RetroReasoner: A Reasoning LLM for Strategic Retrosynthesis Prediction

Hanbum Ko[1], Chanhui Lee[1], Ye Rin Kim[1], Rodrigo Hormazabal[3], Sehui Han[3], Sungbin Lim[2], Sungwoong Kim[1],

[1]Department of Artificial Intelligence, Korea University,
[2]Department of Statistics, Korea University,
[3]Materials Intelligence Lab, LG AI Research,

Correspondence: Sungbin Lim sungbin@korea.ac.kr, Sungwoong Kim swkim01@korea.ac.kr

Abstract

Retrosynthesis prediction aims to identify reactants that can synthesize a given product molecule. Although molecular large language models (LLMs) have recently shown promising results, most existing methods either generate reactants directly or provide only generic product-level analysis, without explicitly reasoning about bond-disconnection strategies that justify specific reactant choices. This paper proposes RetroReasoner, a retrosynthetic reasoning model that captures chemists' strategic disconnection-based thinking. RetroReasoner is trained with supervised fine-tuning and reinforcement learning. For supervised fine-tuning, SyntheticRetro generates structured disconnection rationales paired with reactant predictions. For reinforcement learning, a round-trip reward evaluates predicted reactants by passing them through a forward synthesis model and rewarding predictions that reconstruct the original product. RetroReasoner can also be applied to multi-step retrosynthetic planning by incorporating it into a parallelized Monte Carlo tree search framework, reducing search time while increasing the number and diversity of valid synthetic pathways. Experimental results show that RetroReasoner outperforms prior baselines, including not only molecular LLMs but also retrosynthesis-specific expert models, and generates a broader range of feasible reactant proposals, especially for challenging reaction instances. The code is available at https://github.com/KU-AGI/RetroReasoner.

1 Introduction

Retrosynthesis prediction is an important task in organic synthesis that aims to identify reactants for a desired target molecule. A representative chemist strategy is to infer likely bond formations, disconnect the corresponding bonds into hypothetical reactant fragments called synthons (Corey, 1967, 1988; Corey and Cheng, 1995), and then assign standalone synthetic equivalents to each synthon. Although effective, this process requires expert knowledge and substantial validation time.

Recent advances in large language models (LLMs) (Yu et al., 2024; Fang et al., 2023; Pei et al., 2024; Lee et al., 2025) have led to molecular LLMs that process molecular representations such as SMILES (Weininger, 1988) and SELFIES (Krenn et al., 2020), enabling their application to reactant prediction. However, most molecular LLMs either predict reactants without explicit reasoning or provide reasoning that remains a generic product-level analysis, such as functional group enumeration and reaction mechanism explanation (Wang et al., 2025; Zhao et al., 2024, 2025; Narayanan et al., 2025; Zhang et al., 2025; Li et al., 2025). Such reasoning omits intermediate disconnection steps that are essential for retrosynthesis, causing a logical gap between product analysis and the selection of specific reactants. An illustrative example is shown in Figure 1.

To address this limitation, RetroReasoner reasons through four stages: product analysis, key-substructure identification, strategic bond disconnection, and synthetic-equivalent mapping. It is trained by supervised fine-tuning (SFT) on rationales generated by SyntheticRetro, followed by Group Relative Policy Optimization (GRPO) (Shao et al., 2024) with a round-trip reward (Schwaller et al., 2020). SyntheticRetro uses reaction-derived supporting information to generate accurate intermediate reasoning between the structured stages. By checking whether predicted reactants regenerate the product through a forward synthesis model, the round-trip reward supports accurate, round-trip-consistent alternatives beyond the single reference.

Experimental results show that RetroReasoner outperforms models without strategic reasoning, improving reactant prediction accuracy and round-trip consistency. It also achieves strong performance on challenging evaluation datasets, including those focused on rare reaction types and those

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 49565–49600
October 24-29, 2026 ©2026 Association for Computational Linguistics

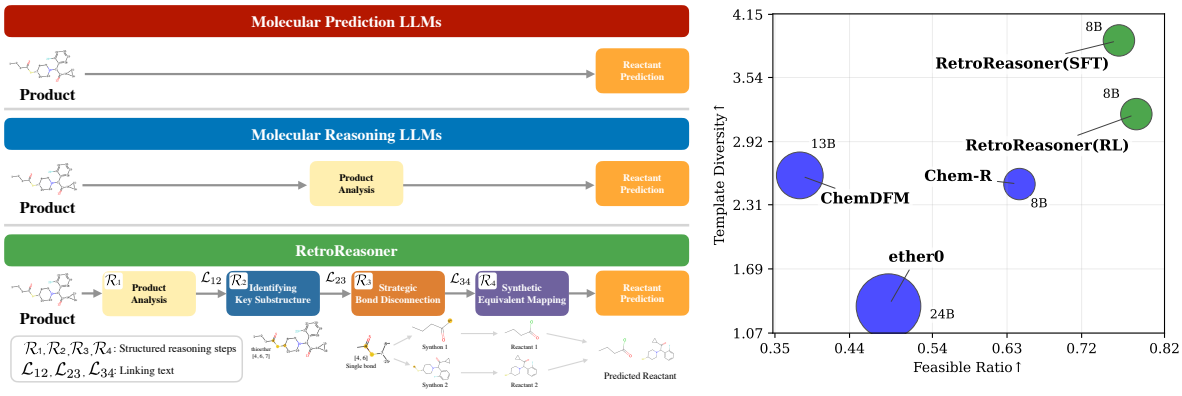


Figure 1: (Left) Comparison of reasoning processes among Molecular Reasoning LLMs, Molecular Prediction LLMs, and RetroReasoner(Ours). RetroReasoner suggest valid reactant given product by explicit strategic disconnection steps. (Right) Comparison of molecular reasoning LLMs. The x-axis represents the feasible ratio of reactant proposals, and the y-axis represents the diversity of proposals. Model sizes are distinguished by the size of the circle.

dominated by rare atoms or rare n-gram tokens. Furthermore, RetroReasoner is applied to multi-step retrosynthetic planning through a parallelized Monte Carlo tree search (MCTS) framework. This addresses a practical bottleneck of LLM-based single-step models, whose slow proposal speed can restrict repeated tree expansion. Through this, RetroReasoner is used efficiently in multi-step planning in terms of runtime and diversity of synthetic paths.

In summary, the main contributions are as follows:

- RetroReasoner, chemist-inspired stepwise reasoning model leveraging strategic bond disconnection, trained with SFT and further optimized with RL using a round-trip reward, improving prediction accuracy and round-trip consistency while achieving a favorable feasibility-diversity trade-off.
- SyntheticRetro, a data generation framework that produces the stepwise reasoning text by using various supporting information to generate accurate intermediate reasoning.
- Empirical validation across challenging single-step retrosynthesis benchmarks and multi-step planning experiments, where simple parallelized Monte Carlo tree search reduces solving time while increasing synthesis routes diversity.

A detailed discussion of related work is provided in Appendix A.

2 RetroReasoner: A Reasoning LLM for Strategic Retrosynthesis Prediction

Retrosynthesis prediction aims to infer reactants that can synthesize a given target product, where molecules and reactions are represented as SMILES (Weininger, 1988) or reaction (RXN) SMILES (Daylight, 2001). Chemists approach this task through backward reasoning: identifying plausible bond disconnections, deriving hypothetical fragments called synthons, and assigning practically accessible synthetic equivalents. Since multiple disconnections and synthon-to-equivalent mappings can be valid, a single product may correspond to multiple feasible reactant sets. Multi-step retrosynthetic planning extends this process into a search problem by repeatedly applying single-step retrosynthesis to intermediate molecules until the terminal precursors are available building blocks. Thus, single-step retrosynthesis acts as the expansion operation for multi-step planning.

