SUPPORTING INFORMATION: RE-EVALUATING RETROSYNTHESIS ALGORITHMS WITH SYNTHESEUS

Krzysztof Maziarz[1*] **Austin Tripp**[2*†] **Guoqing Liu**[1] **Megan Stanley**[1]
Shufang Xie[1] **Piotr Gaiński**[3†] **Philipp Seidl**[4†] **Marwin Segler**[1*]
[1]Microsoft Research AI4Science [2]University of Cambridge
[3]Jagiellonian University [4]Johannes Kepler University Linz

1 PISTACHIO PREPROCESSING

The raw Pistachio data (version 2023Q2, released in June 2023) contained 15 684 711 raw reactions; however, this included many duplicates, outliers (e.g. reactions with extremely large products), and potentially samples that are erroneous. To ensure the test data is of high quality, we performed careful filtering and processing of the raw Pistachio data. We applied the following steps in order:

- Remove duplicate reactions.
- Remove reactions with more than 4 reactants.
- Compute occurrence count of each product molecule across the dataset (counting individual products in multi-product reactions separately). For every reaction with products $[p_1, ..., p_m]$ (including reactions with a single product i.e. $m = 1$), remove all side products. Product p_i is considered a side product if it either has less than 5 atoms, or appears at least 1000 times across the dataset. The latter condition allows us to remove common side products, which may have 5 or more atoms but are still uninteresting. Retain only those reactions where exactly one p_i remained after this procedure (i.e. those with a well-defined main product).
- Remove reactions where the (now unique) product has more than 100 atoms.
- Remove reactions where the ratio of the number of reactant atoms to the number of product atoms exceeds 20.
- Remove reactions where the product appears as one of the reactants.
- Refine reactions by removing the atom mapping numbers that appear only on one side.
- Remove reactions that have double-mapped atoms on either the main product side or the reactants side, or those that lack atom mapping numbers entirely.
- Refine reactions by removing reactants that do not contribute atoms to the product.

We chose the processing steps above such that we exclude erroneous reactions, extreme outliers (i.e. those that are either very large or have an extreme imbalance between the size of the reactants and the size of the product), and reactions with no clearly defined main product. These processing steps (and the particular constants used therein) were informed by expert qualitative analysis of the reactions, as well as practical considerations. For example, we found that several very large outliers in raw Pistachio data seem to cause `rdkit`'s template extraction routines to hang; however, these reactions did not survive our filtering. Further discussion on preprocessing chemical reaction data for use in Deep Learning can be found in Wigh et al. (2023).

After the preprocessing we obtained 3 445 833 single-product samples, which we grouped by their product, and split into train, validation and test sets following a 90/5/5 ratio, making sure the groups of samples with the same product are placed into the same fold. We used a random split, except for those products which were found in USPTO-50K data; in those cases, we attempt to place the corresponding group of samples in the same fold as it appears in USPTO (this limits overlap between training set of one dataset and test set of another, which could distort our generalization results). As the USPTO-50K split from Dai et al. (2019) contains a small amount of product overlap between folds, this process of ensuring a "compatible" Pistachio split was imperfect; the product overlap between USPTO-50K training set and Pistachio test set is non-zero, but it is negligibly small. Note that an alternative approach to preventing overlap would be to completely remove USPTO products from Pistachio before splitting the dataset, but we did not want to artificially exclude (valuable) products present in USPTO-50K.

In this work we use Pistachio solely for testing generalization, thus we only used the test fold, which we randomly subsampled to 20 000 samples for faster evaluation (note that this is still 4 times larger than the test set of USPTO-50K). We described the full procedure to generate all folds to facilitate future work.

