DATA FIRST, REWARDS LATER: GFLOWNETS ACROSS PRE-TRAINING AND POST-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Discrete diffusion models and Generative Flow Networks (GFlowNets) are two seemingly alternative approaches to modelling discrete data. They differ in terminology and parametrisation, but most critically in how they are trained. While discrete diffusion models are primarily trained from data, GFlowNets are primarily trained from rewards. We show that this separation is not fundamental: both model classes admit a common flow–score formulation that allows training objectives to transfer between them. Building on this connection, we propose a pre-training/post-training regime analogous to the typical LLM training pipeline. A GFlowNet can first be pre-trained directly from data, without an explicit reward model, and subsequently post-trained with standard reward-based GFlowNet objectives without changing the underlying generative model. This combines the flexible construction spaces and structured generation native to GFlowNets with efficient learning from existing data characteristic of discrete diffusion models. On peptide generation, data-based pretraining substantially improves subsequent reward-query efficiency under identical post-training, reaching 88.6% high-reward samples compared with 73.5% when training from untrained initialization.

1 INTRODUCTION

Generative Flow Networks (GFlowNets) (Bengio et al., 2021; 2023) are generative models that learn a stochastic policy for sampling compositional objects with probability proportional to a given reward function. GFlowNets have found applications in a variety of domains that require samples that are both diverse and high-quality, including molecular design (Bengio et al., 2021), biological sequence design (Jain et al., 2022), and combinatorial optimization (Zhang et al., 2023b).

Discrete diffusion methods, in contrast, extend the notion of score matching (Hyvärinen, 2005; Song & Ermon, 2019) from continuous spaces (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021) to discrete spaces (Meng et al., 2022). In particular, the *Concrete Score* (Meng et al., 2022) has been proposed as an extension of the Stein score. The concrete score quantifies the rate of change in probabilities with relation to local directional changes.

As defined, GFlowNets and Discrete diffusion appear to be rather disparate approaches. Some previous work by Zhang et al. (2023a) has explored the connection of GFlowNets to other generative models—including diffusion—at a high level, focusing mostly on overall structure and (continuous-time) training. In this work, we go beyond this perspective and demonstrate that a natural consequence of this correspondence enables improved training pipelines for GFlowNets.

Recent work on discrete diffusion has revealed that although the models are parametrized with an explicit time dimension, masked diffusion models can be equivalently defined as inherently time-agnostic (Zheng et al., 2025; Ou et al., 2025). We explore a fundamental connection between the two, showing that both models rely on learning the same underlying object: conditional distributions over clean data. This perspective reveals a fundamental structural equivalence between discrete scores and GFlowNets.

Based on this connection, we introduce a score formulation for GFlowNets, defined analogously to discrete scores in diffusion (Lou et al., 2024). We show that standard GFlowNet objectives can be expressed in terms of these local flow ratios, enabling the transfer of data-driven score-learning objectives to GFlowNets. This gives a practical bridge between two complementary training regimes.

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.

		Diffusion	GFlowNet
Concrete Score for GFlowNets	Theorem 3.1	Forward Process $q(x_t\|x_{t-1})$	Backward policy $P_B(x_{t-1}\|x_t)$
Score matching inherent to training			
Losses as score-based objectives		Reverse Process $p(x_{t-1}\|x_t)$	Forward Policy $P_F(x_t\|x_{t-1})$
Masked Diffusion and GFlowNets			
Structural similarities	Section 3.2	State Space	
Time agnosticity	Section 3.3	Layered by time	Acyclicity
GFlowNet Score Entropy	Section 4.1	Marginal Probabilities $p_t(x_t)$	Flows $F(x_t) \propto p_t(s_t)$
Denoising score entropy	Equation (8)		
Transition-sampled score entropy	Equation (9)	Support of $q(\cdot\|x)$, $p(\cdot\|x)$	Child and Parent nodes

Figure 1: Main contributions of this work (left) unravel fundamental connections between discrete diffusion and GFlowNets, beyond the high-level similarity noted by Zhang et al. (2023a) (right).

Data provides normalised samples from the target distribution without requiring explicit reward values or estimation of their global scale, while reward-based training can directly optimise beyond the empirical data distribution. We derive two objectives, $\mathcal{L}_{\mathrm{DSE}}$ and a transition-sampled variant $\mathcal{L}_{\mathrm{DSE}'}$, for pre-training a GFlowNet directly from data under a fixed backward policy, without a proxy reward or energy model. The resulting model remains a standard GFlowNet and can subsequently be post-trained with existing reward-based objectives. On antimicrobial-peptide generation, this data-first initialization substantially improves reward-query efficiency under identical Replay-TB post-training. We summarize our key contributions in Figure 1.

2 Background

Discrete generative modelling techniques exist in a variety of formalisms and parametrisations. At a fundamental level, however, we can view them as (discrete) *transport problems* from a source distribution to a target distribution $\mathcal{D}$ in some discrete space $\mathcal{S}$, where we denote the support of $\mathcal{D}$ as the *terminal states* $\mathcal{X} \subset \mathcal{S}$. We are given implicit access to $\mathcal{D}$ either via a dataset drawn from $\mathcal{D}$ or an unnormalized black-box reward function $R : \mathcal{X} \to \mathbb{R}_{\geq 0}$ such that $\mathcal{D}(x) \propto R(x)$. We can then formally define a wide class of discrete models as learning a *path measure* over $\mathcal{S}$. We now introduce the two model classes and their relevant concepts in more detail.

2.1 Generative Flow Networks

Generative Flow Networks (GflowNets, or GFNs) (Bengio et al., 2023) are conceptualized to sample proportional to a given reward function $R : \mathcal{X} \to \mathbb{R}_{\geq 0}$. The sampling process considers a directed acyclic construction graph (DAG) $\mathcal{G} = (\mathcal{S}, \mathcal{A})$, where the arcs $(s \to s') \in \mathcal{A}$ denote valid state transitions. Starting from a universal source s_0, GFNs sample *trajectories* $\tau = (s_0, s_1, \ldots, s_n = x) \in \mathcal{T}$. We refer to the *children* or *neighborhood* of a state s as $\mathrm{N}(s) \coloneqq \{s' \in \mathcal{S} : (s, s') \in \mathcal{A}\}$, and its parents as $\mathrm{Par}(s) \coloneqq \{s' \in \mathcal{S} : (s', s) \in \mathcal{A}\}$. Denoting $Z = \sum_{x \in \mathcal{X}} R(x)$, the formal objective of a GFN is then to assign transition probabilities, or a *forward policy* P_F such that

$$\Pr_{P_F}(x) \coloneqq \sum_{\tau \in \mathcal{T} :\, s_n = x} \prod_{(s,s') \in \tau} P_F(s' \mid s) = R(x)/Z.$$

The transition likelihoods induce a conservative *flow function* $\forall s \in \mathcal{S} : F(s) \coloneqq Z \Pr_{P_F}(s)$ over $\mathcal{G}$, as well as a corresponding backward policy P_B.

Depending on the training objective, GFNs are often parametrised by the partition function Z and the forward and backward policies P_F and P_B. Existing objectives enforce flow consistency at different granularities, including *flow matching* over individual states, *trajectory balance* (Malkin et al., 2022), *sub-trajectory balance* (Madan et al., 2023), and related objectives (Richter et al., 2020). A detailed overview of these learning objectives is provided in Section C.1. All these objectives aim to recover a balanced flow whose terminal values match the rewards, $F(x) = R(x)$.

2.2 DISCRETE DIFFUSION

In contrast to GFlowNets, discrete diffusion models typically learn from a fixed dataset $D \sim \mathcal{D}$. In the diffusion literature, clean samples are commonly denoted $x_0 \in \mathcal{X}$, where the subscript indicates the number of noising steps. The noisy training data $x_{1:T}$ is then obtained via explicit corruption with $q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1})$. The goal is to learn a reverse Markov process $p(x_{t-1} \mid x_t)$ from a known prior distribution $p(x_T)$ to clean data, such that the learned marginal distribution p_θ approximates $\mathcal{D}$, where

$$p(x_0) \coloneqq \sum_{x_{1:T}} p(x_T) \prod_{t=1}^{T} p(x_{t-1} \mid x_t).$$

In continuous-time settings of discrete diffusion Campbell et al. (2022), diffusion matrices $Q_t(x, y)$ are used to define the rate of transition from a state x to y, The diffusion matrices govern the evolution of a family of probability distributions indexed by time $[p_t]_{t\geq 0}$. The reversal of the process defined by the diffusion matrix is given by $\tilde{Q}_t(x, y) \coloneqq \frac{p_t(y)}{p_t(x)} Q_t(y, x)$.

More recent approaches to discrete diffusion generalize the notion of score functions to discrete settings, defining corresponding losses that minimize expected score discrepancy. The *concrete score* (Meng et al., 2022) and *score entropy* (Lou et al., 2024) are two prominent instances of discrete scores, which we discuss in detail in the next subsection.

2.3 RELATED WORK

Unification of GFlowNets and diffusion models: In Zhang et al. (2023a), the GFlowNet was explored as a unifying framework for generative modeling. It is proposed, that the discrete denoised diffusion model (Austin et al., 2021; Hoogeboom et al., 2021) can be considered to be a special case of a GFlowNet. In such a framework, the forward policy of the GFlowNet is defined to be the learned denoising process, and the backward process is set to the noising process:

$$P_F(s_{t+1} \mid s_t) = \mathcal{C}(s_{t+1}; \mathbf{p} = \mu_i(s_i)), \quad P_B(s_t \mid s_{t+1}) = \mathcal{C}(s_t; \mathbf{p} = s_{t+1}\mathbf{Q}_t), \tag{1}$$

The concrete score for discrete data: In Meng et al. (2022) the concrete score is proposed as the generalization of the Stein score in discrete spaces. The concrete score is defined for the neighborhood-induced graph. The Neighborhood-induced graph consists of a set of nodes that the neighbor-mapping function N maps any given state to.