2.1 Strategic Reasoning Design

The reasoning process of RetroReasoner consists of four structured reasoning steps as shown in Figure 1: product analysis $\mathcal{R}_1$, identifying key substructures for bond formation $\mathcal{R}_2$, strategic bond disconnection $\mathcal{R}_3$, and mapping synthons to synthetic equivalents $\mathcal{R}_4$. These steps are linked by natural-language linking texts $(\mathcal{L}_{12}, \mathcal{L}_{23}, \mathcal{L}_{34})$, which provide logical connections between the structured reasoning steps. In $\mathcal{R}_1$, basic product information is listed that is used throughout the subsequent steps and content. Specifically, it pro-

vides (1) atom-mapped SMILES (atoms indexed from 1 to n), (2) functional group information for the product (name, corresponding SMILES, and corresponding positions), and (3) statistics for the product SMILES string itself (number of rings, number of carbons, and the number of characters representing stereochemistry). The linking text $\mathcal{L}_{12}$ transitions from this initial product information to the next step by inferring a reaction mechanism and the substructure formed during the reaction. In the next step, $\mathcal{R}_2$, a candidate substructure is identified, narrowing the set of candidates for strategic bond disconnection. The linking text $\mathcal{L}_{23}$ infers which bond from the candidate set should be selected. In $\mathcal{R}_3$, a bond is selected from the candidates, and the corresponding synthons are generated. The linking text $\mathcal{L}_{34}$ explains where the synthon can typically be obtained. Finally, in $\mathcal{R}_4$, each synthon is mapped to its synthetic equivalent. Each of these four structured steps is encapsulated in the <PRODUCT_INFO>, <CANDIDATE_STRUCTURE>, <STRATEGIC_BOND_DISCONNECTION>, and <SYNTHETIC_EQUIVALENT> tags, as shown in Figure 6 of Appendix B.

2.2 Supervised Fine-tuning with Strategic Reasoning

2.2.1 SyntheticRetro: Synthetic Retrosynthesis Rationale Generator

SyntheticRetro generates a single-step retrosynthesis prediction rationale from the RXN SMILES. SyntheticRetro first extracts three types of supporting information: direct-usable information, model-predicted information, and rule-derived information. Direct-usable information is information that can be obtained directly from the RXN SMILES. Model-predicted information is information obtained from deep learning-based models. Rule-derived information is information obtained through rule-based post-processing. Using these three types of supporting information, SyntheticRetro leverages a general-purpose LLM (GPT-oss-20B) (Agarwal et al., 2025) to restructure them into a reasoning text. An illustrative example is shown in Figure 2 and details of SyntheticRetro are described in Appendix B with an example.

Synthetic Generation SyntheticRetro leverages GPT-oss-20B to generate the linking texts $\mathcal{L}_{12}, \mathcal{L}_{23}, \mathcal{L}_{34}$. SyntheticRetro provides the general LLM with all supporting information and two consecutive structured reasoning steps $\mathcal{R}_i, \mathcal{R}_{i+1}$, instructs it to connect the content between the two steps, and then inserts the contents between the two steps. First, it fills in the contents between $\mathcal{R}_1$ and $\mathcal{R}_2$, then it provides all contents up to $\mathcal{R}_2$ and $\mathcal{R}_3$ to the general LLM, instructs it to generate the contents connecting $\mathcal{R}_2$ and $\mathcal{R}_3$, and inserts the contents between $\mathcal{R}_2$ and $\mathcal{R}_3$. Finally, it uses the general LLM to fill in all contents up to $\mathcal{R}_3$ and the contents between $\mathcal{R}_3$ and $\mathcal{R}_4$, completing the generation. This generation process is repeated $n = 15$ times for one reaction instance to obtain various paths to the next structured reasoning steps. The generated data statistics are summarized in Table 9, and the generation prompt is described in Appendix H.1.

2.2.2 Training Objective

Let π_θ denote an LLM policy, $\mathbf{x}$ a product, and $\mathbf{y} = (\mathbf{y}^{\text{reasoning}}, \mathbf{y}^{\text{reactant}}) = (y_1, y_2, \ldots, y_T)$ the corresponding ground-truth token sequence, which consists of reasoning text tokens $\mathbf{y}^{\text{reasoning}}$ and reactant SMILES tokens $\mathbf{y}^{\text{reactant}}$. SFT minimizes the standard cross-entropy loss, defined as

$$\mathcal{L}_{\text{SFT}}(\theta) = -\frac{1}{T}\sum_{t=1}^{T} \log \pi_\theta(y_t|\mathbf{x}, y_{<t}). \quad (1)$$

2.3 Reinforcement Learning with Round-trip Accuracy

The training adopts the reinforcement learning with verifiable rewards (RLVR) paradigm, in which an automatic verifier assigns rewards to policy-generated outputs that serve as the learning signal for policy updates. For each training instance, a group of G output sequences $\{\mathbf{y}_i\}_{i=1}^{G}$ is sampled from a behavior policy $\pi_{\theta_{\text{old}}}$, where $\mathbf{y}_i = (y_{(i,1)}, y_{(i,2)}, \ldots)$ denotes the i-th sampled sequence. Each sequence is assigned a scalar reward R_i, and token-level advantages are computed by normalizing rewards across the group: $\hat{A}_{i,t} = \frac{R_i - \text{mean}(\{R_i\}_{i=1}^{G})}{\text{std}(\{R_i\}_{i=1}^{G})}$, shared across all token positions t within the same output sequence. The RL objective is defined as

$$\mathcal{J}_{\text{RL}}(\theta) = \mathbb{E}_{\{\mathbf{y}_i\}\sim\pi_{\theta_{\text{old}}}}\left[\frac{1}{G}\sum_{i=1}^{G}\sum_{t=1}^{|\mathbf{y}_i|}\ell_{i,t}(\theta)\right], \quad (2)$$

where the per-token loss follows a PPO (Schulman et al., 2017)-style clipped objective $\ell_{i,t}(\theta) = \min\big(r_{i,t}(\theta)\hat{A}_{i,t}, \text{clip}(r_{i,t}(\theta), 1 - \epsilon_{\text{low}}, 1 + \epsilon_{\text{high}})\hat{A}_{i,t}\big)$, with probability ratio $r_{i,t}(\theta) = \frac{\pi_\theta(y_{i,t}|q, y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t}|q, y_{i,<t})}$ and q representing a user query. In addition to the clipped term in Equa-

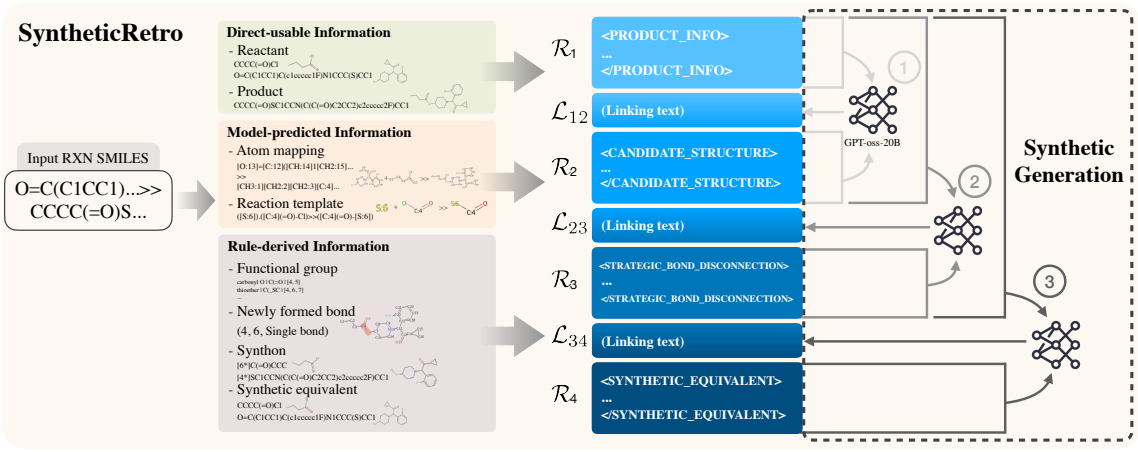


Figure 2: A schematic diagram of the generation process of SyntheticRetro, a chemist's strategy-based reasoning data generation process.

tion (2), training applies a KL-divergence penalty toward the reference policy and an entropy bonus to maintain exploration. The KL-loss and entropy coefficients are both set to 0.05, as reported in Table 11. The objective in Equation (2) is optimized using round-trip accuracy as the reward. Specifically, the policy generates a sequence of reactant SMILES tokens, $\hat{\mathbf{y}}^{\text{reactant}}$. A separately trained round-trip model f_ϕ then takes $\hat{\mathbf{y}}^{\text{reactant}}$ as input and predicts the corresponding product SMILES. To account for equivalent SMILES representations, molecular identity is determined after converting SMILES to standard InChI (Heller et al., 2013). Let $h(\cdot)$ denote the standard InChI conversion and define $\mathbb{I}(a, b) = \mathbf{1}[h(a) = h(b)]$. The round-trip reward is computed as

$$R^{\text{round-trip}} = \mathbb{I}\Big(\mathbf{x}, f_\phi(\hat{\mathbf{y}}^{\text{reactant}})\Big). \tag{3}$$

Training details, hyperparameters, and the cached round-trip verifier are described in Appendix C; details of the forward model f_ϕ are provided in Appendix D.

