RETROGFN: DIVERSE AND FEASIBLE RETROSYNTHESIS USING GFLOWNETS

Piotr Gaiński
Doctoral School of Exact and Natural Sciences
Faculty of Mathematics and Computer Science
Jagiellonian University
Cracow, Poland
piotr.gainski@doctoral.uj.edu.pl

Michał Koziarski
Mila - Quebec AI Institute
Université de Montréal
Montreal, Quebec, Canada

Krzysztof Maziarz & Marwin Segler
Microsoft Research AI4Science
Cambridge, United Kingdom

Jacek Tabor & Marek Śmieja
Faculty of Mathematics and Computer Science
Jagiellonian University
Cracow, Poland

ABSTRACT

Single-step retrosynthesis aims to predict a set of reactions that lead to the creation of a target molecule, which is a crucial task in molecular discovery. Although a target molecule can often be synthesized with multiple different reactions, it is not clear how to verify the feasibility of a reaction, because the available datasets cover only a tiny fraction of the possible solutions. Consequently, the existing models are not encouraged to explore the space of possible reactions sufficiently. To resolve these issues, we first propose a Feasibility Thresholded Count (FTC) metric that estimates the reaction feasibility with a machine-learning model. Second, we develop a novel retrosynthesis model, RetroGFN, which can explore outside the limited dataset and return a diverse set of feasible reactions. We show that RetroGFN outperforms existing methods on the FTC metric by a large margin while maintaining competitive results on the widely used top-k accuracy metric.

1 INTRODUCTION

The rising interest in machine learning led to the development of many deep generative models for de novo drug design Segler et al. (2017); Gómez-Bombarelli et al. (2018); Maziarka et al. (2020); Maziarz et al. (2022); Meyers et al. (2021). Such approaches can propose novel molecules with promising predicted property profiles, however, these virtual compounds eventually need to be synthesized and evaluated in the wet lab. This motivates the development of reliable (retro)synthesis planning algorithms able to design a synthesis route for an input molecule Coley et al. (2019); Schwaller et al. (2020); Chen et al. (2020); Xie et al. (2022); Tripp et al. (2022; 2023).

A crucial component of retrosynthesis is the so-called single-step retrosynthesis, which predicts a reaction that is likely to synthesize a given molecule Coley et al. (2017); Liu et al. (2017); Dai et al. (2019); Sun et al. (2020); Schwaller et al. (2020). In practice, many feasible reactions can lead to a given product. Since the success of a synthesis plan depends on factors that may vary over time

(e.g. the availability or cost of reactants), the retrosynthesis model should ideally return all possible reactions. In other words, we would like to produce a diverse set of feasible reactions leading to the requested product. However, the available datasets cover only a fraction of feasible reactions, so for many of the included products, a lot of alternative reactions are missing. Furthermore, current datasets contain only positive reactions, i.e. those that were successfully conducted in the lab, whereas in reality, reactions can fail, which unfortunately is seldom reported. This limitation of current reaction datasets causes two major issues which we address in this paper.

So far, the feasibility of a reaction is often not taken into account when evaluating retrosynthesis models. The most commonly used metric in the field of retrosynthesis, top-k accuracy, disregards that problem and accounts only for the recall of ground-truth reactions from the dataset. In this paper, we propose the Feasibility Thresholded Count (FTC) that reports the percentage of reactions with sufficiently high feasibility scores. To predict the scores we use a reaction feasibility model. In this way, we can assess both the diversity and feasibility of reactions returned by the model. We measure the diversity simply with the number of unique reactions, as in the context of retrosynthesis the reactions for a given product tend to be similar and differ only in small yet crucial details.

The second problem with missing reactions is that the existing models do not explore the space of feasible reactions well. Therefore, we propose a RetroGFN model that goes beyond the dataset and returns a diverse set of feasible reactions. RetroGFN is based on the recent GFlowNet framework Bengio et al. (2021; 2023) which enables exploration of solution space and sampling from that space with probability proportional to the reward function, e.g. reaction feasibility. In consequence, GFlowNets can sample a large number of highly scored and diverse solutions. Our RetroGFN model leverages this property, sampling a large number of feasible reactions. It outperforms existing methods on the FTC metric by a large margin while achieving competitive results on the top-k accuracy for $k > 3$.

To summarize, our contributions are:

1. We introduce a novel metric called Feasibility Thresholded Count (FTC) which assesses the diversity and feasibility of reactions proposed by backward reaction prediction models, mitigating the limitations of the standard top-k accuracy metric.
2. We design RetroGFN: a model based on the GFlowNet framework that generates diverse and feasible reactions. It achieves competitive results on the top-k accuracy metric while outperforming all considered models on the top-k FTC metrics by a large margin.

2 Related Work

Single-step Retrosynthesis. The single-step retrosynthesis problem is well-known in the drug-discovery community. The methods in this field can be roughly divided into template-based Segler & Waller (2017); Coley et al. (2017); Dai et al. (2019); Baylon et al. (2019) and template-free Sacha et al. (2021); Zhong et al. (2022); Yan et al. (2020); Somnath et al. (2021); Wang et al. (2021); Irwin et al. (2022); Schwaller et al. (2020); Zheng et al. (2019); Mao et al. (2021).

Feasibility-oriented Metrics. There was some effort in the community to develop the notion of feasibility and a metric that measures precision Schwaller et al. (2019; 2020); Chen & Jung (2021); Maziarz et al. (2023a). The most known precision metric is top-k round-trip accuracy. It measures the percentage of top-k reactions that can be translated back by a forward reaction prediction model (for a given product, the retrosynthesis model returns reactants; the reactants are valid if the forward reaction model fed with them returns "back" the original product). The main issue with round-trip accuracy is that the reaction feasibility is defined by a forward reaction model which is of limited reliability Maziarz et al. (2023b), as it only trained on real/positive reactions and lacks the notion of non-feasible/negative reactions; such forward models will always return a product even for non-reactive input molecules which may lead to overly optimistic results or "hallucinations".

GFlowNets. GFlowNets Bengio et al. (2023) are a type of generative methods devoted to sampling from high-dimensional distributions. GFlowNets were originally proposed as an alternative to MCMC (offering the benefits of amortization) and reinforcement learning (displaying a mode-seeking behavior, that is the ability to discover multiple diverse modes), and later shown to be equivalent with special cases of other generative methods Malkin et al. (2022b); Zhang et al. (2022).

The diversity, in particular, is a desired property in multiple scientific discovery tasks Jain et al. (2023a); Bengio et al. (2021); Nica et al. (2022); Roy et al. (2023).

3 FEASIBILITY THRESHOLDED COUNT METRIC

Single-step retrosynthesis is focused on predicting reactions that could lead to the given product. The retrosynthesis model is evaluated with a reaction dataset $D = \{(R_1, p_1), ..., (R_n, p_n)\}$ containing reaction tuples where p_i denotes a product and R_i is a set of reactants that can synthesize the product p_i. During inference, the model is requested to return at most k reactions for every product from the dataset, which are expected to be sorted from the most to the least probable.