The concrete score is defined as the rate of probabilities related to a directional change within the neighborhood induced graph. Meng et al. (2022) propose the following loss objective called *Concrete Score Matching* for learning the score.

$$\mathcal{L}_{\text{CSM}}(\theta) = \mathbb{E}_{x\sim\mathcal{D}} \|c_\theta(x) - c_{\mathcal{D}}(x)\|_2^2 \qquad \text{where} \quad c_{\mathcal{D}}(x) \coloneqq [\Pr(y)/\Pr(x) - 1]_{y\in N(x)}\,. \tag{2}$$

Score Entropy Discrete Diffusion models (SEDD): In Lou et al. (2024), score matching is extended to discrete spaces for diffusion through the introduction of Score Entropy Discrete Diffusion models (SEDD). The following denoising score entropy loss is introduced for learning the concrete score, parametrizing the reverse discrete diffusion process:

$$\int_0^T \mathbb{E}_{x_t\sim p_{t|0}(x_t|x_0)} \sum_{y_t \neq x_t} Q_t(y_t, x_t) \left(c_\theta(x_t, t)_{y_t} - \frac{p_{t|0}(y_t \mid x_0)}{p_{t|0}(x_t \mid x_0)} \log c_\theta(x_t, t)_{y_t} + C \right) dt$$

where $Q_t(\cdot, \cdot)$ corresponds to a diffusion matrix, $C \coloneqq K\left(\frac{p_{t|0}(y_t|x_0)}{p_{t|0}(x_t|x_0)}\right)$ and $K(a) \coloneqq a \log a - a$.

Data-based and reward-based training. Reward-based GFlowNet learning is closely related to entropy-regularized reinforcement learning: for a fixed backward policy, the GFlowNet target can be formulated as the solution of an entropy-regularized control problem (Tiapkin et al., 2024). In contrast, data-based generative modelling learns a normalized target distribution from samples, typically through likelihood- or score-based objectives. These therefore provide two different supervision interfaces to the same underlying generative problem: samples from $\mathcal{D}$ on one hand, and pointwise evaluations of an unnormalized $R(x) \propto \mathcal{D}(x)$ on the other.

Previous work has connected GFlowNets to data-based learning by jointly learning an energy model and a GFlowNet sampler (Zhang et al., 2022). Our setting instead keeps the GFlowNet itself as the generative model and learns its local flow ratios directly from data. This makes it possible to pre-train from samples and subsequently continue training the same model with standard reward-based GFlowNet objectives.

3 Equivalence between Generative Flows and Discrete Scores

In this section, we thoroughly and constructively describe a functional correspondence between discrete diffusion literature and GFlowNets, as well as how concepts transfer between these domains. Zhang et al. (2023a) proposed, that discrete diffusion models could be considered special formulations of GFlowNets. In recent literature Carter et al. (2026), the connection was further elaborated on for masked diffusion, yet, to our knowledge, this connection has been merely implied rather than explained in detail. We now provide an explicit mapping between these two domains, enabling the transfer of techniques and learning objectives between model types.

3.1 Correspondence between Domain-Specific Concepts

We first discuss how concepts—and the surrounding assumptions—discussed in discrete diffusion literature and GFlowNet literature transfer. We assume that, in both settings, we aim to learn a sampler for the same distribution $\mathcal{D}$ over the same terminal space $\mathcal{X}$. We are also given a fixed initial (dummy) state s_0 or a fixed and known prior distribution $p(x_T)$ over some states $\mathcal{S}_0$. To map iteratively from terminal states to initial states, we use the *backward policy/noising process* $P_B \leftrightarrow q$, and conversely, try to learn the corresponding *forward policy/denoising process* $P_F \leftrightarrow p$.

The idea of concrete score matching is to construct a surrogate for the gradient in discrete spaces by leveraging local directional changes to the input. This is done by exploiting special neighborhood structures in the data. We propose that *together, the forward and backward policy induce a score function* as in Meng et al. (2022). We further argue, that score matching is inherent to all GFlowNet training objectives.

In the corresponding domains, the forward policy is induced either via the flow F_θ or the concrete score c_θ. We can now define their canonical correspondence

Definition 3.1 (GFlowNet Score)**.** Let $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ be a GFlowNet state-space with the forward policy $P_F(s' \mid s) = F(s \to s')/F(s)$ and the backward policy $P_B(s \mid s') = F(s \to s')/F(s')$. We define the neighborhood of a state $N(x)$ as the set of its children. We then define the *concrete GFlowNet score* $c_F \in \mathbb{R}_{\geq 0}^{|N(s)|}$ as

$$c_F(s) \coloneqq \left[\frac{P_F(s' \mid s)}{P_B(s \mid s')}\right]_{s' \in N(s)} = \left[\frac{F(s')}{F(s)}\right]_{s' \in N(s)} = c_{\mathcal{D}}(s) + 1, \tag{3}$$

where $c_{\mathcal{D}}(s)$ is the concrete score as defined in Equation (2) and (Zhang et al., 2025). With slight abuse of notation, we will denote the score over one arc (s, s') with $c_F(s, s') = \frac{F(s')}{F(s)}$.

The GFlowNet score c_F has a direct correspondence to indicators of the direction of greatest ascent. The score is directly proportional to the flow through the state it points to. Although the flow through a state F does not directly correspond to a probability distribution over states, it is proportional to the probability of a trajectory visiting the given state.

Theorem 3.2. *For a GFlowNet state-space $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ with a consistent score function $c_F(s, s') = \frac{F(s')}{F(s)}$ and a consistent backward policy $P_B(s \mid s')$ uniquely identify a valid flow F up to a multiplicative constant $Z > 0$.*

We further show, that the GFlowNet score can be utilized in the regular learning objectives and that further, the notion of score matching is inherent to training. Details can be found in the Appendix.

Proposition 3.3 (GFlowNet Training Objectives via score matching)**.** *Let $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ be the state-space of a GFlowNet with the flow function F and the concrete GFlowNet score c_F, as defined in Theorem 3.1, such that $\sum_{s' \in N(s)} P_B(s|s') c_{F\theta}(s, s') = 1$. Then we can equivalently express all GFlowNet objectives in terms of c_F as shown in Table 1.*

3.2 Masked Diffusion and GFlowNets

We propose, that there exists a fundamental parallel between the GFlowNet and the Masked Diffusion model for discrete data.

The purpose of the GFlowNet is to learn to iteratively construct compositional objects, which can equivalently be viewed as the removal of mask tokens. The iterative deconstruction of objects by the backward policy then corresponds directly to a masking process. Conversely, the support of an absorbing masking process induces a DAG structure corresponding to that underlying the GFlowNet. As is stated in Zheng et al. (2025), whenever $x_t^i \neq \mathbf{M}$, and $\hat{x}_t^i \neq x_t^i$, we have $Q^{absorb}(\hat{x}_t^i, x_t^i) = 0$. For $\hat{x}_t^i = x_t^i$, the score evaluates to 1, leaving only the transformations with Hamming-distance 1 to be evaluated. The set of relevant transformations then yields a GFlowNet, where each action corresponds to removing a mask-token. Here, we are using the parametrization of a continuous-time process - for a discrete-time process the correspondence is similar and equivalent in logic.

A notable distinction between the two models, however, arises from the diffusion models' seemingly explicit dependence on time. The GFlowNets aren't necessarily indexed by an absolute dimension of time. However, they also rely on an implicit time dimension, given by the strict partial relation induced on the states by the acyclicness property. Trajectories are not allowed to contain cycles, implying that there exists a dimension of time, that imposes a notion of direction within the model. In the following, we further analyze the time dimensionality of the two models.

3.3 Dependence on Time

Ou et al. (2025) argue, that the quantity implicitly modeled by absorbing state discrete diffusion corresponds to the conditional distribution of clean data. This observation follows from the following decomposition of the absorbing state diffusion score to the conditional distribution at $t = 0$ multiplied by an analytic time-dependent term. We denote here by x a sequence of unmasked tokens, and by $\hat{x}$ a single unmasked token not in x. Further, $y := (x, \hat{x})$.

$$\frac{p_t(y)}{p_t(x)} = \frac{e^{-\sigma(t)}}{1 - e^{-\sigma(t)}} p_0(\hat{x} \mid x) \tag{4}$$

An analogous decomposition appears in GFlowNets. The flow function $F : S \to \mathbb{R}^+$, induces a probability measure over trajectories and terminal states, corresponding to clean data. The probability assigned to an intermediate state is the probability that a trajectory sampled from the reward distribution visits it:

$$\forall s \in S : \quad P(s) = \sum_{s \in \tau : \tau \in \mathcal{T}} P(\tau) = \sum_{x_T} P_B(s \mid x_T) P_T(x_T).$$

To further elaborate on the similarities between the two models, we explore the flow function from the perspective of the previously defined score function. Assuming all trajectories have length d and the backward policy is set to uniformly mask one token at each step, the derivation provided in the Appendix gives

$$\frac{P(s_{t+1})}{P(s_t)} = \frac{t+1}{T-t} P_T(s_{t+1}|s_t), \quad P_F(s_{t+1} \mid s_t) = \frac{1}{(T-t)} P_T(s_{t+1}|s_t) \tag{5}$$

The discrete GFlowNet learns a time-agnostic deconstruction of the reward distribution, as does the absorbing state discrete diffusion model. Both model families take the original data distribution and learn the underlying conditional probabilities augmented by a weight with an analytical form depending on the steps or time taken, differing in the way that the deconstruction process is formulated. Both models learn the same information in a different format.

