Evaluating GFlowNet from partial episodes for stable and flexible policy-based training

Puhua Niu

Department of Electrical and Computer Engineering, Texas A&M University

March 30, 2026

github.com/niupuhua1234/Sub-EB

Overview

Table of Contents

Motivation

1. Existing Generative Flow Networks (GFlowNets) training methods fall into two paradigms: value-based methods enforce flow balance on subtrajectories (partial episodes) for policy optimization.[1] In contrast, policy-based methods alternate between estimating the policy divergence and updating the policy.
2. These two approaches appear fundamentally different, and their connection remains unclear.
3. This work bridges these two perspectives by showing that flow balance naturally induces a principled policy evaluator for measuring policy divergence. Consequently, an evaluation balance objective defined over subtrajectories is proposed to effectively learn the evaluator.

[1] Yoshua Bengio et al. "Gflownet foundations". In: *Journal of Machine Learning Research* 24.210 (2023), pp. 1–55.

Background & Motivation

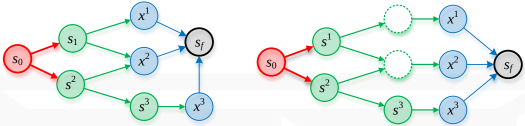

Figure 1: A graphical illustration of a DAG (left) and its graded version (right). Dotted circles represent dummy states, added during the conversion.

GFlowNets are generative models on the space of combinatorial objects $\mathcal{X}$. They describe the process of generating x as a trajectory $\tau = (s_0 \to \cdots \to s_h \to \cdots \to s_H \to s_f)$ along a **graded** Directed Acyclic Graph (DAG) $\mathcal{G} = (\mathcal{S}, \mathcal{E})$, whose topological ordering is indexed by $[H] := [0, \ldots, H]$.

1. The process starts at the unique initial state s_0 (the empty structure) and ends at the unique final state s_f.
2. Each transition $(s \to s')$ corresponds to adding a building component into the current state (structure) s.
3. Given the reward function $R(x)$, we expect τ to return the terminating state $s_H := x$ (a complete structure) with probability $P^*(x) = R(x)/Z^*$, where $Z^* = \sum_x R(x)$ is assumed to be computationally infeasible.

Value-based training

◇ GFlowNet training aims to minimize $\mathcal{L}_{KL} := D_{KL}(P_F(\tau) \| P_B(\tau))$ with $P_F(\tau) := \prod_{i=0}^{H} \pi_F(s_{h+1}|s_h)$, $P_B(\tau) := \prod_{i=0}^{H} \pi_B(s_h|s_{h+1})$.

1. π_F and π_B are forward and backward *Markovian* policies (transition distributions), respectively, and $\pi_B(x|s_f) := R(x)/Z^*$.
2. $P_B(\tau|x)$ can be **freely chosen** as $P^*(x) = \sum_{\tau|x} P_B(\tau)$ always holds.
3. The optimal forward policy π_F^*, satisfying $P_{F^*}(\tau) = P_B(\tau)$ for all $\tau \in \mathcal{T}$, induces its marginal $P_{F^*}(x) = R(x)/Z^*$.

◇To avoid intractable Z^* in $\mathcal{L}_{KL}$, value-based methods enforce the balance of unnormalized trajectory distributions (flows), $F(\tau) := P_F(\tau)Z$ and $F^*(\tau) := P_B(\tau|x)R(x)$, where $F(s_0) := Z$ is the total flow of F.
◇ Based on this principle, the flow imbalance can be generalized and reformulated as the Sub-Trajectory Balance (Sub-TB) objective:[2]

$$\mathcal{L}_F(\theta) := \mathbb{E}_{P_{\mathcal{D}}(\tau)}\Big[\sum_{\tau_{i:j}} w_{j-i}\left(\delta_F(\tau_{i:j};\theta)\right)^2\Big], \quad \delta_F(\tau_{i:j};\theta) := \log \frac{P_F(\tau_{i:j}|s_i;\theta)F(s_i;\theta)}{P_B(\tau_{i:j}|s_j)F(s_j;\theta)},$$

where $F(s_f)P_B(x|s_f) := R(x)$, $F(s_i) := P(s_i)Z$ is the state flow, and w_{j-i} denotes the non-zero weight for subtrajectories with $j-i$ edges.

[2]Kanika Madan et al. "Learning GFlowNets from partial episodes for improved convergence and stability". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 23467–23483.

Table of Contents

Subtrajectory Evaluation Balance

◇ **Critic:** The critic (true evaluation function) $V^\dagger$ of actor π_F captures the policy divergences over subtrajectories, thereby facilitating the computation of $\mathcal{L}_{KL}$. Let $F^*(s) := P_B(s)Z^*$ be the optimal state flow. For any $h \in [H]$,

$$V^\dagger(s_h;\theta) : \log F^*(s_h) - D_{KL}(P_F(\tau_{h:}|s_h;\theta)\|P_B(\tau_{h:}|s_h)).$$

◇ **Actor:** To minimize $\mathcal{L}_{KL}(\theta)$, it is noted that $\nabla_\theta V^\dagger(s_0;\theta) = -\nabla_\theta \mathcal{L}_{KL}(\theta)$. Therefore, the policy gradient $\nabla_\theta V^\dagger(s_0;\theta)$ expressed in terms of $V^\dagger(s_h)$ provides a valid update rule for π_F.

