GFlowNet Pretraining with Inexpensive Rewards

Mohit Pandey*
Vancouver Prostate Centre
The University of British Columbia
mkpandey@student.ubc.ca

Gopeshh Subbaraj
Mila - Quebec AI Institute
Université de Montréal
gopeshh.subbaraj@mila.quebec

Emmanuel Bengio
Recursion Pharmaceuticals
emmanuel.bengio@recursionpharma.com

Abstract

Generative Flow Networks (GFlowNets), a class of generative models have recently emerged as a suitable framework for generating diverse and high-quality molecular structures by learning from unnormalized reward distributions. Previous works in this direction often restrict exploration by using predefined molecular fragments as building blocks, limiting the chemical space that can be accessed. In this work, we introduce Atomic GFlowNets (A-GFNs), a foundational generative model leveraging individual atoms as building blocks to explore drug-like chemical space more comprehensively. We propose an unsupervised pre-training approach using offline drug-like molecule datasets, which conditions A-GFNs on inexpensive yet informative molecular descriptors such as drug-likeliness, topological polar surface area, and synthetic accessibility scores. These properties serve as proxy rewards, guiding A-GFNs towards regions of chemical space that exhibit desirable pharmacological properties. We further our method by implementing a goal-conditioned fine-tuning process, which adapts A-GFNs to optimize for specific target properties. In this work, we pretrain A-GFN on the ZINC15 offline dataset and employ robust evaluation metrics to show the effectiveness of our approach when compared to other relevant baseline methods in drug design.

1 Introduction

GFlowNets are amortized samplers that learn stochastic policies to sequentially generate compositional objects from a given unnormalized reward distribution. They can generate diverse sets of high-reward objects, which has a demonstrated utility in small molecules drug-discovery tasks [1]. Traditionally, GFlowNets for molecules generation have focused on fragment-based drug discovery i.e. the GFlowNet action space comprises of some predetermined fragments as building blocks for molecules. While producing diverse candidates, fragment-based approach limits the pockets of chemical space assessible by GFlowNet policy [2, 1, 3, 4]. A truly explorative generative policy, tapping into the full potential of GFlowNet would be realizable when using atoms instead of fragments as the action space [5, 6, 7]. On the other hand, the vastness of accessible state space makes training atom-based GFlowNets susceptible to collapse. Earlier atom-based GFlowNets attempts overcame this issue by limiting to small trajectories [3]. However, since most commercially available drugs have molecular weights ranging from 200 to 600 daltons[8, 9], the molecules generated from these small trajectories are unlikely to possess the drug-like characteristics necessary for therapeutic efficacy. In this work, we propose to mitigate the small trajectory length constraint for atom-based GFNs by

*Work done as a part of research internship at Recusion Pharmaceuticals

Foundation Models for Science Workshop,38th Conference on Neural Information Processing Systems (NeurIPS 2024).

pretraining them with expert demonstrations, coming in the form of offline drug-like molecules. Our main contributions in this paper are as following:

- We introduce A-GFN-an atom-based GFlowNet for sampling molecules proportional to rewards governed by inexpensive molecular properties.
- We propose a novel strategy for unsupervised pretraining of A-GFNs by leveraging drug-like molecules using offline, off-policy training. This pretraining enables broader exploration of chemical space while maintaining diversity and novelty relative to existing drug-like molecule datasets.
- Through extensive experimentation, we demonstrate that goal-conditioned fine-tuning of A-GFNs for sampling molecules with desired properties offers significant computational advantages over training A-GFNs from scratch for the same objectives.

2 Related Works

2.1 GFlowNet

GFlowNets were introduced by Bengio et al. in 2021 [1] as a framework for training energy-based generative models that learn to sample diverse candidates in proportion to a given reward function. Unlike traditional Reinforcement Learning (RL) methods, which focus on maximizing rewards through a sequence of actions, GFlowNets aim to generate samples with probabilities proportional to their associated rewards. This distinction allows GFlowNets to explore a broader solution space, facilitating the discovery of novel, high-quality, and diverse objects across various domains. Recent works have been conducted along a multitude of directions, some focusing on theoretical aspects, such as connections to variational methods [10, 11]), and some focused on improved training methods for better credit assignment and sample efficiency [12, 13]. Due to its flexibility, GFlowNets have also been applied successfully to different settings such as biological sequences[14], causal discovery [15, 16], discrete latent variable modeling[17], and computational graph scheduling [18].

2.2 Unsupervised pretraining in RL and GFlowNets

Pretraining models in machine learning has virtually become the default way to obtain powerful models [19, 20]. Specifically, unsupervised pretraining in reinforcement learning (RL) has emerged as a promising strategy to enhance data efficiency and improve agent performance across various tasks. Recent works have explored different methodologies to leverage unsupervised interactions with the environment before fine-tuning on specific objectives. For instance, Liu and Abbeel (2020) introduced Active Pre-Training [21], a reward-free pre-training method that maximizes particle-based entropy in a contrastive representation space, achieving human-level performance on several Atari games and significantly enhancing data efficiency in the DMControl suite. Similarly, Mutti et al. [22] addressed unsupervised RL in multiple environments, proposing a framework that allows for pre-training across diverse scenarios to improve the agent's adaptability. Additionally, the Unsupervised-to-Online RL framework [23] was developed, which replaces domain-specific offline RL with unsupervised pre-training, demonstrating that a single pre-trained model can be effectively reused for multiple downstream tasks, often outperforming traditional methods. Similarly, In the context of GFlowNets, recent advancements have introduced unsupervised pre-training strategies, such as the outcome-conditioned GFlowNet [24], which enables reward-free pre-training by framing the task as a self-supervised problem. This approach allows GFlowNets to learn to explore the candidate space and adapt efficiently to downstream tasks, showcasing the potential of unsupervised pre-training in enhancing the performance of generative models in molecular design.

2.3 Goal-Conditioned and Multi-Objective Gflownets

Recent advancements in GFlowNets have focused on goal-conditioned and multi-objective frameworks, enhancing their applicability in complex generative tasks. Goal-conditioned GFlowNets enable the generation of diverse outputs tailored to specific objectives, improving sample efficiency and generalization across different goals, as demonstrated by [14] and further refined through methods like Retrospective Backward Synthesis [25] to address sparse reward challenges. Roy et al., [26], impose hard constraints on a GFlowNet model by employing focus regions as a goal-design strategy which makes it comparable to a form of goal-conditional reinforcement learning [27]. Additionally, the development of multi-objective GFlowNets [3] allows for simultaneous optimization of multiple criteria, providing practitioners with greater control over the generative process and the ability to explore trade-offs between competing objectives.

2.4 Molecule Generation

To contextualize our approach, we provide a brief review of prior research that utilized atom-based vocabularies in molecular generative modeling. Reinforcement learning (RL) studies optimal decision-making methodologies to maximize cumulative rewards. Given the shared notion of object-constructing Markov decision policies between GFlowNets and RL, we focus on the latter literature for molecule generation. For a comprehensive review of methods in molecule generation, we point the readers to [28]. Recently, RL has been frequently employed in de novo design tasks due to its capability to explore chemical spaces beyond the compounds present in existing datasets. Moreover, it allows for targeted molecule generation for constrained property optimization. Early works in this domain focused on the auto-regressive generation of SMILES within an RL-loop for molecular property optimization[29, 30, 31]. MolDQN[32] employs deep Q-Networks and multiobjective molecular properties for scalarization of rewards to generate 100% valid molecules without pretraining on a dataset. You et al.[33] and Atance et al.[34] have employed graph neural networks, trained on offline molecular datasets, to generate molecular graphs. They simultaneously applied policy-gradient reinforcement learning to ensure that the generated molecules adhere to specified property profiles.

