Digital Discovery

ROYAL SOCIETY OF CHEMISTRY

PAPER

View Article Online
View Journal | View Issue



Cite this: *Digital Discovery*, 2026, **5**, 348

# Retrosynformer: planning multi-step chemical synthesis routes *via* a decision transformer

Emma Granqvist, [ab] Rocío Mercado [a] and Samuel Genheden [b]

We present RetroSynFormer, a novel approach to multi-step retrosynthesis planning. Here, we express the task of iteratively breaking down a compound into building blocks as a sequence-modeling problem and train a model based on the Decision Transformer. The synthesis routes are generated by iteratively predicting chemical reactions from a set of predefined rules that encode known transformations, and routes are scored during construction using a novel reward function. RetroSynFormer was trained on routes extracted from the PaRoutes dataset of patented experimental routes. On targets from the PaRoutes test set, the RetroSynFormer could find routes to commercial starting materials for 92% of the targets, and we show that the produced routes on average are close to the reference patented route and of good quality. Furthermore, we explore alternative model implementations and discuss the robustness of the model with respect to beam width, reward function, and template space size. We also compare RetroSynFormer to AiZynthFinder, a conventional retrosynthesis algorithm, and find that our novel model is competitive and complementary to the established methodology, thus forming a valuable addition to the field of computer-aided synthesis planning.

Received 16th April 2025
Accepted 15th November 2025

DOI: 10.1039/d5dd00153f

rsc.li/digitaldiscovery



## 1 Introduction

Organic chemical synthesis is fundamental to molecular design and discovery, yet designing efficient synthetic routes remains a significant challenge and is often a bottleneck.[1] Retrosynthesis, an approach that dates back to the 1960s,[2,3] addresses this by systematically deconstructing target compounds into readily available starting materials. Recent advances in artificial intelligence (AI) and deep learning (DL) have greatly contributed to the fields of retrosynthesis prediction and computer-aided synthesis planning,[4–7] leveraging expanding reaction datasets.[4,8] However, there are outstanding challenges, especially for efficient search algorithms in multi-step retrosynthesis planning.

In this work, we introduce the RetroSynFormer, a novel approach to multi-step retrosynthesis prediction guided by a Decision Transformer (DT).[9] By framing retrosynthesis as a sequence modeling problem, RetroSynFormer predicts reaction steps autoregressively, conditioning each decision on previous steps to capture reaction patterns across datasets. Unlike conventional search-based methods, RetroSynFormer inherently learns context-dependent synthesis strategies, enabling the model to utilize the full history of the path it has taken to synthesize a compound at each step and thereby potentially improving route prediction.

We summarize the three main contributions of our work as follows:

- This is the first example of the DT being used for retrosynthesis planning problems, where we recast retrosynthesis optimization as a sequence modeling task;
- This is one of the first demonstration of true multi-step retrosynthesis planning using DL models, as opposed to prior work which has focused on the sequential application of single-step models;
- We thoroughly benchmark our model using meaningful metrics, such as the success rate and top-1 accuracy, providing a detailed comparison to the current SOTA: AiZynthFinder.

## 2 Background

### 2.1 Retrosynthesis prediction

Organic synthesis plays a central role in nearly all chemical industries, from drug discovery to material discovery.[1] The aims of organic synthesis are to efficiently manufacture organic compounds through a series of chemical reactions; this is a complex problem as each compound can be assembled in multiple ways, creating a huge chemical space and making for a challenging search problem. The synthesis of chemical compounds therefore represents a critical bottleneck in molecular design, a challenge which motivated the conceptualization of retrosynthesis by E. J. Corey in 1967[3] following earlier developments.[2] Retrosynthesis refers to the idea of iteratively breaking down a target compound until all building blocks are readily available starting materials. Since then,

[a]*Department of Computer Science and Engineering, Section for Data Science and AI, Chalmers University of Technology and University of Gothenburg, Gothenburg, Sweden*
[b]*Molecular AI, Discovery Sciences, R&D, AstraZeneca, Gothenburg, Sweden*

emergence of computational technologies and resources has enabled computer-assisted synthesis planning for more productive and efficient retrosynthesis prediction. In recent years, research on retrosynthesis has been further expedited due to the recent developments of AI and DL,[4–7] and the emergence of larger collections of reaction data on which the AI-driven retrosynthesis can be trained.[4,8] This development has enabled chemists to save valuable time and effort when designing synthetic experiments which can lead to finding the right compounds faster.[10]

Retrosynthesis can broadly be categorized into template-free and template-based methods. Template based methods apply predefined reaction rules, templates, which generally result in reliable reactions grounded in known chemistry, while the template-free methods learn the reactions directly from the data and thus are not bounded by the templates. The template-free methods can then generalize beyond the known reaction rules with the risk of generating chemically unfeasible reactions. Furthermore, retrosynthesis prediction is typically separated into two tasks, single-step retrosynthesis prediction and multi-step retrosynthesis planning. In single-step retrosynthesis, the task is to decompose a compound to one or more precursor molecules, or reactants. Multi-step retrosynthesis aims to find a sequence of reactions—a synthesis route—which describes how to make a chemical compound from a set of readily available starting materials. The task is typically approached by iteratively using a single-step model to break down a compound into precursors until all starting materials belong to a set of available building blocks. Single-step models have been extensively researched, and we refer to a recent review for an overview.[11] Conversely, multi-step retrosynthesis typically employs a single-step model with a search algorithm such as Monte Carlo tree search (MCTS)[12,13] or A* search;[14,15] reinforcement learning (RL) has also been proposed for this task.[16] However, methods such as MCTS typically only consider the current state (*i.e.*, a single molecule) when making the predictions for the next reaction. Some work has proposed including additional context into models, such as the parent reactions, when making the predictions.[17] We hypothesize here that including the additional route context in the predictions would be beneficial to retrosynthesis modeling, and that this could result in a model which can learn common patterns or combinations of reactions across the entire route dataset.

### 2.2 Language models in synthesis planning

In recent years, a variety of transformer models have been widely adopted for various different tasks including machine translation, natural language processing and computer vision. Nonetheless, these models have also gained widespread use in computer-aided synthesis planning. One such example is the Chemformer, a molecular transformer model that has been trained for multiple tasks, including retrosynthesis planning, forward synthesis, and property prediction.[18] The Chemformer model has also been integrated with AiZynthFinder for multi-step retrosynthesis planning.[6] One other application that uses large language models for multi-step retrosynthesis prediction is DirectMultiStep, which predicts routes as single strings and thus bypasses the need for single-step methods.[19] Although an interesting approach, predicting a route as a single string presents a unique set of challenges which are difficult to overcome, such as the inability to condition routes on available starting materials and the generation of invalid routes.

### 2.3 Decision transformer

Here we present the RetroSynFormer, a novel approach to multi-step retrosynthesis prediction where the search is guided by a DT model.[9] The DT framework, originally proposed for RL tasks, reformulates decision-making as a sequence modeling problem, leveraging the success of transformers in natural language processing. Instead of learning a traditional value function or policy, a DT models trajectories of states, actions, and rewards as sequences, predicting future actions autoregressively based on past context. This formulation makes DT particularly well-suited for offline RL settings, where a fixed dataset of trajectories is available, and direct interaction with the environment is costly or impractical.

In retrosynthesis prediction, running additional experiments to test new chemical reactions and explore the search space is often infeasible on a short time-scale, making offline learning essential. By using a DT, our approach conditions action predictions (*i.e.*, reaction templates) on previous states and actions in a retrosynthetic route, allowing it to capture common reaction patterns and dependencies observed in historical data. This enables the RetroSynFormer to effectively generalize across diverse reaction pathways without requiring explicit exploration through new experiments. Starting from a target molecule, the model autoregressively predicts the next reaction step until reaching a stopping criterion, constructing complete retrosynthetic routes in a flexible and data-driven manner. Unlike other sequence-based RL approaches such as the Searchformer,[20] which integrates A*-like search with a transformer-based policy, our approach directly models retrosynthetic trajectories with learnable rewards, making it more flexible for data-driven generalization. As the DT leverages the full route trajectory information, making it well-suited for retrosynthesis planning scenarios where long-horizon dependencies are critical, we believe it offers advantages to policy-gradient approaches that may require reward shaping or explicit policy optimization.[21]

## 3 Methodology

RetroSynFormer is a DT model to predict the next reaction (action) in a synthesis route. During inference the model uses a retrosynthesis environment to generate the next reactant molecule(s) (state) and a reward for each action, iteratively generating the retrosynthesis route until one of the stopping criteria is met (Fig. 1).