◇ The Subtrajectory Evaluation Balance (Sub-EB) objective for learning $V(s;\phi)$, which approximates $V^\dagger(s)$, and $\pi_B(s|s';\phi)$ is defined as:

$$\mathcal{L}_V(\phi) := \mathbb{E}_{P_F(\tau)}\Big[\sum_{\tau_{i:j}} w_{j-i}\,(\delta_V(\tau_{i:j};\phi))^2\Big], \quad \delta_V(\tau_{i:j};\phi) = \log \frac{P_F(\tau_{i:j}|s_i)\exp V(s_i;\phi)}{P_B(\tau_{i:j}|s_j;\phi)\exp V(s_j;\phi)}$$

where $P_B(x|s_f)\exp V(s_f) := R(x)$ for $x \in \mathcal{X}$. While the traditional λ-Temporal-Difference (TD) objective operates on only edge-wise difference $\delta_V(s \to s')$, the Sub-EB objective enforces consistency over subtrajectories, thereby improving learning **stability**.

Theorem 1

Suppose V is an arbitrary evaluation function over $\mathcal{S}$. Given a forward policy π_F, $\forall h \in [H]: V(s_h) = V^\dagger(s_h)$ if and only if V satisfies $\mathcal{L}_V = 0$.

◇ **Critic:** We define $W^\dagger(s_0) := \log F(s_0)$. For any $h \in [H]\backslash\{0\}$,

$$W^\dagger(s_h;\phi) =: \log F(s_h) - D_{KL}(P_B(\tau_{:h}|s_h;\phi) \| P_F(\tau_{:h}|s_h)).$$

◇ **Actor:** Minimizing $\mathbb{E}_{P_{\mathcal{D}(x)}}[\mathcal{L}_{KL}(\phi|x)] := \mathbb{E}_{P_{\mathcal{D}(x)}}[(P_B(\tau|x;\phi)\|P_F(\tau|x))]$ reduces the gap between $P_B(\tau)$ and $P_F(\tau)$, where $P_{\mathcal{D}}(x)$ is induced by an **offline** data-collection policy $\pi_{\mathcal{D}}$. Moreover, since $-\nabla_\phi \mathbb{E}_{P_{\mathcal{D}(x)}}[W^\dagger(x;\phi)] = \nabla_\phi \mathbb{E}_{P_{\mathcal{D}(x)}}[\mathcal{L}_{KL}(\phi|x)]$, the expected policy gradient $\nabla_\phi \mathbb{E}_{P_{\mathcal{D}(x)}}[W^\dagger(x;\phi)]$ expressed in terms of $W^\dagger(s_h)$ provides a valid update rules for π_B.

◇ We present the backward Sub-EB objective for jointly learning $W(s;\theta)$ that approximates $W^\dagger(s)$, and $\pi_F(s'|s;\theta)$ as follows:

$$\mathcal{L}_W(\theta) := \mathbb{E}_{P_B^{\mathcal{D}}(\tau)}\Big[\sum_{\tau_{i:j}} w_{j-i}\,(\delta_W(\tau_{i:j};\theta))^2\Big], \quad \delta_W(\tau_{i:j};\theta) = \log \frac{P_F(\tau_{i:j}|s_i;\theta)\exp W(s_i;\theta)}{P_B(\tau_{i:j}|s_j)\exp W(s_j;\theta)}$$

where $P_B(x|s_f)\exp W(s_f) := R(x)$ and $P_B^{\mathcal{D}}(\tau) := P_B(\tau|x)P_{\mathcal{D}}(x)$. The backward Sub-EB objective enables the integration of an offline sampler $\pi_{\mathcal{D}}$, thereby increasing the **flexibility** of policy-based training.

Theorem 2

Suppose W is an arbitrary evaluation function over $\mathcal{S}\backslash\{s_f\}$. Given a backward policy π_B, $\forall h \in [H-1]: W(s_{h+1}) = W^\dagger(s_{h+1})$ and $W(x) = \log R(x)$ if and only if W satisfies $\mathcal{L}_W = 0$.