3 Preliminaries

GFlowNets are generative models designed to sample structured objects from a state space $\mathcal{S}$ in proportion to a reward function $R(s_T)$ assigned to terminal states s_T. The model generates trajectories $\tau = (s_0, a_0, \ldots, s_T)$ by transitioning between states through actions, guided by a forward policy $P_F(s' \mid s)$. The key idea is to learn a flow function $F(s)$ that ensures the total flow into each state equals the flow out, so the probability of reaching any terminal state is proportional to its reward. This is achieved by aligning the forward and backward policies P_F and P_B, ensuring the final distribution over terminal states reflects the desired reward distribution.

Existing extensions of GFlowNets that handle multiple, potentially conflicting objectives [3, 2], along with the benefits of scalability to large action spaces and credit assignment makes them well-suited for molecular generation tasks; the primary focus of our work. We consider molecules as their topological graphs $G = (A, E, \mathcal{X})$ such that $\mathcal{X} \in \mathbb{R}^{n \times d}$ define d-dimensional atomic features for the n nodes in G, $E \in \{0,1\}^{b \times n \times n}$ is edge-adjacency tensor for b types of edges connecting n nodes and finally $A \in \{0,1\}^{n \times n}$ is the adjacency matrix for n nodes. The corresponding trajectory τ for G ($\tau = (s_0, a_0), ...(s_n, a_n)$) is generated by sampling actions a from a conditional learning policies $P_F(. \mid c; \theta)$ and $P_B(. \mid c, G; \theta)$ of the GFlowNet $\mathcal{G}_\theta$, where state s_i is a partially constructed subgraph of G, and s_n = G.

Our primary objective is to learn π such that the generated G are chemically valid molecular graphs satisfying some defined molecular property conditional ranges c, and exhibiting sufficient diversity. Specifically, we want to train a conditional policy which samples molecular graphs G with probability proportional to $R(G|c)$, where $R(G|c)$ is a reward function measuring how well generated G satisfies c. Our secondary objective involves fine-tuning this pretrained $\mathcal{G}_\theta = (P_F(;\theta), P_B(;\theta))$ to achieve drug discovery tasks with certain molecular property constraints and establish the benefits of fine-tuning GFlowNets. Trajectory balance [12] being the most common learning objective designed to improve credit assignment in GFlowNets is used as the training method in this work.

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

where, trajectory $\tau = (s_0, s_1, ..., s_n)$ such that $s_n = x$ is a fully constructed object. Within the context of our work, $x \in \mathcal{X}$ are samples from the combinatorial space of all possible molecules using actions $a_t \sim \mathcal{A}$. This equation ensures that the product of forward policy probabilities along a trajectory is proportional to the product of backward policy probabilities and the reward along the same trajectory.

4 Unsupervised Pretraining with Inexpensive Rewards

To construct molecular graphs, we design an action space with 5 action types. The agent can add a node (heavy atom) to the graph, add an edge between two nodes, set a node's properties (e.g. its chirality), set a bond's properties (i.e. its bond order), or stop the trajectory. We use a graph neural network [35] to parameterize a policy with such actions, using the GNN's invariances to produce per-node and per-edge logits. We are also careful to mask these logits such that the produced molecules always have valid valences (i.e. by design the molecules are always convertible to RDKit

molecules and SMILES). Since our method generates molecules atom by atom, we refer to the approach as **Atomic GFlowNet (A-GFN)**.

4.1 Inexpensive Molecular Rewards

In the context of pre-training A-GFN models for molecular design, we utilize inexpensive molecular rewards—such as Topological Polar Surface Area (TPSA), Quantitative Estimate of Drug-likeness (QED), synthetic accessibility (SAS), and the number of five or six-membered rings in a molecule. These rewards are computationally cheap to evaluate and serve as proxies for more complex properties. Fine-tuning the models is then conducted on more expensive and computationally intensive tasks, such as predicting binding affinity or toxicity (e.g., LD50), which are crucial for drug discovery but require significant computational resources or experimental data to assess accurately.

4.1.1 Reward Function

In order to pre-train a goal-conditioned A-GFN for learning molecular properties $p \in P$, we need to define the property-specific conditional ranges $c_p = (c_{low}, c_{high})$ from which molecules are generated. We use the following goal-conditioned reward function for property p and molecule x:

$$R_p(x|c_p, \vec{d} > 0) = \text{reward}(p_x \mid c_p, \vec{d} > 0) = \begin{cases} 0.5 * \exp\left(-\frac{(\mathrm{c}_{low} - p_x)}{\lambda}\right) & \text{if } p_x < \mathrm{c}_{low} \\ \exp\left(-\frac{(p_x - \mathrm{c}_{high})}{\lambda}\right) & \text{if } p_x > \mathrm{c}_{high} \\ \frac{0.5*(p_x - \mathrm{c}_{low})}{(\mathrm{c}_{high} - \mathrm{c}_{low})} + 0.5 & \text{otherwise} \end{cases} \tag{2}$$

here, λ controls the decay rate, c_{low} and c_{high} are the lower and upper bounds for property p, and $\vec{d} \in \mathbb{R}$ represents the $preference\_direction$ hyperparameter, indicating whether lower or higher property values within the range c_p are preferred (for further details, see appendix A,B). Note that during training, we sample these ranges so that the model learns to be robust to inference-time queries (sec 4.1.2).

Drug discovery is inherently a multiobjective optimization problem where the drug candidates are expected to simultaneously have several desired properties, such as a high drug-likeliness (QED), high SAS, and reasonably low TPSA, among other criteria. In our de novo molecular generation setup, we wish to satisfy the same multiobjective desiderata [3]. We choose this aggregated scalarization of the multiobjective reward over property set P as

$$R(x|c_{p_1}, ..., c_{p_{|P|}}) = \prod_{p \in P} R_p(x|c_p) \tag{3}$$

4.1.2 Reward conditioning A-GFN

To alleviate the problem of sparse rewards in training A-GFNs for molecular graph generation with some hard constraints, we condition the sampler by using a distribution of goals derived from reasonable lower and upper bounds of the molecular property ranges we care about. This conditioning effectively narrows down the search space to regions of interest defined by these property ranges, ensuring that the generated molecules are not only diverse but also relevant to the specific objectives of the drug discovery process. In order to ensure that the A-GFN does not develop a selective bias towards specific values within a property range and instead explores the full range of possible values for each molecular property p, we sample the conditional vectors $c_{p,j}$ uniformly across their respective ranges. Specifically, for the j^{th} online trajectory in the batch, the lower and upper bounds $c_{p,j}^{low}, c_{p,j}^{high}$ for the property p are drawn from a uniform distribution as $c_{p,j}^{low}, c_{p,j}^{high} \sim U(c_{low}, c_{high})$, where c_{low} and c_{high} are the predefined desired lower and upper bounds for the property p. In cases where the values for properties p for the trajectory j leading to a valid molecular graph are known *a priori* (p_j) (e.g., from molecules in an offline dataset), $c_{p,j}$ are centered around these known values, i.e. $c_{p,j}^{low}, c_{p,j}^{high} \sim \mathcal{T}(p_j, \sigma_p, c_{low}, c_{high})$, where $\mathcal{T}$ is a truncated normal distribution centered at the value of property p calculated for trajectory j, σ_p is a hyperparmeter controlling the variance of $\mathcal{T}$ for property p.