In contrast to top-k accuracy, the proposed top-k FTC can account for feasible reactions outside the evaluation dataset. For a single product, the top-k FTC value denotes the percentage of feasible reactions among top k reactions returned by the model. The feasibility is estimated with an auxiliary model RFM described in Appendix B.1. The exact formula for top-k FTC calculated on a product p and retrosynthesis model f is given by:

$$\mathrm{F}_{\mathrm{FTC}}(f, p, k) = \frac{1}{k} \sum_{i=1}^{k} \mathbb{1}[\mathrm{RFM}(f(p)_i) \geq t], \tag{1}$$

where $\mathrm{RFM}(f(p)_i) \in [0, 1]$ is the output of the reaction feasibility model for the i-th reaction proposed by f, and t is a feasibility threshold given by the user. We assume that $\mathrm{RFM}(x) = 1$ for reaction $x \in D$. We report the top-k FTC for the entire dataset D, which can be written as $FTC(f, k) = \frac{1}{n} \sum_{i}^{n} \mathrm{F}_{\mathrm{FTC}}(f, p_i, k)$. Therefore, top-k FTC assesses both the diversity and feasibility of the returned reactions.

4 RETROGFN

4.1 REACTION TEMPLATES AND PATTERNS

Several existing single-step retrosynthesis models, including ours, work on the (backward) reaction templates. A reaction template can be seen as a regular expression on graphs (see Figure 1). It describes the transformation of a product into the reactants and consists of the product pattern (left side of the regular expression) and a set of reactants' patterns (right side). The atoms of the product pattern are mapped to atoms of reactants' patterns. Reaction templates provide a strong inductive bias to the model while limiting it to a fixed set of possible transformations. However, we extend the covered reaction space by introducing a template composition process inspired by RetroComposer Yan et al. (2022). In this approach, we choose the reaction center where the template is going to be applied and compose a concrete template step by step using the building blocks, called patterns.

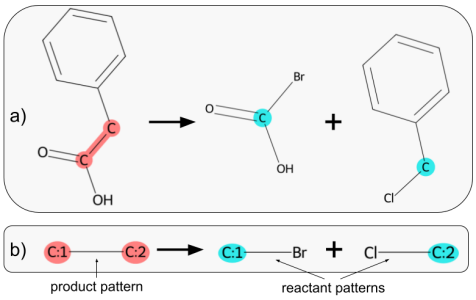

Figure 1: Illustration of a single-step retrosynthesis (a), and a corresponding reaction template (b). Atoms from a product pattern on the left side of the template are mapped to atoms from reactant patterns on the right side .

We extract the templates from the train split of USPTO-50k, following Chen & Jung (2021). Each template is then split into product and reactant patterns (see Figure 1 b)). We denote a set of all encountered product patterns PPS and an analogous set of reactant patterns RPS. The patterns do not include any molecular regular expression (SMARTS) and can be represented similarly to molecules - as annotated graphs.

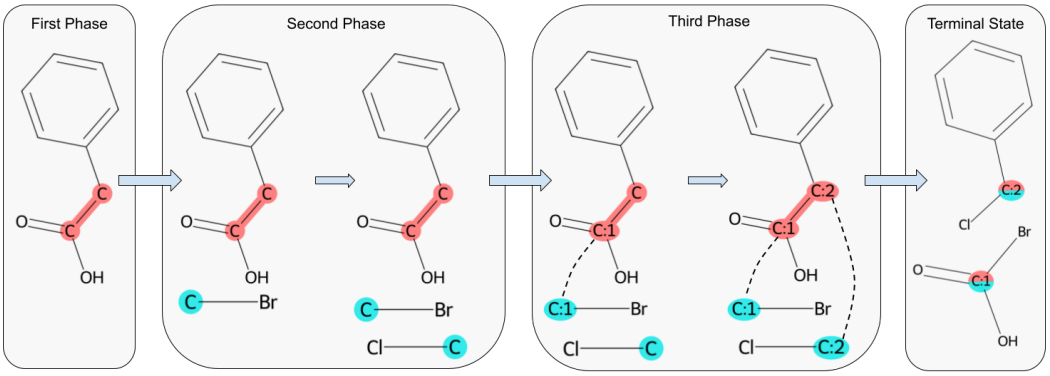

Figure 2: Illustration of the template composition process in RetroGFN for an input product. In the first phase, a product pattern and its concrete match to the atoms of the product is chosen. In the second phase, reactant patterns are gathered until all mappable atoms of the product pattern (highlighted red) can be mapped to mappable atoms of the reactant pattern (highlighted blue). In the third phase, the mapping between mappable product and reactant patterns is created and the obtained template is applied resulting in the reactants.

4.2 Generation Process

Given a product, our RetroGFN composes an appropriate template in three phases:

1. The first phase determines a reaction center: a product pattern matched to the atoms of the product.
2. The second phase gathers the reactant patterns.
3. The third phase constructs atom mapping between the atoms of the product pattern and the reactants' patterns.

In the end, the obtained template is applied to the given product and results in a final set of reactants. Figure 2 shows an example of the composition process while a detailed description of each phase can be found further in the Appendix A.

4.3 Training

We trained our RetroGFN with a modified version of Trajectory Balance Objective from Malkin et al. (2022a), which for a trajectory $\tau = (s_1, a_1, s_2, a_2, ..., s_k, a_k, t)$ is given with the formula:

$$\mathcal{L}(\tau) = \left(\log \frac{F(s_1) \prod_{i=1}^{k} P_F(a_i|s_i)}{R(t) P_B(a_k|t) \prod_{i=2}^{k} P_B(a_{i-1}|s_i)} \right)^2 .$$

The main difference from the original formulation comes from the fact that our RetroGFN is conditioned Bengio et al. (2023); Pan et al. (2023); Kim et al. (2023); Jain et al. (2023b); Roy et al. (2023) on the product from the initial state s_1. Therefore, for every initial state, we estimate the incoming flow separately using $F(s_1)$ function which is essentially an index embedding $F(s) = E_P(s) \in \mathbb{R}$ that looks up the set of training products P and returns a learnable scalar (note that we only evaluate $F(s)$ during training). As a backward policy $P_B(a|s)$, we use a uniform distribution over the possible actions that could lead to state s. The reward is an exponential reward of the form $R(x) = \exp(\beta f(x))$ where f is a reaction feasibility model with output $f(x) \in [0, 1]$. The feasibility model used during training is distinct from the one used in the FTC metric evaluation (details are described in Appendix B.1).

During training, we used a combination of three sampling methods: 1) standard exploratory sampling from the forward policy P_F with some ϵ probability of taking random actions, 2) backward sampling from reward-prioritized replay buffer Shen et al. (2023); Fedus et al. (2020); Jain et al. (2022), and 3) backward sampling from the dataset D. Backward sampling starts with a terminal state and samples the trajectory in the backward direction using the backward policy P_B.

5 EXPERIMENTS

Datasets. We compared the considered methods on two datasets: USPTO-50k, a default choice for benchmarking retrosynthesis models, and USPTO-MIT, which we use as a generalization benchmark for models trained on USPTO-50k. We used commonly used splits for both datasets Coley et al. (2017); Jin et al. (2017). We refined the USPTO-MIT to ensure there is no overlap between it and the USPTO-50k train split.

Retrosynthesis Models. We compared our RetroGFN to well-known and recent state-of-the-art models: GLN Dai et al. (2019), MEGAN Sacha et al. (2021), MHNreact Seidl et al. (2021), LocalRetro Chen & Jung (2021), RootAligned Zhong et al. (2022), RetroKNN Xie et al. (2023), and Chemformer Irwin et al. (2022). We used the wrappers of the original implementations and checkpoints from the Syntheseus repository[1]. We used the evaluation procedure from Syntheseus that queries the model for 100 reactions per product, removes the duplicates, and truncates the list of reactions to be no larger than 50. The same output was used both to calculate standard and FTC metrics.