Algorithm 1 Online Policy-based Workflow

Require: $\pi_F(\cdot\,|\cdot\,;\theta)$, $\pi_B(\cdot\,|\cdot\,;\phi)$, $V(\cdot\,;\phi)$, batch size K, number of total iterations N

for $n = 1, \ldots, N$ **do**

$\mathcal{D} \leftarrow \{\tau^k : \tau^k \sim P_F(\tau)\}_{k=1}^K$

Based on $\mathcal{D}$, update ϕ by $\nabla_\phi \widehat{\mathcal{L}}_V(\phi)$

Based on $\mathcal{D}$ and V, update θ by $-\widehat{\nabla}_\theta V^\dagger(s_0;\theta)$

end for

Algorithm 2 Offline Policy-based Workflow

Require: $\pi_B(\cdot\,|\cdot\,;\phi)$, $\pi_F(\cdot\,|\cdot\,;\theta)$, $W(\cdot\,;\theta)$, $\pi_{\mathcal{D}}$, K, N

for $n = 1, \ldots, N$ **do**

$\mathcal{D}^\top \leftarrow \{x^k : x^k \in \tau^k, \tau^k \sim P_{\mathcal{D}}(\tau)\}_{k=1}^K$

$\mathcal{D} \leftarrow \{\tau^k : x^k \in \mathcal{D}^\top, \tau^k \sim P_B(\tau|x^k)\}_{k=1}^K$

Based on $\mathcal{D}$, update θ by $\nabla_\theta \widehat{\mathcal{L}}_W(\theta)$

Based on $\mathcal{D}$ and W, update ϕ by $-\widehat{\nabla}_\phi \mathbb{E}_{P_{\mathcal{D}}(x)}[W^\dagger(x;\phi)]$

end for

Table of Contents

Experiment

$\diamond$ We consider two policy-based methods that learn V by either the λ-TD objective or the proposed Sub-EB objective, denoted RL and Sub-EB (Alg.1). We also include Sub-TB as representative baselines for value-based methods. By default, $\pi_{\mathcal{D}} = (1-\alpha)\pi_F + \alpha \mathrm{Uniform}(0,1)$, and π_B is an uniform policy. The weight w_{j-i} for $i < j \in [H+1]$ is set to $\lambda^{j-i} / \sum_{i<j\in[H+1]} \lambda^{j-i}$

Hypergrid

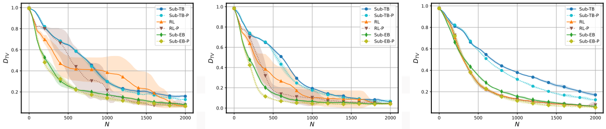

Figure 2: Plots of the means and standard deviations (represented by the shaded area) of D_{TV} for different training methods with parameterized π_B and uniform π_B on the 256×256 (left) and 128×128 (middle) and $64 \times 64 \times 64$ (right) grids, based on five randomly started runs for each method. **By default**, metric values are recorded every 20 iterations over $N = 2000$ training iterations and smoothed by a sliding window of length 5.

$\diamond$ States are the coordinate tuples of an D-dimensional hyper-cubic grid of height H. Starting from $s_0 = (0, \ldots, 0)$, actions correspond to increasing one of D coordinates by 1 for the current state or stopping the process at the current state and designating it as the terminating state x. We perform experiments on 256×256, $128 \times 128 \times 128$ and $64 \times 64 \times 64$ grids, and compute the exact $D_{\text{TV}} := |\mathcal{X}|^{-1} \sum_x |P_F(x) - P^*(x)|$. In addition to the uniform π_B, we also consider parameterized backward policies π_B, and denote the corresponding variants as Sub-EB-P, Sub-TB-P, and RL-P.

Bayesian network structure learning

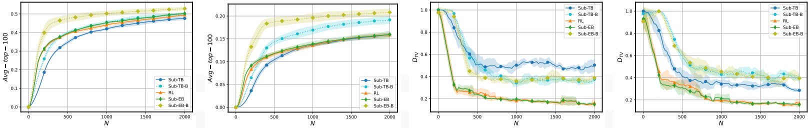

Figure 3: Plots of the mean and standard deviation values (represented by the shaded area) of average reward (left) of the top 100 unique candidate graphs and approximated D_{TV} (right) over 10 and 15 nodes, based on five randomly started runs for each method.

$\diamond$ The object space $\mathcal{X}$ corresponds to the space of Bayesian Network structures. The generative process begins with an empty graph, and at each step, a valid edge is added to the graph or terminates the process, yielding the current graph as a generated BN structure. We consider the cases of 10 and 15 nodes, where the sizes of $\mathcal{X}$ are approximately 4.18×10^{18} and 2.38×10^{35}. We report the average reward of the top 100 unique graphs that are discovered during the training process, and the samples-based approximation of the total variation D_{TV}. Besides Sub-EB and Sub-TB, we further consider the variants Sub-EB-B (Alg. 2) and Sub-TB-B, which employ a $\pi_{\mathcal{D}}$ designed to explicitly promote exploration of high-reward states.

Table of Contents

Conclusions & Future Work

Conclusions & Future Work:

- In this work, we propose the Sub-EB objective, which adopts a balance-style formulation for learning the evaluation function $V(s)$, enabling stable and flexible policy-based training.
- In principle, the Sub-EB objective goes beyond the basic policy-based methods. Integrating the Sub-EB objective into more advanced policy-based methods is left for future work.

Acknowledgements: This work was supported in part by the U.S. National Science Foundation (NSF) grants SHF-2215573 and IIS-2212419. Portions of this research were conducted with the advanced computing resources provided by Texas A&M High Performance Research Computing.

Thank You!