Such a probabilistic goal-sampling strategy ensures that each trajectory within the batch is conditioned on a randomly selected sub-range within the broader property range. This enables our laid out objective of promoting exploration across the entire desired chemical space and preventing the A-GFN from overfitting to narrow regions within the chemical space. In order to further prevent

A-GFN from memorizing the specified property ranges but rather encourage it to learn to sample molecules within any arbitrary range, we provide negative samples by sampling out-of-bounds conditionals with a small probability. To this end, with $\epsilon \approx 0$, we sometimes sample $c_{p,j}^{low}, c_{p,j}^{high}$ from $\mathcal{U}(c_p^{low}, c_p^{high})$ and $\mathcal{U}(c_{p,j}^{low}, c_p^{high})$ respectively for the online trajectories, while for offline trajectories, these conditional bounds are sampled as following

$$(c_{p,j}^{low}, c_{p,j}^{high}) = \begin{cases} (c_p^{low}*, \mathcal{U}(c_p^{low}*, p_j)) & \text{if } B_i = 0 \\ (\mathcal{U}(p_j, c_p^{high}*), c_p^{high}*) & \text{if } B_i = 1 \end{cases}$$

where B_i is a Bernoulli random variable with $P(B_i = 0) = P(B_i = 1) = 0.5$ and $c_p^{low}*$ and $c_p^{high}*$ are minimum and maximum permissible values for molecular property p. Finally, $c_{p,j}$ is constructed as a vector by applying thermometer encoding [36] to the sampled scalar bounds $c_{p,j}^{low}$ and $c_{p,j}^{high}$.

4.2 Pretraining GFN with expert offline trajectories

Bengio et al. [1] employ replay buffer-based off-policy training for GFlowNets to generate novel molecules. However, such online off-policy methods can suffer from high variance [37, 38] and and a lock-in of suboptimal trajectories, particularly in sparse reward settings, where the agent struggles to adequately explore rare, high-reward regions of the state space, resulting in slow convergence and suboptimal performance. To mitigate these challenges, we propose leveraging the vast amounts of readily available inexpensive and unlabelled molecular data to perform a hybrid online-offline off-policy pretraining of A-GFN. This data provides valuable expert trajectories and the molecular properties derived from this data can provide inexpensive extrinsic rewards, giving a better starting point for exploration. We form our training batches by integrating offline expert trajectories from the ZINC dataset ($\mathcal{D}_{ZINC}$) with online updates ($\tau = \tau_{online} \oplus \tau_{offline}$). $\tau_{offline}$ are generated from these molecules, $x \in \mathcal{D}_{ZINC}$ by sampling deleterious actions {`deleteNode`, `deleteEdge`, `removeNodeAttribute`, `removeEdgeAttribute`} according to a conditional backward policy P_B as $\tau_{offline} \sim P_B(. \mid x, c_p; \theta)$. Likewise, online trajectories are created by sampling constructive actions {`addNode`, `addEdge`, `addNodeAttribute`, `addEdgeAttribute`, `stop`} from conditional forward policy P_F as $\tau_{online} \sim P_F(. \mid c_p; \theta)$.

4.3 Regularized Loss Balancing for Exploration and Synthesis Feasibility

The primary allure of molecular generative models with large explorative capacities such as A-GFN is their ability to navigate the near-infinite possibilities of chemical structures. On the other hand, ensuring that generated molecules lie close to the chemical space feasible for synthesis, particularly within the constraints of make-on-demand (MOD) libraries such as ZINC, is crucial for synthesis and *in vitro* validation in drug discovery. To balance these two seemingly conflicting goals, we introduce a regularization term in the pretraining objective. Specifically, we combine the exploration objective with a Maximum Likelihood Estimation (MLE) loss over the offline dataset, leading to the following regularized loss function:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{TB} + \lambda_2 \mathcal{L}_{MLE} \tag{4}$$
$$\textit{where, } \mathcal{L}_{MLE} = -\log(P_F(. \mid x, c_p)) \forall x \in D_{ZINC}$$

$\mathcal{L}_{TB}$ encourages exploration of the chemical space, $\mathcal{L}_{MLE}$ is the distributional learning loss term ensuring proximity to MOD libraries, and λ_1, λ_2 are hyperparameters controlling the trade-off between exploration and adherence to the MOD space (Tab.8). With the online and offline trajectories described in sec4.2 and molecular property rewards in sec4.1, we optimize the prior A-GFN ($\mathcal{G}_\theta$) by minimizing Eq.4 until convergence in reward.

5 Finetuning

In this section, we investigate the methodology for utilizing the pre-trained A-GFN model ($\mathcal{G}_\theta$) and its subsequent adaptation to downstream drug discovery tasks based on harder reward functions. To fine-tune the model, the pre-trained parameters of $\mathcal{G}_\theta$ serve as the initialization for task-specific adaptation. This initialization enables the model to retain useful structural priors from pretraining, thus improving sample efficiency and convergence speed during fine-tuning. In particular, we retrain $\mathcal{G}_\theta$ by integrating task-specific reward R_{ext}. This modifies eq.3 as

$$R(x|c_{p_1}, ..., c_{p_{|P|}}) = \prod_{p \in P} R_p(x) \times R_{ext}(x) \tag{5}$$

Such a reward formulation ensures that the GFlowNet receives a high reward for generating molecules that simultaneously follow the desired molecular properties and are highly suitable for the downstream

task. It should be noted that Z_θ in eq 1 is a global scalar that estimates the normalization constant for the unnormalized reward function $R(x)$ $\big($i.e, $Z_\theta = \sum_{x \in \mathcal{X}} R(x)\big)$. Thus, to enable Z_θ to generalize to the new $R(x)$, we inject noise into the pretrained A-GFN model's parameters, a common strategy in pretrain-then-finetune approaches [39, 40].

6 Experiments

We evaluate A-GFN's effectiveness during both pretraining and fine-tuning using comprehensive metrics such as novelty, diversity, uniqueness, success rate, validity, L1-distance, and number of modes. Full details are provided in appendix B.

For pretraining, we compare the performance of A-GFN against fragment-based GFlowNet, conditioned on the same molecular properties. Our results show that A-GFN significantly outperforms fragment-GFN in exploring drug-like chemical space across multiple objectives. For fine-tuning, we benchmark the pre-trained A-GFN against A-GFN trained from scratch (task-trained A-GFN) on several downstream drug discovery tasks. Following the task setup in [33], we focus on the following finetuning objectives:

Property Optimization: The goal here is to generate molecules that maximize or minimize a specified physicochemical, structural, or binding property while ensuring diversity among the generated molecules. In this unconstrained optimization setting, we compare the fine-tuned A-GFN to the task-trained A-GFN and other state-of-the-art methods for unconstrained molecule generation. For fairness, we only include atom-based generative methods in the baselines, excluding fragment-based approaches.

Property Targeting: In this task, the objective is to generate molecules that adhere to predefined molecular property ranges while being structurally distinct from the training (pretraining) set.

Property Constrained Optimization: This task requires the generation of molecules that simultaneously meet both property optimization and property targeting criteria i.e. molecules must lie within the specified property ranges while also maximizing or minimizing the targeted property.

6.1 Pretraining

A-GFN pretraining setup: To create a versatile foundation for a range of downstream molecular generation tasks, we train our A-GFN using a hybrid online-offline off-policy strategy. For our pretraining, we utilize ZINC250K, a curated subset of the ZINC database, which consists of 250,000 commercially accessible drug-like compounds drawn from over 37 billion molecules available in ZINC [41]. The primary goal during pretraining is to optimize the A-GFN for generic yet critical drug-like properties: TPSA, QED, SAS, and the number of rings. The desired ranges for these properties are enumerated in Tab.5. These properties are chosen based on their established relevance in guiding molecular design towards compounds with desirable pharmacokinetic and pharmacodynamic profiles, following the framework of [42]. By optimizing for these properties, we aim to equip A-GFN with the capability to generate molecules that strike a balance between drug-likeness and structural diversity. We restrict atom types to a core set commonly found in drug-like molecules: C, S, P, N, O, F, and implicit hydrogen (H). This ensures that the generated molecules are synthetically relevant and pharmacologically plausible, avoiding rare or exotic atom types that are less likely to lead to viable drug candidates. For benchmarking fragment-based GFlowNets, we adapt the purely online training setup proposed in [1], conditioning it on the same molecular properties as A-GFN. To align with our A-GFN setup, we generate a fragment vocabulary from the BRICS decomposition of ZINC250K, selecting the 73 (following [1]) most common fragments. This equips the fragment-based GFlowNet with a diverse and representative set of building blocks for molecular generation. Through pretraining, we observe that A-GFN effectively adapts to the specified property ranges while maintaining high molecular diversity and uniqueness. All molecules generated by A-GFN are valid, adhering to chemical rules, and novel, as they do not replicate any molecules in the ZINC250K dataset (Fig.2). For the same set of property conditionals, A-GFN demonstrates superior chemical scaffold exploration compared to the fragment-based GFlowNet, covering nearly twice as many distinct scaffolds (Tab.1). While the fragment-based method has a higher success rate and better control over specific molecular properties, A-GFN excels in uniqueness and novelty, making it a more powerful tool for exploring uncharted chemical space in drug discovery. This highlights the model's capacity to explore novel regions of chemical space while adhering to fundamental molecular design principles.