### 3.1 Data

RetroSynFormer is trained on a subset of routes from the PaRoutes dataset[4] where the routes have been derived from the reactions in the USPTO dataset provided by Lowe.[22] The PaRoutes dataset includes 457 166 routes as JSON data objects with nested dictionaries, where the molecules are represented

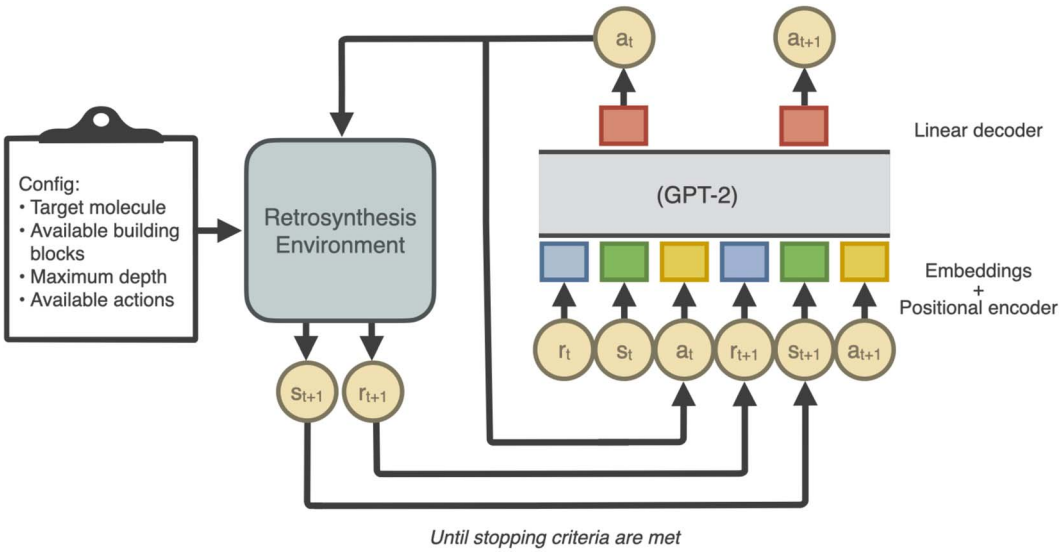


Fig. 1 Overview of the RetroSynFormer method. The Decision Transformer, based on GPT-2 architecture, predicts an action, *a*, (reaction template) which is passed to the Retrosynthesis Environment that generates the new state, *s*, along with the corresponding reward, *r*. The environment keeps track of the chemical context, such as available building blocks, reaction templates, and maximum depth allowed, as well as the current route, its status, and stop criteria for when to end the predictions.

as SMILES and the reactions are atom-mapped reaction SMILES with an associated reaction template extracted *via* RDChiral.[23,24] The full PaRoutes dataset includes a total of 42 551 reaction templates and routes for 175 164 unique targets.

Here, we formulate the retrosynthesis task as a classification task and at each step we aim to predict the correct action, *i.e.*, a reaction template. By reducing the number of possible templates, the complexity of the task can be reduced. Therefore, we sorted the templates based on the frequency in the routes and extracted routes with only the most frequently occurring templates. We have created three datasets: small, standard and large, each based on different frequency-cut-offs for the included templates. For the standard dataset, all routes with only the 3000 most commonly used templates were taken, giving in total 247 531 (54%) routes, and for the small and large dataset we instead extracted the 1000 and 6000 most common templates, respectively (for detailed numbers, see Table 1). To further reduce the number of templates, we followed the procedure of Heid *et al.*[25] to identify templates that are subgraphs of other templates, which correspond to the final size of the action space. For the standard dataset this gives 1572 templates. The process of creating the dataset is illustrated in Fig. 2a.

The standard dataset is used in all experiments while the small and large datasets are only used for the results in Section 4.6. Note that the standard dataset is a subset of the large dataset and that the small dataset is a subset of the standard dataset, as illustrated in Fig. 2b.

Synthesis routes are naturally tree-like data structures. In order to easily process them using the DT, we need to convert the tree-like routes into sequences of states, actions, and rewards, illustrated in Fig. 3. Where the molecules are the states, the reaction template is the action and for each step, a reward is calculated. When a reaction decomposes a molecular state into more than one intermediate reactant, a branch is formed in the route. To standardize the way branching points are handled in the data, reactant SMILES are sorted according to length and the molecule with the longest SMILES is expanded first, while the other molecule is added to a stack. In this way, we follow a depth-first order when transforming the route into a sequence: only after a given branch is completely rolled out (all leaves are building blocks) do we continue with the next branch.

Finally, the set of available starting materials, *i.e.*, building blocks, is defined by the set of all leaf molecules from the extracted routes. Details can be found in Table 1.

### 3.2 Data splitting

The same data splitting is followed three times for each of the large, standard, and small datasets. These datasets enable us to

Table 1 Details of the datasets. The first three rows show the number of unique routes in the training, validation, and test sets for the three main datasets curated in this work. All datasets were created from PaRoutes[4] data. The last row includes the mean pairwise Tanimoto similarity between 10 000 random targets from the training set and test set

| Dataset | Small | Standard | Large |
|---|---|---|---|
| Training set | 44 736 | 67 180 | 86 048 |
| Valid set | 5362 | 8222 | 10 645 |
| N1 test set | 2168 | 4320 | 5631 |
| N5 test set | 1732 | 3569 | 5260 |
| # Templates | 588 | 1572 | 2986 |
| # Building blocks | 38 521 | 58 251 | 72 737 |
| Total # unique targets | 53 626 | 82 222 | 106 452 |
| Total # routes | 144 812 | 326 294 | 442 844 |
| Test–train mean similarity | 0.127 | 0.123 | 0.126 |

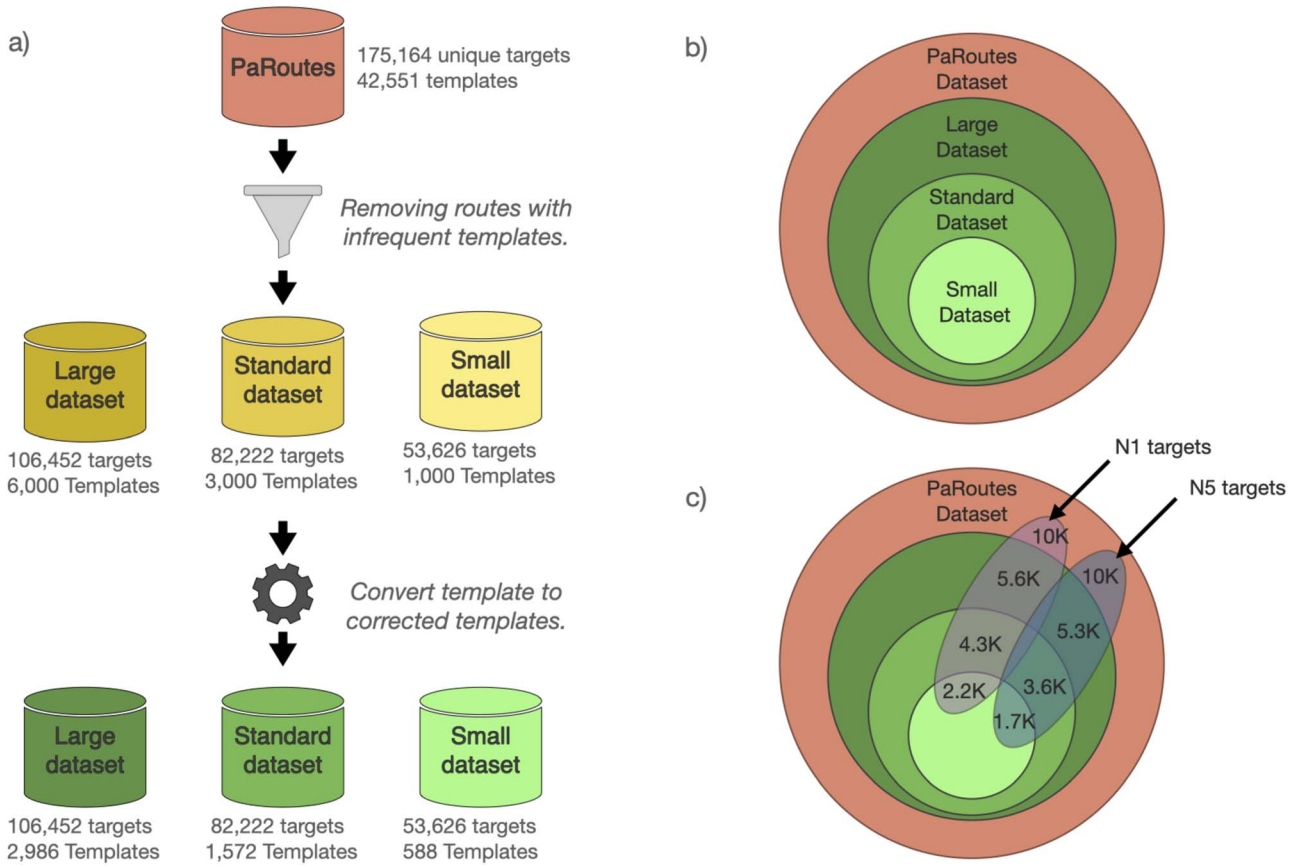


Fig. 2 Overview of the data processing steps to create the small, standard and large datasets. (a) The filtering process. First, we filter routes from PaRoutes to keep only those using the 3000 most common templates. After filtering, the remaining reaction templates are further reduced by converting the templates to the "corrected" templates described in ref. 25. (b) Illustration of the relationship between PaRoutes and the three datasets, large, standard, and small, showing how the large dataset is a subset of the PaRoutes dataset, the standard dataset is a subset of the large dataset, and the small dataset is a subset of the standard dataset. (c) Illustration showing how the N1 and N5 sets relate to the three datasets in (b); as in (b) the N1 and N5 sets in the smaller datasets are subsets of the larger ones.

study the effect of training set size on model performance; details for each dataset can be found in Table 1.