2 Extended Single-step Results

2.1 Trade-off Between Quantitative Performance and Speed

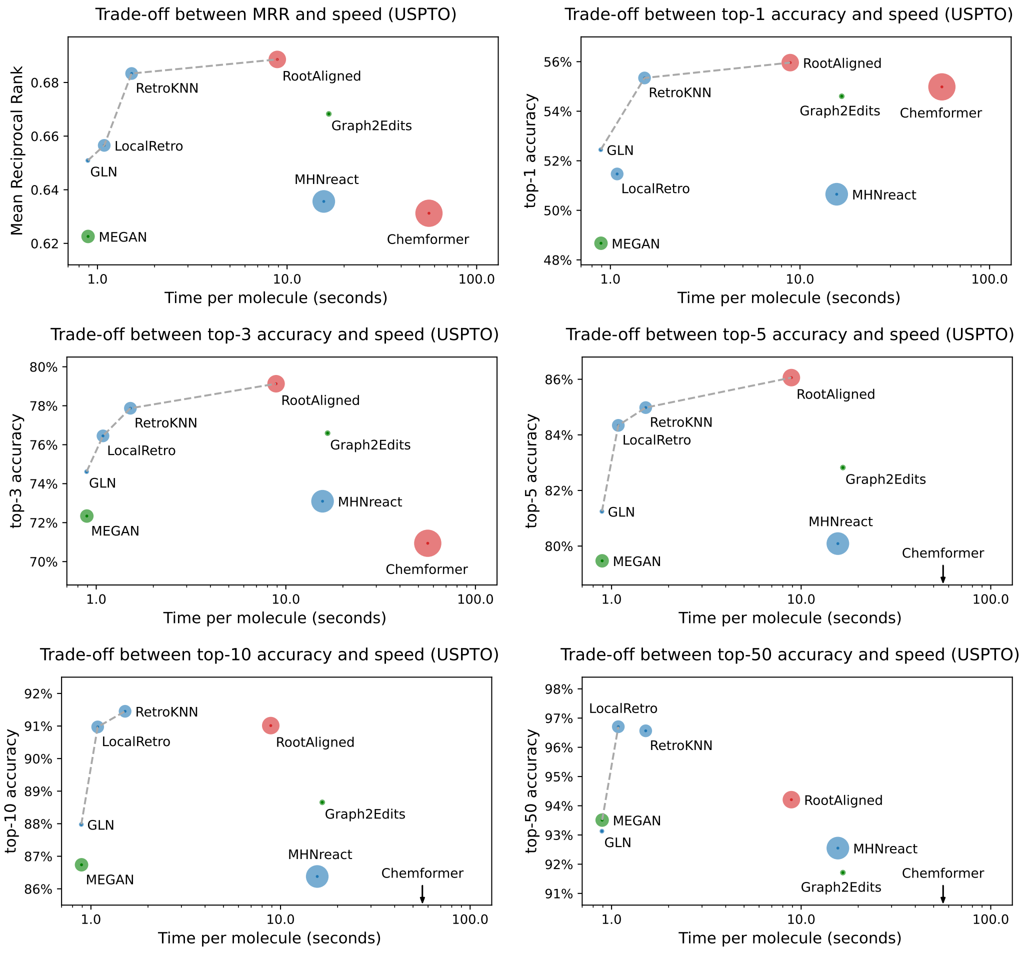

Figure 1: Results on USPTO-50K in same format as Main Text Fig. 2 but extended with top-1, top-3, top-10, top-50, and MRR. Plot for top-5 shown in Main Text Fig. 2 is reprinted here for convenience.