6.2 Fine-tuning

Fine-tuning setup We split the fine-tuning tasks' objectives as property optimization, property targeting, and property constrained optimization. The primary distinction between these three

Table 1: Comparing pretraining A-GFN to pertaining a fragment-based GFlowNet for same molecular property conditionals. Pretraining was performed on Nvidia A100-40G GPUs.

Method	**N_modes**	**Diversity**	**Success %**	**L1-dist (↓)**				**Uniqueness**	**Novelty**	**Validity**	**Scaffolds (n=200)**	**Time (GPU-hours)**
				TPSA	Num Rings	SAS	QED					
Fragment GFlowNet	0	0.002	**50.00**	0.00	1.50	0.00	0.71	0.001	0.97	0.99	113	514.0
A-GFN	**1252**	**0.88**	45.09	0.03	0.49	0.24	0.67	**0.96**	**1.0**	1.0	**196**	

Table 2: Comparing fine-tuned A-GFN with other baselines for logP and molecular weight property optimization task.

Task	**Method**	**N_modes**	**Diversity**	**Success %**	**L1-dist (↓) task**	**Uniqueness**	**Novelty**	**Validity**	**Time (GPU-hours)**
logP	GCPN	154	**0.91**	2.54	0.45	**1.0**	**1.0**	1.0	40.0
	Reinvent	28	0.90	0.34	0.41	0.93	0.99	0.99	4.0
	A-GFN Task trained	928	0.89	13.02	0.17	**1.0**	**1.0**	1.0	20.0
	A-GFN Finetuned	**1199**	0.87	**16.67**	**0.19**	0.99	**1.0**	1.0	20.0
Molecular Weight	GCPN	223	0.92	4.02	0.14	**1.0**	**1.0**	1.0	40.0
	Reinvent	1	0.88	0.02	0.33	0.97	0.99	0.99	4.0
	A-GFN Task trained	9	0.93	0.88	0.40	**1.0**	**1.0**	1.0	8.0
	A-GFN Finetuned	**1157**	**0.94**	**6.58**	**0.05**	0.95	**1.0**	1.0	8.0

setups is in terms of the corresponding conditionals and reward function. For the tasks where some small labeled dataset is accessible, we investigate the benefits of offline data on fine-tuning atomic-GFlowNets. Similar to GCPN [33], we consider two structural tasks, molecular weight (mol.wt.), and logP (partition coefficient for drug's water to octanol concentration [43]). In addition, we also consider other standard drug-discovery tasks where rewards are based on empirical models[44] such as the LD50 (toxicity) task of [45]. For each task and objective pair, we aim to show that fine-tuning a pretrained GFlowNet achieves the task's objectives more quickly than training a new GFlowNet from scratch. We now define each objective in detail.

6.2.1 Property Optimization

In this task, the goal is to generate novel molecules that minimize specific physicochemical properties. While previous works have commonly benchmarked their models on QED optimization [46, 32, 33], we exclude QED from our evaluation, as our pretrained A-GFN is already optimized for this metric. Instead, we focus on mol.wt. and logP—two critical properties in drug discovery. To further challenge the models, we set the target property ranges to be slightly below the minimum values found in the Zinc250K dataset, with logP in the range [-5, -4.5] and molecular weight in the range [100, 110]. The success percentage is calculated based on the proportion of generated molecules that fall within these strict property ranges. The low success percentages across all methods reflect the challenging nature of the task (Tab.2). However, the fine-tuned A-GFN shows a relatively higher success rate compared to other methods, combined with a lower-L1 distance from the target ranges. This indicates that even in cases where exact matches are not achieved, the fine-tuned A-GFN generates molecules that are close to the desired property values. A-GFN's ability to sample from these "out-of-domain" chemical spaces—where molecules fall outside the property ranges in MOD datasets like Zinc250K—suggests that fine-tuned A-GFN excels in generating diverse molecules that could extend the scope of current chemical libraries. Moreover, the high diversity, novelty, and validity of the molecules generated by A-GFN indicate its capacity to produce a broad spectrum of unique, structurally valid compounds, essential for mitigating structural redundancy in drug candidates. Fine-tuning further enhances these aspects, highlighting A-GFN's potential for property-driven molecular design.

6.2.2 Property Targeting

Here, we generate molecules that satisfy specific molecular property constraints, demonstrating the model's capacity for fine-tuning and adaptation to new objectives. Compared to training A-GFN from scratch, fine-tuning a pretrained A-GFN leads to significantly faster convergence, even when property ranges are altered. In particular, we focus on modifying the TPSA property while keeping the constraints for other molecular properties the same as those used in pretraining.

To evaluate the model's ability to generalize to new property constraints, we alter the TPSA range from its pretraining interval of [60, 100] to both lower [40, 60] and higher [100, 120] intervals. Our experiments demonstrate that the fine-tuned A-GFN consistently outperforms the model trained from scratch in terms of convergence speed and overall performance. This is particularly evident in the number of distinct high-reward molecular modes discovered (N_modes), molecular diversity, and success percentage, all of which show substantial improvements after fine-tuning (Tab.6). The ability of the fine-tuned A-GFN to adapt to modified property constraints highlights the robustness of the pretrained $\mathcal{G}_\theta$ model, which has not merely memorized specific property ranges but has learned a more generalizable sampling strategy. This adaptability is crucial in real-world drug discovery applications, where molecular property requirements often shift during the drug discovery process.

6.2.3 Property Constrained Optimization

This comprehensive benchmarking task aims to generate molecules that minimize a target property (e.g., mol.wt., logP, or toxicity) within a predefined range, while maintaining the drug-like characteristics encoded during pretraining. We evaluate two key scenarios: conditionals-preserved fine-tuning and Dynamic Range adjustment (DRA). In the conditionals-preserved fine-tuning setup, we retain the same conditional property ranges (c_p) used during pretraining, setting the task's $preference\_direction$ to -1, which directs the model to minimize the target property. The desired ranges for these task properties are set between the 25^{th} percentile of the ZINC dataset and a predetermined maximum threshold (see Tab.3 and 7 for specifics). The fine-tuned A-GFN significantly outperforms the A-GFN trained from scratch, achieving superior results within the same computational budget. In the dynamic range adjustment scenario, we modify one of the pretraining conditionals (TPSA) by shifting its range from $60 \leq TPSA \leq 100$ to both lower ($40 \leq TPSA \leq 60$) and higher ($100 \leq TPSA \leq 120$) values. In this case, fine-tuned A-GFN again surpasses its scratch-trained counterpart, demonstrating faster convergence and higher success rates (Tab.4). We also explore a hybrid online-offline fine-tuning approach, where A-GFN leverages offline task-specific data with the desired c_p, similar to its pretraining setup. In most tasks, hybrid fine-tuning shows comparable performance to fully online fine-tuning, with notable exceptions in more complex tasks like logP optimization when $100 \leq TPSA \leq 120$; it is plausible that in such a case grounding the model in data stabilizes early learning when rewards are low. In this challenging case, the goal was to generate molecules with logP$\approx$1.5 while maintaining the conditional properties outlined in Tab.5. The inherent difficulty of this task is underscored by the fact that only 0.002% of molecules in the ZINC250K dataset meet these stringent criteria. Despite the challenge, A-GFN fine-tuned with a small offline dataset of expert trajectories achieve a respectable success rate and uncover diverse modes in the chemical space. This underscores the potential of hybrid online-offline fine-tuning to unlock otherwise inaccessible regions of the chemical landscape, offering a promising strategy for tackling difficult-to-sample molecular spaces.