4 Transfer of Objectives between GFlowNets and Discrete Diffusion Models

In this section, we use the equivalence between GFlowNets and discrete diffusion to show how learning objectives naturally transfer.

Objective	Flow Condition	Score Condition
$s \in \mathcal{S} : \mathcal{L}_{\text{FM}}(s)$	$F(s) = \sum_{s' \in \text{N}(s)} F(s \to s') = \sum_{s' \in \text{Par}(s)} F(s' \to s)$	$\sum_{s' \in \text{N}(s)} c_F(s, s') P_B(s \mid s') = 1$ $\sum_{s' \in \text{Par}(s)} P_B(s' \mid s) = 1$
$(s, s') \in \mathcal{A} : \mathcal{L}_{\text{DB}}(s, s')$	$P_B(s \mid s')F(s') = P_F(s' \mid s)F(s)$	$c_F(s, s') = \frac{F(s')}{F(s)}$
$\tau = (s_0, \ldots, s_n) \in \mathcal{T} : \mathcal{L}_{\text{TB}}$	$\prod_{t=0}^{n-1} \frac{P_F(s_{t+1} \mid s_t)}{P_B(s_t \mid s_{t+1})} = \frac{R(s_n)}{Z}$	$\prod_{t=0}^{n-1} c_F(s_t, s_{t+1}) = \frac{R(s_n)}{Z}$
$\tau_{m:k} = (s_m, \ldots, s_k) : \mathcal{L}_{\text{sub-TB}}$	$\prod_{t=m}^{k-1} \frac{P_F(s_{t+1} \mid s_t)}{P_B(s_t \mid s_{t+1})} = \frac{F(s_k)}{F(s_m)}$	$\prod_{t=m}^{k-1} c_F(s_t, s_{t+1}) = \frac{F(s_k)}{F(s_m)}$

Table 1: Reward-based consistency conditions for GFlowNets under the score convention $c_F(s, s') = P_F(s' \mid s)/P_B(s \mid s') = F(s')/F(s)$, when enforcing $\forall x \in \mathcal{X} : F(x) = R(x)$.

4.1 Training GFlowNets with Data

In the previous section, we proposed a score formulation for GFlowNets and transferred their reward-based learning objectives to this parametrization. We also showed that GFlowNets and absorbing discrete diffusion models are equivalent in this formulation. In this section, we utilize this formalism to unify the learning objectives for GFlowNets, especially DB, with the purely data-based objectives used in discrete diffusion. To connect these settings, we first clearly state our objective. When we have access to an unnormalized reward $R(x) \propto \mathcal{D}(x)$, as well as a consistent flow function F, then $\forall s \in \mathcal{S} : F(s) \propto \Pr_{\tau \sim D \# P_B}\{s \in \tau\} \coloneqq P_B(s)$, i.e., the flow value of s is proportional to our terminal distribution $\mathcal{D}$ under our backward policy P_B. Thus $\mathcal{D}\#P_B$ define the normalized target of a consistent flow, and we can use this observation, combined with Theorem 3.1 to apply standard discrete diffusion objective functions to learn from data. In principle, we can learn with the flow formulation directly, but we motivate why this does not work well in practice.

Proposition 4.1 (Vertex Flow Marginalization with data)**.** *Let $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ be the state-space of a gflownet, $\mathcal{D}$ a data distribution and P_B be a fixed backward policy. For F_θ and P_B pointwise strictly positive,*

$$\mathcal{L}_{DFM}(s) = \left[F_\theta(s) - Z_\theta \mathbb{E}_{x \sim \mathcal{D}} P_B(s \mid x)\right]^2 \tag{6}$$

results in a balanced GFlowNet with $F_\theta(x) \propto \mathcal{D}(x)$.

However, Theorem 4.1 is a hard learning target for two reasons. First, close to terminal states, we have weak supervision from our dataset, and crucially, without explicit enumeration of unreachable states $\forall x \notin D \sim \mathcal{D}^n$ and even $\forall s \in \mathcal{S} : P_B(s \mid D) = 0$, it holds that $\mathcal{L}_{\text{DFM}}(F_\theta(s))$ is not optimized, we cannot utilize the local geometry of $\mathcal{G}$. Furthermore, as vertex flows are an energy field, we implicitly need to learn the global partition function $Z_\theta = F_\theta(s_0)$, which is notoriously hard.

Consequently, we propose the GFlowNet score from Theorem 3.1 as a well-behaved learning target for learning from data. In our score estimation, the partition functions cancel, but we still define a flow function F which is unique up to scale, unlike learning unrestricted edge-flows.

A straight-forward score-based objective is to minimize $\mathcal{L}_{\text{DB}}$ over $\mathcal{D}\#P_B$, yielding

$$\mathcal{L}_{\text{GSM}}(s, s') = \left(c_{F\theta}(s, s') - \frac{\mathbb{E}_{x \sim \mathcal{D}} P_B(s' \mid x)}{\mathbb{E}_{x \sim \mathcal{D}} P_B(s \mid x)}\right)^2 = \left(c_{F\theta}(s, s') - \frac{\Pr_{\tau \sim D \# P_B}\{s' \in \tau\}}{\Pr_{\tau \sim D \# P_B}\{s \in \tau\}}\right)^2. \tag{7}$$

However, this objective does not yield a straight-forward learning algorithm, as we would need to estimate visitation probabilities for s and s' over independent trajectories. Variance in these estimates causes the objective to become unstable. A much stabler optimization target based on Score Entropy and analogous to SEDD, instead tries to find $c_{F\theta} \geq 0$ such that

$$\arg\min_\theta : c_{F\theta}(s, s') - \frac{\Pr\{s' \in \tau\}}{\Pr\{s \in \tau\}} \log c_{F\theta}(s, s') = \arg\min_\theta : \mathcal{L}_{\text{GSM}}(s, s').$$

This objective now admits decomposition into loss functions over sampled or generated states.

Theorem 4.2 (Score Entropy for GFlowNets). *Let $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ be the state-space of a gflownet, $\mathcal{D}$ a data distribution and P_B be a fixed backward policy. For $c_{F\theta}$ and P_B pointwise strictly positive, and let $N(s, s_n) \coloneqq \{s' \in N(s) : \exists \tau : s', s_n \in \tau\}$, define the* GFlowNet Score Entropy Loss *as*

$$\mathcal{L}_{DSE} = \mathbb{E}_{\tau \sim \mathcal{D}\# P_B} \sum_{s \in \tau} \left[\sum_{s' \in N(s)} c_{F\theta}(s, s') - \sum_{s' \in N(s, s_n)} \frac{P_B(s' \mid s_n)}{P_B(s \mid s_n)} \log c_{F\theta}(s, s') \right]. \tag{8}$$

Then, $\mathcal{L}_{DSE}$ is a strictly proper loss function with the minimizer $c^*_{F\theta}(s, s') = \frac{\Pr(s')}{\Pr(s)}$.

The proof is provided in Section A.1.

In settings where $P_B(s \mid x)$ is known for all $(s, x) \in \mathcal{S} \times \mathcal{X}$, we can now optimize this loss directly. When $P_B(s \mid x)$ is not known analytically, we can reorganize terms to estimate it implicitly:

Corollary 4.3 (Transition-sampled Score Entropy). *Under the assumptions of Theorem 3.3, assume that $P_B(s \mid s') > 0$ for all sampled edges (s, s'). For $\tau = (s_0, \ldots, s_n) \sim \mathcal{D}\# P_B$ and $s \in \tau$, let s' denote the unique child of s in τ. Then the loss*

$$\mathcal{L}_{\mathrm{DSE}'} = \mathbb{E}_{\tau \sim \mathcal{D}\# P_B} \sum_{s \in \tau \setminus \{s_n\}} \left[\sum_{s'' \in N(s)} c_{F\theta}(s, s'') - \frac{\log c_{F\theta}(s, s')}{P_B(s \mid s')} \right] \tag{9}$$

is an unbiased transition-sampled estimator of $\mathcal{L}_{DSE}$. Consequently, its pointwise minimizer is

$$c^*_{F\theta}(s, s') = \frac{\Pr_{\tau \sim \mathcal{D}\# P_B}\{s' \in \tau\}}{\Pr_{\tau \sim \mathcal{D}\# P_B}\{s \in \tau\}}.$$

The proof is provided in Section A.1. This objective directly implies a training algorithm analogous to the typical score matching for discrete diffusion. We can sample trajectories under our backward policy, sample one noisy state, and then enforce detailed balance over the data.

5 Experiments with Data-Pretrained GFlowNets

5.1 Data Pretraining and Reward Post-Training

We study antimicrobial-peptide design, following established GFlowNet sequence-design benchmarks (Jain et al., 2022). Terminal states are peptides over the canonical amino-acid alphabet with maximum length 10. Our construction graph uses arbitrary-position residue insertions, with a fixed backward policy that uniformly deletes one residue, yielding a non-trivial DAG with many construction trajectories for the same terminal peptide.

Our main question is whether data-based pretraining provides a useful advantage compared with strong reward-based alternatives. We compare $\mathcal{L}_{\mathrm{DSE}'}$ pretraining against offline Trajectory Balance on the same pretraining data. For TB, we consider both the standard setting in which the partition function Z must be learned jointly with the policy and an intentionally stronger oracle baseline in which Z is known and fixed. The latter removes global reward-scale estimation from the learning problem and therefore tests whether this is an important source of difficulty in reward-based training.