2.4 Parallelized Multi-step Retrosynthesis Prediction

Given a target molecule m_{target} and a set of commercially available stock molecules $\mathcal{B}$, multi-step retrosynthesis prediction aims to find a synthesis route whose leaf molecules all belong to $\mathcal{B}$. The search is conducted over an AND-OR tree, where OR nodes denote molecules and AND nodes denote reactions, and is performed via MCTS consisting of selection, expansion, evaluation, and backpropagation stages. During selection, the tree is traversed from the root using a UCB score that balances exploitation and exploration over reaction nodes, and the expansion stage queries an LLM-based single-step retrosynthesis model π_θ at the leaf molecule to generate candidate reactant sets. However, LLM-based single-step retrosynthesis models are computationally expensive in MCTS-based multi-step planning, since the proposal model π_θ must be repeatedly queried during tree expansion. To mitigate this expansion bottleneck, multiple expansion requests are generated in parallel and served as batched LLM queries, leveraging inference optimizations such as continuous batching (Yu et al., 2022), paged key-value caching (Kwon et al., 2023), and optimized attention kernels (Dao et al., 2022). Parallel expansion follows the core idea of WU-UCT (Liu et al., 2018), in which each node maintains an unobserved count O representing the number of in-flight simulations assigned to that node. The standard UCB score is augmented with these unobserved counts as

$$\text{UCB}(r)_{\text{WU}} = \frac{V_r}{N_r} + c \cdot p_r \cdot \frac{\sqrt{N_{\text{parent}} + O_{\text{parent}}}}{1 + N_r + O_r}, \tag{4}$$

where V_r denotes the cumulative value backed up through reaction node r, N_r its visit count, N_{parent} the visit count of the parent molecule node, p_r the reaction prior, and c the exploration constant. The term O_r lowers the exploration bonus of branches already being expanded, encouraging different workers to explore different tree regions. As a result, parallel MCTS reduces redundant LLM calls, better utilizes batched inference, and improves route diversity. See Appendix E for details.

3 Experiments

3.1 Baseline Models

RetroReasoner is evaluated against four categories of baselines: *Molecular Prediction LLMs*, *Molecular Reasoning LLMs*, *General Purpose LLMs*, and *Expert Models*. *Molecular Prediction LLMs* directly generate reactants without explicit reasoning and include LlaSMol (Yu et al., 2024), Mol-Instructions (Fang et al., 2023), BioT5+ (Pei et al., 2024), PRESTO (Cao et al., 2024), and Mol-LLM (Lee et al., 2025). *Molecular Reasoning LLMs* incorporate reasoning into retrosynthesis prediction, including Chem-R (Wang et al., 2025), ChemDFM (Zhao et al., 2024), and ether0 (Narayanan et al., 2025). *General Purpose LLMs* are prompted foundation models, including OpenAI-o3 (OpenAI, 2025), GPT-5-mini (Singh et al., 2025), GPT-oss-120B (Agarwal et al., 2025), Qwen3-8B, and Qwen3-235B-A22B (Yang et al., 2025). For OpenAI-o3, only greedy metrics are reported due to the high cost of sampling 100 outputs per instance (>500$), and GPT-5-mini results are used for sampling-based comparison (Singh et al., 2025). *Expert Models* are non-LLM retrosynthesis models, including RetroSynFlow (Yadav et al., 2026) and G2S-HCVAE (Wang et al., 2026), retrained on the same training data as RetroReasoner. A Prediction-Only variant is also included as a controlled baseline; it shares the same architecture, data, and optimization procedure as RetroReasoner but removes explicit reasoning generation. Prompts for *General Purpose LLMs* are provided in Appendix H. Training details for the expert models are provided in Appendix F.

For multi-step retrosynthesis, RetroInText (Kang et al., 2025) is additionally included as a reproducible planning baseline, using MolT5 (Edwards et al., 2022) for single-step retrosynthesis and MCTS with a fusion-based value function. Evaluation is conducted on the same fixed PaRoutes (Genheden and Bjerrum, 2022) set-n1 and set-n5 target subsets used for RetroReasoner. Retro-R1 (Liu et al., 2026) is excluded because its trained search-policy weights are not publicly available, preventing faithful reproduction of the complete method.

3.2 Evaluation Metrics

Single-step Retrosynthesis Single-step retrosynthesis is evaluated under both greedy decoding and sampling. Greedy metrics assess the top-1 proposal, while sampling metrics evaluate whether valid and diverse solutions can be found among 100 candidates. Exact@1↑ and Exact@100↑ measure matches to the labeled reactants based on the standard InChI identity function $\mathbb{I}$ defined in Equation (3), for the top-1 prediction and among 100 samples, respectively. Round-trip@1↑ and Round-trip@100↑ measure whether the predicted reactants are mapped back to the input product by a forward reaction model f_ϕ. Feasible Ratio↑ is a forward-model-based proxy for reaction feasibility, measuring the fraction of sampled reactants that reconstruct the target product under the forward verifier. Template Diversity↑ counts the number of distinct canonical reaction templates among feasible candidates, reflecting the diversity of valid disconnection patterns. All metrics are averaged over evaluation instances, with detailed calculations provided in Appendix F.1.

Multi-step Retrosynthesis Multi-step retrosynthesis is evaluated using three PaRoutes metrics (Genheden and Bjerrum, 2022). Solved Targets↑ counts targets for which at least one fully purchasable route is found. Routes Extracted↑ counts all complete routes extracted from the search trees. Number of Clusters↑ measures route diversity by clustering extracted routes based on pairwise route distances and summing the number of distinct clusters. Further details are provided in Appendix F.1.

3.3 Evaluation Dataset

We adopt ORDerly (Wigh et al., 2024), a curated collection of high-quality reaction instances, as the base dataset. To mitigate evaluation bias arising from the multi-label nature of retrosynthesis, instances in which a single product corresponds to multiple distinct valid reactant sets are excluded from the validation and test sets. Instances used to train the baseline models are also excluded from the test set. The main evaluation dataset is constructed by randomly selecting 500 instances from the remaining filtered test set. To assess generalization ability for challenging cases, two additional hard evaluation subsets are also constructed, consisting of rare reaction template instances and rare atom or token instances. For the rare reaction template subset, canonical reaction template frequencies are computed over all ORDerly train and test instances, and 100 instances are sampled in total, with 50 drawn from cases whose canonical reaction template frequency lies in the range one to three and 50

drawn from those in the range four to six. For the rare atom or token subset, each product SMILES is assigned a rarity score based on the extent to which it is composed of rare n-gram tokens, with higher scores given to sequences containing a larger proportion of rare n-grams. Instances are then sorted by this score, and 100 instances are selected by taking the top 50 under the rare 2-gram scoring and the top 50 under the rare 3-gram scoring. For multi-step retrosynthesis evaluation, 500 target molecules are sampled from each of the PaRoutes set-n1 and set-n5 benchmarks (Genheden and Bjerrum, 2022). A fixed subset is used to ensure a controlled and repeatable comparison across methods under the same MCTS search budget. The details of the evaluation dataset creation are described in Appendix F.2.

3.4 Hyperparameters

BFloat16 (Wang and Kanwar, 2019) weight precision is used with full parameter tuning on 8 NVIDIA H100 GPUs. Identical training hyperparameters are applied across all ablation studies and the Prediction-Only baseline. For greedy metrics, greedy decoding is used to obtain a single top-1 reactant suggestion per instance. For sampling-based metrics, 100 reactant suggestions are sampled per instance with temperature 1.2, matching the setting used by RetroReasoner, unless the corresponding open evaluation code specifies different sampling parameters. For all MCTS-based comparisons, including RetroInText, MCTS is run with a maximum of 300 iterations, a beam width of 10, and a maximum search depth of 8. Training hyperparameters are provided in Appendices C and D, MCTS settings in Appendix E, and evaluation and expert-model settings in Appendix F.