7 Conclusion

In this work, we introduced Atomic GFlowNets (A-GFN), an extension of the GFlowNet framework that leverages atoms as fundamental building blocks to explore molecular space. By shifting the action space from predefined molecular fragments to individual atoms, A-GFN is able to explore a much larger chemical space, enabling the discovery of more diverse and pharmacologically relevant molecules. To address the challenges posed by the vastness of this atomic action space, we propose a pretraining strategy using datasets of drug-like molecules. This off-policy pretraining approach conditions A-GFN on informative molecular properties such as drug-likeness, topological polar surface area, and synthetic accessibility, allowing it to effectively explore regions of chemical space that are more likely to yield viable drug candidates.

Our experimental results demonstrate that pretraining A-GFN with these expert trajectories leads to improved diversity and novelty in the generated molecules. Furthermore, we show that fine-tuning A-GFN for specific property optimization tasks offers significant computational efficiency compared to training from scratch. However, akin to challenges in fine-tuning RL models [47], A-GFN is prone

Table 3: Comparing the effectiveness of finetuned A-GFN over A-GFN trained from scratch for conditional preserved finetuning.

Task	**Method**	**N_modes**	**Diversity**	**Success %**	**L1-dist (↓)**					**Uniqueness**	**Novelty**	**Validity**	**Time (GPU-hours)**
					TPSA	Num Rings	SAS	QED	Task				
Molecular Weight	A-GFN Task Trained	0	**0.92**	15.71	0.33	1.50	1.96	2.32	0.14	0.95	1.0	1.0	24.0
	A-GFN Finetuned	340	0.71	**43.09**	0.06	0.75	0.12	0.99	0.04	0.62	1.0	1.0	
	A-GFN Finetuned w/ data	**963**	0.86	35.52	0.03	0.39	0.41	0.94	0.08	**1.0**	1.0	1.0	24.0
logP	A-GFN Task Trained	0	**0.96**	20.32	0.64	1.18	1.98	2.59	0.48	0.75	1.0	1.0	1.0
	A-GFN Finetuned	**1286**	0.86	41.20	0.04	0.45	0.31	0.49	0.20	**1.0**	1.0	1.0	
	A-GFN Finetuned w/ data	1080	0.87	**43.47**	0.05	0.49	0.28	0.65	0.31	0.98	1.0	1.0	1.0

Table 4: Comparing the effectiveness of finetuned A-GFN over A-GFN trained from scratch for dynamic range adjustment property constrained optimization finetuning.

Task	**Adjusted range**	**Method**	**N_modes**	**Diversity**	**Success %**	**L1-dist (↓)**					**Uniqueness**	**Novelty**	**Validity**	**Time (GPU-hours)**
						TPSA	Num Rings	SAS	QED	Task				
Molecular Weight	TPSA $\in [100, 120]$	A-GFN Task Trained	0	**0.94**	16.80	1.71	1.5	1.87	2.48	0.21	0.88	1.0	1.0	16.0
		A-GFN Finetuned	**873**	0.79	**51.70**	0.16	0.34	0.10	0.59	0.06	0.94	1.0	1.0	
		A-GFN Finetuned w/ data	401	0.87	34.70	0.23	0.34	0.36	0.88	0.09	**1.0**	1.0	1.0	16.0
	TPSA $\in [40, 60]$	A-GFN Task Trained	0	**0.93**	11.88	0.72	1.50	1.87	2.02	0.20	0.89	1.0	1.0	16.0
		A-GFN Finetuned	**1034**	0.82	**57.06**	0.10	0.26	0.15	0.66	0.04	0.91	1.0	1.0	
		A-GFN Finetuned w/ data	256	0.87	38.14	0.09	0.35	0.49	0.41	0.09	**1.0**	1.0	1.0	16.0
logP	TPSA $\in [100, 120]$	A-GFN Task Trained	25	0.74	16.64	0.47	1.5	0.62	0.94	0.30	0.84	1.0	1.0	72.0
		A-GFN Finetuned	1	0.02	20.08	0.09	1.5	0.11	2.17	0.04	0.01	1.0	1.0	
		A-GFN Finetuned w/ data	**1309**	**0.86**	**35.73**	0.32	0.26	0.30	1.37	0.26	**1.0**	1.0	1.0	24.0
	TPSA $\in [40, 60]$	A-GFN Task Trained	0	0.65	27.50	1.00	1.50	0.02	0.86	0.36	0.35	1.0	1.0	24.0
		A-GFN Finetuned	257	0.60	**58.0**	0.03	1.14	0.03	0.87	0.07	0.17	1.0	1.0	
		A-GFN Finetuned w/ data	**1253**	**0.89**	46.29	0.13	0.61	0.28	0.50	0.39	**0.91**	1.0	1.0	28.0

to catastrophic forgetting of pre-trained knowledge during extended fine-tuning. This suggests a need for methods like relative trajectory balance [48] to regularize the fine-tuning process, ensuring the policy stays anchored to the pretrained policy while still adapting to task-specific goals. Within the context of A-GFN, an obvious future direction is to extend it to lead optimization in drug discovery, where the goal is to improve candidate molecules' binding affinity, toxicity profiles, and synthetic accessibility.

References

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

[2] Yiheng Zhu, Jialu Wu, Chaowen Hu, Jiahuan Yan, Tingjun Hou, Jian Wu, et al. Sample-efficient multi-objective molecular optimization with gflownets. *Advances in Neural Information Processing Systems*, 36, 2024.

[3] Moksh Jain, Sharath Chandra Raparthy, Alex Hernández-Garcıa, Jarrid Rector-Brooks, Yoshua Bengio, Santiago Miret, and Emmanuel Bengio. Multi-objective gflownets. In *International conference on machine learning*, pages 14631–14653. PMLR, 2023.

[4] Tony Shen, Seonghwan Seo, Grayson Lee, Mohit Pandey, Jason R Smith, Artem Cherkasov, Woo Youn Kim, and Martin Ester. TacoGFN: Target-conditioned GFlownet for structure-based drug design. *Transactions on Machine Learning Research*, 2024.

[5] Varnavas D Mouchlis, Antreas Afantitis, Angela Serra, Michele Fratello, Anastasios G Papadiamantis, Vassilis Aidinis, Iseult Lynch, Dario Greco, and Georgia Melagraki. Advances in de novo drug design: from conventional to machine learning methods. *International journal of molecular sciences*, 22(4):1676, 2021.

[6] Joshua Meyers, Benedek Fabian, and Nathan Brown. De novo molecular design and generative models. *Drug discovery today*, 26(11):2707–2715, 2021.

[7] Yoshihiko Nishibata and Akiko Itai. Automatic creation of drug candidate structures based on receptor structure. starting point for artificial lead generation. *Tetrahedron*, 47(43):8985–8990, 1991.

[8] Jan D Bos and Marcus MHM Meinardi. The 500 dalton rule for the skin penetration of chemical compounds and drugs. *Experimental Dermatology: Viewpoint*, 9(3):165–169, 2000.

[9] G Richard Bickerton, Gaia V Paolini, Jérémy Besnard, Sorel Muresan, and Andrew L Hopkins. Quantifying the chemical beauty of drugs. *Nature chemistry*, 4(2):90–98, 2012.

[10] 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, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 26412–26428. PMLR, 17–23 Jul 2022.

[11] Nikolay Malkin, Salem Lahlou, Tristan Deleu, Xu Ji, Edward Hu, Katie Everett, Dinghuai Zhang, and Yoshua Bengio. Gflownets and variational inference, 2023.