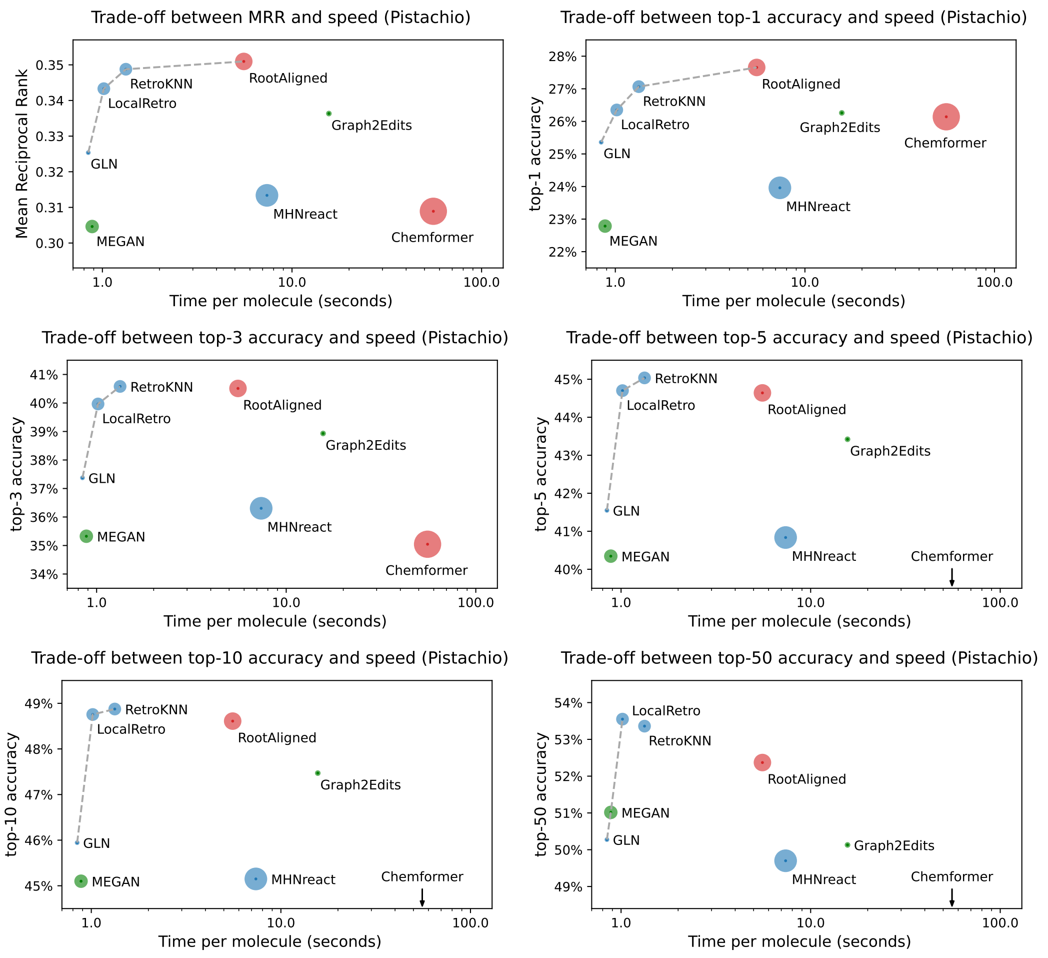

Figure 2: Results on Pistachio in same format as Main Text Fig. 3 but extended with top-1, top-3, top-10, top-50, and MRR. Plot for top-5 shown in Main Text Figure 3 is reprinted here for convenience.

2.2 Comparison with Published Results

In Table 1, we present the results from Figure 1 in numeric form, as well as contrast them with the published numbers. For results produced with SYNTHESEUS we additionally investigate the effect of deduplication.

Table 1: Results on USPTO-50K compared to the numbers reported in the literature. SYNTH denotes whether we used SYNTHESEUS to produce the result (as opposed to copying the published number, or, in case of LocalRetro with exact match, generating the number ourselves using authors' code), D denotes whether deduplication was performed (which in SYNTHESEUS is enabled by default, but can be turned off). We underline values that differ significantly from the previous row (at least 0.7% for top-k or 0.003 for MRR), and use colors to distinguish whether the value is better (green) or worse (red) than the row directly above.