Reaction Feasibility Models. We used different checkpoints of RFM for training and evaluation. Details can be found in Appendix B.

5.1 RESULTS ON USPTO-50K

We can find the standard metrics evaluated on USPTO-50k in Table 1. Our method achieves relatively low top-k accuracy for $k = 1$ (46.9%), but it becomes significantly more competitive for $k > 5$ which is arguably more important for retrosynthesis search than $k = 1$.

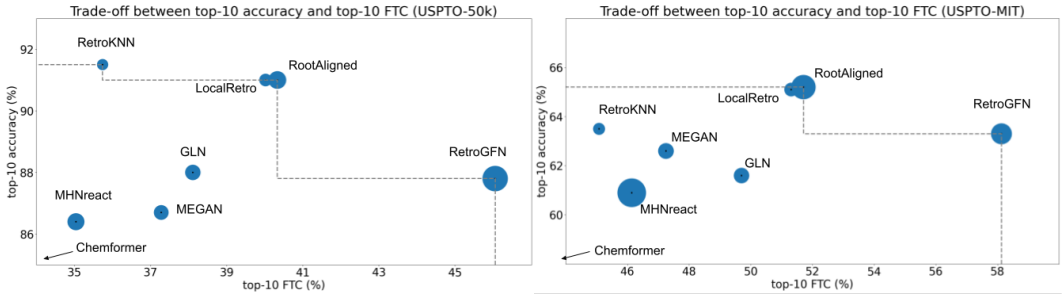

Figure 3: Trade-off between top-10 accuracy and top-10 FTC metric for feasibility threshold=0.9. The circle area is proportional to the model's inference time. The dashed line denotes the Pareto front - the best result for any trade-off coefficient between accuracy and FTC. We observe that our RetroGFN maximizes the top-10 FTC while having competitive results on top-10 accuracy.

However, the actual power of our RetroGFN is evidenced in Table 3 which shows the top-k FTC metric results. Note that the feasibility model used during the training of RetroGFN was trained on a different data split than the one used for evaluation. We observe that for $k > 1$, our model consistently outperforms other models, producing plenty of feasible reactions. The absolute and relative advantage of RetroGFN over the second-best model on top-k FTC increases with k: from 3.6%p and 5.9% for k=3 to 9%p and 30.8% for k=50. The good results of RetroGFN on standard metrics and its exceptional performance on FTC evidence that one can greatly improve the results on FTC without sacrificing the performance on standard metrics. The trade-off between top-10 accuracy and FTC metric can be found in Figure 3 (for other values of k, see Appendix C.5).

5.2 GENERALIZATION RESULTS ON USPTO-MIT

We evaluated the models trained on USPTO-50k further on the USPTO-MIT dataset to assess their generalization properties. To compute top-k FTC results we used a reaction feasibility model fine-tuned on the test split of USPTO-MIT. The evaluation of both standard and FTC metrics (Appendix C.2) echoes the results of USPTO-50k: RootAligned is the best on standard metrics, while

[1] https://github.com/microsoft/syntheseus

our model achieves SOTA results on FTC metrics. As in the USPTO-50k case, the absolute and relative advantage of RetroGFN over the second-best model on top-k FTC increases with k: from 3%p and 4.4% for k=3 to 10.5%p and 25.8% for k=50. The trade-off between top-10 accuracy and FTC metric can be found in Figure 3.

We include the top-k FTC metrics on USPTO-50k and USPTO-MIT for different thresholds in Appendix C. Regardless of the chosen threshold, our RetroGFN model consistently outperforms the rest of the models.

6 CONCLUSIONS

In this paper, we proposed top-k FTC, a novel metric for single-step retrosynthesis model evaluation, that mitigates the limitations of widely used top-k accuracy and recently proposed round-trip accuracy. Leveraging the GFlowNet framework which is designed for tasks where plenty of sensible solutions are desired, we developed a RetroGFN model that achieves competitive results on top-k accuracy and performs outstandingly on the top-k FTC metric. We opened a few interesting paths for further research: 1) improvement of the RFM model, 2) improvement on FTC metric, and 3) leveraging off-dataset training using the RFM model. We discuss the limitations of the paper in the Appendix D.

7 ACKNOWLEDGEMENT

The research of J. Tabor was supported by the Foundation for Polish Science co-financed by the European Union under the European Regional Development Fund in the POIR.04.04.00-00-14DE/18-00 project carried out within the Team-Net program. The research of P. Gaiński and M. Śmieja was supported by the National Science Centre (Poland), grant no. 2022/45/B/ST6/01117. The research of M. Koziarski was funded by the Genetech-Mila partnership. For the purpose of Open Access, the author has applied a CC-BY public copyright license to any Author Accepted Manuscript (AAM) version arising from this submission.

REFERENCES

Javier L Baylon, Nicholas A Cilfone, Jeffrey R Gulcher, and Thomas W Chittenden. Enhancing retrosynthetic reaction prediction with deep learning using multiscale reaction classification. *Journal of chemical information and modeling*, 59(2):673–688, 2019.

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.

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.

Binghong Chen, Chengtao Li, Hanjun Dai, and Le Song. Retro*: learning retrosynthetic planning with neural guided A* search. In *International Conference on Machine Learning*, pp. 1608–1616. PMLR, 2020.

Shuan Chen and Yousung Jung. Deep retrosynthetic reaction prediction using local reactivity and global attention. *JACS Au*, 1(10):1612–1620, 2021.

Connor W. Coley, Luke Rogers, William H. Green, and Klavs F. Jensen. Computer-assisted retrosynthesis based on molecular similarity. *ACS Central Science*, 3(12):1237–1245, 2017.

Connor W Coley, Dale A Thomas III, Justin AM Lummiss, Jonathan N Jaworski, Christopher P Breen, Victor Schultz, Travis Hart, Joshua S Fishman, Luke Rogers, Hanyu Gao, et al. A robotic platform for flow synthesis of organic compounds informed by ai planning. *Science*, 365(6453): eaax1566, 2019.

Hanjun Dai, Chengtao Li, Connor Coley, Bo Dai, and Le Song. Retrosynthesis prediction with conditional graph logic network. *Advances in Neural Information Processing Systems*, 32, 2019.

Vijay Prakash Dwivedi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Graph neural networks with learnable structural and positional representations. *arXiv preprint arXiv:2110.07875*, 2021.

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*, pp. 3061–3071. PMLR, 2020.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pp. 1263–1272. PMLR, 2017.

Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

Ross Irwin, Spyridon Dimitriadis, Jiazhen He, and Esben Jannik Bjerrum. Chemformer: a pre-trained transformer for computational chemistry. *Machine Learning: Science and Technology*, 3 (1):015022, 2022.

Moksh Jain, Emmanuel Bengio, Alex Hernandez-Garcia, Jarrid Rector-Brooks, Bonaventure FP Dossou, Chanakya Ajit Ekbote, Jie Fu, Tianyu Zhang, Michael Kilgour, Dinghuai Zhang, et al. Biological sequence design with gflownets. In *International Conference on Machine Learning*, pp. 9786–9801. PMLR, 2022.

Moksh Jain, Tristan Deleu, Jason Hartford, Cheng-Hao Liu, Alex Hernandez-Garcia, and Yoshua Bengio. Gflownets for ai-driven scientific discovery. *Digital Discovery*, 2(3):557–577, 2023a.