[12] Nikolay Malkin, Moksh Jain, Emmanuel Bengio, Chen Sun, and Yoshua Bengio. Trajectory balance: Improved credit assignment in gflownets. *Advances in Neural Information Processing Systems*, 35:5955–5967, 2022.

[13] 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.

[14] 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 *Proceedings of the 39th International Conference on Machine Learning*, pages 9786–9801. PMLR, 2022.

[15] Tristan Deleu, António Góis, Chris Emezue, Mansi Rankawat, Simon Lacoste-Julien, Stefan Bauer, and Yoshua Bengio. Bayesian structure learning with generative flow networks. In James Cussens and Kun Zhang, editors, *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, volume 180 of *Proceedings of Machine Learning Research*, pages 518–528. PMLR, 2022.

[16] Lazar Atanackovic, Alexander Tong, Bo Wang, Leo J. Lee, Yoshua Bengio, and Jason Hartford. Dyngfn: Towards bayesian inference of gene regulatory networks with gflownets, 2023.

[17] Edward J. Hu, Nikolay Malkin, Moksh Jain, Katie Everett, Alexandros Graikos, and Yoshua Bengio. Gflownet-em for learning compositional latent variable models, 2023.

[18] David W. Zhang, Corrado Rainone, Markus Peschl, and Roberto Bondesan. Robust scheduling with gflownets, 2023.

[19] Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[20] András Kalapos and Bálint Gyires-Tóth. Cnn-jepa: Self-supervised pretraining convolutional neural networks using joint embedding predictive architecture. *arXiv preprint arXiv:2408.07514*, 2024.

[21] Hao Liu and Pieter Abbeel. Behavior from the void: Unsupervised active pre-training, 2021.

[22] Mirco Mutti, Mattia Mancassola, and Marcello Restelli. Unsupervised reinforcement learning in multiple environments, 2021.

[23] Junsu Kim, Seohong Park, and Sergey Levine. Unsupervised-to-online reinforcement learning, 2024.

[24] Ling Pan, Moksh Jain, Kanika Madan, and Yoshua Bengio. Pre-training and fine-tuning generative flow networks. *arXiv preprint arXiv:2310.03419*, 2023.

[25] Haoran He, Can Chang, Huazhe Xu, and Ling Pan. Looking backward: Retrospective backward synthesis for goal-conditioned gflownets, 2024.

[26] Julien Roy, Pierre-Luc Bacon, Christopher Pal, and Emmanuel Bengio. Goal-conditioned gflownets for controllable multi-objective molecular design, 2023.

[27] Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *International conference on machine learning*, pages 1312–1320. PMLR, 2015.

[28] Camille Bilodeau, Wengong Jin, Tommi Jaakkola, Regina Barzilay, and Klavs F Jensen. Generative models for molecular discovery: Recent advances and challenges. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 12(5):e1608, 2022.

[29] Marcus Olivecrona, Thomas Blaschke, Ola Engkvist, and Hongming Chen. Molecular de-novo design through deep reinforcement learning. *Journal of cheminformatics*, 9:1–14, 2017.

[30] Mariya Popova, Olexandr Isayev, and Alexander Tropsha. Deep reinforcement learning for de novo drug design. *Science advances*, 4(7):eaap7885, 2018.

[31] Manan Goel, Shampa Raghunathan, Siddhartha Laghuvarapu, and U Deva Priyakumar. Molegular: molecule generation using reinforcement learning with alternating rewards. *Journal of Chemical Information and Modeling*, 61(12):5815–5826, 2021.

[32] Zhenpeng Zhou, Steven Kearnes, Li Li, Richard N Zare, and Patrick Riley. Optimization of molecules via deep reinforcement learning. *Scientific reports*, 9(1):10752, 2019.

[33] Jiaxuan You, Bowen Liu, Zhitao Ying, Vijay Pande, and Jure Leskovec. Graph convolutional policy network for goal-directed molecular graph generation. *Advances in neural information processing systems*, 31, 2018.

[34] Sara Romeo Atance, Juan Viguera Diez, Ola Engkvist, Simon Olsson, and Rocío Mercado. De novo drug design using reinforcement learning with graph-based deep generative models. *Journal of chemical information and modeling*, 62(20):4863–4872, 2022.

[35] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[36] Jacob Buckman, Aurko Roy, Colin Raffel, and Ian Goodfellow. Thermometer encoding: One hot way to resist adversarial examples. In *International conference on learning representations*, 2018.

[37] William Fedus, Prajit Ramachandran, Rishabh Agarwal, Yoshua Bengio, Hugo Larochelle, Mark Rowland, and Will Dabney. Revisiting fundamentals of experience replay. In *International conference on machine learning*, pages 3061–3071. PMLR, 2020.

[38] Nikhil Vemgal, Elaine Lau, and Doina Precup. An empirical study of the effectiveness of using a replay buffer on mode discovery in gflownets. *arXiv preprint arXiv:2307.07674*, 2023.

[39] Hongyi Yuan, Zheng Yuan, Chuanqi Tan, Fei Huang, and Songfang Huang. HyPe: Better pre-trained language model fine-tuning with hidden representation perturbation. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3246–3264, Toronto, Canada, July 2023. Association for Computational Linguistics.

[40] Shoujie Tong, Qingxiu Dong, Damai Dai, Tianyu Liu, Baobao Chang, Zhifang Sui, et al. Robust fine-tuning via perturbation and interpolation from in-batch instances. *arXiv preprint arXiv:2205.00633*, 2022.

[41] Benjamin I Tingle, Khanh G Tang, Mar Castanon, John J Gutierrez, Munkhzul Khurelbaatar, Chinzorig Dandarchuluun, Yurii S Moroz, and John J Irwin. Zinc-22-a free multi-billion-scale database of tangible compounds for ligand discovery. *Journal of chemical information and modeling*, 63(4):1166–1176, 2023.

[42] James Wellnitz, Holli-Joi Martin, Mohammad Anwar Hossain, Marielle Rath, Colton Fox, Konstantin I Popov, Timothy M Willson, Eugene N Muratov, and Alexander Tropsha. Stoplight: A hit scoring calculator. *Journal of Chemical Information and Modeling*, 2024.

[43] James M Sangster. *Octanol-water partition coefficients: fundamentals and physical chemistry*, volume 1. John Wiley & Sons, 1997.

[44] Kexin Huang, Tianfan Fu, Wenhao Gao, Yue Zhao, Yusuf Roohani, Jure Leskovec, Connor W Coley, Cao Xiao, Jimeng Sun, and Marinka Zitnik. Therapeutics data commons: Machine learning datasets and tasks for drug discovery and development. *arXiv preprint arXiv:2102.09548*, 2021.

[45] Hao Zhu, Todd M Martin, Lin Ye, Alexander Sedykh, Douglas M Young, and Alexander Tropsha. Quantitative structure- activity relationship modeling of rat acute toxicity by oral exposure. *Chemical research in toxicology*, 22(12):1913–1921, 2009.

[46] Woosung Jeon and Dongsup Kim. Autonomous molecule generation using reinforcement learning and docking to develop potential novel inhibitors. *Scientific reports*, 10(1):22104, 2020.

[47] Maciej Wołczyk, Bartłomiej Cupiał, Mateusz Ostaszewski, Michał Bortkiewicz, Michał Zając, Razvan Pascanu, Łukasz Kuciński, and Piotr Miłoś. Fine-tuning reinforcement learning models is secretly a forgetting mitigation problem. *arXiv preprint arXiv:2402.02868*, 2024.

[48] Siddarth Venkatraman, Moksh Jain, Luca Scimeca, Minsu Kim, Marcin Sendera, Mohsin Hasan, Luke Rowe, Sarthak Mittal, Pablo Lemos, Emmanuel Bengio, et al. Amortizing intractable inference in diffusion models for vision, language, and control. *arXiv preprint arXiv:2405.20971*, 2024.