We split the data by the target compounds rather than by the routes and in this way ensure that there is no overlap between the targets in the hold-out test set and the training and validation sets. The mean Tanimoto similarity between the compounds in the test set and training set (based on 1024-bit Morgan Fingerprints) is also reported. First, we created a hold-

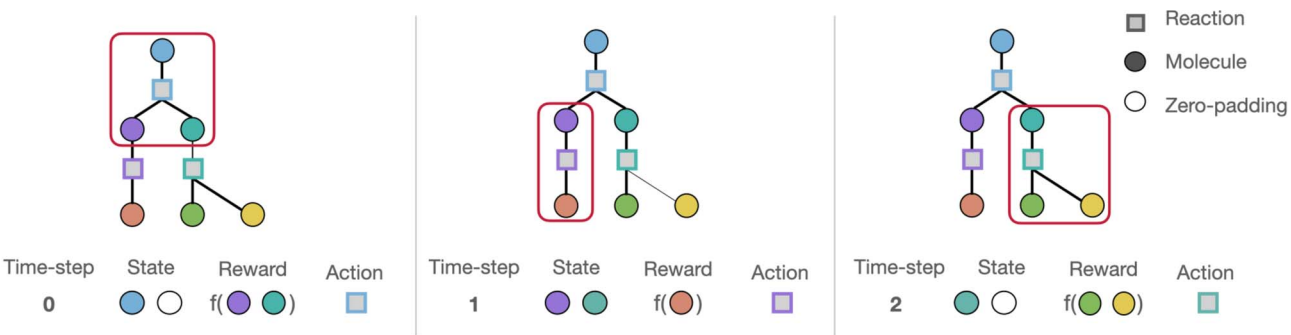


Fig. 3 Overview of how the synthesis routes are converted from tree structures to state, reward and action sequences for the Decision Transformer model. In the example, a route is decomposed into a 3-step state, reward and action sequences. Circles represent the molecules forming the states and squares represent the reactions forming the actions. The reward at each time-step is a function of the reactants given by a reaction, *i.e.*, the next state. A state is defined by either a single or pair of molecules, where in the case of a single molecule, zero-padding is used for the second "empty" molecule (represented by the empty circle in time step 0 and 2). After each time-step, the status of each reactant is evaluated and if the compound is an intermediate (turquoise and purple circles) it will be put in a stack of unexplored states and decomposed in the next time-steps. If the compound is a building block (red, green, and yellow circles) then it is the end of a branch.

out test set from the N1 and N5 targets of the PaRoutes dataset,[4] and because we subsample the templates, we also needed to subsample the N1 and N5 sets (see Fig. 2c). For the standard dataset the number of targets are 4320 and 3569 for N1 and N5, respectively. Here, the N1 set consists of relatively simple, predominantly linear synthetic routes, reflecting the broader distribution of extracted routes. The N5 set also contains routes, generally longer and more complex, and was designed to be a more challenging benchmark for retrosynthesis prediction.

After removing the N1 and N5 targets and their corresponding routes, we randomly selected 10% (8222 for the standard dataset) of the total number of unique target compounds from the remaining target compounds to create the validation set. Finally, the rest of the target compounds formed the training set (67 180 target compounds for the standard dataset).

### 3.3 Decision transformer

The RetroSynFormer architecture leverages the DT to autoregressively predict the next chemical reaction given all previous reactions, molecules, and rewards. Here, we used the GPT-2 DT model implemented in the Transformers repository.[26]

The input to the DT model consists of three vectors: actions, $A = [a_0,\ldots, a_t]$, states, $S = [s_0,\ldots, s_t]$, and rewards, $R = [r_0,\ldots, r_t]$. Each element in $A$ is a one-hot vector encoding a reaction template where $a_i \in \{0,1\}^{1,572}$ for the standard dataset. Each template determines the chemical transformation to apply to the target molecule. Each element in $S$ represents the target molecule(s), so that $s_{i+1} = a_i\ (s_i)$; here, $s_i \in \mathbb{R}^d$ where $d$ is the length of the molecular fingerprint used. Finally, each element in $R$ is a reward such that $r_i = f(s_i) \in \mathbb{R}$, providing an estimate of the quality of the route at the current time-step, $i$. The total number of time-steps is denoted by $t$ in a reaction sequence. We used Optuna for hyperparameter optimization to determine the model parameters; details and optimal parameters are provided in Table S1 in the SI.

All the presented models have been trained on only one route per target, where, for each target, the route was randomly selected from all available routes.

**3.3.1 Actions.** The reaction templates used here are the "corrected" templates following Heid *et al.*.[25] The number of available actions is determined by the number of templates that are used in the routes in our dataset. As described in Section 3.1, we are using a set of 1572 reaction templates for the standard dataset, which is also the size of our action space, $A \in \{0,1\}^{t\times 1572}$, where $t$ is the total number of reactions (time-steps) sampled in a given route. During inference, RDChiral[23] is used to get the reactants for the next state given the template and target molecule. The size of the action space for small and large datasets is given in Table 1 and corresponds to the number of available templates in the respective dataset.

**3.3.2 States.** The initial state contains a Morgan fingerprint representation[27] of the target molecule, generated using RDKit.[28] The next states contain the Morgan fingerprints for the unexplored intermediate reactant(s) generated by the next reaction. The state dimension needs to be constant when predicting each timestep even though the actual numbers of molecules in the state will vary. We have therefore truncated the state vector which is passed to the model to include two molecules. If only one molecule is in the state (as in the initial state or for uni-molecular reactions), zero padding is added; if there are three or more molecules, only the two largest molecules are included in the state vector. However, all reactants will of course be explored by the model. This restriction only impacts the few templates which have three reactants but does not impact the overall performance. Importantly, the molecular states are treated as a stack, meaning that the last molecule added to the state will be the first explored in the next step, while any other reactants are put in a stack of all unexplored intermediates. This leads to a depth-first search exploration of the states. When calculating the reward for a given step, all reactants predicted by the reaction are considered. Each molecule is represented by a 1024-bit Morgan fingerprint, such that the total dimension of the state is 2048 bits.

**3.3.3 Reward function.** In contrast to other route scoring methods[29,30] that evaluate the complete route, here we need a reward function that can evaluate the goodness of an intermediate route after each reaction. This poses a challenge, as assessing route quality before it is complete is inherently more difficult and we cannot use any of the commonly used route scorers. Instead, we chose here to include two aspects of the states when calculating the reward at each time-step: (1) the condition of the molecule(s) in the current state and (2) the depth. For a molecule in a route there are three distinct conditions it can meet: (1) a building block, (2) an intermediate, or (3) a dead end. A dead end means that either no reaction template can be applied or that the maximum depth has been exceeded. The second component of the reward function, the depth, simply measures how many transformations the current molecule is away from the target. The depth is included in the reward to incentivize the model to not only focus on finding building block compounds but also to favor shorter routes by penalizing greater route depths. See Algorithm 1 in the SI and Fig. 4 for an example with a 4-step route. To determine how much each condition should contribute to the reward and how to incorporate the depth, we performed an optimization using Optuna;[31] see Table S2 for details.

### 3.4 Inference

The DT model can be viewed as a policy which predicts the next action. However, it lacks the chemical context such as knowledge about the building blocks, the route, what actions are applicable to what molecule, when a route is solved, *etc.*, which is needed during inference. Therefore, we have implemented a retrosynthesis environment to be used in combination with the DT, as is typically done for RL. During inference the environment is used to apply the predicted templates to get the new reactants, evaluate the status of the reactants, calculate the reward function, and build and evaluate the synthesis route. In more detail, the probability for each reaction template is predicted by the DT, which determines the first applicable template with the highest likelihood as the next action. The action is used to take a step in the environment using RDChiral,[23] which returns the reactants given

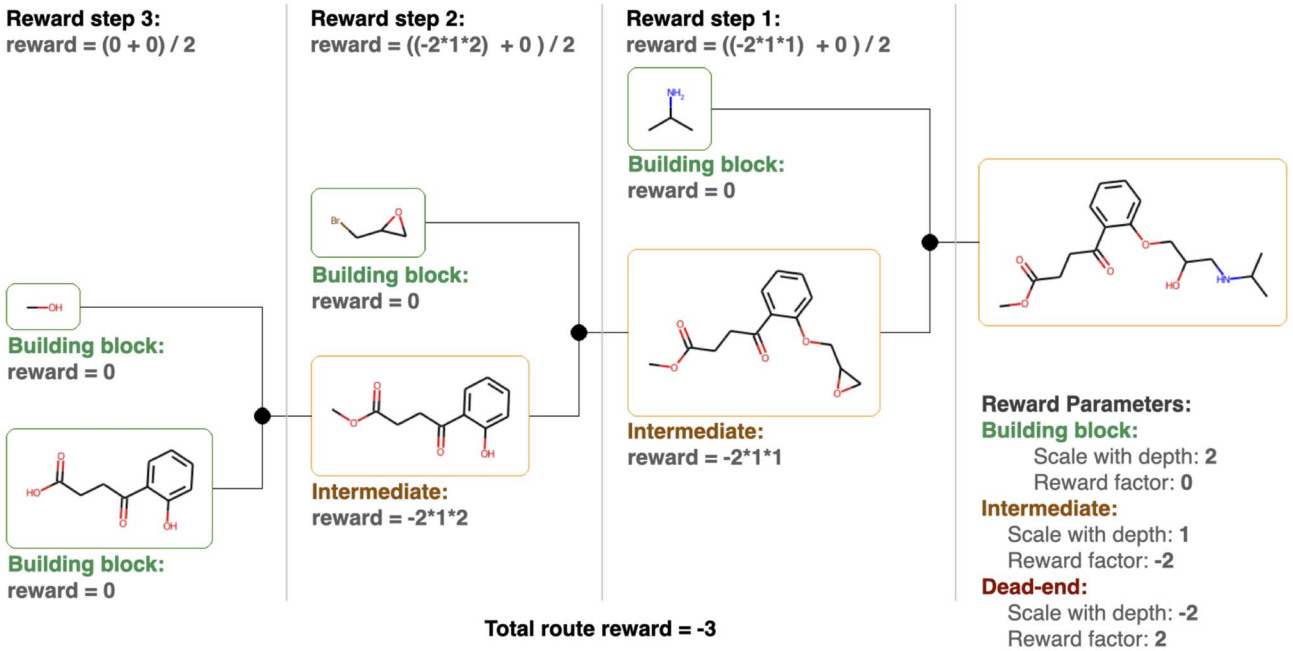


Fig. 4 Example of a synthesis route annotated with the associated reward calculations for each step, showing how the route reward is calculated using the reward parameters. The reward for each step is the average of all reactant molecules in that step and the total route reward is the sum of the reward over all steps.

the reaction template (action) and the next molecule (first molecule in the state). Thereafter, the molecules in the new state are sorted based on size and evaluated. If a molecule in the state is part of the building block set, that branch is solved and thus the molecule does not need to be further expanded. However, if there are no applicable actions (reaction templates) for that molecule or the route exceeds the maximum depth, the molecule is labeled as a dead end and the route is unsolved. If the molecule, however, is determined to be an intermediate, then we add it to the stack of states to be expanded.