After pretraining, all methods are post-trained with Replay TB, which reuses previously discovered high-reward samples from a replay buffer (Shen et al., 2023). This provides a strong reward-based training baseline for evaluating the benefit of data pretraining. Thus, our comparison asks whether data pretraining remains useful when followed by a strong reward-training procedure, rather than only relative to a weak online baseline. Table 2 and Figure 2 show that it does: data pretraining gives the strongest overall initialization for subsequent reward optimization, while the oracle fixed-Z baseline substantially closes the gap compared with standard learned-Z TB.

5.2 Data Only Generative Modelling with GFlowNets

We empirically demonstrate that our formalism enables efficient data-based training for typical GFlowNet tasks. We show this for two discrete generative tasks: generating Ising states (Ising, 1925)

Table 2: Peptide generation before and after Replay TB post-training. All evaluations use 2,048 samples. Values are mean ± standard deviation over 10 seeds. Offline TB with fixed partition function Z uses information about the reward function typically unavailable during training. *Discovered* denotes the number of unique, novel, high-reward peptides discovered by the model.

	Before post-training		After post-training	
Pretraining	High-reward (%)	Discovered	High-reward (%)	Discovered
Untrained	0.00 ± 0.00	0.0 ± 0.0	73.48 ± 3.37	1474 ± 115
Data-Based $\mathcal{L}_{DSE'}$ (ours)	$\mathbf{67.34 \pm 1.55}$	154 ± 18	$\mathbf{88.58 \pm 0.94}$	$\mathbf{1802 \pm 19}$
Offline TB (fixed Z)	53.39 ± 1.40	$\mathbf{215 \pm 21}$	86.96 ± 0.75	1773 ± 17
Offline TB (learned Z)	11.20 ± 0.86	147 ± 15	79.14 ± 1.12	1617 ± 23

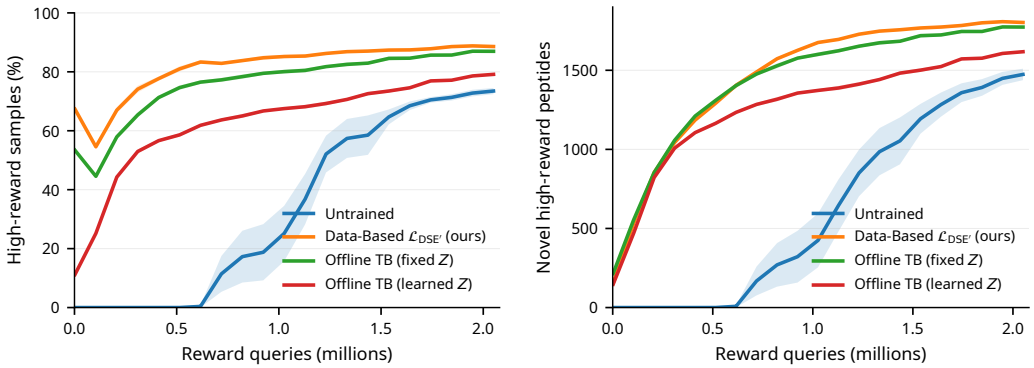

Figure 2: Reward-query efficiency during Replay-TB post-training of peptide GFlowNets with different initializations. Curves show mean ± SEM over 10 seeds; each evaluation uses 2,048 samples. Reward-query accounting includes offline selection and Z-warmup queries and excludes evaluation queries. Data-based $\mathcal{L}_{\mathrm{DSE}'}$ pretraining provides a large initial advantage and retains the strongest performance throughout post-training.

and generating binarized MNIST (LeCun et al., 1998) digits, by painting individual pixels: We define our state-space on these tasks for some $k \in \mathbb{N}$ as $\mathcal{S} = \{-1, 0, 1\}^{k \times k}$, $\mathcal{X} = \{0, 1\}^{k \times k}$, $\mathcal{A}(s) = \{(i, j, -1) \to (i, j, p) : i, j \in [1, k], p \in \{0, 1\}\}$. That is, in each partial state in $\mathcal{S}$, the model sets one unset pixel (-1) to either binary state, and terminates when all pixels are set. For our experiments, we extend the `torchgfn` library (Lahlou et al., 2023) with a score-parametrized GFlowNet, and implement our loss functions $\mathcal{L}_{DSE}$ and $\mathcal{L}_{DSE'}$. In both settings, the training process looks as follows: We take a dataset of terminal states, and then sample backward traces from the fixed (uniform) backward policy P_B. We then train the model with $\mathcal{L}_{DSE}$ with the analytic form of $P_B(s \mid x)$ for $\mathcal{L}_{DSE}$ as well as the sampled form $\mathcal{L}_{DSE'}$, which does not require a known P_B beyond the transition kernel.

Table 3 shows our experimental results on Ising grids of different sizes. In both settings, the sampling quality, in terms of state energy, is very similar to that of the base dataset generator. Across all energy parameters β and different sizes, the model learns to minimize energy almost as well as the ground truth simulator, without access to the energy function. In a slightly more complex data-generation task, we use GFlowNets to generate MNIST digits without a formal reward function. We show qualitative results in Figure 3: Qualitatively, we achieve both high-quality and visually diverse MNIST digits, with the analytic $\mathcal{L}_{DSE}$ yielding a stronger training supervision signal and lower variance. Furthermore, the last three rows show that our data-trained model can perform inpainting.

Table 3: Experimental results for simulating Ising models from simulated states, at different simulation temperatures β. E is the size-normalized final-state energy, $|m|$ reports the mode bias in the final state of the simulation. Values are averaged over 4096 simulations.

Size	Source	$\beta = 0.2$		$\beta = 0.4$		$\beta = 0.6$	
		$E \downarrow$	$\lvert m \rvert$	$E \downarrow$	$\lvert m \rvert$	$E \downarrow$	$\lvert m \rvert$
4×4	GFlowNet	-0.213	0.334	-0.677	0.759	-0.948	0.970
	Exact Solution	-0.228	0.343	-0.690	0.765	-0.954	0.973
16×16	GFlowNet	-0.201	0.084	-0.548	0.375	-0.953	0.973
	Gibbs Sampler	-0.214	0.084	-0.565	0.388	-0.954	0.973

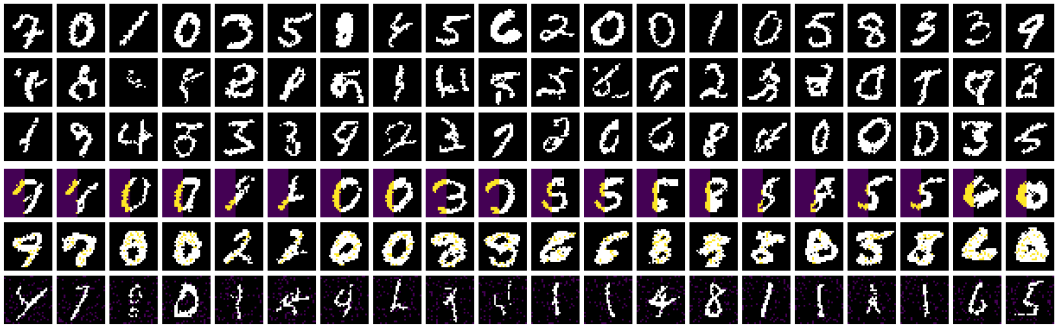

Figure 3: Generating and Inpainting MNIST digits. The first row shows actual samples from the MNIST dataset, the second row is samples generated after training with $\mathcal{L}_{DSE'}$, the third row with $\mathcal{L}_{DSE}$. In the last three rows, we perform different inpainting tasks with preset (colored) pixels.

6 Conclusion

We establish a common flow-score perspective on GFlowNets and discrete diffusion, revealing a shared representation underlying their traditionally distinct training regimes. This correspondence allows objectives developed for data-based generative modeling to be transferred to GFlowNets. In particular, we derive score-entropy objectives that train a GFlowNet directly from terminal samples, including a transition-sampled variant requiring only local backward probabilities. More broadly, this enables a pre-training/post-training regime in which the same GFlowNet is first trained from data and subsequently optimized using standard reward-based objectives. On peptide generation, data-based pretraining substantially improves subsequent reward-query efficiency and final high-reward generation under identical Replay-TB post-training. These results suggest that data-based and reward-based training should be viewed as complementary supervision regimes rather than separate model paradigms.

AI use statement

In this work, we used generative AI tools to assist with writing experiment code, including implementing methods specified by the authors. We also used these tools to identify relevant literature, brainstorm, draft parts of the paper, give feedback on correctness of mathematical statements and proofs and edit the text to improve readability. The authors reviewed all AI-assisted work, checked AI-generated code for correctness, and finalized all manuscript text.

We have *not* used generative AI tools to help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, design or provide feedback on research methodology or experiments, support qualitative and thematic data analysis, or interpret results. The following tasks are not applicable to this work: generating synthetic datasets, assisting with translation, and cleaning and reformatting datasets.

We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

Reproducibility statement

Complete proofs are provided in Appendix A, and experimental details are provided in Section 5 and Appendix B. An anonymized repository contains the implementations of the proposed objectives and baselines, experiment configurations, data processing and evaluation scripts, hyperparameters, random seeds, and scripts for reproducing the reported tables and figures. The peptide experiments use 10 random seeds, with uncertainty and reward-query accounting reported alongside the results. Code is available anonymously.

References

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 17981–17993. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/958c530554f78bcd8e97125b70e6973d-Paper.pdf.

Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. Flow network based generative models for non-iterative diverse candidate generation. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 27381–27394. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/e614f646836aaed9f89ce58e837e2310-Paper.pdf.