[49] Nathan Brown, Marco Fiscato, Marwin HS Segler, and Alain C Vaucher. Guacamol: benchmarking models for de novo molecular design. *Journal of chemical information and modeling*, 59(3):1096–1108, 2019.

Appendix

A Conditionals and Rewards

When higher values of a molecular property p or task, within desired range $c_p = (c_{low}, c_{high})$ are preferred, $preference\_direction$ is set to $\vec{d} > 0$ and the corresponding reward is as defined in eq.2. Similarly, when lower values are preferred,

$$R_p(x|c_p, \vec{d} < 0) = \text{reward}(p_x \mid c_p, \vec{d} < 0) = \begin{cases} \exp\left(-\frac{(\text{c}_{low} - p_x)}{\lambda}\right) & \text{if } p_x < \text{c}_{low} \\ 0.5 * \exp\left(-\frac{(p_x - \text{c}_{high})}{\lambda}\right) & \text{if } p_x > \text{c}_{high} \\ \frac{-0.5*(p_x - \text{c}_{low})}{(\text{c}_{high} - \text{c}_{low})} + 1 & \text{otherwise} \end{cases} \tag{6}$$

When there is no preference i.e. $\vec{d} = 0$, the reward is defined as

$$R_p(x|c_p, \vec{d} < 0) = \text{reward}(p_x \mid c_p, \vec{d} < 0) = \begin{cases} \exp\left(-\frac{(\text{c}_{low} - p_x)}{\lambda}\right) & \text{if } p_x < \text{c}_{low} \\ \exp\left(-\frac{(p_x - \text{c}_{high})}{\lambda}\right) & \text{if } p_x > \text{c}_{high} \\ 1 & \text{otherwise} \end{cases} \tag{7}$$

The rate parameter λ is set for each property individually- λ_{QED}=λ_{SAS}=$\lambda_{NumRings} = 1$, and λ_{TPSA}=20.

Table 5: Task desired property ranges for different fine-tuning objectives. The QED, SAS, and Num Rings conditional ranges are [0.65, 0.8, 0], [1,3,0], and [1,3,1] respectively across all experiments where the ordering is [c_{low},c_{high},$\vec{d}$]

Task	Finetuning Objective	TPSA	Task Range
Mol.Wt.	Property Optimization	[60,100,0]	[100,800,-1]
	Preserved Property Constrained Optimization	[60,100,0]	[302,800,-1]
	DRA Property Constrained Optimization	[100,120,0]	[340,800,-1]
		[40,60,0]	[300,800,-1]
logP	Property Optimization	[60,100,0]	[-5,6,-1]
	Preserved Property Constrained Optimization	[60,100, 0]	[1.65,5,-1]
	DRA Property Constrained Optimization	[100,120,0]	[1.5,5,-1]
		[40,60,0]	[2.4,5,-1]
LD50	Property Optimization	[60,100,0]	[2,6,-1]
	Preserved Property Constrained Optimization	[60,100,0]	[2,6,-1]
	DRA Property Constrained Optimization	[100,120,0]	[1.68, 4.4,-1]
		[40,60,0]	[2,6,-1]

Table 6: Comparing the effectiveness of finetuned atomic-GFlowNet over atomic-GFlowNet trained from scratch for TPSA property targeting objective.

Adjusted Range	Method	N_modes	Diversity	Success %	L1-dist (↓)				Uniqueness	Novelty	Validity	Time (GPU-hours)
					TPSA	Num Rings	SAS	QED				
TPSA ∈ [100,120]	A-GFN Task Trained	0	0.01	24.71	0.01	1.5	0.62	2.40	0.01	1.0	1.0	73.0
	A-GFN Finetuned	254	0.60	46.30	0.07	1.50	0.01	1.23	0.50	1.0	1.0	
TPSA ∈ [40,60]	A-GFN Task Trained	0	0.39	46.83	0.01	1.5	0.00	1.35	0.04	1.0	1.0	10.0
	A-GFN Finetuned	616	0.83	55.60	0.10	0.22	0.13	0.55	0.80	1.0	1.0	

Table 7: Property constrained optimization for toxicity (LD50) task

Adjusted range	Method	N_modes	Diversity	Success %	L1-dist (↓)					Uniqueness	Novelty	Validity	Time (GPU-hours)
					TPSA	Num Rings	SAS	QED	Task				
Preserved	A-GFN tasktrained	0	0.37	57.91	0.02	1.2	0.01	1.56	0.74	0.24	0.24	1.0	24.0
	A-GFN finetuned	682	0.61	55.16	0.02	0.30	0.11	3.68	0.49	0.99	1.0	1.0	24.0
TPSA ∈ $[100, 120]$	A-GFN tasktrained	0	0.75	28.95	0.20	1.49	0.40	2.789	0.73	0.99	0.99	1.0	24.0
	A-GFN finetuned	308	0.70	38.52	0.30	1.49	0.05	4.06	0.67	0.96	1.0	1.0	24.0
TPSA ∈ $[40, 60]$	A-GFN tasktrained	0	0.04	20	0.48	1.5	0.039	2.38	0.61	0.002	0.002	1.0	24.0
	A-GFN finetuned	493	0.63	47.61	0.36	0.09	0.07	2.68	0.47	0.70	1.0	1.0	24.0

B Evaluation Metrics

In line with earlier works [49, 33], we employ following set of metrics to ensure a robust comparison across different generative methods.

Validity: A molecule is considered valid if it successfully passes RDKit's sanitization checks. Note that in the proposed method, masking makes this trivially 1.

Diversity: For a set of generated molecules, diversity is defined based on the Tanimoto similarity of Morgan Fingerprint representation of molecules.

$$Diversity = 1 - \frac{2}{N(N-1)} \sum_{1 \leq i \leq j \leq N} \frac{|M_i \cap M_j|}{|M_i \cup M_j|}$$

Uniqueness: The ratio of distinct canonicalized SMILES strings (without stereochemistry) to the total number of generated molecules, after filtering out duplicates and invalid structures.

Novelty: The ratio of unique generated molecules, that are not present in the pretraining ZINC dataset, to the total number of generated molecules.

Time: For each task, a fixed time budget is allotted to A-GFN for fine-tuning and task-training. Other baseline methods are allowed to run for time needed to run their default configuration.

N_Modes: This metric quantifies the number of distinct, high-reward molecular modes identified by generative model. A mode is defined as a molecule whose reward exceeds a threshold, typically set at the 75^{th} percentile of the reward distribution achieved by the best-performing model for the task. For a molecule to be counted as a new mode, its Tanimoto similarity to any previously identified mode must be less than a specified threshold (typically 0.7), ensuring that only sufficiently diverse molecules are counted.

Normalized L1-dist: This measures how far a generated molecule's properties deviate from their target ranges. For each property, the L1 distance is calculated by taking the absolute difference between the generated property value and the 10^{th} percentile value in desired range, and then normalized by the respective property range to ensure comparability across properties. This allows us to assess how well the generated molecules meet multi-objective constraints in a unified metric.

Success Percent: For a set of N generated graphs $\{G\}_{i=1}^{N}$ and conditionals C_{task} for given task, we

define success percentage as

$$S_{\text{task}} = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{1}{|\mathcal{C}_{\text{task}}|} \sum_{c \in \mathcal{C}_{\text{task}}} \mathbb{I}_c(G_i) \right) \times 100$$

where, $\mathbb{I}_c(G)$ counts the molecules within desired conditional ranges and defined as

$$\mathbb{I}_c(G) = \begin{cases} \mathbb{I}\left(|c(G) - c_{\text{low}}| \leq 0.1 \cdot |c_{\text{low}}|\right), & \text{if preference\_direction}(c) < 0 \\ \mathbb{I}\left(|c(G) - c_{\text{high}}| \leq 0.1 \cdot |c_{\text{high}}|\right), & \text{if preference\_direction}(c) > 0 \\ \mathbb{I}\left(c_{\text{low}} \leq c(G) \leq c_{\text{high}}\right), & \text{if preference\_direction}(c) = 0 \end{cases}$$

where, for any considered task, $c(G)$ is the the value of the conditional property c for the molecular graph G, c_{low} and c_{high} are the lower and upper bounds of c, $preference\_direction(c)$ indicates whether lower values (< 0), higher values (> 0), or values within a range ($= 0$) are preferred.