3.5 Results

This section presents the main evaluation, rare template and rare atom/token evaluations, multi-step retrosynthesis results, the round-trip reward ablation in Section 3.5.4, and the human expert evaluation in Section 3.5.5. Appendix G provides supplementary reasoning, linking-text, model-size, and sampling ablations, as well as forward-model, inference-speed, confidence-interval, benchmark, qualitative, intermediate-step, and LLM-judge analyses. It supplements the round-trip reward results with forward-model error and external-verifier analyses, and the human evaluation with detailed procedures and qualitative expert feedback.

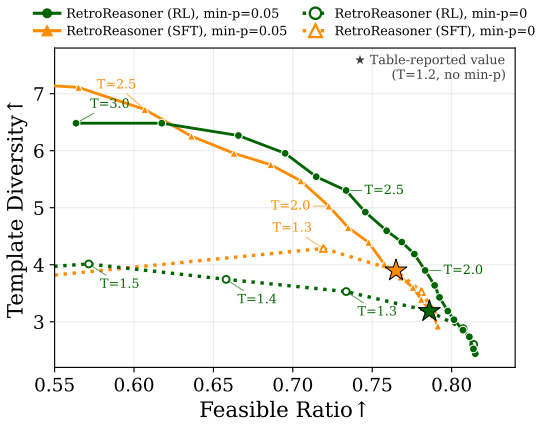


Figure 3: Feasibility-diversity trade-off of RetroReasoner (SFT) and RetroReasoner (RL). Feasible Ratio↑ and Template Diversity↑ are measured while varying the sampling temperature from 1.2 to 3.0 under min-p decoding ($p = 0.05$). RetroReasoner (RL) achieves higher Template Diversity↑ at matched Feasible Ratio↑ levels, forming a more favorable empirical Pareto frontier than SFT.

3.5.1 Main Evaluation

RL improves accuracy and round-trip consistency while shifting the feasibility-diversity trade-off. Table 1 shows that RetroReasoner generally outperforms *Molecular Reasoning LLMs* and Prediction-Only, with particularly clear gains in Exact@100↑ and Template Diversity↑. These results indicate that explicit disconnection reasoning broadens the coverage of plausible reactant solutions beyond direct prediction. RL further improves Exact@1↑, Round-trip@1↑, and Feasible Ratio↑, but decreases Template Diversity↑ from 3.898 to 3.186 under the default sampling configuration. To determine whether this reduction reflects an inherent loss of diversity or a decoding-dependent trade-off, Figure 3 compares SFT and RL across a broader range of sampling conditions. Min-p decoding is applied to suppress low-probability out-of-distribution tokens while allowing the sampling temperature to vary. At comparable Feasible Ratio↑ levels, RetroReasoner (RL) consistently achieves higher Template Diversity↑ than RetroReasoner (SFT), forming a more favorable empirical Pareto frontier. Thus, the lower diversity at the default operating point does not persist when feasibility and diversity are compared under controlled decoding conditions. Detailed token-entropy and decoding analyses are provided in Appendix G.1.5.

	Greedy Metrics		Sampling Metrics			
	Exact@1↑	Round-trip@1↑	Exact@100↑	Round-trip@100↑	Feasible Ratio↑	Template Diversity↑
Molecular Prediction LLMs						
LlaSMol (Yu et al., 2024)	**0.254**	**0.668**	**0.584**	**0.926**	**0.437**	**7.814**
Mol-Instructions (Fang et al., 2023)	0.008	0.036	0.038	0.188	0.007	0.290
BioT5+ (Pei et al., 2024)	0.098	0.388	0.166	0.660	0.366	1.198
PRESTO (Cao et al., 2024)	0.006	0.186	0.012	0.458	0.434	0.496
Mol-LLM (Lee et al., 2025)	0.016	0.036	0.066	0.154	0.015	0.192
Molecular Reasoning LLMs						
Chem-R (Wang et al., 2025)	**0.214**	**0.624**	**0.382**	**0.902**	**0.645**	2.510
ChemDFM (Zhao et al., 2024)	0.140	0.360	0.338	0.864	0.380	**2.594**
ether0 (Narayanan et al., 2025)	0.034	0.488	0.128	0.758	0.487	1.330
General Purpose LLMs						
OpenAI-o3 (OpenAI, 2025)	0.010	0.022	-	-	-	-
GPT-5-mini (Singh et al., 2025)	0.010	**0.038**	**0.152**	**0.498**	**0.031**	**0.794**
GPT-oss-120B (Agarwal et al., 2025)	**0.014**	0.028	0.070	0.164	0.004	0.190
Qwen3-8B (Yang et al., 2025)	0.000	0.008	0.006	0.096	0.006	0.074
Qwen3-235B-A22B (Yang et al., 2025)	0.006	0.024	0.002	0.006	0.000	0.004
Expert Models						
RetroSynFlow (Yadav et al., 2026)	0.424	0.528	0.692	0.702	**0.533**	3.358
G2S-HCVAE (Wang et al., 2026)	**0.492**	**0.814**	**0.704**	**0.944**	0.482	**3.872**
Prediction-Only (SFT)	0.482	0.784	0.678	0.950	0.774	2.562
Prediction-Only (RL)	0.486	0.802	0.662	0.936	0.785	2.324
RetroReasoner (SFT)	0.512	0.812	**0.734**	0.944	0.765	**3.898**
RetroReasoner (RL)	**0.526**	**0.826**	0.724	**0.952**	**0.786**	3.186

Table 1: Main evaluation comparison. Performance in each category is shown. The highest value within each category is highlighted in **bold**. Sampling metrics for RetroReasoner and Prediction-Only use the default decoding setting with temperature 1.2; the feasibility-diversity trade-off across decoding settings is analyzed in Figure 3.

3.5.2 Rare Template and Rare Atom/Token

RetroReasoner is more robust on hard instances. Table 2 shows evaluation results using the rare reaction template and atom/token instances. The performance gap between RetroReasoner and Prediction-Only widens in this challenging problem setting. Notably, in the rare atom/token evaluation, RetroReasoner demonstrates a higher Feasible Ratio↑ and Template Diversity↑ compared to Prediction-Only. These results indicate that RetroReasoner is more robust in handling challenging cases, likely due to the inclusion of strategic reasoning in its design.

3.5.3 Multi-step Retrosynthesis

The lower single-step diversity under default RL decoding does not translate into lower multi-step route diversity. Although RetroReasoner (RL) shows lower Template Diversity↑ than its SFT counterpart under the default single-step decoding setting, this reduction does not propagate to final route diversity. As shown in Table 3, the Number of Clusters↑ increases from 1,130 to 1,150 on set-n1 and from 1,243 to 1,332 on set-n5 after RL. Routes Extracted↑ also increases on set-n1 (4,432 to 4,612), although it slightly decreases on set-n5 (5,246 to 5,080). In addition, RL increases Solved Targets↑ relative to SFT on both PaRoutes subsets. These results indicate that the lower raw single-step Template Diversity↑ at one decoding operating point does not imply reduced diversity of the complete routes discovered by search. RetroInText is additionally included as a reproducible external multi-step planning baseline. RetroReasoner (RL) solves fewer targets than RetroInText (423 vs. 449 on set-n1 and 416 vs. 451 on set-n5), but discovers substantially more complete routes (4,612 vs. 2,659 and 5,080 vs. 2,651) and route clusters (1,150 vs. 694 and 1,332 vs. 742). Accordingly, the multi-step advantage of RetroReasoner is primarily reflected in broader route exploration and route diversity rather than uniformly higher target solvability.

Optimizing LLM inference within parallelized tree search improves both search efficiency and synthetic route diversity. Figure 4 shows performance on the PaRoutes set-n1 and set-n5 subsets when using a single MCTS search worker and when applying the parallelization method introduced in Section 2.4 with varying numbers of workers. While the Solved Targets↑ is largely maintained as the number of workers increases, the average solving time is reduced by nearly sixfold. Moreover, by introducing the unobserved count, workers are encouraged to explore branches that are not currently being searched by other workers. As a result, Routes Extracted↑ and Number of Clusters↑, which measure the diversity of synthetic routes, gradually increase. This suggests that the proposed parallelization strategy is also beneficial in terms of route diversity.