Yoshua Bengio, Salem Lahlou, Tristan Deleu, Edward J. Hu, Mo Tiwari, and Emmanuel Bengio. Gflownet foundations. *Journal of Machine Learning Research*, 24(210):1–55, 2023. URL http://jmlr.org/papers/v24/22-0364.html.

Andrew Campbell, Joe Benton, Valentin De Bortoli, Thomas Rainforth, George Deligiannidis, and Arnaud Doucet. A continuous time framework for discrete denoising models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 28266–28279. Curran Associates, Inc., 2022. doi: 10.52202/068431-2049. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/b5b528767aa35f5b1a60fe0aaeca0563-Paper-Conference.pdf.

Arran Carter, Sanghyeok Choi, Kirill Tamogashev, Víctor Elvira, and Esmeralda S. Whitammer. Discrete diffusion samplers and bridges: Off-policy algorithms and applications in latent spaces. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=bYg45bvuV9.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 12454–12465. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/67d96d458abdef21792e6d8e590244e7-Paper.pdf.

Aapo Hyvärinen. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(24):695–709, 2005. URL http://jmlr.org/papers/v6/hyvarinen05a.html.

Ernst Ising. Beitrag zur theorie des ferromagnetismus. *Zeitschrift für Physik*, 31(1):253–258, 1925. doi: 10.1007/BF02980577.

Moksh Jain, Emmanuel Bengio, Alex Hernandez-Garcia, 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 Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 9786–9801. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/jain22a.html.

Salem Lahlou, Joseph D. Viviano, Victor Schmidt, and Yoshua Bengio. torchgfn: A PyTorch GFlowNet library. *arXiv preprint arXiv:2305.14594*, 2023. doi: 10.48550/arXiv.2305.14594.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32819–32848. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/lou24a.html.

Kanika Madan, Jarrid Rector-Brooks, Maksym Korablyov, Emmanuel Bengio, Moksh Jain, Andrei Cristian Nica, Tom Bosc, Yoshua Bengio, and Nikolay Malkin. Learning GFlowNets from partial episodes for improved convergence and stability. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 23467–23483. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/madan23a.html.

Nikolay Malkin, Moksh Jain, Emmanuel Bengio, Chen Sun, and Yoshua Bengio. Trajectory balance: Improved credit assignment in GFlownets. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=5btWTw1vcw1.

Chenlin Meng, Kristy Choi, Jiaming Song, and Stefano Ermon. Concrete score matching: Generalized score matching for discrete data. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 34532–34545. Curran Associates, Inc., 2022. doi: 10.52202/068431-2502. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/df04a35d907e894d59d4eab1f92bc87b-Paper-Conference.pdf.

Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=sMyXP8Tanm.

Lorenz Richter, Ayman Boustati, Nikolas Nüsken, Francisco Ruiz, and Omer Deniz Akyildiz. Vargrad: A low-variance gradient estimator for variational inference. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 13481–13492. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/9c22c0b51b3202246463e986c7e205df-Paper.pdf.

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.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 2256–2265, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/sohl-dickstein15.html.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/3001ef257407d5a371a96dcd947c7d93-Paper.pdf.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=PxTIG12RRHS.

Daniil Tiapkin, Nikita Morozov, Alexey Naumov, and Dmitry P Vetrov. Generative flow networks as entropy-regularized RL. In Sanjoy Dasgupta, Stephan Mandt, and Yingzhen Li (eds.), *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pp. 4213–4221. PMLR, 02–04 May 2024. URL https://proceedings.mlr.press/v238/tiapkin24a.html.

Dinghuai Zhang, Nikolay Malkin, Zhen Liu, Alexandra Volokhova, Aaron Courville, and Yoshua Bengio. Generative flow networks for discrete probabilistic modeling. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 26412–26428. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/zhang22v.html.

Dinghuai Zhang, Ricky T. Q. Chen, Nikolay Malkin, and Yoshua Bengio. Unifying generative models with gflownets and beyond, 2023a. URL https://arxiv.org/abs/2209.02606.

Dinghuai Zhang, Hanjun Dai, Nikolay Malkin, Aaron C Courville, Yoshua Bengio, and Ling Pan. Let the flows tell: Solving graph combinatorial problems with gflownets. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 11952–11969. Curran Associates, Inc., 2023b. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/27571b74d6cd650b8eb6cf1837953ae8-Paper-Conference.pdf.

Ruixiang Zhang, Shuangfei Zhai, Yizhe Zhang, James Thornton, Zijing Ou, Joshua M. Susskind, and Navdeep Jaitly. Target concrete score matching: A holistic framework for discrete diffusion. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=ZMrdvSm7xi.

Kaiwen Zheng, Yongxin Chen, Hanzi Mao, Ming-Yu Liu, Jun Zhu, and Qinsheng Zhang. Masked diffusion models are secretly time-agnostic masked models and exploit inaccurate categorical sampling. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=CTC7CmirNr.

A Technical appendices and supplementary material

A.1 Proofs for Theorems

Proof of Theorem 4.2. As shorthand, let $\Pr(s) = \Pr_{\tau\sim D\#P_B}\{s \in \tau\}$. We want to find

$$\arg\min_{\theta} : \sum_{s,s'\in\mathcal{S}} c_{F\theta}(s,s') - \frac{\Pr(s')}{\Pr(s)} \log c_{F\theta}(s,s') \tag{10}$$

which has the same minimizer as

$$\arg\min_\theta : \sum_{s,s'\in\mathcal{S}} c_{F\theta}(s,s')\Pr(s) - \Pr(s')\log c_{F\theta}(s,s') \tag{11}$$

$$\arg\min_\theta : \sum_{s,s'\in\mathcal{S}} \Pr(s)\left[c_{F\theta}(s,s') - \frac{\Pr(s')}{\Pr(s)}\log c_{F\theta}(s,s')\right] \tag{12}$$

$$\arg\min_\theta : \mathbb{E}_{\tau\sim\mathcal{D}\#P_B}\sum_{s\in\tau}\sum_{s'\in\mathcal{S}}\left[c_{F\theta}(s,s') - \frac{\Pr(s')}{\Pr(s)}\log c_{F\theta}(s,s')\right] \tag{13}$$

Finally, $c_{F\theta}(s,s')$ is only defined for $(s,s')\in\mathcal{A}$, so we can restrict the scope of s'. Similarly, for each edge (s,s'), $\Pr(s') = \mathbb{E}_{\tau\sim\mathcal{D}\#P_B}\mathbf{1}\{s\in\tau\}\frac{\Pr(s'|s_n)}{\Pr(s|s_n)} = \mathbb{E}_{\tau\sim\mathcal{D}\#P_B}\mathbf{1}\{s\in\tau\}\frac{P_B(s'|s_n)}{P_B(s|s_n)}$. This gives

$$\arg\min_\theta : \mathbb{E}_{\tau\sim\mathcal{D}\#P_B}\sum_{s\in\tau}\sum_{s'\in N(s)}\left[c_{F\theta}(s,s') - \frac{\Pr(s'\mid s_n)}{\Pr(s\mid s_n)}\log c_{F\theta}(s,s')\right] \tag{14}$$

$$\arg\min_\theta : \mathbb{E}_{\tau\sim\mathcal{D}\#P_B}\sum_{s\in\tau}\left[\sum_{s'\in\mathrm{N}(s)} c_{F\theta}(s,s') - \sum_{s'\in\mathrm{N}(s,s_n)}\frac{P_B(s'\mid s_n)}{P_B(s\mid s_n)}\log c_{F\theta}(s,s')\right] \tag{15}$$

□

Proof of Theorem 4.3. We want to show that for fixed s, s_n $\frac{\log c_{F\theta}(s,s')}{P_B(s|s')}$ is an unbiased estimator for

$$\sum_{s'\in\mathrm{N}(s,s_n)}\frac{P_B(s'\mid s_n)}{P_B(s\mid s_n)}\log c_{F\theta}(s,s'). \tag{16}$$

Since $s'\in\mathrm{N}(s)$, it holds that $\Pr(s'\in\tau\mid s\in\tau,s_n) = \frac{P_B(s'|s_n)P_B(s|s')}{P_B(s|s_n)}$. Thus,

$$\sum_{s'\in\mathrm{N}(s,s_n)}\Pr(s'\in\tau\mid s\in\tau,s_n)\frac{\log c_{F\theta}(s,s')}{P_B(s\mid s')} = \sum_{s'\in\mathrm{N}(s,s_n)}\frac{P_B(s'\mid s_n)}{P_B(s\mid s_n)}\log c_{F\theta}(s,s'). \tag{17}$$

And finally

$$\mathbb{E}\left[\frac{\log c_{F\theta}(s,s')}{P_B(s\mid s')}\mid s\in\tau,s_n\right] = \sum_{s'\in\mathrm{N}(s,s_n)}\Pr(s'\in\tau\mid s\in\tau,s_n)\frac{\log c_{F\theta}(s,s')}{P_B(s\mid s')} \tag{18}$$