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*, pp. 14631–14653. PMLR, 2023b.

Wengong Jin, Connor Coley, Regina Barzilay, and Tommi Jaakkola. Predicting organic reaction outcomes with weisfeiler-lehman network. *Advances in neural information processing systems*, 30, 2017.

Steven Kearnes, Kevin McCloskey, Marc Berndl, Vijay Pande, and Patrick Riley. Molecular graph convolutions: moving beyond fingerprints. *Journal of computer-aided molecular design*, 30:595–608, 2016.

Minsu Kim, Joohwan Ko, Dinghuai Zhang, Ling Pan, Taeyoung Yun, Woochang Kim, Jinkyoo Park, and Yoshua Bengio. Learning to scale logits for temperature-conditional gflownets. *arXiv preprint arXiv:2310.02823*, 2023.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Bowen Liu, Bharath Ramsundar, Prasad Kawthekar, Jade Shi, Joseph Gomes, Quang Luu Nguyen, Stephen Ho, Jack Sloane, Paul Wender, and Vijay Pande. Retrosynthetic reaction prediction using neural sequence-to-sequence models. *ACS Central Science*, 3(10):1103–1113, 2017.

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, 2022a.

Nikolay Malkin, Salem Lahlou, Tristan Deleu, Xu Ji, Edward Hu, Katie Everett, Dinghuai Zhang, and Yoshua Bengio. Gflownets and variational inference. *arXiv preprint arXiv:2210.00580*, 2022b.

Kelong Mao, Xi Xiao, Tingyang Xu, Yu Rong, Junzhou Huang, and Peilin Zhao. Molecular graph enhanced transformer for retrosynthesis prediction. *Neurocomputing*, 457:193–202, 2021.

Łukasz Maziarka, Agnieszka Pocha, Jan Kaczmarczyk, Krzysztof Rataj, Tomasz Danel, and Michał Warchoł. Mol-cyclegan: a generative model for molecular optimization. *Journal of Cheminformatics*, 12(1):1–18, 2020.

Krzysztof Maziarz, Henry Richard Jackson-Flux, Pashmina Cameron, Finton Sirockin, Nadine Schneider, Nikolaus Stiefl, Marwin Segler, and Marc Brockschmidt. Learning to extend molecular scaffolds with structural motifs. In *International Conference on Learning Representations*, 2022.

Krzysztof Maziarz, Austin Tripp, Guoqing Liu, Megan Stanley, Shufang Xie, Piotr Gaiński, Philipp Seidl, and Marwin Segler. Re-evaluating retrosynthesis algorithms with syntheseus. *arXiv preprint arXiv:2310.19796*, 2023a.

Krzysztof Maziarz, Austin Tripp, Guoqing Liu, Megan Stanley, Shufang Xie, Piotr Gaiński, Philipp Seidl, and Marwin Segler. Re-evaluating retrosynthesis algorithms with syntheseus. In *NeurIPS 2023 AI for Science Workshop*, 2023b. URL https://openreview.net/forum?id=W5U18rgtpg.

Joshua Meyers, Benedek Fabian, and Nathan Brown. De novo molecular design and generative models. *Drug Discovery Today*, 26(11):2707–2715, 2021.

Andrei Cristian Nica, Moksh Jain, Emmanuel Bengio, Cheng-Hao Liu, Maksym Korablyov, Michael M Bronstein, and Yoshua Bengio. Evaluating generalization in gflownets for molecule design. In *ICLR2022 Machine Learning for Drug Discovery*, 2022.

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

Julien Roy, Pierre-Luc Bacon, Christopher Pal, and Emmanuel Bengio. Goal-conditioned gflownets for controllable multi-objective molecular design. *arXiv preprint arXiv:2306.04620*, 2023.

Mikołaj Sacha, Mikołaj Błaz, Piotr Byrski, Paweł Dabrowski-Tumanski, Mikołaj Chrominski, Rafał Loska, Paweł Włodarczyk-Pruszynski, and Stanisław Jastrzebski. Molecule edit graph attention network: modeling chemical reactions as sequences of graph edits. *Journal of Chemical Information and Modeling*, 61(7):3273–3284, 2021.

Philippe Schwaller, Vishnu H Nair, Riccardo Petraglia, and Teodoro Laino. Evaluation metrics for single-step retrosynthetic models. In *Second Workshop on Machine Learning and the Physical Sciences*. NeurIPS Vancouver, Canada, 2019.

Philippe Schwaller, Riccardo Petraglia, Valerio Zullo, Vishnu H Nair, Rico Andreas Haeuselmann, Riccardo Pisoni, Costas Bekas, Anna Iuliano, and Teodoro Laino. Predicting retrosynthetic pathways using transformer-based models and a hyper-graph exploration strategy. *Chemical science*, 11(12):3316–3325, 2020.

Marwin HS Segler and Mark P Waller. Neural-symbolic machine learning for retrosynthesis and reaction prediction. *Chemistry–A European Journal*, 23(25):5966–5971, 2017.

Marwin HS Segler, Thierry Kogej, Christian Tyrchan, and Mark P Waller. Generating focussed molecule libraries for drug discovery with recurrent neural networks. *arXiv preprint arXiv:1701.01329*, 2017.

Philipp Seidl, Philipp Renz, Natalia Dyubankova, Paulo Neves, Jonas Verhoeven, Marwin Segler, Jörg K Wegner, Sepp Hochreiter, and Günter Klambauer. Modern hopfield networks for few-and zero-shot reaction template prediction. *arXiv preprint arXiv:2104.03279*, 2021.

Max W Shen, Emmanuel Bengio, Ehsan Hajiramezanali, Andreas Loukas, Kyunghyun Cho, and Tommaso Biancalani. Towards understanding and improving gflownet training. *arXiv preprint arXiv:2305.07170*, 2023.

Vignesh Ram Somnath, Charlotte Bunne, Connor Coley, Andreas Krause, and Regina Barzilay. Learning graph models for retrosynthesis prediction. *Advances in Neural Information Processing Systems*, 34:9405–9415, 2021.

Ruoxi Sun, Hanjun Dai, Li Li, Steven Kearnes, and Bo Dai. Energy-based view of retrosynthesis. *arXiv preprint arXiv:2007.13437*, 2020.

Austin Tripp, Krzysztof Maziarz, Sarah Lewis, Guoqing Liu, and Marwin Segler. Re-evaluating chemical synthesis planning algorithms. In *NeurIPS 2022 AI for Science: Progress and Promises*, 2022. URL https://openreview.net/forum?id=8VLeT8DFeD.

Austin Tripp, Krzysztof Maziarz, Sarah Lewis, Marwin Segler, and José Miguel Hernández-Lobato. Retro-fallback: retrosynthetic planning in an uncertain world. *arXiv preprint arXiv:2310.09270*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, et al. Deep graph library: A graph-centric, highly-performant package for graph neural networks. *arXiv preprint arXiv:1909.01315*, 2019.

Xiaorui Wang, Yuquan Li, Jiezhong Qiu, Guangyong Chen, Huanxiang Liu, Benben Liao, Chang-Yu Hsieh, and Xiaojun Yao. Retroprime: A diverse, plausible and transformer-based method for single-step retrosynthesis predictions. *Chemical Engineering Journal*, 420:129845, 2021.

Shufang Xie, Rui Yan, Peng Han, Yingce Xia, Lijun Wu, Chenjuan Guo, Bin Yang, and Tao Qin. Retrograph: Retrosynthetic planning with graph search. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 2120–2129, 2022.