The environment also monitors for loops in the route that could stem from recurring intermediates. Loops are not a desirable behavior in a route, so if an intermediate occurs more than once in a linear route, the route is terminated and labeled as unsolved.

### 3.5 Beam search

Since beam search has proven to be a useful strategy for various natural language tasks, often resulting in improved performance, we have implemented it for the RetroSynFormer.[32] Using beam search, the top $n$ templates given a specific compound (state) are applied at each time-step and evaluated; the same procedure is then applied to each of the $n$ resulting new states, leading to $n^2$ new states. These $n^2$ states are sorted based on the cumulative route likelihood and the top $n$ beams kept for the next iteration. This procedure is repeated until either a route is solved or the maximum depth is reached. The beam search enables the RetroSynFormer to consider multiple routes for each target and this way increase the likelihood of finding a solved route. The special case of a beam search using $n = 1$ is equivalent to a greedy search.

### 3.6 Evaluation metrics

To assess the model's performance on the retrosynthesis task, we computed standard metrics which evaluate a model's ability to find solved routes for a given target. Specifically, we measured success rate, defined as the percentage of targets for which the model identifies a route where all starting materials are available in stock. Additionally, we assessed top-1 accuracy, which calculates the percentage of targets for which the predicted retrosynthetic route is identical to the ground-truth route. To further compare predicted and target routes, we also evaluated the model using distance metrics to compare the similarity of the predicted routes with the target routes. We calculated the route similarity using the tree edit distance (TED). Since exact TED computation is often infeasible, we used an approximation based on an LSTM model, following Genheden *et al.*[33]

To further evaluate the predictions made by the model we also calculate the action accuracy and reaction class accuracy. Here, we compare each action in the predicted routes to the corresponding step in the target route. If the predicted and target routes are not the same length, we exclude the additional actions of the longer route in order to calculate these accuracies.

Finally, we assess the route quality using the DeepSet route score described in Yujia *et al.*[34] and implemented in rxnutils.[35] This score has incorporated human expert assessments of synthesis routes and can be used to categorize routes as either "good" for scores between 0 and 5, "plausible" if the score is between 5 and 9, or "bad" if the score is between 9 and 15.

### 3.7 Baseline

As a baseline, we trained AiZynthFinder, a template-based one-step retrosynthesis model on the reactions from the same

routes as the RetroSynFormer model using AiZynthTrain.[24] AiZynthFinder is a template-based Monte Carlo tree search method. The search is guided by a policy to suggest possible precursors at each step. The policy here is a neural network that has been trained on reaction templates. We then performed retrosynthesis experiments with the template-based one-step model using AiZynthFinder using the same templates as used by RetroSynFormer. The baseline was evaluated on the same N1 and N5 target sets from PaRoutes.[4] In these experiments, we used the same set of building blocks (stock) as in the RetroSynFormer experiments. We used the default AiZynthFinder settings, *i.e.*, 100 iterations of Monte Carlo tree search, and a maximum depth of 6 (the same as RetroSynFormer). Rigorous comparisons of the standard AiZynthFinder with a template-free single-step model (Chemformer) have been made previously, demonstrating that the template-free model solves more routes but with a longer search time.[6]

# 4 Results

The RetroSynFormer was used to predict routes for 1500 random targets from the N1 test set, and 1500 random targets from the N5 test set. The training and evaluation steps were repeated three times for statistical analysis. For these experiments, RetroSynFormer was trained on one randomly sampled route per unique target (*e.g.*, 67 180 routes for the standard dataset). By default, the RetroSynFormer returns only the first route found *via* the beam search, although it can also generate multiple routes. Unless otherwise stated, we use a beam width of 50 and return the route with the greatest cumulative likelihood; we call this model RetroSynFormer50. As a baseline, AiZynthFinder was trained as described in Section 3.7 and evaluated on the same test sets. For comparison with the baseline, the highest ranked route from AiZynthFinder was extracted for each target.

## 4.1 Retrosynthesis performance

There are many approaches for evaluating the performance of retrosynthesis algorithms,[4] and one of the most common and also most important metrics is the success rate (the percentage of targets for which a solved route is found). This is a prerequisite for evaluating other aspects of the predictions such as the quality of the sampled routes. However, because we have reference routes for the targets as extracted from patents, we can not only calculate if the predicted route is identical to the target route (the accuracy), but also the similarity.

A summary of the RetroSynFormer50 performance on the N1 and N5 targets is presented in Table 2. Although RetroSynFormer50 results in a slightly worse success rate than the AiZynthFinder baseline, we can see that it performs comparably in many other aspects. For example, the success rate for the N1 test set between the models differs by less than 2%, and the top-1 accuracy differs by only 0.039 and is nevertheless rather low for both models. Compared to the N1 set, the N5 set is more difficult for both models as the success rate and top-1 accuracy is lower. The TED and average route length in Table 2 are only calculated for the solved routes. Here we only see some smaller differences between the models. It seems like the difference between the target and predictions in general are larger for the N5 set and also that on average the routes for the N5 targets are slightly longer, although the routes are on average short.

In Fig. 5a we can see the distribution of route lengths for the predictions compared to the targets. We observe that the distributions for the AiZynthFinder routes and RetroSynFormer routes are very similar and, interestingly, that the target routes are in general longer compared to the predicted routes. This suggests that it is possible to find shorter routes with the available stock than what was used for the original patent routes. In Fig. 5b, we plot the distribution of the route reward in the predicted routes from RetroSynFormer compared to the target routes. The predicted routes show a wider distribution than the target routes, and the median reward is rather different. Equivalent figures for results on the N5 set are available in Fig. S1a and b – and we observe the similar trends as for the N1 set. In addition to the evaluation on the test set, an additional evaluation on a separate set of targets has been included in SI D.

## 4.2 Complementarity of retrosynthesis approaches

As demonstrated, the RetroSynFormer is almost comparable to AiZynthFinder with respect to the success rate. A natural question that arises is if the solved targets are the same across the models, or if the targets solved by each model are complementary. As described above, the two models have been evaluated on the same N1 and N5 target sets, and the number of

Table 2 Performance of the RetroSynFormer compared to the AiZynthFinder baseline on retrosynthesis planning tasks using a diverse suite of evaluation metrics. Arrows indicate the direction of better performance for each metric

| | RetroSynFormer50[a] | | AiZynthFinder | |
|---|---|---|---|---|
| Test set | N1 | N5 | N1 | N5 |
| Success rate (%) ↑ | 0.924 | 0.899 | 0.939 | 0.924 |
| Top-1 accuracy ↑ | 0.106 | 0.058 | 0.143 | 0.071 |
| Mean time per route(s) ↓ | 68.1 | 81.9 | 5.45 | 5.8 |
| Mean TED ↓ | 5.58 | 7.08 | 5.40 | 7.12 |
| Mean # reactions per route ↓ | 2.34 | 2.55 | 2.52 | 2.92 |

[a] RetroSynFormer results show averages over three runs using beam width 50. Standard deviation for the RetroSynFormer models is ≤ 0.004 for the success rate, ≤ 0.001 for the top-1 accuracy, ≤ 1.7 for the mean time per route, ≤ 0.05 for the mean tree edit distance (TED), and ≤ 0.01 for the mean # reactions per route.

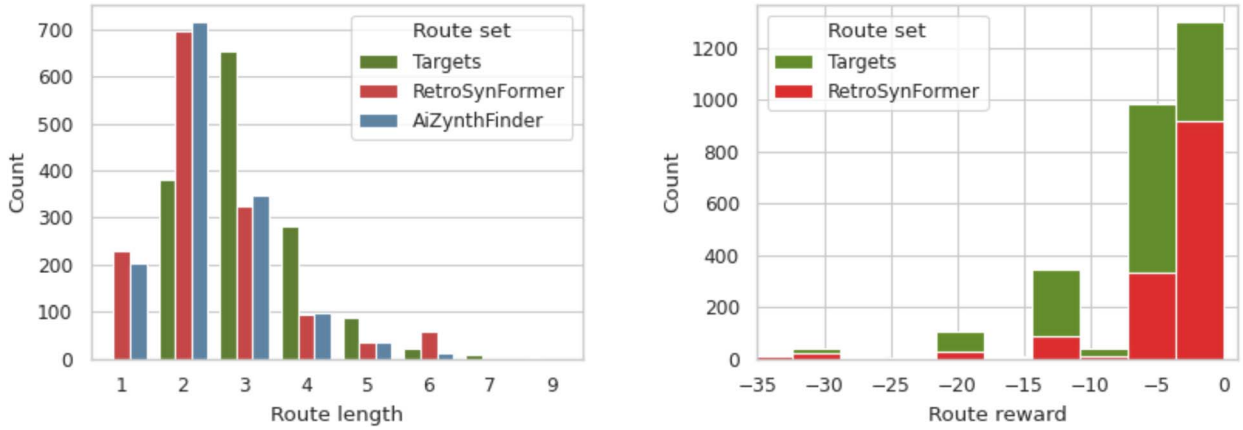


Fig. 5 Route characteristics for the N1 test set target routes compared to the RetroSynFormer and AiZynthFinder predictions (including only the solved routes) for the same targets. (a) The median route length is 3 for the target routes and 2 for the RetroSynFormer and AiZynthFinder routes, indicating that solved routes are generally shorter than those in the reference dataset. (b) The median reward for the route reward is −6 for the target routes and −2 for the RetroSynFormer routes. (a) Histograms of N1 route lengths, measured as the number of actions per route, for each route set. (b) Stacked bar plot showing the distribution of N1 route rewards in the target *versus* RetroSynFormer-predicted routes.

targets solved by each model is reported in Table 3. Here, we can see that 88.5% of the N1 routes are solved by both models and that 3.9% and 5.4% are solved by only RetroSynFormer and AiZynthFinder, respectively. This means that only 32 routes (2.1%) could not be solved by either model. Similarly, for the N5 set only 3.1% of targets could not be solved by any model. To conclude, by combining both methods we can solve 97–98% of the routes and potentially reduce the error compared to using an individual model. We have also compared the tree edit distance between the routes solved by both models and observe that the TED for the solved routes are lower than the observed TED for the reference routes presented in 2. However, we clearly see that the different methods suggest different routes.