	Exact@1↑	Round-trip@1↑	Exact@100↑	Round-trip@100↑	Feasible Ratio↑	Template Diversity↑
Rare Template Evaluation						
Prediction-Only (SFT)	0.12	0.65	0.24	0.92	0.560	3.20
Prediction-Only (RL)	0.12	0.64	0.22	0.89	0.621	2.66
RetroReasoner (SFT)	**0.14**	0.63	**0.41**	0.92	0.587	**5.20**
RetroReasoner (RL)	0.13	**0.70**	0.39	**0.95**	**0.625**	4.25
Rare Atom/Token Evaluation						
Prediction-Only (SFT)	0.36	0.49	0.52	0.82	0.458	2.02
Prediction-Only (RL)	0.35	0.49	0.49	0.78	0.478	1.97
RetroReasoner (SFT)	**0.43**	**0.64**	0.66	0.91	0.530	**3.43**
RetroReasoner (RL)	0.38	0.60	**0.67**	**0.92**	**0.557**	2.97

Table 2: Performance comparison of hard instances consisting of rare reaction templates and rare atoms/tokens. The highest metric performance for each evaluation dataset is highlighted in **bold**.

Single-step Model	Search Method	Value Function	Solved Targets↑	Routes Extracted↑	Number of Clusters↑
PaRoutes (500 subset of set-n1)					
MolT5 (RetroInText) (Kang et al., 2025)	MCTS	RetroInText fusion	**449**	2659	694
Prediction-Only (SFT)	MCTS	Retro* default	*438*	*3298*	*874*
Prediction-Only (RL)	MCTS	Retro* default	*414*	*2659*	*752*
RetroReasoner (SFT)	MCTS	Retro* default	*415*	*4432*	*1130*
RetroReasoner (RL)	MCTS	Retro* default	*423*	***4612***	***1150***
PaRoutes (500 subset of set-n5)					
MolT5 (RetroInText) (Kang et al., 2025)	MCTS	RetroInText fusion	**451**	2651	742
Prediction-Only (SFT)	MCTS	Retro* default	429	3636	888
Prediction-Only (RL)	MCTS	Retro* default	420	3289	915
RetroReasoner (SFT)	MCTS	Retro* default	398	**5246**	1243
RetroReasoner (RL)	MCTS	Retro* default	416	5080	**1332**

Table 3: Multi-step performance comparison. The highest metric performance is highlighted in **bold**.

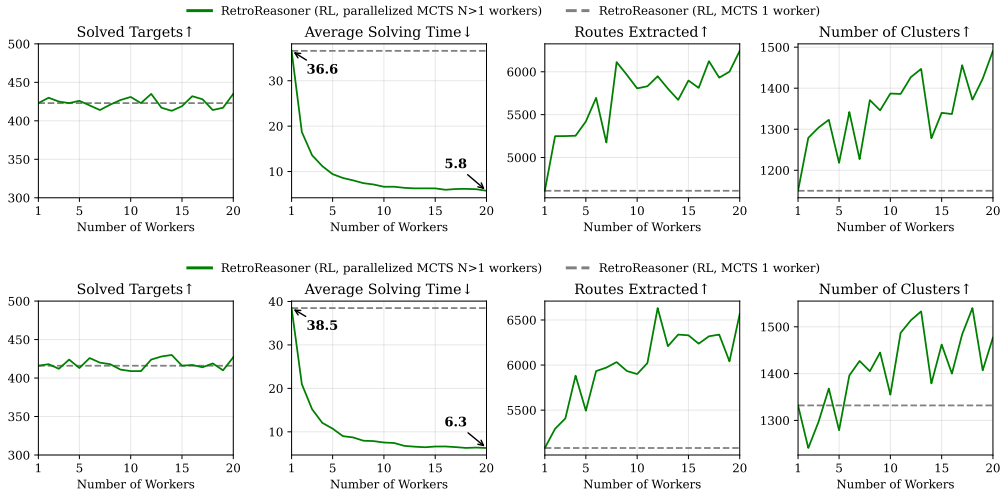


Figure 4: Performance comparison across different numbers of MCTS workers on the PaRoutes set-n1 and set-n5 subsets.

3.5.4 Effect of Round-trip Reward

Round-trip reward enhances feasible molecule space exploration. Table 4 compares the results of using the round-trip reward versus the exact match reward for the reactant. When $R^{\text{round-trip}}$ is not used (i.e., R^{exact} is used), Feasible Ratio↑ increases, but Exact@100↑, Round-trip@100↑, and Template Diversity↑ decrease significantly compared with the results obtained using $R^{\text{round-trip}}$. These findings suggest that the $R^{\text{round-trip}}$ helps explore a broader feasible reactant space during policy updates. Without it, the model fails to explore this broader space and instead narrows its focus, ultimately handling a smaller reactant space. This comparison isolates the effect of the reward function within RL and should be distinguished from the SFT-RL comparison in Section 3.5.1: round-trip-based RL preserves substantially more diversity than exact-match-based RL, even though its default decoding point remains less diverse than the SFT model.

	Exact@1↑	Round-trip@1↑	Exact@100↑	Round-trip@100↑	Feasible Ratio↑	Template Diversity↑
RetroReasoner (RL)	**0.526**	**0.826**	**0.724**	**0.952**	0.786	**3.186**
RetroReasoner (RL, w/ R^{exact})	0.524	0.824	0.696	0.934	**0.812**	2.430

Table 4: Performance comparison of round-trip reward $R^{\text{round-trip}}$ in RL. When $R^{\text{round-trip}}$ are not used, the reward is the exact match between the model's predicted reactant and the reactant labeled in the data instance (R^{exact}). The best performance is highlighted in **bold**.

	Our round-trip model	MolecularTransformer (MT)	Our round-trip model	MolecularTransformer (MT)
	Round-trip@1↑	Round-trip@1↑	Round-trip@100↑	Round-trip@100↑
Prediction-Only (RL)	0.802	0.740	0.936	0.862
RetroReasoner (RL)	**0.826**	**0.748**	**0.952**	**0.896**

Table 5: Comparison of round-trip metrics under our internal forward verifier and MolecularTransformer (MT), an independent external verifier not trained on ORDerly.

	RetroReasoner win rate	**Chem-R win rate**	**Tie**
Person A	88%	4%	8%
Person B	64%	12%	24%
Person C	20%	80%	0%
Person D	84%	10%	6%
Person E	20%	28%	52%
Average	55.2%	26.8%	18%

Table 6: Comparison of win rates between Chem-R, the best-performing *Molecular Reasoning LLMs*, and RetroReasoner through A/B testing of the reasoning contents.

The effectiveness of the round-trip reward is consistent under an independent forward verifier. Because the round-trip reward and several evaluation metrics rely on a learned forward model, predictions are additionally evaluated with Molecular Transformer (MT) (Schwaller et al., 2019), an independent verifier not trained on ORDerly. Table 5 reports Round-trip@1↑ and Round-trip@100↑ under both validators. RetroReasoner (RL) maintains the same advantage over Prediction-Only (RL) under MT, improving Round-trip@1↑ from 0.740 to 0.748 and Round-trip@100↑ from 0.862 to 0.896. These results suggest that the observed gains are not specific to the internal verifier, while round-trip consistency remains a proxy for laboratory feasibility. Detailed results are provided in Appendix G.7.

3.5.5 Human Expert Evaluation

To further assess whether the improved prediction performance is accompanied by higher-quality chemical reasoning, a human evaluation is performed against Chem-R, the strongest reasoning baseline in the comparison except RetroReasoner. Table 6 reports the results of a blind pairwise evaluation conducted by five experts. The evaluation set includes 33 instances where both methods are correct, 9 instances where only RetroReasoner is correct, and 8 instances where only Chem-R is correct. In each case, both the reasoning process and the predicted reactants are provided, and the evaluators are asked to choose the response that demonstrates better chemical reasoning. As shown in Table 6, RetroReasoner is preferred over Chem-R on average, supporting the higher overall quality of its single-step retrosynthetic reasoning and reactant predictions. Detailed evaluation instructions and qualitative expert feedback are provided in Appendix G.14.

4 Conclusion

This study presents RetroReasoner, a retrosynthetic reasoning model that follows chemists' disconnection-based strategies for organic synthesis. SyntheticRetro is introduced to generate structured reasoning data for supervised fine-tuning, and RetroReasoner is further optimized through reinforcement learning with a round-trip reward. Experimental results show that the proposed strategic reasoning model outperforms direct reactant prediction as well as previous reasoning approaches, particularly on challenging cases involving rare reaction types and rare atoms or tokens. RetroReasoner also extends to multi-step retrosynthetic planning, where LLM inference optimization combined with parallelized MCTS reduces solving time and improves route diversity. These results connect strategic single-step reasoning to broader route exploration and highlight RetroReasoner's potential for future agent systems in organic synthesis prediction and planning.