Shufang Xie, Rui Yan, Junliang Guo, Yingce Xia, Lijun Wu, and Tao Qin. Retrosynthesis prediction with local template retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

Chaochao Yan, Qianggang Ding, Peilin Zhao, Shuangjia Zheng, Jinyu Yang, Yang Yu, and Junzhou Huang. Retroxpert: Decompose retrosynthesis prediction like a chemist. *Advances in Neural Information Processing Systems*, 33:11248–11258, 2020.

Chaochao Yan, Peilin Zhao, Chan Lu, Yang Yu, and Junzhou Huang. Retrocomposer: Composing templates for template-based retrosynthesis prediction. *Biomolecules*, 12(9):1325, 2022.

Dinghuai Zhang, Ricky TQ Chen, Nikolay Malkin, and Yoshua Bengio. Unifying generative models with gflownets. *arXiv preprint arXiv:2209.02606*, 2022.

Shuangjia Zheng, Jiahua Rao, Zhongyue Zhang, Jun Xu, and Yuedong Yang. Predicting retrosynthetic reactions using self-corrected transformer neural networks. *Journal of chemical information and modeling*, 60(1):47–55, 2019.

Zipeng Zhong, Jie Song, Zunlei Feng, Tiantao Liu, Lingxiang Jia, Shaolun Yao, Min Wu, Tingjun Hou, and Mingli Song. Root-aligned smiles: a tight representation for chemical reaction prediction. *Chemical Science*, 13(31):9023–9034, 2022.

A RetroGFN Details

The core component of a GFlowNet model is a forward policy $P_F(a|s)$ describing the probability of taking action a in the state s. The generation process samples a sequence of states and actions $\tau = (s_1, a_1, ..., s_k, a_k, t)$ called a trajectory, where t is a terminal state. In RetroGFN, an initial state s_1 is an input product, the intermediate states s_i correspond to the partially constructed template, and the terminal state t stores a final template along with a result of its application to the product. We group the states into three phases and the specific definition of $P_F(a|s)$ depends on the phase i:

$$P_F^i(a|s) = \frac{\exp(\text{score}_i(s,a)\alpha)}{\sum_{a' \in A^i(s)} \exp(\text{score}_i(s,a')\alpha)},$$

where score_i is a phase-specific score function parameterized with a neural network and $A^i(s)$ is a set of possible actions that can be taken from s in the i-th phase. The policy is simply a softmax with temperature coefficient α over the scores of all possible actions $A^i(s)$.

Score functions for all the phases share a common Graph Neural Network (GNN) encoder, denoted as gnn_1 that given a product $p = (V, E, T)$, embeds its nodes' features: $\text{gnn}_1(p) \in \mathbb{R}^{n \times d}$, where n is the number of product nodes and d is the embedding size. We overload the notation and let $\text{gnn}_1(v_j)$ denote the embedding of a product node $v_j \in V$. The GNN architecture we use is similar to the one from LocalRetro: a stack of MPNN layers with a single Transformer layer Vaswani et al. (2017) on top. Details can be found in the Appendix A.

A.1 RETROGFN PHASES

First Phase. A state s in the first phase is an input product p. The action space $A^1(s)$ contains all possible atom matchings of product patterns from PPS to the product p. An action $a \in A^1(s)$ contains the matched product pattern $pp \in PPS$ and the matched atom indices $I = \{i_1, ..., i_m\}$. The value of i_j is an index of the product atom matched with j-th product pattern atom. To compute the $\text{score}_1(s, a)$, we aggregate the representation of matched product's nodes and put them into multi-layer perceptron $\text{MLP}_1 : \mathbb{R}^d \to \mathbb{R}$:

$$\text{score}_1(s, a) = \text{MLP}_1 \left(\sum_{i \in I} \text{gnn}_1(v_i) \right).$$

After the action is chosen and applied, the generation process transitions directly to the second phase.

Second Phase. The second phase iteratively adds reactant patterns to the composed template. At the beginning of the phase, the list of reactant patterns is empty. The second phase action a is a reactant pattern $rp_j \in RPS$ that is going to be added to the template. The $\text{score}_2(s, a)$ concatenates the information from the previous phase and the reactant patterns collected so far (denoted as R) and feeds it to $\text{MLP}_2 : \mathbb{R}^{3d} \to \mathbb{R}^{|RPS|}$ that predicts the score for all the reactant patterns in RPS:

$$\begin{aligned} &\text{score}_2(s, a) = \\ &= \text{MLP}_2 \left(\sum_{i \in I} \text{gnn}_1(v_i) \mid \text{E}_{PPS}(pp) \mid \sum_{rp \in R} \text{E}_{RPS}(rp) \right)_j. \end{aligned}$$

Here we select the jth score returned by the MLP_2 as it corresponds to rp_j reactant pattern from the action. Index embedding $e = \text{E}_A(a)$ is a function that looks up the index of the element a in the set A and assigns the index a learnable embedding $e \in \mathbb{R}^d$ (e.g. E_{PPS} assign a unique learnable embedding to every $pp \in PPS$).

At the end of this phase, we want to be sure that every atom from the product pattern can be mapped to some atom of the reactant pattern. Originally, each pattern had some atom mapping in the template it comes from (see Figure 1). Although those explicit mappings are inadequate in the novel-composed template, we can leverage the knowledge that an atom was originally mapped. For every pattern, we construct a set of mappable atoms that consists of the pattern's atoms that were mapped in the original template (see Figure 4). The composed template is allowed to map only the mappable atoms. We ensure that all mappable atoms in the composed template can be mapped by properly restricting the action space $A^2(s)$.

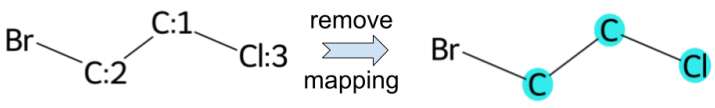

Figure 4: Illustration of a pattern before (left) and after (right) mapping removal. The mappable atoms of the pattern are colored blue.

Third Phase. The third phase creates a mapping between atoms of product and reactant patterns. An action a is an atom mapping $(j, k, l) \in M$ that links the j-th node from the product pattern pp

with the l-th mappable node of the k-th reactant pattern from the list of reactant patterns R. The $\mathrm{score}_3(s, a)$ is given with the formula:

$$\mathrm{score}_3(s, a) = \mathrm{MLP}_3(\mathrm{gnn}_1(v_{i_j}) \mid \mathrm{gnn}_2(v_{kl})),$$

where v_{i_j} is a product node matched with the j-th node of the product pattern, and v_{kl} is the l-th node of the k-th reactant pattern from R. To embed the reactant pattern nodes, we introduce a GNN gnn_2 with the same architecture as gnn_1.

The action space $A^3(s)$ contains all possible atom mappings. We call an atom mapping between two nodes possible when the atom symbols of the nodes are the same and neither of the nodes was previously mapped.

The third phase ends when every node from the product pattern is mapped, resulting in a template that can be applied to the reaction center chosen in the first phase. The obtained reaction forms the terminal state t.