In Fig. 6 we compare an example route generated by RetroSynFormer to the route generated by AiZynthFinder for the same substituted benzamide target from the patent US-8680159-B2, a patent for bradykinin 1 receptor modulating compounds. In this example, RetroSynFormer generates a route in two steps and is very efficient. The route starts with a reductive amination, a step with some support from the literature (*e.g.*, US8519124B2 or WO/2020/103896 patents as identified by a search in the Pistachio database[36,37]), followed by a deprotection of the amine group. The patented target route also ends with a reduction of a protected aminocyclohexanone and naphthyridine, followed by deprotection, but takes some steps to build up the aminocyclohexanone intermediate from cheaper starting materials. AiZynthFinder, on the other hand, fails to employ the template for the reductive amination, and thus is unable to break the bond between the naphthyridine and cyclohexane rings. After many unproductive steps, the search hits the maximum tree depth and stops at a starting material not in the commercial stock. Nevertheless, we want to stress that this example only serves to illustrate a scenario where RetroSynFormer produces a route complementary to AiZynthFinder, and we could likely identify other examples where AiZynthFinder is better suited to solve a given target than RetroSynFormer.

Table 3 Comparison in terms of number and percentage of the N1 and N5 targets successfully solved, showing how many are solved by only RetroSynFormer, only AiZynthFinder, both models, or by neither of the models. There are 1500 targets in each of these N1 and N5 sets. The TED column refers to the tree edit distance between the solved routes

| | N1 | | | N5 | | |
|---|---|---|---|---|---|---|
| Solved by | Count | Percent | TED | Count | Percent | TED |
| Only RetroSynFormer | 59 | 3.9% | | 67 | 4.5% | |
| Only AiZynthFinder | 81 | 5.4% | | 112 | 7.5% | |
| Both | 1328 | 88.5% | 3.81 | 1274 | 84.9% | 4.81 |
| None | 32 | 2.1% | | 47 | 3.1% | |

### 4.3 Analysis of chemistry in predicted routes

To gain a deeper understanding of the model behavior, we further analyzed the reactions in the solved predicted routes, *i.e.*, the routes that terminated in purchasable (in-stock) starting material(s). In Table 4, we show that the total number of reactions observed in the solved routes is on the order of 3000 compared to 4000 in the target routes, an observation that is also reflected by the shorter average route lengths of RetroSynFormer-predicted routes (Table 2). Furthermore, these reactions are represented by about 650 and 700 unique templates for the RetroSynFormer and AiZynthFinder, respectively, which is fewer than the approximately 800 unique templates in the target routes. However, the unique number of reaction classes is about 60 for both the predicted and target routes. This shows that there is a great redundancy in the templates, *i.e.*, several templates represent similar reactions,

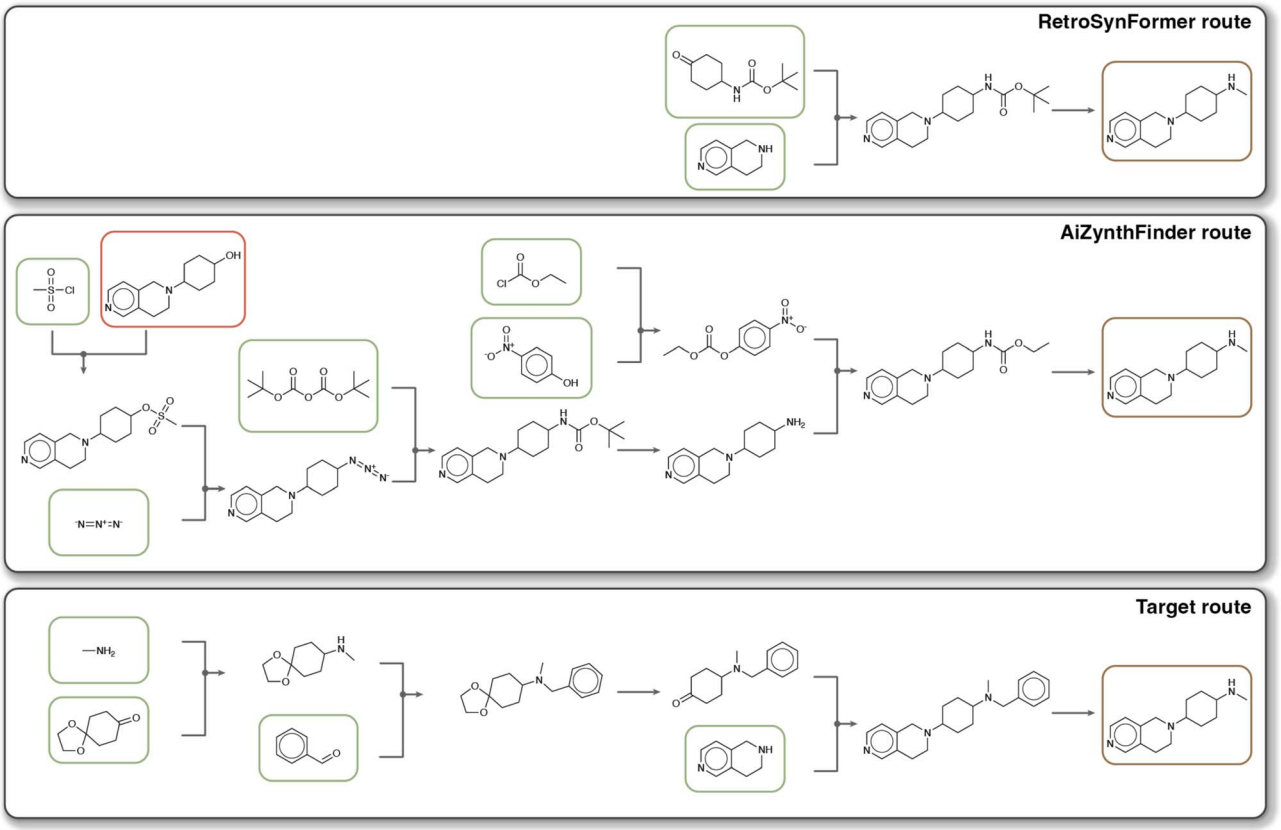


Fig. 6 Comparison of an example route generated by RetroSynFormer for a substituted benzamide target from the patent US-8680159-B2 (top panel) to the route generated by AiZynthFinder (middle panel); the target route is shown for reference (bottom panel). The final product is the same in all three panels and denoted with a golden box. The reactions are visualized as arrows, the green boxes indicate purchasable building blocks and the red box indicates a dead-end state that is not a purchasable building block (*i.e.*, the maximum route depth was reached).

Table 4 Summary statistics, action/class/route accuracies, and route scores for the solved routes. DeepSet route score estimates the route quality, and routes with scores between 0 and 5 are considered to be of high quality. Arrows indicate the direction of better performance for each metric

| | RetroSynFormer50 | | AiZynthFinder | | Targets | |
|---|---|---|---|---|---|---|
| | N1 | N5 | N1 | N5 | N1 | N5 |
| Number of total reactions | 3047 | 3189 | 3234 | 3556 | 4093 | 4567 |
| Number of unique templates | 669 | 650 | 712 | 706 | 795 | 812 |
| Number of unique reaction classes | 63 | 61 | 62 | 63 | 62 | 61 |
| Action accuracy ↑ | 0.266 | 0.223 | 0.312 | 0.239 | — | — |
| Class accuracy ↑ | 0.374 | 0.321 | 0.415 | 0.333 | — | — |
| Route accuracy ↑ | 0.127 | 0.084 | 0.163 | 0.096 | — | — |
| Unordered route accuracy ↑ | 0.177 | 0.125 | 0.216 | 0.147 | — | — |
| DeepSet route score[34] ↓ | 3.139 | 3.313 | 3.336 | 3.531 | 3.137 | 3.411 |

such that the model can predict different templates that lead to identical disconnections.

In Fig. S2a and S3a we plot histograms of the 15 most common reaction templates in the N1 and N5 target routes, respectively, and compare them to the histograms from the predicted routes. We can observe that there is a significant discrepancy—popular templates in the target routes are not necessarily the most frequently used templates in the predicted routes. Furthermore, we plot histograms of the 15 most common reaction classes in the target routes in Fig. S2b and S3b, and we observe that there is significantly less discrepancy in the templates between the reference N1 and N5 routes compared to those in the predicted routes for the same sets. This is also reflected in the higher class accuracies for the RetroSynFormer and AiZynthFinder compared to the action accuracies (Table 4). This indicates that routes generated by the respective methods differ in the preferred templates and that combining both methods might increase the diversity of routes.