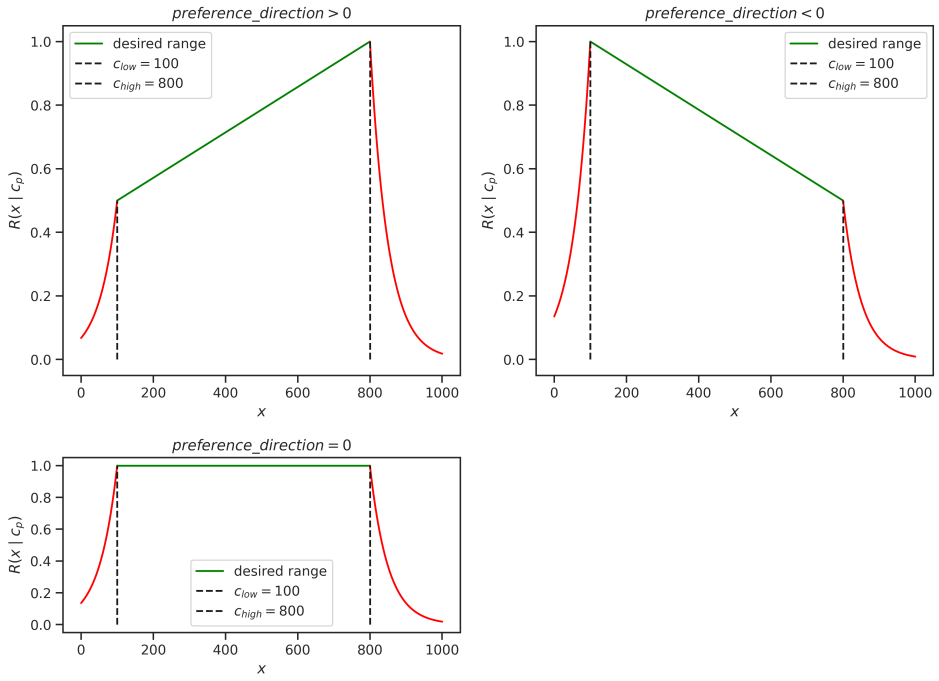

Figure 1: Visual representation of reward as a function of $preference\_direction$.

C Experiments

Table 8: Pretraining and finetuning setups and corresponding hyperparameters

Hyperparameter	**Pretraining Values**	**Finetuning Values**	**FT w/data Values**
max_num_iter	500,000		
bootstrap_own_reward	FALSE		
random_seed	1,428,570		
beta	96	64	
OOB_percent	0.1		
zinc_rad_scale (λ)	1		
gfn_batch_shuffle	FALSE		
reward_aggregation	mul		
sampling_batch_size	2048	1024	
training_batch_size	64		
learning_rate	1.00e-04		
online_offline_mix_ratio	0.5		
num_workers	8	2	
gfn_loss_coeff (λ_1)	0.04	-	0.04
MLE_coeff (λ_2)	20	-	20
num_emb	128		
num_layers	8		
num_mlp_layers	4		
num_heads	2		
i2h_width	1		
illegal_action_logreward	-512		
reward_loss_multiplier	1		
weight_decay	1.00e-08		
num_data_loader_workers	8		
momentum	0.9		
adam_eps	1.00e-08		
lr_decay	20,000		
Z_lr_decay	20,000		
clip_grad_type	norm		
clip_grad_param	10		
random_action_prob	0.001		
random_stop_prob	0.001		
num_back_steps_max	25		
max_traj_len	40		
max_nodes	45		
max_edges	50		
tb_p_b_is_parameterized	TRUE		
num_thermometer_dim	16		
sample_temp	1		
checkpoint_every	1,000	500	
Z_learning_rate	1e-3		

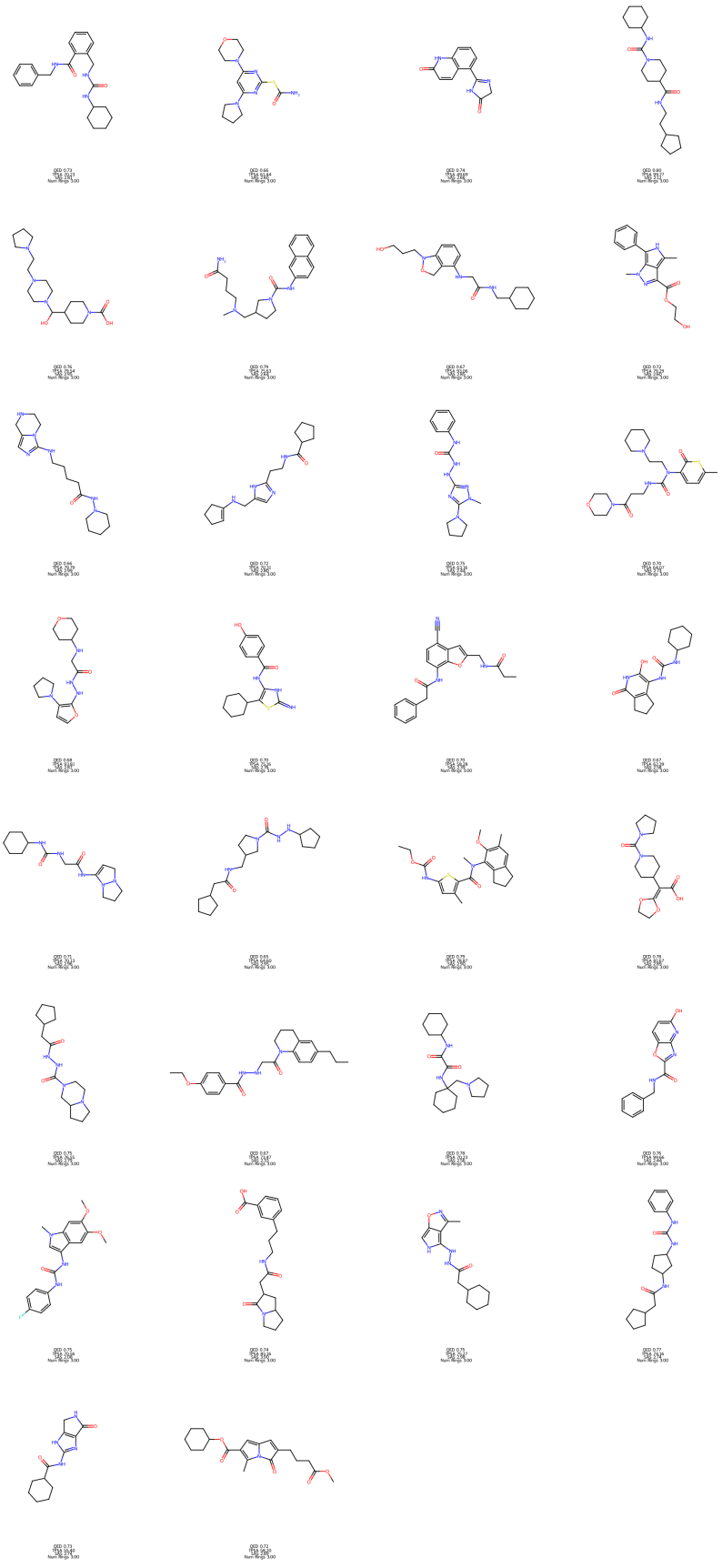

Figure 2: Some randomly chosen successful molecules from the pretrained A-GFN.