Model	SYNTH	D	top-1	top-3	top-5	top-10	top-50	MRR
Chemformer	✗		54.3%	-	62.3%	63.0%	-	-
	✓	✗	55.0%	67.8%	70.5%	72.5%	74.8%	0.6182
	✓	✓	55.0%	70.9%	73.7%	75.4%	76.0%	0.6312
GLN	✗		52.5%	69.0%	75.6%	83.7%	92.4%	-
	✓	✗	52.4%	68.8%	75.4%	83.5%	92.5%	0.6262
	✓	✓	52.4%	74.6%	81.2%	88.0%	93.1%	0.6509
Graph2Edits	✗		55.1%	77.3%	83.4%	89.4%	92.7%	-
	✓	✗	54.6%	76.4%	82.6%	88.5%	91.7%	0.6672
	✓	✓	54.6%	76.6%	82.8%	88.7%	91.7%	0.6683
LocalRetro	✗		53.4%	77.5%	85.9%	92.4%	97.7%	-
(exact match)	✗		52.0%	75.5%	83.4%	90.0%	95.7%	-
	✓	✗	51.5%	75.6%	83.5%	90.6%	96.7%	0.6530
	✓	✓	51.5%	76.5%	84.3%	91.0%	96.7%	0.6565
MEGAN	✗		48.1%	70.7%	78.4%	86.1%	93.2%	-
	✓	✗	48.7%	71.9%	78.9%	86.0%	93.2%	0.6203
	✓	✓	48.7%	72.3%	79.5%	86.7%	93.5%	0.6226
MHNreact	✗		50.5%	73.9%	81.0%	87.9%	94.1%	-
	✓	✗	50.6%	73.1%	80.1%	86.4%	92.6%	0.6356
	✓	✓	50.6%	73.1%	80.1%	86.4%	92.6%	0.6356
RetroKNN	✗		57.2%	78.9%	86.4%	92.7%	98.1%	-
	✓	✗	55.3%	76.9%	84.3%	90.8%	96.5%	0.6796
	✓	✓	55.3%	77.9%	85.0%	91.5%	96.6%	0.6834
RootAligned	✗		56.3%	79.2%	86.2%	91.0%	94.6%	-
	✓	✗	56.0%	79.1%	86.1%	91.0%	94.2%	0.6886
	✓	✓	56.0%	79.1%	86.1%	91.0%	94.2%	0.6886

Focusing on the most significant differences between the results, we make the following observations:

- Chemformer's results are improved when switching to SYNTHESEUS, and then further when turning on deduplication. The former could be explained by the fact that SYNTHESEUS removes invalid molecules, which Chemformer (as a SMILES-based model) can produce.
- GLN's published results match those obtained with SYNTHESEUS with no deduplication. However, its top-k accuracies for $k > 1$ improve significantly with deduplication turned on.
- Graph2Edits' results are slightly worse than originally published, which may be explained by differences in hyperparameters such as the number of beams.
- LocalRetro (and by extension RetroKNN) used a relaxed notion of success, and we see that the results deteriorate significantly when using SYNTHESEUS. For LocalRetro, we additionally measured accuracy using authors' original code but replacing the relaxed match

with an exact one (see row labelled with "(exact match)"), which caused a similar drop in performance, confirming that the way of measuring accuracy is indeed responsible for the difference. Both LocalRetro and RetroKNN improve due to deduplication, but the final results still fall short of the originally reported numbers.

- MEGAN's published results improve slightly after moving to SYNTHESUES, and then there is a small further improvement from deduplication. We hypothesize the former might be a result of retraining the model (while the authors did release a checkpoint trained on USPTO-50K, our analysis seemed to indicate that model used a different data split for training, as the performance on our USPTO-50K test set was unrealistically high).
- MHNreact's results are not affected by deduplication, but the numbers we obtain with SYNTHESEUS are worse than those originally published; this may be explained by either the fact that we retrained the model or implementation details.
- RootAligned's published results closely match those obtained with SYNTHESEUS and are unaffected by deduplication, showing this model likely already conforms to many of the best practices from Section **??**.

Next, in Table 2 we present the exact numbers corresponding to the results from Figure 2. However, here we cannot compare to published results, as to the best of our knowledge these are not available.

Table 2: Generalization results on Pistachio in numeric form.