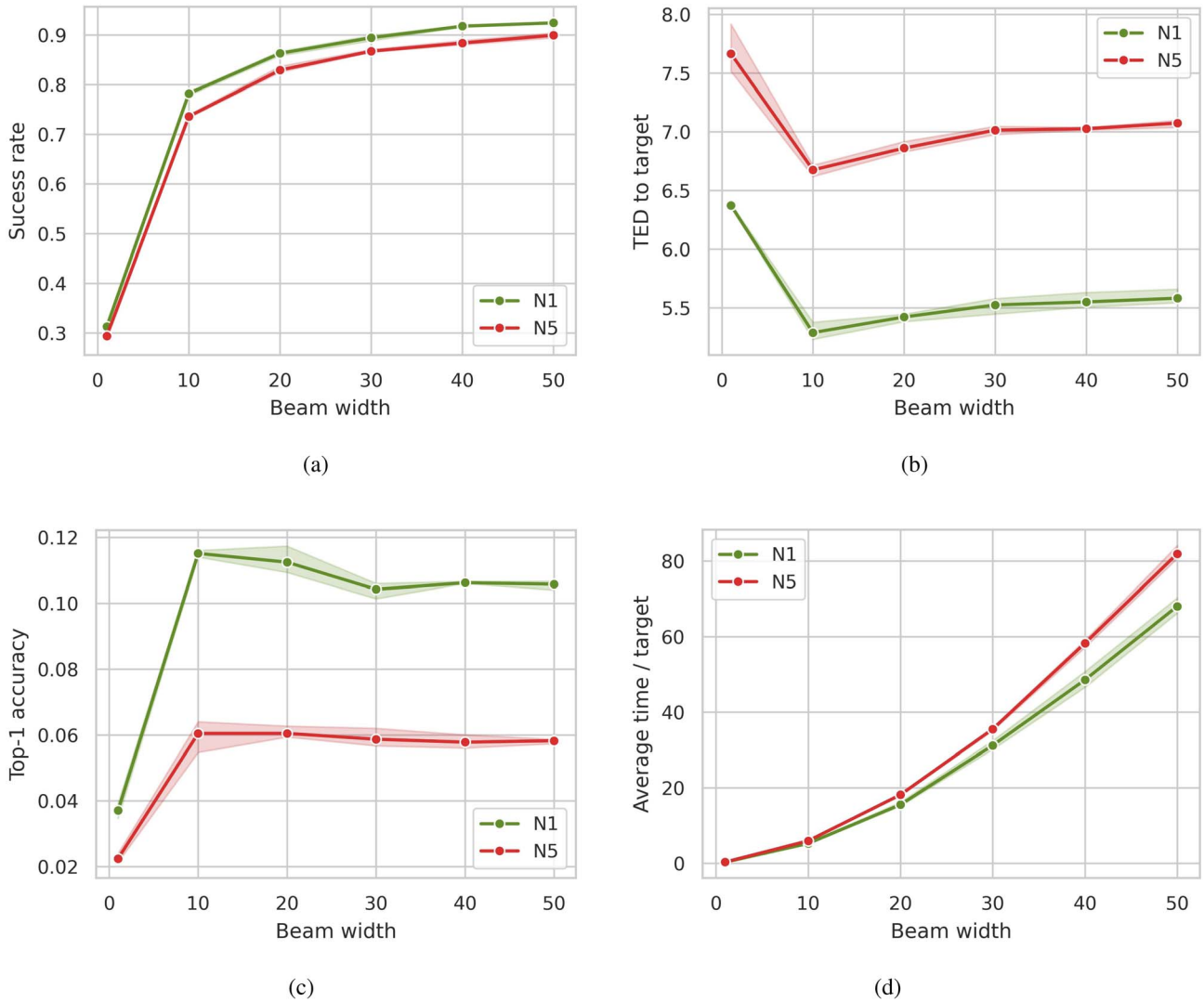


Fig. 7 Effect of increasing the beam width on RetroSynFormer performance on the N1 and N5 sets. (a) Success rate *versus* beam width. (b) Tree edit distance (TED) *versus* beam width. (c) Top-1 accuracy *versus* beam width. (d) Average search time per target *versus* beam width. Error bars indicate the standard deviation of three separate model predictions in all plots.

In general, both the action and class accuracies are lower for the RetroSynFormer than for AiZynthFinder, and, generally speaking, the accuracy is lower for the N5 set than for the N1 set.

We also looked at the solved routes separately and calculated the route accuracy for solved routes (Table 4). Naturally, we observe that this accuracy is slightly higher than when we also include unsolved routes (Table 2). If we disregard the order of the steps, we also observe a slight increase in accuracy (*e.g.*, +0.05 for route accuracy on the N1 set using the RetroSynFormer), indicating that although some of the predicted routes are overall correct, the exact order of the individual actions might differ compared to the target routes.

Finally, we estimated route quality by the recently proposed DeepSet route score[34] for both the predicted and target routes as an additional quality assessment. We observe that scores for the RetroSynFormer are marginally better (lower) than for AiZynthFinder on both the N1 and N5 datasets. However, as averages for predicted and patented reference routes in both datasets are <3.5, they can all be classified as "good". This quality assessment indicates that on average the routes can be used as-is to plan the wet-lab experiments and that they do not require modification by an experienced scientist.

### 4.4 Model exploration 1: beam search

The results presented above were achieved using a beam width of 50. A high beam width naturally results in a larger search space and thus a higher probability of success; however, for the same reason, it also increases the search time as we can see in Fig. 7c. Thus, there is a trade-off between optimizing success rate and search time. Using the trained model, we have evaluated the effect of the beam width on route predictions for the targets in the test set (Fig. 7). We can here clearly see that the success rate increases logarithmically with the beam width, as expected. However, when increasing the beam width, the search time increases exponentially and becomes a limiting factor for choosing the beam width. Interestingly, we don't see the

Table 5 Different reward functions and their impact on the model performance

| | Success rate[a] | | Top-1 accuracy[a] | |
|---|---|---|---|---|
| Reward function | N1 | N5 | N1 | N5 |
| Default | **0.924** | **0.899** | 0.106 | 0.058 |
| Increasing building block reward | 0.916 | 0.889 | 0.099 | 0.054 |
| Decreasing building block reward | 0.916 | 0.891 | 0.107 | 0.056 |
| Remove scaling building block reward | 0.912 | 0.890 | **0.110** | **0.064** |
| Flipping sign intermediate score | 0.914 | 0.890 | 0.106 | 0.054 |
| Remove scaling intermediate | 0.920 | 0.898 | 0.105 | 0.060 |
| Flipping sign dead end score | 0.921 | **0.899** | 0.107 | 0.058 |
| Remove scaling dead end | 0.915 | 0.898 | 0.103 | 0.059 |

[a] RetroSynFormer results show averages over three runs. Standard deviations for all RetroSynFormer models are ≤ 0.006 for success rate and ≤ 0.005 for top-1 accuracy.

expected increase in top-1 accuracy or decrease in TED with increasing beam width. Both of these metrics show an optimal beam width of 10 and they deteriorate slightly for high beam widths. This shows that there is not a clear correlation between the success rate and the top-1 accuracy, and highlight the importance of considering both metrics for retrosynthesis.

### 4.5 Model exploration 2: reward function

A hyperparameter search was performed as described in Section 3.3.3. Surprisingly, we observed that the model was not very sensitive to the choice of reward function. In Table 5 we demonstrate this by showing the success rate and top-1 accuracy for models trained using alternative reward functions. Here, we have changed the signs and removed the scaling with the depth parameter. Details for all the reward functions can be found in Table S3. The results show that the default reward used in the main results presented does indeed give the highest success rate. However, the difference between the other rewards is in general small and if the top-1 accuracy is considered instead, the best reward function is another one denoted as remove scaling building block reward. This is because the reward was optimized with regards to the success rate rather than the top-1 accuracy.

### 4.6 Model exploration 3: action space size

All results presented above have used the standard dataset which includes 1572 templates. We noted that the size of the action space scales with the number of reaction templates as the templates are encoded as one-hot vectors. To evaluate the scalability of the method with respect to the reaction templates and show that the approach is also useful for other action spaces that are of larger or smaller sizes, we have trained RetroSynFormer and AiZynthFinder with two other datasets, denoted as small and large (see Section 3.1 and Fig. 2). For the evaluation, we constructed different N1 and N5 test sets for the different action space sizes by randomly sampling 1500 routes from each N1 and N5. This is done to ensure that we evaluate the models on targets with reference routes consisting of the full template space. The results of this evaluation can be found in Table 6. The results indicate that the model is able to solve the majority of the targets for all action space sizes and that the success rate does not differ significantly depending on the template set. We can also observe that the top-1 accuracy decreases when the number of available templates increases. This is not surprising as when there are more disconnections to choose from, the probability of predicting the correct one decreases and it is likely that there are more reaction templates that are plausible and/or very similar. In addition, the additional templates which are added when scaling up the template space will be less frequently used and might be only used in very few routes, something which may also explain the drop in top-1 accuracy.

We further sub-sampled the targets in the N1 and N5 sets to select the targets that are common between the small, medium, and large datasets; there are 232 and 248 such targets for N1 and N5, respectively. The performances of RetroSynFormer and AiZynthFinder on those targets are shown in Table S4.