Limitation

Although RetroReasoner, a reasoning LLM based on chemists' strategy, is developed in this study, several limitations exist. First, it does not consider the various environmental conditions in actual synthesis, such as temperature, humidity, and pressure. Second, the model primarily handles organic molecules and small molecules, but is not capable of dealing with polymers or crystals used in the development of new materials. Lastly, the current reasoning process remains at the level of reaction mechanisms and does not extend to more sophisticated reasoning, such as electron transfer. If these limitations are addressed, the model could evolve into a more advanced chemical agent system that would be valuable for organic synthesis research and could be effectively applied in an industrial environment.

Ethical Considerations

Retrosynthesis prediction with RetroReasoner can suggest feasible organic synthesis processes for a given target molecule. However, the model may also generate predictions involving hazardous or toxic compounds. Therefore, RetroReasoner is intended for research use at the current stage, and any future practical deployment should be accompanied by expert review and appropriate safety assessment. Another ethical consideration is the computational cost of generating data with SyntheticRetro and training LLMs on the generated data. Such costs may contribute to increased carbon emissions. This concern could become more pronounced in future development if models larger than the current 8B-scale model or larger training datasets are used.

Acknowledgements

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) in 2026 (Grant No. RS-2026-25613293, Project Title: Physics- and Mathematics-Informed Causal AI Models for Robust Analysis and Prediction under Geometry, Boundary-Condition, Parameter, and Multi-Environment Shifts, 20%) and (Grant No. RS-2026-25473475, Project Title: Functional Analytic Approach to Foundational Theory of Stochastic PDE based Probabilistic Generative Models, 20%), and was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korean government (MSIT) (IITP-2026-RS-2025-02304828, artificial intelligence star fellowship support program to nurture the best talents, 20%), (No. RS-2026-25507543, Development of AI Co-Scientist based on Scientific Causal World Model, 20%), and (No. RS-2022-II220612, Project Title: Geometric and Physical Commonsense Reasoning based Behavior Intelligence for Embodied AI, 20%), and was also supported by LG AI Research.

References

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, and 1 others. 2025. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*.

He Cao, Yanjun Shao, Zhiyuan Liu, Zijing Liu, Xiangru Tang, Yuan Yao, and Yu Li. 2024. Presto: Progressive pretraining enhances synthetic chemistry outcomes. *arXiv preprint arXiv:2406.13193*.

Binghong Chen, Chengtao Li, Hanjun Dai, and Le Song. 2020. Retro*: Learning retrosynthetic planning with neural guided a* search. In *The 37th International Conference on Machine Learning (ICML 2020)*.

Shuan Chen, Sunggi An, Ramil Babazade, and Yousung Jung. 2024. Precise atom-to-atom mapping for organic reactions via human-in-the-loop machine learning. *Nature Communications*, 15(1):2250.

Elias James Corey. 1967. General methods for the construction of complex molecules. *Pure and Applied chemistry*, 14(1):19–38.

Elias James Corey. 1988. Robert robinson lecture. retrosynthetic thinking—essentials and examples. *Chemical society reviews*, 17:111–133.

Elias James Corey and Xue-Min Cheng. 1995. *The logic of chemical synthesis*. Wiley.

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

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35:16344–16359.

Daylight. Smarts - a language for describing molecular patterns.

Daylight. 2001. Reaction smiles and smirks.

Joseph L Durant, Burton A Leland, Douglas R Henry, and James G Nourse. 2002. Reoptimization of mdl

keys for use in drug discovery. *Journal of chemical information and computer sciences*, 42(6):1273–1280.

Carl Edwards, Tuan Lai, Kevin Ros, Garrett Honke, Kyunghyun Cho, and Heng Ji. 2022. Translation between molecules and natural language. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 375–413.

William Falcon and The PyTorch Lightning team. 2019. PyTorch Lightning.

Yin Fang, Xiaozhuan Liang, Ningyu Zhang, Kangwei Liu, Rui Huang, Zhuo Chen, Xiaohui Fan, and Huajun Chen. 2023. Mol-instructions: A large-scale biomolecular instruction dataset for large language models. *arXiv preprint arXiv:2306.08018*.

Samuel Genheden and Esben Bjerrum. 2022. Paroutes: towards a framework for benchmarking retrosynthesis route predictions. *Digital Discovery*, 1(4):527–539.

Stephen Heller, Alan McNaught, Stephen Stein, Dmitrii Tchekhovskoi, and Igor Pletnev. 2013. Inchi-the worldwide chemical structure identifier standard. *Journal of cheminformatics*, 5(1):7.

Chenglong Kang, Xiaoyi Liu, and Fei Guo. 2025. Retrointext: A multimodal large language model enhanced framework for retrosynthetic planning via in-context representation learning. In *International Conference on Learning Representations*, volume 2025, pages 71919–71936.

Mario Krenn, Florian Häse, AkshatKumar Nigam, Pascal Friederich, and Alan Aspuru-Guzik. 2020. Self-referencing embedded strings (selfies): A 100% robust molecular string representation. *Machine Learning: Science and Technology*, 1(4):045024.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Greg Landrum. 2013. Rdkit documentation. *Release*, 1(1-79):4.

Chanhui Lee, Hanbum Ko, Yuheon Song, YongJun Jeong, Rodrigo Hormazabal, Sehui Han, Kyunghoon Bae, Sungbin Lim, and Sungwoong Kim. 2025. Mol-llm: Multimodal generalist molecular llm with improved graph utilization. *arXiv preprint arXiv:2502.02810*.

Xinyi Li, Sai Wang, Yutian Lin, Yu Wu, and Yi Yang. 2025. Retro-expert: Collaborative reasoning for interpretable retrosynthesis. *arXiv preprint arXiv:2508.10967*.

Anji Liu, Jianshu Chen, Mingze Yu, Yu Zhai, Xuewen Zhou, and Ji Liu. 2018. Watch the unobserved: A simple approach to parallelizing monte carlo tree search. *arXiv preprint arXiv:1810.11755*.

Wei Liu, Jiangtao Feng, Hongli Yu, Yuxuan Song, Yuqiang Li, Shufei Zhang, Lei Bai, Wei-Ying Ma, and Hao Zhou. 2026. Retro-r1: Llm-based agentic retrosynthesis. *Advances in Neural Information Processing Systems*, 38:70709–70737.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Siddharth M Narayanan, James D Braza, Ryan-Rhys Griffiths, Albert Bou, Geemi Wellawatte, Mayk Caldas Ramos, Ludovico Mitchener, Samuel G Rodriques, and Andrew D White. 2025. Training a scientific reasoning model for chemistry. *arXiv preprint arXiv:2506.17238*.

OpenAI. 2025. Openai o3 and o4-mini system card. *OpenAI*.

Qizhi Pei, Lijun Wu, Kaiyuan Gao, Xiaozhuan Liang, Yin Fang, Jinhua Zhu, Shufang Xie, Tao Qin, and Rui Yan. 2024. Biot5+: Towards generalized biological understanding with iupac integration and multi-task tuning. *arXiv preprint arXiv:2402.17810*.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 3505–3506.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Philippe Schwaller, Teodoro Laino, Théophile Gaudin, Peter Bolgar, Christopher A Hunter, Costas Bekas, and Alpha A Lee. 2019. Molecular transformer: a model for uncertainty-calibrated chemical reaction prediction. *ACS central science*, 5(9):1572–1583.

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

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

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Yu Shee, Anton Morgunov, Haote Li, and Victor S Batista. 2025. Directmultistep: Direct route generation for multistep retrosynthesis. *Journal of Chemical Information and Modeling*, 65(8):3903–3914.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2024. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*.

Chence Shi, Minkai Xu, Hongyu Guo, Ming Zhang, and Jian Tang. 2020. A graph to graphs framework for retrosynthesis prediction. In *International conference on machine learning*, pages 8818–8827. PMLR.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, and 1 others. 2025. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*.

Xiaozhuang Song, Xuanhao Pan, Xinjian Zhao, Hangting Ye, Shufei Zhang, Jian Tang, and Tianshu Yu. 2025. Aot*: Efficient synthesis planning via llm-empowered and-or tree search. *arXiv preprint arXiv:2509.20988*.

