 0.01)$, the performance of GFlowNets trained additionally with our proposed regularization via OT does not seem to be significantly different from the baseline model's performance, which holds for both UB OT, Min OT, and Max OT. This may be resulted from the small contribution of regularization to the regularized training objective, which is caused by the not large enough value of λ. Specifically, when $\lambda = 0.01$, we can still see that the performance of GFlowNets trained by minimizing the upper bound is slightly better than the baseline's result.

In addition, when λ is relatively large ($\lambda = 0.4$), the result of learned GFlowNets is even worse than the baseline, which may be due to the large value of λ forces the model's learning focus on the regularization part more than necessary, which badly affects the optimization of the main training objective (trajectory balance objective). Specifically, this can be observed in the lower KL divergence of both UB OT, Min OT, and Max OT compared to the baseline.

Meanwhile, when $\lambda = 0.1$, GFlowNets trained by minimizing the OT regularization and its upper bound clearly perform better than the baselines regarding the number of modes found and KL divergence between the actual and empirical distribution, which proves our motivation that minimizing the proposed path regularization is more beneficial in this circumstances.

F.2 ADDITIONAL RESULTS OF THE HYPERGRID ENVIRONMENT

We also plot the mean results over 10 runs for each configuration with variance in Fig. 4.

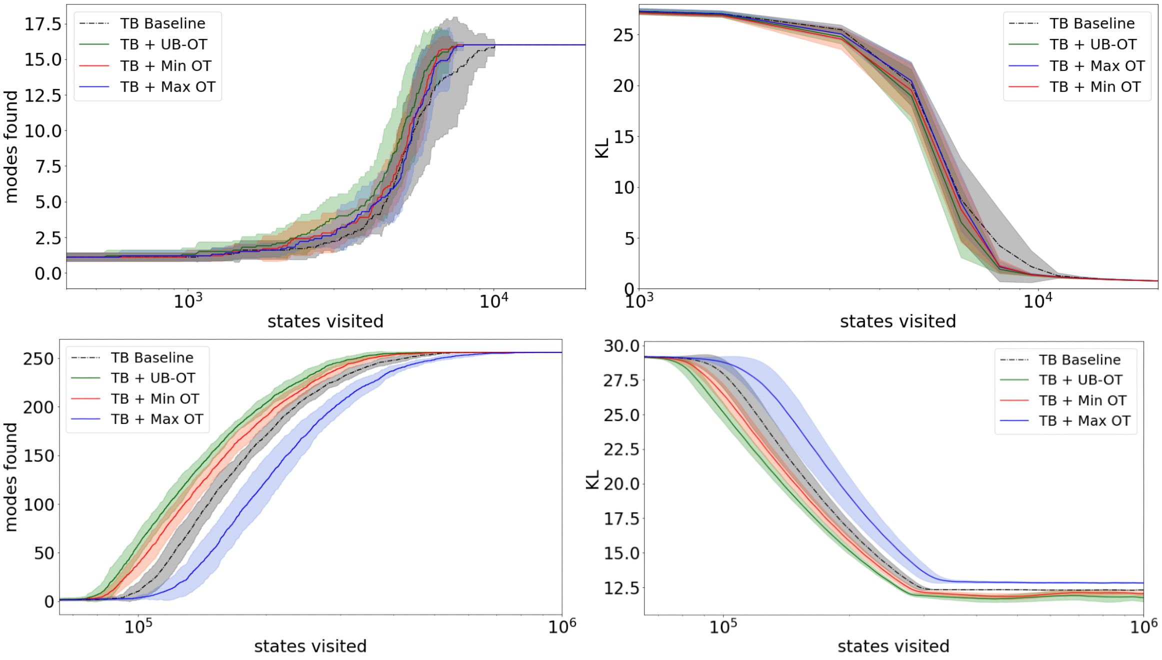

Figure 4: Results with variance on the $4-D$ (upper) and $8-D$ (lower) hyper-grid environment. Left: Number of modes found during training. Right: KL divergence between the true and empirical distribution.