Table 6 Performance of RetroSynFormer compared to AiZynthFinder for the different datasets

| Dataset | Model | Test set | Success rate[a] | Top-1 accuracy[a] | TED[a] | Avg. Route length[a] |
|---|---|---|---|---|---|---|
| Small | RetroSynFormer50 | N1 | **0.950** | 0.182 | 4.426 | 2.291 |
| | | N5 | 0.833 | 0.101 | 5.428 | 2.472 |
| | AiZynthFinder | N1 | 0.923 | **0.223** | 4.065 | 2.343 |
| | | N5 | **0.917** | **0.125** | 7.121 | 2.923 |
| Standard | RetroSynFormer50 | N1 | 0.924 | 0.106 | 5.5836 | 2.337 |
| | | N5 | 0.899 | 0.058 | 7.075 | 2.548 |
| | AiZynthFinder | N1 | **0.939** | **0.143** | 5.398 | 2.517 |
| | | N5 | **0.924** | **0.071** | 7.120 | 2.923 |
| Large | RetroSynFormer50 | N1 | 0.929 | 0.082 | 6.168 | 2.276 |
| | | N5 | 0.887 | 0.045 | 7.54 | 2.531 |
| | AiZynthFinder | N1 | **0.939** | **0.115** | 6.112 | 2.538 |
| | | N5 | **0.925** | **0.073** | 7.713 | 3.019 |

[a] RetroSynFormer results show averages over three runs. Standard deviations of all RetroSynFormer models are ≤ 0.004 for success rate, ≤ 0.003 for top-1 accuracy, ≤ 0.09 for TED, and ≤ 0.01 for route length.

Interestingly, the largest drop in performance is observed for the models trained on the small dataset, where the success rate of RetroSynFormer drops by almost 10% compared to the entire small target set. The success rate of AiZynthFinder trained on the small dataset does not show the same drop in success rate, but the top-1 accuracy is noticeably lower.

# 5 Discussion

We have presented a novel approach, RetroSynFormer, for retrosynthesis prediction that uses the recently developed DT to generate synthesis routes conditioned on a target compound. The DT model is one of the few DL models that recast an RL problem as a sequence modeling problem. We have herein shown for the first time that it is possible to adapt the DT model for problems in the chemical domain. Nevertheless, RetroSynFormer shares many implementation details with established retrosynthesis methods: it uses a fixed set of templates to break down molecules into reactants—just as template-based single-step models—and it uses a fixed stock of building blocks to indicate the termination of retrosynthesis pathways—as do the majority of published multi-step retrosynthesis algorithms. In contrast, although RetroSynFormer predicts the next reaction autoregressively, it utilizes at each step the entirety of the previously predicted route and thus has the potential to optimize the sequence of reactions over a larger context window than is possible with existing algorithms.

We conducted a comprehensive benchmarking of RetroSynFormer against AiZynthFinder, a well-established, template-based method that represents a conventional retrosynthesis approach. In terms of success rate, *i.e.*, for how many targets the retrosynthesis produces a route that leads to commercial building blocks, the two approaches are comparable (see Table 2). AiZynthFinder outperforms RetroSynFormer by a few percentage points, but it is unclear if this difference is of practical importance. When comparing the predicted routes to the reference patent routes for each target, AiZynthFinder more often reproduces the patented routes—but if we look at the route similarity as computed by TED, RetroSynFormer is indistinguishable from AiZynthFinder on average. The two approaches also find routes of comparable length, and the average route score is similar. This indicates that RetroSynFormer produces routes that are different than the patented route but of similar quality to the routes produced by AiZynthFinder. Encouragingly, the predicted routes by either algorithm are of similar quality to the patented reference routes, and can in both cases be classified as "good" according to a recently established route scoring method.[34] Furthermore, we have shown that the two approaches to retrosynthesis are complementary, and together they can predict routes to the commercial starting material for >97% of the targets investigated (see Table 3). Such a result has been observed for other retrosynthesis models with AiZynthFinder before[6] and point to a real use-case for RetroSynFormer where the combination of different algorithms has a higher chance of producing valuable results and can be used in a staged fashion.

The exact reproducibility of the reference patented route is also not a necessity because of the redundancy in the template set, *i.e.*, different templates could translate into identical or near identical disconnections. We have shown that both RetroSynFormer and AiZynthFinder show a greater discrepancy from the patented routes if one looks at the exact predicted template rather than the reaction classes represented by the templates. If we instead evaluate route quality based on class accuracy or disregard the order of the reactions in the route, we generally observe greater agreement with the patented routes (Table 4). This complexity of calculating route similarity was recently discussed in Genheden *et al.*.[29] Furthermore, we have again shown that only a fraction of the templates are practically needed to find synthesis routes, *e.g.*, RetroSynFormer uses only 669 out of the 1572 available templates (standard dataset) to find synthesis routes for the N1 targets. This has been shown previously for AiZynthFinder[38] and here we have shown it again for RetroSynFormer. Hence, it is clear that we either (1) should evaluate retrosynthesis planning on a different target set where more templates are needed, or (2) need better retrosynthesis models that can better employ rarely used templates.

Designing a novel algorithm is not straightforward, and herein we have highlighted a few explorations on the model design. First, beam search was essential for finding routes to commercial materials; in Fig. 7 we show that with a greedy algorithm we only reach about 30% success rate. Unfortunately, the scaling factor of the beam search is considerable, especially compared to a search algorithm such as the one in AiZynthFinder where additional iterations come at basically a constant cost. However, the success rate increases more steeply with increased beam width in the RetroSynFormer than with additional iterations in AiZynthFinder.[4] This indicates that the effort of increased beam width could make a practical difference to the produced synthesis routes, compared to AiZynthFinder where it takes many additional iterations to find additional solutions. We acknowledge that there are still a lot of engineering improvements possible to increase the efficiency of RetroSynFormer, which is currently much slower than AiZynthFinder as a result of using a high beam width. We would like to argue that currently the reported times in Table 2 are not a fair comparison as the performance of AiZynthFinder has been optimized over the course of >5 years.

Furthermore, a key challenge in developing RetroSynFormer has been the design of the reward function. As presented in Section 4.5, we observed that changing the reward function in what seems to be a suboptimal way does not have a significant negative impact on the results, which may seem counter-intuitive. First of all, it is not straightforward to evaluate synthesis routes in a step-by-step fashion as we do here as it is not evident how good a single reaction is until we have the full route. We thus believe that further investigation of different reward set-ups could potentially be beneficial and it is possible that one could design a better reward function than the one used here. One possibility, for instance, could involve using a machine learning model to estimate the future reward of a partially constructed route.[14] As a final exploration, we increased and decreased the action space and found that the

performance of the RetroSynFormer compared to AiZynthFinder does not change noticeably in these experiments. It will be up to future work to investigate if the model scales well to the sizes available in proprietary datasets.[24] Considering the low fraction of templates used in practice, one could nevertheless argue that having the ability to scale to larger template spaces is not a requirement for a useful retrosynthesis algorithm.

Herein, we have specifically evaluated the RetroSynFormer on targets from the PaRoutes dataset because we are interested in developing a novel algorithm and benchmarking it. Rather than focusing on improving the state-of-the-art success rate by a few percentage points, we have explored here a completely novel architecture in this domain—the Decision Transformer—and evaluated how well it works for retrosynthesis tasks and what its limitations are. The PaRoutes dataset is ideal in this scenario as it is robust, has been used in several previous publications, and also provides reference patented routes. However, additional experiments using the full PaRoutes dataset or alternatives would be an interesting complement to our assessment of the generalizability. In this direction, we have presented in SI D results of the out-of-distribution evaluation of the method. Further investigation through a more comprehensive evaluation of the RetroSynFormer's generalizability on other target datasets, *e.g.*, ChEMBL,[39] GDB,[40] or compound ideas from internal drug discovery projects, could be interesting to explore in future work to further explore the generalizability of the model. Since the conventional AiZynthFinder is already successfully deployed in drug discovery projects,[10] it is worthwhile to identify areas where a different algorithm like RetroSynFormer could bring additional value. We have started in this work to identify that RetroSynFormer is in many ways complementary to AiZynthFinder, although further investigations are required. It would be especially interesting to show the advantage of the memory inheritance feature of the Decision Transformer, where we may base the prediction of the next action on all previously predicted actions (beyond a single route). This will require the design of a special target set as it is likely that this feature may only be important for certain classes of compounds.

## 6 Conclusion

We have presented RetroSynFormer, a novel approach to retrosynthesis that models the task as a sequence modeling problem. We have demonstrated that the model can find valid synthesis routes for 92% of the N1 targets and 90% of the N5 targets from the PaRoutes suite of retrosynthesis benchmarks, and that—using it complementarily with AiZynthFinder—up to >97% of targets can be solved. Our method is unique in its approach as it is the first time a DT has been used in the chemical domain. The RetroSynFormer formulates retrosynthesis prediction as sequence modeling and conditions its predictions on previous reactions, targets, and rewards, thus suggesting disconnections using greater global awareness than more conventional approaches. With further development and benchmarking, RetroSynFormer has the potential to become a valuable tool for computer-aided synthesis planning.

## Author contributions

EG: conceptualization, methodology, software, formal analysis, investigation, data curation, visualization, writing – original draft. SG: conceptualization, data curation, supervision, methodology, funding acquisition, resources, writing – review & editing. RM: conceptualization, supervision, methodology, funding acquisition, visualization, resources, writing – review & editing. All authors discussed the results, contributed to interpretation and refinement of the study, and approved the final manuscript.

## Conflicts of interest

EG and SG are employees of AstraZeneca. The authors declare no competing interests.