Zhengkai Tu and Connor W Coley. 2022. Permutation invariant graph-to-sequence model for template-free retrosynthesis and reaction prediction. *Journal of chemical information and modeling*, 62(15):3503–3513.

Huibin Wang, Yueqing Zhang, Zehui Wang, Jiaxi Zhuang, Zixian Cheng, Ying Qian, Aimin Zhou, Sihua Peng, and Xiao He. 2026. Enhancing diversity of template-free retrosynthesis prediction via hierarchical latent variables. *Journal of Chemical Information and Modeling*, 66(6):3074–3090.

Shibo Wang and Pankaj Kanwar. 2019. Bfloat16: The secret to high-performance on cloud tpus.

Weida Wang, Benteng Chen, Di Zhang, Wanhao Liu, Shuchen Pu, Ben Gao, Jin Zeng, Xiaoyong Wei, Tianshu Yu, Shuzhou Sun, and 1 others. 2025. Chem-r: Learning to reason as a chemist. *arXiv preprint arXiv:2510.16880*.

David Weininger. 1988. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *J. Chem. Inf. Comput. Sci.*, 28(1):31–36.

Daniel S Wigh, Joe Arrowsmith, Alexander Pomberger, Kobi C Felton, and Alexei A Lapkin. 2024. Orderly: data sets and benchmarks for chemical reaction data. *Journal of Chemical Information and Modeling*, 64(9):3790–3798.

Nguyen Xuan-Vu, Daniel P Armstrong, Zlatko Jončev, and Philippe Schwaller. 2025. Tempre: Template generation for single and direct multi-step retrosynthesis. *arXiv preprint arXiv:2507.21762*.

Robin Yadav, Qi Yan, Guy Wolf, Joey Bose, and Renjie Liao. 2026. Retro synflow: Discrete flow-matching for accurate and diverse single-step retrosynthesis. *Advances in Neural Information Processing Systems*, 38:55229–55258.

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

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Botao Yu, Frazier N Baker, Ziqi Chen, Xia Ning, and Huan Sun. 2024. Llasmol: Advancing large language models for chemistry with a large-scale, comprehensive, high-quality instruction tuning dataset. *arXiv preprint arXiv:2402.09391*.

Gyeong-In Yu, Joo Seong Jeong, Geon-Woo Kim, Soojeong Kim, and Byung-Gon Chun. 2022. Orca: A distributed serving system for {Transformer-Based} generative models. In *16th USENIX symposium on operating systems design and implementation (OSDI 22)*, pages 521–538.

Situo Zhang, Hanqi Li, Lu Chen, Zihan Zhao, Xuanze Lin, Zichen Zhu, Bo Chen, Xin Chen, and Kai Yu. 2025. Reasoning-driven retrosynthesis prediction with large language models via reinforcement learning. *arXiv preprint arXiv:2507.17448*.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, and 1 others. 2023. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*.

Zihan Zhao, Bo Chen, Ziping Wan, Lu Chen, Xuanze Lin, Shiyang Yu, Situo Zhang, Da Ma, Zichen Zhu, Danyang Zhang, and 1 others. 2025. Chemdfm-r: A chemical reasoning llm enhanced with atomized chemical knowledge. *arXiv preprint arXiv:2507.21990*.

Zihan Zhao, Da Ma, Lu Chen, Liangtai Sun, Zihao Li, Yi Xia, Hongshen Xu, Zichen Zhu, Su Zhu, Shuai Fan, and 1 others. 2024. Chemdfm: A large language foundation model for chemistry. In *Neurips 2024 Workshop Foundation Models for Science: Progress, Opportunities, and Challenges*.

Model Name	Data Source	Model Size	Public Release	Training Objective	Reward Function	Task Scope	Supporting Information
Chem-R (Wang et al., 2025)	ChemLLMBench (USPTO 50k, mixed)	Llama-3.1-8B	O	GRPO (Task-balanced sampling)	Answer accuracy	Forward prediction, Retrosynthesis, Reagent selection, Yield prediction	SMILES, Functional group
ChemDFM (Zhao et al., 2024)	ChemLLMBench (USPTO 50k, mixed)	Llama3-13B	O	Domain pretraining, Instruction tuning	-	Forward prediction, Retrosynthesis, Reagent selection, Yield prediction	SMILES, Functional group
ether0 (Narayanan et al., 2025)	25 of USPTO 50k	Mistral-Small-24B	O	GRPO	Answer format, Answer accuracy	Forward prediction, Retrosynthesis	SMILES, Functional group
Retro-Expert (Li et al., 2025)	USPTO-50k(5k)	Qwen2.5-7B	X	GRPO	Answer format, Answer accuracy, Stage	Retrosynthesis	Reaction type, Reaction center, External knowledge
RetroDFM-R (Zhang et al., 2025)	USPTO-50k(5k), USPTO-FULL (100k of 1M)	Llama3-8B	O	DAPO	Answer format, Answer accuracy	Retrosynthesis	SMILES, IUPAC name
ChemDFM-R (Zhao et al., 2025)	USPTO-50k(5k), USPTO-FULL (100k of 1M), USPTO-MIT (40k)	Qwen2.5-14B	O	DAPO	Answer format, Answer accuracy	Retrosynthesis	SMILES, Functional group
RetroReasoner (Ours)	**ORDerly**	**Qwen3-8B**	**O**	**GRPO**	**Round-trip accuracy**	**Retrosynthesis**	**SMILES, Functional group, Reaction template, Reaction center, Synthon, Synthetic equivalent**

Table 7: Comparison of *Molecular Reasoning LLMs*. The Supporting Information column lists inputs used to generate reasoning. Chem-R, ChemDFM, ether0, RetroDFM-R, and ChemDFM-R rely mainly on SMILES, functional groups, and IUPAC names, so their rationales stay at generic product analysis. Retro-Expert adds reaction type and reaction center, but does not derive synthons or synthetic equivalents from the disconnection. Most models use reactant correctness as the reward, which can bias learning and does not reflect feasibility, since even reactants that can genuinely synthesize the target product may receive no positive reward if they do not match the single reference reactant. RetroReasoner instead uses a forward model based round-trip reward to better target feasible reactants.

A Related Works

A.1 Disconnection-aware Retrosynthesis Before LLMs

Prior to LLM-based approaches, semi-template retrosynthesis models that explicitly leverage reaction centers and intermediate structures have been proposed to mimic chemists' retrosynthetic strategies. For example, GLN (Dai et al., 2019) combines reaction templates with graph neural networks to learn whether a specific transformation rule can be applied to a given product, complementing the chemical validity of template-based retrosynthesis with neural networks. RetroXpert (Yan et al., 2020) proposes a two-stage architecture that first identifies potential reaction centers in the product, then generates intermediate synthons based on these centers, and finally predicts the reactants. Similarly, G2G (Shi et al., 2020) formulates retrosynthesis as a graph-to-graphs translation problem by decomposing the product graph into multiple synthons based on reaction centers, and then converting each synthon into a reactant graph. These studies demonstrate the importance of disconnection-aware intermediate representations in retrosynthesis prediction. However, in existing semi-template or graph-based models, the intermediate states are typically used only as prediction outputs of specific modules and are not trained as human-readable natural language reasoning trajectories. RetroReasoner is distinguished from these approaches in that it follows such chemist-like decomposition processes while transforming product analysis, candidate substructure identification, strategic bond disconnection, synthon generation, and synthetic-equivalent mapping into a single structured reasoning trace, which then serves as a direct training target for both supervised fine-tuning and verifier-based reinforcement learning. Specifically, the SyntheticRetro component of RetroReasoner is designed to generate reasoning text composed of $\mathcal{R}_1$ product analysis, $\mathcal{R}_2$ candidate substructure, $\mathcal{R}_3$ strategic bond disconnection, and $\mathcal{R}_4$ synthetic equivalent mapping, along with linking texts $\mathcal{L}_{12}$, $\mathcal{L}_{23}$, and $\mathcal{L}_{34}$ that connect these reasoning stages.