□

Proof: Concrete score matching and score entropy. Again, let $\Pr(s) = \Pr_{\tau\sim D\#P_B}\{s\in\tau\}$. For any distribution $\Pr$ with full support, we note that the score entropy loss:

$$L_{SE} = \mathbb{E}_{\Pr}\left[\sum_{y\in\text{Child(x)}}(c_{F\theta}(x,y) - \frac{\Pr(y)}{\Pr(x)}\log c_{F\theta}(x,y))\right] \tag{19}$$

is minimized whenever

$$(c_{F\theta}(x,y) - \frac{\Pr(y)}{\Pr(x)}\log c_{F\theta}(x,y)) \tag{20}$$

is minimized. Then,

$$\frac{\partial}{\partial\theta}(c_{F\theta}(x,y) - \frac{\Pr(y)}{\Pr(x)}\log c_{F\theta}(x,y) = 0 \tag{21}$$

$$c'_{F\theta}(x,y) - \frac{\Pr(y)}{\Pr(x)}\frac{1}{c_{F\theta}(x,y)}c'_{F\theta}(x,y) = 0 \tag{22}$$

$$c_{F\theta}(x,y) = \frac{\Pr(y)}{\Pr(x)} \tag{23}$$

Which is the solution minimizing the concrete score matching / detailed balance loss. □

Proof for Theorem 4.2: The earlier proof follows Lou et al. (2024), in proving that the score matching loss is equivalent to the score entropy loss. Further, training with the detailed balance objective has the same minimizer as training with score matching as seen in A.5. Thus we prove the validity of the proposed loss function by using the score matching objective as a starting point. We again note, that

$$P(x) = \frac{F(x)}{Z}$$

We start from the score entropy loss:

$$L_{SE} = \mathbb{E}_{x\sim \Pr}\left[\sum_{y\in Child(x)} (c_{F\theta}(x,y) - \frac{\Pr(y)}{\Pr(x)}\log c_{F\theta}(x,y))\right]$$

We use the following identity:

$$\begin{aligned}
&\mathbb{E}_{x\sim \Pr}\left[\sum_{y\in Child(x)} f(x,y)\frac{\Pr(y)}{\Pr(x)}\right] = \sum_x \Pr(x) \sum_{y\in Child(x)} f(x,y)\frac{\Pr(y)}{\Pr(x)} \\
&= \sum_x \sum_{y\in Child(x)} f(x,y)\Pr(y) = \sum_x \sum_{y\in Child(x)} \sum_{x_T} P(x_T)P_B(x|x_T)f(x,y)\frac{\Pr(y|x_T)}{\Pr(x|x_T)} \\
&= \mathbb{E}_{x_T\sim P_T}\mathbb{E}_{x\sim P_B(\cdot|x_T)} \sum_{y\in Child(x)} f(x,y)\frac{P_B(y|x_T)}{P_B(x|x_T)}
\end{aligned}$$

Further we note:

$$\begin{aligned}
\mathbb{E}_{x\sim p}\left[\sum_{y\in Child(x)} c_{F\theta}(x,y)\right] &= \sum_x p(x) \sum_{y\in Child(x)} c_{F\theta}(x,y) \\
&= \sum_x \sum_{x_T} p_T(x_T)p(x|x_T) \sum_{y\in Child(x)} c_{F\theta}(x,y) \\
&= \sum_{x_T} p_T(x_T) \sum_x p(x|x_T) \sum_{y\in Child(x)} c_{F\theta}(x,y) \\
&= \mathbb{E}_{x_T\sim p_T,\, x\sim p(x|x_T)}\left[\sum_{y\in Child(x)} c_{F\theta}(x,y)\right]
\end{aligned}$$

Applying this identity, we get:

$$\begin{aligned}
L_{SE} &= \mathbb{E}_{x\sim \Pr}\left[\sum_{y\in Child(x)} c_{F\theta}(x,y) - \frac{\Pr(y)}{\Pr(x)}\log c_{F\theta}(x,y))\right] \\
&= \mathbb{E}_{x\sim \Pr}\left[\sum_{y\in Child(x)} c_{F\theta}(x,y)\right] - \mathbb{E}_{x\sim \Pr}\left[\frac{\Pr(y)}{\Pr(x)}\log c_{F\theta}(x,y))\right] \\
&= \mathbb{E}_{x\sim \Pr}\left[\sum_{y\in Child(x)} c_{F\theta}(x,y)\right] - \mathbb{E}_{x_T\sim P_T}\mathbb{E}_{x\sim P_B(\cdot|x_T)}\left[\sum_{y\in Child(x)} \log c_{F\theta}(x,y)\frac{P_B(y|x_T)}{P_B(x|x_T)}\right] \\
&= \mathbb{E}_{x_T\sim P_T,\, x\sim P_B(\cdot|x_T)}\left[\sum_{y\in Child(x)} c_{F\theta}(x,y)\right] - \mathbb{E}_{x_T\sim P_T}\mathbb{E}_{x\sim P_B(\cdot|x_T)}\left[\sum_{y\in Child(x)} \log c_{F\theta}(x,y)\frac{P_B(y|x_T)}{P_B(x|x_T)}\right] \\
&= \mathbb{E}_{x_T\sim P_T,\, x\sim P_B(\cdot|x_T)}\left[\sum_{y\in Child(x)} c_{F\theta}(x,y) - \frac{P_B(y|x_T)}{P_B(x|x_T)}\log c_{F\theta}(x,y)\right]
\end{aligned}$$

In the diffusion framework the state space has a structure layered by time. We assume a layered GFlowNet, such that for a fixed number of steps $P_B(x_t|x_T)$ corresponds to a probability distribution. This is without loss of generality for our setting: variable-length trajectories can be padded by deterministic dummy states, so that all trajectories have length T. These padding states preserve the terminal distribution and only serve to define a common time index, provided each non-dummy transition represents an increment of the same size. □

Proof of Equation (7).

$$\begin{aligned} c_{F\theta}(y,x) &= \frac{P_F(x|y)}{P_B(y|x)} = \frac{F_\theta(x)}{F_\theta(y)} = \frac{\sum_{x_T} F(x,x_T)}{\sum_{x_T} F(y,x_T)} \\ &= \frac{\sum_{x_T} R(x_T)P_B(x|x_T)}{\sum_{x_T} R(x_T)P_B(y|x_T)} = \frac{Z\sum_{x_T} P_T(x_T)P_B(x|x_T)}{Z\sum_{x_T} P_T(x_T)P_B(y|x_T)} \\ &= \frac{\mathbb{E}_{x_T\sim P_T(x_T)}[P_B(x|x_T)]}{\mathbb{E}_{x_T\sim P_T(x_T)}[P_B(y|x_T)]} \end{aligned}$$

Given

$$c_{F\theta}(y,x) = \frac{\mathbb{E}_{x_T\sim P_T}[P_B(x|x_T)]}{\mathbb{E}_{x_T\sim P_T}[P_B(y|x_T)]}$$

we note the following:

$$c_{F\theta}(y,x)P_B(y|x) = \frac{\mathbb{E}_{x_T\sim P_T}[P_B(x|x_T)]}{\mathbb{E}_{x_T\sim P_T}[P_B(y|x_T)]}P_B(y|x) \tag{24}$$

$$P_{F,\theta}(x|y)\cdot\mathbb{E}_{x_T\sim P_T}[P_B(y|x_T)] = P_B(y|x)\cdot\mathbb{E}_{x_T\sim P_T}[P_B(x|x_T)] \tag{25}$$

$$P_{F,\theta}(x|y)F_\theta(y) = P_B(y|x)F_\theta(x) \tag{26}$$

That is, the optimal solution to 7 results in the detailed balance conditions

Further we show that the forward transition probability corresponds to a valid probability kernel.

$$\begin{aligned} \sum_{x\in Child(y)} P_{F,\theta}(x|y)F_\theta(y) &= \sum_{x\in Child(y)} P_B(y|x)F_\theta(x) && (27)\\ &= \sum_{x\in Child(y)} P_B(y|x)\sum_{x_T} R(x_T)P_B(x|x_T) && (28)\\ &= \sum_{x_T} R(x_T)\sum_{x\in Child(y)} P_B(y|x)P_B(x|x_T) && (29)\\ &= \sum_{x_T} R(x_T)\sum_{x\in Child(y)} P_B(y,x|x_T) && (30)\\ &= \sum_{x_T} R(x_T)P_B(y|x_T) = F_\theta(y) && (31) \end{aligned}$$

This is the necessary condition for the forward process to correspond to a probability kernel, as can be seen by dividing both sides by the flow through the state y. □

Proof of Equation (6). Assume, that

$$\forall x: \quad L_{DFM}(x;\theta) = 0$$

Further we define:

$$F_\theta(x\to y) = F_\theta(y)P_B(x|y)$$

Then

$$\forall x: \quad F_\theta(x) = Z_\theta \cdot \mathbb{E}_{x_T \sim p_T}[P_B(x|x_T)] \tag{32}$$

$$= Z_\theta \cdot \sum_{x_T} p_T(x_T) P_B(x|x_T) = Z_\theta \cdot \sum_{x_T} p_T(x_T) \sum_{y \in Child(x)} P_B(x, y|x_T) \tag{33}$$

$$= Z_\theta \cdot \sum_{x_T} p_T(x_T) \sum_{y \in Child(x)} P_B(x|y) P_B(y|x_T) \tag{34}$$

$$= Z_\theta \cdot \sum_{y \in Child(x)} P_B(x|y) \sum_{x_T} p_T(x_T) P_B(y|x_T) \tag{35}$$

$$= \sum_{y \in Child(x)} P_B(x|y) F_\theta(y) \tag{36}$$

Which is the flow matching condition. Thus training to zero results in balancing the GFlowNet. □

Proof for Theorem 3.2: For a GFlowNet state-space $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ with a consistent score function $c_F(s, s') = \frac{F(s')}{F(s)}$ and a consistent backward policy $P_B(s \mid s')$ uniquely identify a valid flow F up to a multiplicative constant $Z > 0$.