## Data availability

The code used to produce these results is available here: https://doi.org/10.5281/zenodo.17588966. The data are available at https://doi.org/10.5281/zenodo.17177425, where the templates, building blocks and routes can be found. The routes are a subset from the PaRoutes datasets available at https://doi.org/10.5281/zenodo.6275421.

Supplementary information is available. See DOI: https://doi.org/10.1039/d5dd00153f.

## Acknowledgements

EG and RM acknowledge the funding provided by the Wallenberg AI, Autonomous Systems, and Software Program (WASP), supported by the Knut and Alice Wallenberg Foundation. The computations and data storage were partially enabled by resources provided by Chalmers e-Commons. The computations and data storage were partially enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS), partially funded by the Swedish Research Council through grant agreement no. 2022-06725. We thank Mike Preuss for fruitful discussions.

## References

1 K. Nicolaou, Organic synthesis: the art and science of replicating the molecules of living nature and creating others like them in the laboratory, *Proc. R. Soc. A*, 2014, **470**(2163), 20130690.
2 G. E. Vleduts, Concerning one system of classification and codification of organic reactions, *Inf. Storage Retr.*, 1963, **1**, 117–146.
3 E. J. Corey and W. T. Wipke, Computer-assisted design of complex organic syntheses: Pathways for molecular synthesis can be devised with a computer and equipment for graphical communication, *Science*, 1969, **166**(3902), 178–192.
4 S. Genheden and E. Bjerrum, PaRoutes: towards a framework for benchmarking retrosynthesis route predictions, *Digital Discovery*, 2022, **1**(4), 527–539.

5 K. Maziarz, G. Liu, H. Misztela, A. Kornev, P. Gaiński, H. Hoefling, M. Fortunato, R. Gupta, and M. Segler: Chimera: Accurate retrosynthesis prediction by ensembling models with diverse inductive biases, *arXiv*, 2024, preprint, arXiv:2412.05269v1, DOI: 10.48550/arXiv.2412.05269v1.

6 A. M. Westerlund, S. Manohar Koki, S. Kancharla, A. Tibo, L. Saigiridharan, M. Kabeshov, R. Mercado and S. Genheden, Do Chemformers dream of organic matter? evaluating a transformer model for multistep retrosynthesis, *J. Chem. Inf. Model.*, 2024, **64**(8), 3021–3033.

7 M. Cretu, C. Harris, J. Roy, E. Bengio, and P. Liò, SynFlowNet: Towards molecule design with guaranteed synthesis pathways, in *ICLR 2024 Workshop on Generative and Experimental Perspectives for Biomolecular Design*, 2024.

8 S. M. Kearnes, M. R. Maser, M. Wleklinski, A. Kast, A. G. Doyle, S. D. Dreher, J. M. Hawkins, K. F. Jensen and C. W. Coley, The open reaction database, *J. Am. Chem. Soc.*, 2021, **143**(45), 18820–18826.

9 L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas and I. Mordatch, Decision transformer: Reinforcement learning *via* sequence modeling, *Adv. Neural Inf. Process. Syst.*, 2021, **34**, 15084–15097.

10 J. D. Shields, R. Howells, G. M. Lamont, L. Yin, A. Madin, C. Reimann, H. Rezaei, T. Reuillon, B. Smith, C. Thomson, Y. Zheng and R. E. Ziegler, AiZynth impact on medicinal chemistry practice at AstraZeneca, *RSC Med. Chem.*, 2024, **15**(4), 1085–1095.

11 Z. Zhong, J. Song, Z. Feng, T. Liu, L. Jia, S. Yao, T. Hou and M. Song, Recent advances in deep learning for retrosynthesis, *Wiley Interdiscip. Rev.: Comput. Mol. Sci.*, 2024, **14**(1), e1694.

12 S. Genheden, A. Thakkar, V. Chadimová, J.-L. Reymond, O. Engkvist and E. Bjerrum, AiZynthFinder: a fast, robust and flexible open-source software for retrosynthetic planning, *J. Cheminf.*, 2020, **12**(1), 70.

13 M. H. Segler, M. Preuss and M. P. Waller, Planning chemical syntheses with deep neural networks and symbolic AI, *Nature*, 2018, **555**(7698), 604–610.

14 B. Chen, C. Li, H. Dai, and L. Song, Retro*: learning retrosynthetic planning with neural guided A* search, in *International Conference on Machine Learning*, PMLR, 2020, pp. 1608–1616.

15 S. Xie, R. Yan, P. Han, Y. Xia, L. Wu, C. Guo, B. Yang, and T. Qin, RetroGraph: Retrosynthetic planning with graph search, in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022, pp. 2120–2129.

16 J. S. Schreck, C. W. Coley and K. J. Bishop, Learning retrosynthetic planning through simulated experience, *ACS Cent. Sci.*, 2019, **5**(6), 970–981.

17 S. Liu, Z. Ying, Z. Zhang, P. Zhao, J. Tang, L. Lin, and D. Wu, *Metro: Memory-enhanced transformer for retrosynthetic planning via reaction tree*, 2022.

18 R. Irwin, S. Dimitriadis, J. He and E. J. Bjerrum, Chemformer: a pre-trained transformer for computational chemistry, *Mach. Learn. Sci. Technol.*, 2022, **3**(1), 015022.

19 Y. Shee, H. Li, A. Morgunov, and V. Batista, DirectMultiStep: Direct route generation for multi-step retrosynthesis, *arXiv*, 2024, preprint, arXiv:2405.13983, DOI: 10.48550/arXiv.2405.13983.

20 L. Lehnert, S. Sukhbaatar, D. Su, Q. Zheng, P. Mcvay, M. Rabbat, and Y. Tian, Beyond A*: Better planning with transformers *via* search dynamics bootstrapping, *arXiv*, 2024, preprint, arXiv:2402.14083, DOI: 10.48550/arXiv.2402.14083.

21 P. Bhargava, R. Chitnis, A. Geramifard, S. Sodhani, and A. Zhang, When should we prefer decision transformers for offline reinforcement learning?, *arXiv*, 2023, preprint, arXiv:2305.14550, DOI: 10.48550/arXiv.2305.14550.

22 D. Lowe, *Chemical reactions from US patents (1976–sep2016)*, figshare, 2017.

23 C. W. Coley, W. H. Green and K. F. Jensen, RDChiral: An RDKit wrapper for handling stereochemistry in retrosynthetic template extraction and application, *J. Chem. Inf. Model.*, 2019, **59**(6), 2529–2537.

24 S. Genheden, P.-O. Norrby and O. Engkvist, AiZynthTrain: robust, reproducible, and extensible pipelines for training synthesis prediction models, *J. Chem. Inf. Model.*, 2023, **63**(7), 1841–1846.

25 E. Heid, J. Liu, A. Aude and W. H. Green, Influence of template size, canonicalization, and exclusivity for retrosynthesis and reaction prediction applications, *J. Chem. Inf. Model.*, 2021, **62**(1), 16–26.

26 T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, Transformers: State-of-the-art natural language processing, in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Association for Computational Linguistics, 2020, pp. 38–45.

27 D. Rogers and M. Hahn, Extended-connectivity fingerprints, *J. Chem. Inf. Model.*, 2010, **50**(5), 742–754.

28 G. Landrum, *RDKit: Open-source cheminformatics*, https://www.rdkit.org.

29 S. Genheden and J. D. Shields, A simple similarity metric for comparing synthetic routes, *Digital Discovery*, 2025, **4**(1), 46–53.

30 J. Li, L. Fang and J.-G. Lou, Retro-BLEU: quantifying chemical plausibility of retrosynthesis routes through reaction template sequence analysis, *Digital Discovery*, 2024, **3**(3), 482–490.

31 T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, Optuna: A next-generation hyperparameter optimization framework, in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 2623–2631.

32 E. Cohen and C. Beck, Empirical analysis of beam search performance degradation in neural sequence models, in *International Conference on Machine Learning*, PMLR, 2019, pp. 1290–1299.

33 S. Genheden, O. Engkvist and E. Bjerrum, Clustering of synthetic routes using tree edit distance, *J. Chem. Inf. Model.*, 2021, **61**(8), 3899–3907.
34 G. Yujia, M. Kabeshov, T. H. D. Le, S. Genheden, G. Bergonzini, O. Engkvist and S. Kaski, An expert-augmented deep learning approach for synthesis route evaluation, *ChemRxiv*, 2025, preprint, DOI: 10.26434/chemrxiv-2024-tp7rh-v2.
35 C. Kannas and S. Genheden, *Rxnutils–a cheminformatics python library for manipulating chemical reaction data*, 2022.
36 https://www.nextmovesoftware.com/pistachio.html, accessed September 2025.
37 S. G. Christoph Bauer, T. Kogej and P.-O. Norrby, Precedent finder – locating pareto-optimal reactions, *ChemRxiv*, 2025, preprint, DOI: 10.26434/chemrxiv-2025-2ld37.
38 L. Saigiridharan, A. K. Hassen, H. Lai, P. Torren-Peraire, O. Engkvist and S. Genheden, AiZynthFinder 4.0: developments based on learnings from 3 years of industrial application, *J. Cheminf.*, 2024, **16**(57).
39 A. Gaulton, L. J. Bellis, A. P. Bento, J. Chambers, M. Davies, A. Hersey, Y. Light, S. McGlinchey, D. Michalovich, B. Al-Lazikani and J. P. Overington, ChEMBL: a large-scale bioactivity database for drug discovery, *Nucleic Acids Res.*, 2011, **40**, 1100–1107.
40 S. Bühlmann and J. Reymond, ChEMBL-likeness score and database GDBChEMBL, *Front. Chem.*, 2020, **8**, 46.