A.2 Inference

During inference, the retrosynthesis model is given a product and requested to output at most N reactions sorted from the most to least promising. RetroGFN samples the reactions using the trained forward policy $P_F(a|s)$ and orders them with the estimated probability. The probability of a reaction represented by a terminal state t is estimated by summing the probabilities of all sampled trajectories that end with t:

$$p(t) = \sum_{\tau : t \in \tau} \prod_{(s,a) \in \tau} P_F(a|s). \tag{2}$$

To increase the accuracy of the estimation, we sample $K \cdot N$ trajectories. We leave the exploration of other estimation methods for future work.

A.3 Architectures

All neural networks in RetroGFN used the same hidden dimension $h = 200$. To obtain initial node and edge features for products, we used featurization from Kearnes et al. (2016) implemented in the DGL library Wang et al. (2019). For the reactant pattern, we used the same edge featurization and a custom node featurization that accounted for atom type, degree, aromaticity, whether the atom was mapped in the original template, relative charge difference between product and reactant atom in the original template, and analogous implicit hydrogen difference. The node features for both products and reactant patterns were enriched with random walk positional encoding Dwivedi et al. (2021) of size $\mathrm{n\_random\_walk} = 16$.

Product node encoder gnn_1 consists of $\mathrm{num\_layer\_1} = 4$ layers of the MPNN convolution Gilmer et al. (2017) and one Transformer layer with $\mathrm{num\_heads} = 8$. The reactant pattern encoder differs only in the number of layers $\mathrm{num\_layer\_2} = 3$. Multi-layer perceptrons $\mathrm{MLP}_1, \mathrm{MLP}_2, \mathrm{MLP}_3$ had one hidden layer (with hidden dimension h) and used the GeLU activation function.

During the training probability of taking random action in the forward policy was set to $\epsilon = 0.05$, the number of sampled forward trajectories in the batch was $\mathrm{n\_forward} = 32$, and the analogous numbers for backward dataset trajectories and backward replay buffer trajectories were $\mathrm{n\_dataset} = 80$ and $\mathrm{n\_replay} = 16$. The reward temperature $\beta = 12$ and the forward policy temperature $\alpha = 1$. The model was trained with Adam optimizer Kingma & Ba (2014) with a learning rate $lr = 0.0005$ (with other parameters set to default values in the torch implementation) for $\mathrm{n_iterations} = 18000$ iterations. In the evaluation, the forward policy temperature was set to $\alpha = 0.7$. During the inference, we sampled $K \cdot N$ trajectories to accurately estimate the reaction probability. For USPTO-50k, we set $K = 10$ while for, due to limited computational resources, we set $K = 6$ for USPTO-MIT.

All the hyperparameters were chosen manually based on the top-k accuracy and top-k FTC metrics estimated on the USPTO-50k validation split.

B REACTION FEASIBILITY MODEL DETAILS

B.1 REACTION FEASIBILITY MODEL

The Reaction Feasibility Model (RFM) is a model that takes reaction x as an input and outputs its feasibility - probability that the reaction is feasible: $\mathrm{RFM}(x) \in [0, 1]$. In this paper, we develop a reliable RFM baseline that can be used as a benchmark in future work. Our RFM implementation consists of two Graph Neural Networks (GNNs) with a Transformer layer and attention pooling at the top that create product and reactant embeddings which are then concatenated and fed into the MLP layer.

B.2 CHECKPOINTS FOR USPTO-50K

To train the model, we augmented the USPTO-50k dataset with negative (non-feasible) reactions using two methods: 1) application of existing forward templates to obtain a novel product from existing reactants, 2) swapping a product in the reaction with another product that is similar to the original one in terms of Tanimoto similarity. Such an approach ensured that the generated negative reactions are not trivially unfeasible (they use an existing template and/or the product is not strikingly different from the reactants), but still are very unlikely to occur in reality (the original reactants were reported to return a different product). We obtained a reaction feasibility dataset with a negative-to-positive ratio of 5:1. We trained two distinct checkpoints of feasibility models: RFM-Train-50k and RFM-Eval-50k. The **RFM-Train** was trained only on the train split of the reaction feasibility dataset and was then used to calculate the reward in the RetroGFN during the training. The **RFM-Eval** was trained on the merged train and test splits of the reaction dataset and used solely during the final evaluation of FTC metrics. Both models achieved good results on the validation split: AUROC=0.96, AUPRC=0.82 and AUROC=0.97, AUPRC=0.85, respectively. To additionally measure the performance of RFM-Eval-50k, we evaluated it on the augmented USPTO-MIT test set. It obtained a strong AUROC=0.93 and AUPRC=0.74, indicating its good generalization properties.

B.3 CHECKPOINT FOR USPTO-MIT

For USPTO-MIT, we used a slightly modified dataset augmentation procedure and replaced the application of existing forward templates by swapping a product in the reaction with another random product. Due to the large size of the augmented USPTO-MIT dataset, we fine-tuned the existing RFM-Eval on the augmented test split of the USPTO-MIT. Obtained **RFM-Eval-MIT** achieves AUROC=0.98 and AUPRC=0.91 on the augmented *subset* of USPTO-MIT valid split. More details can be found in the Appendix B.

B.4 DETAILS AND HYPERPARAMETERS

The architecture of the Reaction Feasibility Model consists of two GNNs that encode the product and reactants. Both GNNs consist of $\mathrm{num\_layers} = 4$ layers of MPNN convolution Gilmer et al. (2017), a Transformer layer with $\mathrm{num\_attention\_heads} = 4$ attention heads, and an attention pooling that aggregates the nodes embeddings to a single graph-level representation. The graph-level representations are then concatenated and fed into an MLP with two hidden layers and ReLU activation. The hidden dimension of all neural networks is $\mathrm{hidden\_dim} = 256$.

The model was trained on the augmented USPTO-50k using Adam optimizer with $lr = 0.0001$ for $\mathrm{n\_epochs} = 100$. To create a checkpoint for USPTO-MIT, we fine-tuned the existing model on the augmented test split of USPTO-MIT using the same optimization parameters for $\mathrm{n\_epochs} = 30$ epochs.

The hyperparameters were chosen manually to maximize the AUPRC metric calculated on the validation set.

C EXTENDED RESULTS

This section shows the extended experimental results.

C.1 USPTO-50K RESULTS

Table 1: Top-k accuracy and mean reciprocal rank (MRR) results on USPTO-50k. The numbers in columns denote k values. The best results in every column are **bolded**. We observe that for $k > 3$ our RetroGFN achieves competitive results.

method	MRR	top-1	top-3	top-5	top-10	top-20	top-50
GLN	0.6509	52.4	74.6	81.2	88.0	91.8	93.1
MEGAN	0.6226	48.7	72.3	79.5	86.7	91.0	93.5
MHNreact	0.6356	50.6	73.1	80.1	86.4	90.3	92.6
LocalRetro	0.6565	51.5	76.5	84.3	91.0	**94.9**	**96.7**
RetroKNN	0.6834	55.3	77.9	85.0	**91.5**	91.6	96.6
RootAligned	**0.6886**	**56.0**	79.1	**86.1**	91.0	93.3	94.2
Chemformer	0.6312	55.0	70.9	73.7	75.4	75.9	76.0
RetroGFN	0.6144	46.9	72.2	80.0	87.8	91.9	94.7

Table 2: Top-k FTC results on USPTO-50k for threshold=0.8. The numbers in columns denote k values. The best results in every column are **bolded**. We observe that for $k > 1$ our RetroGFN consistently outperforms other methods.