Model	top-1	top-3	top-5	top-10	top-50	MRR
Chemformer	26.1%	35.0%	37.1%	38.4%	39.1%	0.3089
GLN	25.4%	37.4%	41.5%	45.9%	50.3%	0.3254
Graph2Edits	26.3%	38.9%	43.4%	47.5%	50.1%	0.3363
LocalRetro	26.4%	40.0%	44.7%	48.8%	53.5%	0.3433
MEGAN	22.8%	35.3%	40.3%	45.1%	51.0%	0.3046
MHNreact	24.0%	36.3%	40.8%	45.1%	49.7%	0.3134
RetroKNN	27.1%	40.6%	45.0%	48.9%	53.4%	0.3488
RootAligned	27.7%	40.5%	44.6%	48.6%	52.4%	0.3510

3 Obtaining Multiple Results from Single-step Models

During evaluation, we need to obtain n results for a given input. It is worth noting that the value of n is used differently depending on model type: models based on templates and local templates (GLN, LocalRetro, MHNreact and RetroKNN) first process the input and then apply the templates until n results are obtained, while models that employ a sequential auto-regressive decoder (Chemformer, Graph2Edits and MEGAN) use beam search with n beams. These two approaches lead to different scaling, as in the former case the bulk of the computation is amortized and does not scale with n, while in the latter case the entire procedure scales with n essentially linearly. Finally, the RootAligned model is a special case, as it uses a combination of beam search and test-time data augmentation; scaling up either of these hyperparameters increases inference time and number of results, but the right balance between them requires careful tuning. In our work we used the default settings (20 augmentations, 10 beams) which correspond to a maximum of $20 \cdot 10 > n$ results being generated (recall that $n = 100$).

4 Search Algorithms Hyperparameter Tuning

To ensure a fair comparison, we tuned the hyperparameters of both MCTS and Retro* separately for each single-step model. For both algorithms we focused on tuning the component that directly interacts with the single-step model: policy in MCTS and cost function in Retro*. Notably, we did not vary many of the other components of the algorithms (e.g. reward function in MCTS or value function in Retro*) to avoid an infeasibly large search space.

All tuning runs used 25 targets from the ChemBL Hard set used in Tripp et al. (2022) and searched under a time limit of 5 minutes. As the primary objective we used the solve rate (i.e. number of solved targets), breaking ties first by the median and then mean number of non-overlapping routes found (formally, these three objectives were combined with weights 1.0, 0.1 and 0.01, respectively). For each search algorithm and single-step model combination we ran 50 trials using the default tuning algorithm in `optuna` (Akiba et al., 2019) to maximize the combined score.

For MCTS, we tuned the clipping range for the single-step model probabilities (lower bound in $[10^{-11}, 10^{-10}, ..., 10^{-5}]$, upper bound in $[0.9999, 0.999, 0.99, 0.9]$), temperature applied to the clipped distribution (in $[0.125, 0.25, ..., 4.0, 8.0]$), bound constant (in $[1, 10, 100, 1000, 10000]$) and node value constant (in $[0.25, 0.5, 0.75]$). For Retro*, we only tuned the clipping range (over the same values as for MCTS), as the temperature would have no effect due to using a constant-0 value function (referred to as Retro*-0 in Chen et al. (2020)).

In general, we found that the single-step probability clipping range has little effect on the algorithms, and so the performance of Retro* was not significantly improved through the hyperparameter tuning. Conversely, in MCTS parameters such as bound constant and temperature can have a sizable effect on the behaviour, and indeed choosing them carefully improved performance with respect to an initial guess. While MCTS seemingly performed worse than Retro* when using untuned hyperparameters, carefully setting the parameters of the former led it to perform on par with Retro*, echoing the conclusions from Tripp et al. (2022).

5 Maintenance Plan for Syntheseus

We intend to actively continue and support the development of SYNTHESEUS going forward, including adding new features, fixing any bugs, and improving documentation. As SYNTHESEUS is an open-source project on GitHub, anybody is free to modify and propose changes by raising an issue or opening a pull request. We are committed to promptly responding to and engaging with all issues and pull requests.

The code to reproduce all experimental results (apart from those utilizing the proprietary Pistachio dataset) is publicly available.

References

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2623–2631, 2019.

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.

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.

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`.

Daniel Wigh, Joe Arrowsmith, Alexander Pomberger, Kobi Felton, and Alexei Lapkin. Orderly: Datasets and benchmarks for chemical reaction data. *ChemRxiv prepring*, 2023.