To prove the statement, we note, that the score and backward policy together provide the necessary information for recovering the normalized flows:

$$\forall x \in \mathcal{S}: \forall \tau = (s_0, .., s_n = x): \quad P(x) = \frac{F(x)}{Z} = \prod_{t=0}^{n-1} c_F(s_t, s_{t+1}) = \prod_{t=0}^{n-1} \frac{F(s_{t+1})}{F(s_t)}$$

Given a known backward policy, we can also then recover the normalized edge flows:

$$\forall x \to y \in \mathcal{A}: \; P(x \to y) \coloneqq P(y) P_B(x|y)$$

□

A.2 Time Agnosticity

Unlike diffusion models, the GFlowNet-framework is free of an explicit time variable. Instead, the concept of time is encoded by the underlying DAG-structure as directionality. Acyclicity induces a strict partial order on the states in the graph, and trajectories follow this order from the initial state toward terminal objects.

In the following, we adapt the derivation of Ou et al. (2025) for the GFlowNet framework. In its general form, GFlowNets distribute the unnormalized probability mass given by the reward function R along the construction graph according to the backward policy, making the state-, and edgeflows quantifications of probability mass distributed to these intermediate objects. In the uniform masking setting below, this position can be summarized by the number t of constructed tokens. The derivation shows that the normalized flow through such a state decomposes into two factors: the terminal probability mass of all complete objects compatible with the partial state, and a purely combinatorial factor induced by the backward policy. Consequently, the score between adjacent states depends on the data distribution through the conditional probability of the partial state with relation to the terminal distribution, and the structural properties of the graph and backward policy which in this case simplify substantially.

Proof of Equation (5). We investigate a setting, where each of the terminal states consists of T tokens, and the backward policy $P_B(\cdot|\cdot)$ is set to uniformly deconstruct the states, that is, to iteratively and uniformly mask tokens.

We then use indices to denote the amount of steps taken: s_t corresponds to a state where t tokens have been added to the empty state. Now, given $s_t \subset x_T$, we note that the marginal probability of reaching s_t from x for some terminal state $x_T \in \mathcal{X}$ is equal to the probability of drawing an unordered set of size t from a set of T items, that is:

$$P_B(s_t|x_T) = \binom{T}{t}^{-1}$$

Symbol	Name	Meaning
GFlowNet construction graph		
$\mathcal{G} = (\mathcal{S}, \mathcal{A})$	DAG	Directed acyclic construction graph.
$\mathcal{S}$	State space	Set of partial and terminal states.
$\mathcal{X} \subseteq \mathcal{S}$	Terminal space	Set of complete objects.
s_0	Initial state	Empty or fully masked starting state.
$\mathcal{N}(s)$	Children	States reachable from s in one forward step.
$\mathrm{Par}(s)$	Parents	States that can transition into s in one forward step.
$\tau = (s_0, \ldots, s_n)$	Trajectory	A valid path through the construction DAG.
$\mathcal{T}$	Trajectory set	Set of valid construction trajectories.
Flows and policies		
$F(s)$	State flow	Unnormalized flow through state s.
$F(s \to s')$	Edge flow	Unnormalized flow through edge (s, s').
$Z = F(s_0)$	Total flow	Global normalizing constant.
$P_F(s'\|s)$	Forward policy	Learned construction or denoising policy.
$P_B(s\|s')$	Backward policy	Fixed deconstruction or noising policy.
$R(x)$	Reward	Unnormalized terminal reward for $x \in \mathcal{X}$.
P_T or $\mathcal{D}$	Terminal distribution	Data or target distribution over terminal states.
Scores and data training		
$c_F(s, s')$	GFlowNet score	Local flow ratio $F(s')/F(s) = P_F(s'\|s)/P_B(s\|s')$.
$\mathcal{D}\#P_B$	Backward trajectory law	Distribution over trajectories obtained by sampling $x \sim \mathcal{D}$ and deconstructing with P_B.
$\mathcal{N}(s, x)$	Compatible children	Children of s that lie on at least one backward trajectory from terminal state x.
$\mathcal{L}_{\mathrm{DSE}}$	Analytic score-entropy loss	Data-training loss using backward marginals $P_B(s\|x)$.
$\mathcal{L}'_{\mathrm{DSE}}$	Sampled score-entropy loss	Transition-sampled loss requiring only sampled backward trajectories and local $P_B(s\|s')$.
Masked diffusion correspondence		
$q(x_t\|x_{t-1})$	Noising process	Diffusion forward corruption process; corresponds to P_B.
$p_\theta(x_{t-1}\|x_t)$	Denoising process	Learned diffusion reverse process; corresponds to P_F.
c_θ	Diffusion score	Concrete score parameterization in discrete diffusion.

Table 4: Table of Notation

Then, we construct the probability given by the normalized flow:

$$P(s_t) = \sum_{x_T} P_T(x_T) P_B(s_t|x_T) = \sum_{x_T} P_T(x_T) \binom{T}{t}^{-1} \delta_{s_t \subset x_T} = P_T(x_T : s_t \subset x_T) \binom{T}{t}^{-1}$$

Next, we investigate the score between two consecutive states.

$$\begin{aligned} \frac{P(s_{t+1})}{P(s_t)} &= \frac{P_T(s_{t+1})\binom{T}{t+1}^{-1}}{P_T(s_t)\binom{T}{t}^{-1}} = P_T(s_{t+1}|s_t)\frac{(t+1)!(T-t-1)!}{t!(T-t)!} \\ &= P_T(s_{t+1}|s_t)\frac{t+1}{T-t} = P_T(s_{t+1}|s_t)\frac{1}{P_B(s_t|s_{t+1})(T-t)} \end{aligned}$$

□

A.3 Constructing the underlying GFlowNet of a masked Diffusion model

Theorem A.1 (Equivalence between GFlowNets and Absorbing State Diffusion Models)**.** *Take an absorbing state diffusion model and a sampling schedule; this uniquely defines a fixed GFlowNet that samples proportionally to the distribution represented by the diffusion model.*

The sampling strategies of masked diffusion models typically happen in some form of discretized time. Considering the time-agnostic underlying process of the CTMC, we propose further that the sampling processes can be used to define a GFlowNet. For a given sampling schedule, $[t_i]_{i\in[0,d]}$, $y = (x, \hat{x})$ where the maximum length of tokens is d and x consists of d_1 tokens, we note that by setting

$$\forall x \in S: \quad P_F(y \mid x) = \frac{1}{d-d_1} P_T(\hat{x} \mid x) = \frac{1}{d-d_1} \frac{1-e^{-\sigma(t_{d_1})}}{e^{-\sigma(t_{d_1})}} \frac{p_{t_{d_1}}(y)}{p_{t_{d_1}}(x)} ,$$

we get a GFlowNet that samples from the same learned underlying distribution.

A.4 On Concrete Scores

There is also a very direct comparison to be made between the Denoising variant of Concrete Score Matching from Meng et al. (2022) and the GFlowNet. Meng et al. (2022) derive a denoising counterpart for the Concrete Score. Using a perturbation kernel $q(\tilde{x}|x)$, the perturbed data distribution is given by $\tilde{p}(x) = \sum_x p_{data}(x)q(\tilde{x}|x)$. For these objects, the Denoising Concrete Score Matching objective is given:

$$\mathcal{J}_{D-CSM}(\theta) = \sum_{x,\tilde{x}} p_{data}(x)\tilde{q}(\tilde{x}|x)||c_\theta(\tilde{x};\mathcal{N}) - c_{q(\tilde{x}|x)}(\tilde{x};\mathcal{N})||_2^2$$

The D-CSM-objective is minimized when $c_\theta(\tilde{x};\mathcal{N}) = c_{p(\tilde{x})}(\tilde{x};\mathcal{N})$

Considering the backward process to be a noising process, or a perturbation kernel, we note a correspondence between the perturbed distribution $\tilde{p}(x)$ of Meng et al. (2022) and the Flow Function. That is,

$$F(x) = \sum_{x_T} R(x_T)P_B(x|x_T), \quad \tilde{p}(x) = \sum_x p_{data}(x)q(\tilde{x}|x)$$

up to notational differences the perturbed variant of the data distribution can be understood as the flow function learned by the GFlowNet.

A.5 The GFlowNet objectives in terms of scores

The GFlowNet training objectives implicitly lead to the concrete score matching objectives.

Trajectory balance

The trajectory balance objective states

$$Z \prod_{t=0}^{n-1} P_F(s_{t+1}|s_t) = R(s_n) \prod_{t=0}^{n-1} P_B(s_t|s_{t+1})$$

For a GFlowNet parametrized in the following way

$$P_F(s_{t+1}|s_t) = c_F(s_t, s_{t+1})P_B(s_t|s_{t+1})$$

with the restriction

$$\sum_{s_{t+1}\in Child(s_t)} P_F(s_{t+1}|s_t) = \sum_{s_{t+1}\in Child(s_t)} P_B(s_t|s_{t+1})c_F(s_t, s_{t+1}) = 1$$

is equivalent to

$$\prod_{t=0}^{n-1} c_{F\theta}(s_t, s_{t+1}) = \frac{R(s_n)}{Z}$$

Proof:

$$Z \prod_{t=0}^{n-1} P_F(s_{t+1}|s_t) - R(s_n) \prod_{t=0}^{n-1} P_B(s_t|s_{t+1}) = 0$$

$$\frac{Z}{R(s_n)} \prod_{t=0}^{n-1} c_F(s_t, s_{t+1}) = 1$$

$$\prod_{t=0}^{n-1} c_F(s_t, s_{t+1}) = \frac{R(s_n)}{Z}$$

Detailed Balance:
The detailed balance conditions

$$P_F(y|x)F(x) = P_B(x|y)F(y) \quad \forall x \in S : \forall y \in Child(x)$$

are equivalent to the expression

$$\forall x : \; y \in \text{Child}(x) : \quad c_F(x, y) - \frac{F(y)}{F(x)} = 0$$

with the restriction

$$\sum_{y \in Child(x)} P_F(y|x) = \sum_{y \in Child(x)} P_B(x|y) c_F(x, y) = 1$$

Proof:
We prove this statement through simple algebraic manipulation of the objectives, starting from the score matching objective:

$$0 = c_F(x, y) - \frac{F(y)}{F(x)} \Leftrightarrow 0 = c_F(x, y) P_B(x|y) - \frac{F(y)}{F(x)} P_B(x|y) \tag{37}$$

$$= P_F(y|x) - \frac{F(y)}{F(x)} P_B(x|y) \tag{38}$$

which is the detailed balance condition.