method	top-1	top-3	top-5	top-10	top-20	top-50
GLN	80.29	66.44	59.01	47.96	36.82	21.92
MEGAN	76.79	63.13	55.98	46.48	36.78	24.67
MHNreact	77.91	63.01	55.00	44.08	33.11	18.95
LocalRetro	80.33	66.27	59.45	49.96	40.44	26.84
RetroKNN	79.07	63.39	56.08	44.22	28.44	12.00
RootAligned	**81.57**	66.11	59.27	49.72	39.70	20.85
Chemformer	80.31	46.89	32.83	18.37	9.75	3.98
RetroGFN	79.49	**70.15**	**64.46**	**56.68**	**48.81**	**37.39**

Table 3: Top-k FTC results on USPTO-50k for threshold=0.9. The numbers in columns denote k values. The best results in every column are **bolded**. We observe that for $k > 1$ our RetroGFN consistently outperforms other methods.

method	top-1	top-3	top-5	top-10	top-20	top-50
GLN	73.96	57.36	49.02	38.11	28.35	16.37
MEGAN	70.38	54.20	46.67	37.28	28.65	18.48
MHNreact	72.00	54.15	45.50	35.04	25.61	14.29
LocalRetro	73.38	57.16	49.78	40.02	31.39	20.12
RetroKNN	71.98	54.73	47.03	35.74	22.53	9.46
RootAligned	**75.93**	57.21	49.68	40.33	31.38	16.15
Chemformer	74.88	41.50	28.48	15.69	8.24	3.36
RetroGFN	72.38	**60.96**	**54.09**	**46.05**	**38.88**	**29.09**

Table 4: Top-k FTC results on USPTO-50k for threshold=0.95. The numbers in columns denote k values. The best results in every column are **bolded**. We observe that for $k > 1$ our RetroGFN consistently outperforms other methods.

method	top-1	top-3	top-5	top-10	top-20	top-50
GLN	68.18	48.31	39.88	29.82	21.44	12.04
MEGAN	64.65	46.00	38.28	29.38	21.91	13.64
MHNreact	66.43	45.97	37.21	27.59	19.62	10.65
LocalRetro	67.45	48.96	41.02	31.58	23.97	14.85
RetroKNN	66.05	46.90	38.87	28.47	17.61	7.37
RootAligned	**70.74**	49.05	41.03	32.16	24.29	12.32
Chemformer	69.82	36.61	24.67	13.43	6.99	2.84
RetroGFN	66.37	**51.96**	**44.58**	**36.56**	**30.03**	**21.83**

C.2 USPTO-MIT RESULTS

Table 5: Top-k accuracy and mean reciprocal rank (mrr) results on USPTO-MIT. The numbers in columns denote k values. The best results in every column are **bolded**. We observe that for $k > 3$ our RetroGFN achieves competitive results.

method	MRR	top-1	top-3	top-5	top-10	top-20	top-50
GLN	0.4480	35.6	51.5	56.5	61.6	64.2	65.3
MEGAN	0.4498	35.3	52.0	57.6	62.6	65.8	68.1
MHNreact	0.4451	35.3	51.3	56.4	60.9	63.7	65.2
LocalRetro	0.4636	36.0	54.2	59.9	**65.1**	**67.9**	**69.7**
RetroKNN	0.4491	34.9	52.5	58.2	63.5	65.3	65.5
RootAligned	**0.4838**	**38.9**	**55.6**	**60.6**	65.2	67.7	68.8
Chemformer	0.4362	37.8	49.1	51.2	52.5	52.9	52.9
RetroGFN	0.4375	33.1	51.3	57.5	63.3	66.7	68.9

Table 6: Top-k FTC results on USPTO-MIT for threshold=0.8. The numbers in columns denote k values. The best results in every column are **bolded**. We observe that for all k our RetroGFN consistently outperforms other methods.

method	top-1	top-3	top-5	top-10	top-20	top-50
GLN	81.54	73.36	68.17	60.04	50.76	35.86
MEGAN	79.82	70.61	65.35	57.26	48.48	35.02
MHNreact	79.99	70.84	64.95	56.04	45.72	29.28
LocalRetro	82.77	74.28	69.12	61.70	53.21	38.83
RetroKNN	81.72	71.53	65.38	53.98	36.38	15.69
RootAligned	83.81	74.56	69.26	61.30	51.47	28.30
Chemformer	80.29	53.45	39.56	23.90	13.24	5.55
RetroGFN	**83.92**	**77.85**	**74.24**	**68.63**	**62.27**	**50.35**

Table 7: Top-k FTC results on USPTO-MIT for threshold=0.9. The numbers in columns denote k values. The best results in every column are **bolded**. We observe that for $k > 1$ our RetroGFN consistently outperforms other methods.

method	top-1	top-3	top-5	top-10	top-20	top-50
GLN	74.82	64.19	58.27	49.70	40.62	27.55
MEGAN	72.90	61.48	55.64	47.25	38.83	27.06
MHNreact	73.34	61.72	55.20	46.14	36.58	22.71
LocalRetro	76.35	65.44	59.41	51.31	42.93	30.24
RetroKNN	75.05	62.87	56.08	45.08	29.84	12.78
RootAligned	**77.63**	66.00	59.96	51.71	42.41	22.80
Chemformer	73.81	47.07	34.18	20.21	11.02	4.59
RetroGFN	77.37	**69.06**	**64.48**	**58.13**	**51.74**	**40.74**

Table 8: Top-k FTC results on USPTO-MIT for threshold=0.95. The numbers in columns denote k values. The best results in every column are **bolded**. We observe that for $k > 1$ our RetroGFN consistently outperforms other methods.

method	top-1	top-3	top-5	top-10	top-20	top-50
GLN	67.70	55.08	48.67	40.08	31.81	20.82
MEGAN	65.90	52.92	46.52	38.26	30.56	20.64
MHNreact	66.47	53.00	46.10	37.17	28.68	17.31
LocalRetro	69.24	56.52	49.90	41.56	33.78	23.08
RetroKNN	67.83	54.26	47.14	36.73	23.87	10.16
RootAligned	**70.86**	57.12	50.57	42.43	33.98	17.97
Chemformer	66.93	40.93	29.14	16.89	9.10	3.76
RetroGFN	70.04	**59.72**	**54.33**	**47.69**	**41.58**	**31.96**

C.3 Ablation Results

To provide more insight into our RetroGFN model performance, we performed an ablation study regarding the choice of the reward function used during RetroGFN training. It is a very important component as it defines the distributions of the reactions that the RetroGFN model aims to model. Let us recall that the reward function is defined as $R(x) = \exp(\beta f(x))$ where f is a non-negative feasibility proxy function. We can define the feasibility proxy function in three main ways:

- Feasibility (Original): $f(x)$ is the output of RFM-Train i.e. $f(x) = \mathrm{RFM}(x)$,
- In-dataset: $f(x)$ is the indicator of x being in the dataset i.e. $f(x) = \mathbb{1}[x \in D]$,
- Feasibility + In-dataset: $f(x)$ mimics the FTC metric i.e. $f(x) = 1$ if $x \in D$ and $f(x) = TFM(x)$ otherwise.

Tables 9 and 10 from show the evaluation results for three types of the reward function. We observe a significant gap between In-dataset and Feasibility + In-dataset in terms of FTC metrics without a decrease in standard metrics. An additional boost in FTC metrics is given by lowering the influence of the dataset during training by not using the dataset in reward at all. It comes with a decrease in top-1 accuracy, which is, however, relatively small concerning gains obtained in the FTC metrics.

The conclusion is that our RetroGFN model can leverage the RFM-Train model to drastically increase the performance of FTC metrics without sacrificing the standard top-k metrics too much.

Table 9: Top-k accuracy and mean reciprocal rank (mrr) results on USPTO-50k for ablation studies. The numbers in columns denote k values. The best results in every column are **bolded**.

method	MRR	top-1	top-3	top-5	top-10	top-20	top-50
Feas. (Original)	0.6144	46.9	72.2	**80.0**	87.8	91.9	**94.7**
In-dataset	**0.6263**	**49.2**	72.3	79.8	87.2	91.3	94.1
Feas. + In-Dataset	0.6253	48.6	**72.6**	80.4	**88.1**	**92.1**	94.5

Table 10: Top-k FTC results on USPTO-MIT for threshold=0.9 for ablation studies. The numbers in columns denote k values. The best results in every column are **bolded**.

method	top-1	top-3	top-5	top-10	top-20	top-50
Feas. (Original)	**72.38** 1	**60.96**	**54.09**	**46.05**	**38.88**	**29.09**
In-dataset	71.66	55.07	46.57	36.25	26.63	16.76
Feas. + In-Dataset	72.22	58.33	51.33	43.34	35.98	26.07

C.4 Novel Templates

Thanks to the generality of local templates, the vast majority of the test split of USPTO-50k can be covered with patterns extracted from the train split of USPTO-50k. However, 20 out of 5007 reactions in the test split are described with patterns unseen in the training set. Our RetroGFN was able to provide an alternative template for 3 of such reactions.

C.5 Extended Trade-off Results

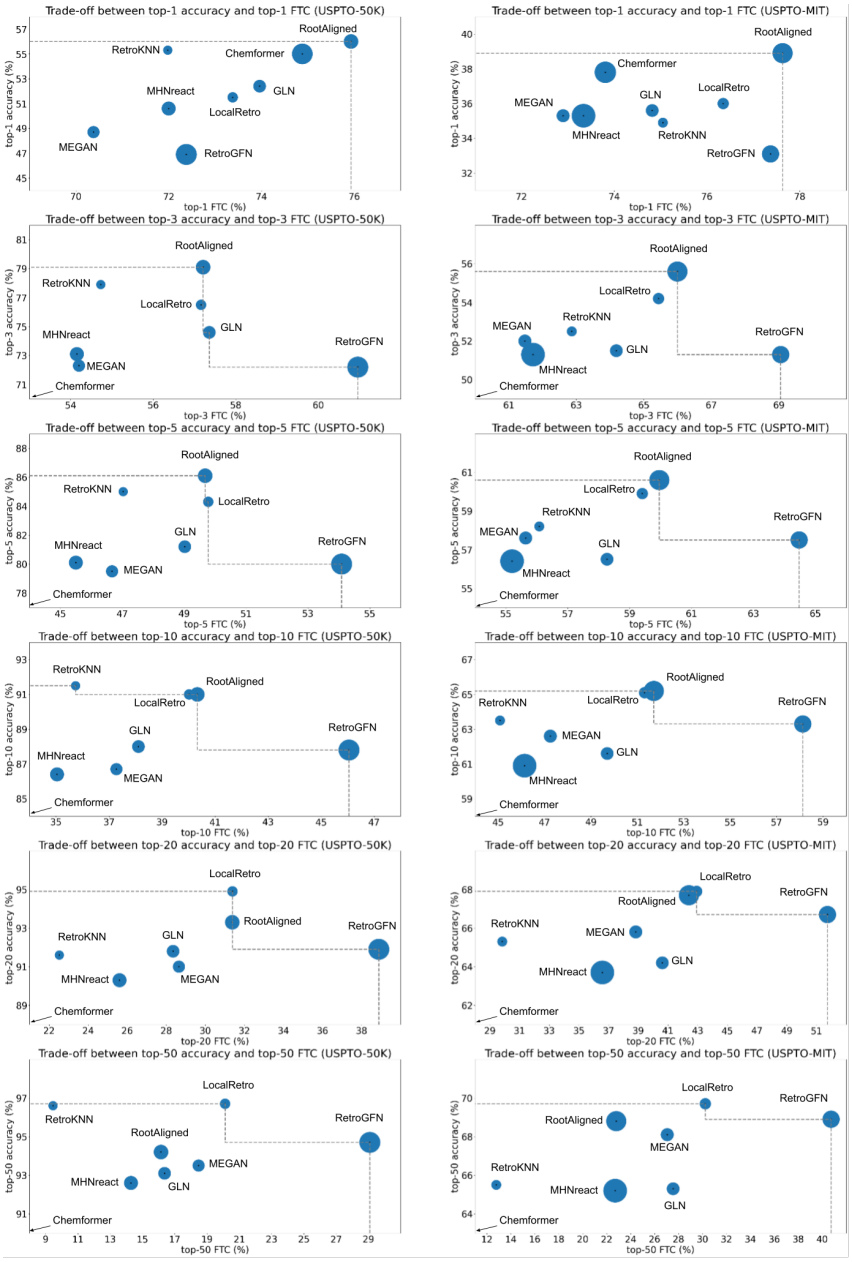

Figure 5: Trade-off between top-k accuracy and top-k FTC metric for feasibility threshold=0.9. The circle area is proportional to the model's inference time. The dashed line denotes the Pareto front - the best result for any trade-off coefficient between accuracy and FTC.

D Limitations and Discussion

This section briefly discusses the limitations of the paper.

D.1 Top-k FTC

The main limitation of the top-k FTC metric is that it relies on the Reaction Feasibility Model (RFM) which suffers both false negative and false positive errors. However, we believe that there is

an inherent epistemic uncertainty within the notion of feasibility (we cannot screen all the reactions) and any sensible retrosynthesis metric will have some portion of false negatives (it will not take all feasible reactions into account). In comparison to top-k accuracy, our Top-k FTC has a strictly lower number of false negatives, while keeping false positives on a decent level. We believe that the FTC metric will benefit from the further improvements of the RFM and we leave it for future work.

D.2 RETROGFN

Top-k Accuracy. The main limitation of our RetroGFN method is its results on top-k accuracy for $k < 5$. At first glance, it looks like a trade-off necessary to achieve excellent results on the top-k FTC metric, but the ablation studies in Appendix C.3 show that the top-k accuracy of RetroGFN does not increase much even if it is trained without RFM-Train using a spiky reward function that has a non-zero value only for reactions for datasets. We argue that it may be caused by two things: 1) other hyperparameters of the model are not optimal for top-k accuracy, 2) the GFlowNet framework struggles with spiky reward function, and 3) the parametrization of the composition process is sub-optimal. It is possible that further refinements of the method could improve the results.

Leveraging RFM-Train. The fact that RetroGFN leverages the RFM-Train checkpoint can be seen as an unfair advantage because a similar RFM-Eval model is used in the FTC metric computation. However, we think that fairness comes from the fact that all models use the same data splits for training or evaluation. The models differ in the way they learn from the training data and leveraging the RFM-Train is yet another way of learning. It does not inject any new knowledge that cannot be extracted from the training data (see Appendix B.1). Once the FTC metric is established, it becomes reasonable to optimize it using RFM-Train. Moreover, we believe that RFM-Eval and RFM-Train are expected to be similar because they have similar goals: 1) to extract as much information from the train and test split as possible, and 2) to extract as much information from the train split as possible. It is sensible then that they share architecture and data augmentation techniques. The difference should come from the data split used for training.