Flow Matching: The GFlowNet Flow-Matching objective

$$F(x) = \sum_{z \in Par(x)} F(z \to x) = \sum_{y \in Child(x)} F(x \to y)$$

Is equivalent to the restriction placed on the parametrization for nonzero flow functions F

$$\sum_{y \in Child(x)} P_F(y|x) = \sum_{y \in Child(x)} P_B(x|y) c_F(x, y) = 1$$

Proof:

$$F(x) = \sum_{z \in Par(x)} F(z \to x) = \sum_{y \in Child(x)} F(x \to y) \tag{39}$$

$$\leftrightarrow \sum_{y \in Child(x)} P_B(x|y) \frac{F(y)}{F(x)} = \sum_{y \in Child(x)} P_B(x|y) c_F(x, y) = 1 \tag{40}$$

Subtrajectory balance

The GFlowNet Partial Trajectory balance objective:

$$\forall \tau \in \mathcal{T}^{partial} \cup \mathcal{T} : \; \tau = (s_m, ...s_n) :$$

$$F(s_m) \prod_{t=m}^{n-1} P_F(s_{t+1}|s_t) = F(s_n) \prod_{t=m}^{n-1} P_B(s_t|s_{t+1}) \tag{41}$$

is equivalent to the objective:

$$\prod_{t=m}^{n-1} c_F(s_t, s_{t+1}) = \frac{F(s_n)}{F(s_m)}$$

restricted by

$$\sum_{y \in Child(x)} P_F(y|x) = \sum_{y \in Child(x)} P_B(x|y) c_F(x, y) = 1$$

Proof: To prove this, it is enough to note that in Madan et al. (2023), it is proven that the detailed balance objective and the partial trajectory balance objective are equivalent. Thus, as the score matching objective leads to the detailed balance objective which leads to the partial trajectory balance, score matching leads to partial trajectory balance.

B Experimental Details

The code for the experiments is available anonymously on Anonymous Github. All experiments were run on a cluster with individual Nvidia V100 GPUs, and typically finished in a few minutes.

We define a score-parameterized GFlowNet class and run an experiment using the `DiscreteEBM` environment provided by the `torchGFN` library.

We define multiple custom datasets: one for MNIST and for Ising, one created analytically, and one with a Gibbs sampler. Each model was trained from 100.000 partial reverse trajectories, with cosine-annealed learning rates. The detailed configs of each run are provided in the repository and allow for exact reproduction.

C Extended Background on Generative Flow Networks

A Generative Flow Network (GFlowNet) (Bengio et al., 2023) is a generative model or probabilistic inference machine trained to sample from $\mathcal{D}$, when given access to the unnormalized reward function R. Structurally, the GFlowNet is defined over a directed acyclic graph (DAG), often denoted $\mathcal{G} = (\mathcal{S}, \mathcal{A})$, outlining all the ways to construct the objects $x \in \mathcal{X}$. The nodes of the graph represent elements of $\mathcal{X}$ as well as *intermediate* compositional objects s in some state space $\mathcal{S} \supset \mathcal{X}$. We assume the terminal samples $x \in \mathcal{X}$ are then iteratively constructed from some initial state s_0, where valid state transitions in the constructions are represented by the arcs $(s, s') \in \mathcal{A}$ of the graph $\mathcal{G}$. A full construction process over n steps is called a *trajectory* and denoted as $\tau = (s_0, s_1, \ldots, s_n) \in \mathcal{T}$, with $s_n \in \mathcal{X}$. We refer to the *children* or *neighborhood* of a state s as $\mathrm{N}(s) \coloneqq \{s' \in \mathcal{S} : (s, s') \in \mathcal{A}\}$, and its parents as $\mathrm{Par}(s) \coloneqq \{s' \in \mathcal{S} : (s', s) \in \mathcal{A}\}$.

To induce a distribution over $\mathcal{T}$ and consequently over $\mathcal{X}$, GFlowNets define a *forward policy* $P_F : \mathcal{S} \times \mathcal{S} \to \mathbb{R}_{\geq 0}$, with $\forall s \in \mathcal{S} \setminus \mathcal{X} : \sum_{s' \in N(s)} P_F(s' \mid s) = 1$, to select, at each intermediate state $s \in \mathcal{S} \setminus \mathcal{X}$, each child state $s' \in \mathrm{N}(s)$ with a probability such that

$$\Pr_{P_F}(x) = \sum_{\tau \in \mathcal{T} :\, s_n = x} \prod_{(s_t, s_{t+1}) \in \tau} P_F(s_{t+1} \mid s_t) \propto R(x).$$

A forward policy is typically expressed via a *flow function* F, that assigns unnormalized probability to states in $\mathcal{S}$, to arcs in $\mathcal{A}$ or to entire trajectories in $\mathcal{T}$.

For a given flow function F, we can define the forward policy $P_F(s' \mid s) \coloneqq F(s \to s')/F(s)$ and the backward policy $P_B(s \mid s') \coloneqq F(s \to s')/F(s')$. Consistency between F and R can be explicitly enforced at terminal states: $\forall x \in \mathcal{X} : F(x) = R(x)$. For intermediate states $s \in \mathcal{S} \setminus \mathcal{X}$, F then has to adhere to the *flow matching* (FM) or *flow conservation* property

$$\forall s \in \mathcal{S} \setminus \mathcal{X} : F(s) = \sum_{s' \in \mathrm{N}(s)} F(s \to s'), \qquad \forall s \in \mathcal{S} \setminus \{s_0\} : F(s) = \sum_{s'' \in \mathrm{Par}(s)} F(s'' \to s). \quad (42)$$

By the flow conservation property, the total flow is given by $Z = F(s_0) = \sum_{x_T \in \mathcal{X}} R(x_T)$. In learning settings, different parametrizations for GFlowNets exist, and we denote directly parametrized functions with the subscript θ in each setting.

C.1 Training GFlowNets

Multiple training objectives for GFlowNets have been introduced (Bengio et al., 2021; 2023; Malkin et al., 2022; Madan et al., 2023). These loss functions all optimize for the flow-matching property in Equation (42) in different ways. Crucially for all of these loss functions, it is necessary to additionally enforce $\forall x \in \mathcal{X} : F(x) = R(x)$.

Flow Matching loss (Bengio et al., 2021): The flow matching loss is based on directly optimizing for FM:

$$\forall s \in \mathcal{S} \setminus \mathcal{X} :\ \mathcal{L}_{\text{FM}}(s) \coloneqq \left(\log \frac{\sum_{s' \in \mathrm{N}(s)} F_\theta(s \to s')}{\sum_{s' \in \mathrm{Par}(s)} F_\theta(s' \to s)} \right)^2$$

$$\forall x \in \mathcal{X} :\ \mathcal{L}_{\text{FM}}(x) \coloneqq \left(\log \frac{R(x)}{\sum_{s' \in \mathrm{Par}(x)} F_\theta(s' \to x)} \right)^2$$

Detailed Balance loss (Bengio et al., 2023): An alternative to enforcing FM directly is the detailed balance (DB) condition, which is sufficient and necessary to ensure FM:

$$\forall (s, s') \in \mathcal{A} : \quad \mathcal{L}_{\text{DB}}(s, s') = \left(\log \frac{F_\theta(s) P_{F_\theta}(s' \mid s)}{F_\theta(s') P_{B_\theta}(s \mid s')} \right)^2.$$

The reward matching condition $F(x) = R(x)$ is enforced similarly to $\mathcal{L}_{\text{FM}}$.

Subtrajectory Balance Loss Madan et al. (2023): The Subtrajectory Balance loss is defined for all trajectories $\tau := (s_m, .., s_n)$ where $0 \leq m \leq n$:

$$\mathcal{L}_{sub-TB}(\tau) = \left(\log \frac{F_\theta(s_m) \prod_{t=m}^{n-1} P_{F_\theta}(s_{t+1} \mid s_t)}{F_\theta(s_n) \prod_{t=m}^{n-1} P_{B_\theta}(s_t \mid s_{t+1})} \right)^2,$$

where $\forall x \in \mathcal{X} : F_\theta(x) = R(x)$ is enforced. The subtrajectory balance constraint holds if and only if the Detailed Balance holds Madan et al. (2023). **Trajectory Balance Loss** (Malkin et al., 2022) is a special case of $\mathcal{L}_{sub-TB}$, defined for complete trajectories $\tau = (s_0, ..., s_n = x)$.

Training the GFlowNet with data: Previous work has largely considered the problem of training a GFlowNet based on a given reward function. Zhang et al. (2022) extend GFlowNets to scenarios where a reward function is not directly available. They introduce an energy model to serve as an intermediate between the implicit reward function given by the true data distribution and the GFlowNet. Interleaved updates are used to train the GFlowNet to sample from the energy distribution, and to train the energy model based on a set of data.