A.2 Reasoning LLMs for Chemical Reaction Prediction

Recently, as LLMs have been applied to chemistry and synthesis problems, research has emerged that goes beyond simple SMILES-to-SMILES translation (Fang et al., 2023; Yu et al., 2024; Cao et al., 2024; Pei et al., 2024; Lee et al., 2025) to generate natural language reasoning or to improve chemical decision-making through reinforcement learning. Chem-R (Wang et al., 2025) is proposed as a general chemical reasoning model that combines chemical foundation training, rea-

soning protocol distillation, and multi-task GRPO to learn chemist-like reasoning abilities across various molecular and reaction-level tasks, including forward prediction, retrosynthesis, reagent selection, and yield prediction. ChemDFM-R (Zhao et al., 2025) enhances chemical reasoning capabilities by leveraging atomized chemical knowledge and domain-specific reinforcement learning, while ether0 (Narayanan et al., 2025) is a 24B-scale reasoning model post-trained to perform natural language reasoning before outputting chemical structures. More directly related to RetroReasoner are RetroDFM-R (Zhang et al., 2025) and Retro-Expert (Li et al., 2025). RetroDFM-R is a reasoning-based LLM specialized for retrosynthesis that improves prediction accuracy and explainability through reinforcement learning with chemically verifiable rewards. Retro-Expert is a collaborative reasoning framework in which a specialized retrosynthesis model constructs the chemical decision space, an LLM performs critical reasoning, and reinforcement learning optimizes an interpretable decision policy. However, compared to these works, RetroReasoner aligns its reasoning more closely with the logical structure intrinsic to retrosynthesis. While existing chemical reasoning LLMs generally provide explanations grounded in product functional groups, reaction types, and general chemical knowledge, RetroReasoner explicitly incorporates into its reasoning trace which bonds should be disconnected, which synthons are formed after disconnection, and which synthetic equivalents each synthon should be mapped to. Thus, the key distinction of RetroReasoner lies not merely in "generating reasoning," but in bridging the logical gap between generic product-level analysis and reactant selection through disconnection-centric reasoning. The differences from these related works are summarized in Table 7.

A.3 Round-trip Rewards and Multi-step Planning

In retrosynthesis, multiple reactant combinations are possible for a single product, so using exact match with a single reference as the sole reward signal may unnecessarily penalize chemically valid alternative pathways. To address this limitation, prior work has employed round-trip accuracy, which verifies whether predicted reactants can reconstruct the target product via a forward prediction model, as an evaluation metric for single-step retrosynthesis (Schwaller et al., 2020). RetroReasoner extends this concept beyond evaluation by incorporating it as a reward signal during the RL stage. Specifically, a reward is granted when the predicted reactants successfully reconstruct the original product through the forward model, enabling the model to learn feasible reactant proposals even when they differ from the reference. In terms of multi-step planning, recent studies have explored directly generating entire routes or coupling LLMs with search algorithms. DirectMultiStep (Shee et al., 2025) proposes generating multi-step synthetic routes directly as a single string, AOT* (Song et al., 2025) integrates LLM-generated synthetic pathways with AND-OR tree search, and Retro-R1 (Liu et al., 2026) trains a multi-step retrosynthesis agent via reinforcement learning. RetroInText (Kang et al., 2025) combines a MolT5 (Edwards et al., 2022)-based single-step model with MCTS and a fusion-based value function for retrosynthetic planning. In contrast, RetroReasoner does not generate entire routes end-to-end; rather, it adopts a hybrid approach that first strengthens a disconnection-centric single-step proposal model and then integrates it into a parallelized MCTS. Accordingly, RetroReasoner is better positioned not as a multi-step planner itself, but as a strategic single-step reasoner that can be plugged into a search framework.

B Details of SyntheticRetro

This section describes SyntheticRetro in detail using an illustrative example.

B.1 Supporting Information

Supporting information refers to the data required to construct reasoning text. This information is used in both structured reasoning steps and the generation of linking text.

B.1.1 Direct-usable Information

Direct-usable information is obtained directly from an RXN SMILES string. For example, given the RXN SMILES `O=C(C1CC1)C(c1ccccc1F)N1CCC(S)CC1.CCCC(=O)Cl»CCCC(=O)SC1CCN(C(C(=O)C2CC2)c2ccccc2F)CC1`, the reactants are `O=C(C1CC1)C(c1ccccc1F)N1CCC(S)CC1` and `CCCC(=O)Cl`, and the product is `CCCC(=O)SC1CCN(C(C(=O)C2CC2)c2ccccc2F)CC1`. These SMILES strings are treated as direct-usable information.

B.1.2 Model-predicted Information

Model-predicted information is obtained from deep-learning-based atom-mapping models. Lo-

calMapper (Chen et al., 2024) is a sequence-to-sequence model that takes an RXN SMILES string as input and outputs an atom-mapped RXN SMILES string. For the example above, LocalMapper produces

```
[O:13]=[C:12]([CH:14]1[CH2:15][CH2:16]1)
    [CH:11]([c:17]1[cH:18][cH:19][cH
    :20][cH:21][c:22]1[F:23])[N:10]1[CH2
    :9][CH2:8][CH:7]([SH:6])[CH2:25][CH2
    :24]1.[CH3:1][CH2:2][CH2:3][C:4](=[O
    :5])Cl>>[CH3:1][CH2:2][CH2:3][C
    :4](=[O:5])[S:6][CH:7]1[CH2:8][CH2
    :9][N:10]([CH:11]([C:12](=[O:13])[CH
    :14]2[CH2:15][CH2:16]2)[c:17]2[cH
    :18][cH:19][cH:20][cH:21][c:22]2[F
    :23])[CH2:24][CH2:25]1.
```

This representation encodes atom-level changes between reactants and products.

B.1.3 Rule-derived Information

From the atom-mapped SMILES, additional information is extracted using rule-based algorithms. A reaction template is obtained by identifying the minimum bond changes between reactants and products. In the example above, sulfur forms a new bond and chlorine is removed from the reactant. This change is represented as `([S:6]).([C:4](=O)-Cl)»([C:4](=O)-[S:6])`. This instance-specific template is then canonicalized as `[S:2].[C:1](=O)-Cl»[C:1](=O)-[S:2]`. Functional group names and positions, as well as substructures, are extracted using SMARTS (Daylight) patterns. Functional groups defined in RDKit (Landrum, 2013) and substructures used in MACCS keys (Durant et al., 2002) fingerprints are employed. The newly formed bond is identified by comparing bond sets between reactants and products and is treated as the strategic bond disconnection. Synthons are then generated by disconnecting this bond in the product, resulting in two molecular fragments. For example, when the disconnected bond is a single bond between atoms 4 and 6, the resulting synthons are `[6*]C(=O)CCC` and `[4*]SC1CCN(C(C(=O)C2CC2)c2ccccc2F)CC1`. Here, `[n*]` denotes a placeholder corresponding to the original atom index. Finally, each synthon is mapped to its synthetic equivalent using atom mapping information. In this example, the synthon `[6*]C(=O)CCC` corresponds to the reactant `CCCC(=O)Cl`, while the synthon `[4*]SC1CCN(C(C(=O)C2CC2)c2ccccc2F)CC1` corresponds to the reactant `O=C(C1CC1)C(c1ccc cc1F)N1CCC(S)CC1`.

B.2 Asynchronous Data Generation

SyntheticRetro uses vLLM (Kwon et al., 2023) for efficient generation of linking text. However, the total number of requests is the product of the number of instances, the number of linking texts per instance, and the number of samples per instance, which results in a very large request volume. To handle this efficiently, SyntheticRetro adopts a producer–consumer pattern for asynchronous data generation. Figure 5 illustrates the overall generation system, which uses eight GPUs, each running a single vLLM serving instance. The producer constructs prompts for linking text generation and pushes them into a global prompt queue. The consumers, corresponding to the vLLM servers, asynchronously retrieve prompts from the queue and process multiple requests in parallel. Each consumer is configured to handle up to four concurrent requests, and the maximum number of active requests per consumer is controlled using a semaphore mechanism.

B.3 Data Filtering

Instances are excluded when LocalMapper fails to produce an atom-mapped output and returns `None`. Instances are also excluded when generation exceeds the maximum allowed token length, which is indicated by a finish reason of length.

B.4 Generation Hyperparameters

Hyperparameters for data generation in SyntheticRetro are shown in Table 8.

B.5 Rationale Example

A representative example of the generated data is shown in Figure 6.

C Details of RetroReasoner and Prediction-Only

C.1 Training Details

Supervised fine-tuning is conducted using distributed data parallelism (DDP) on eight NVIDIA H100 GPUs, with full parameter tuning in BFloat16 precision. Ten different user prompt templates are used, with only the product SMILES modified for each instance. A simple system prompt of `You are a chemist.` is used. Cross-entropy loss is masked for both system and user prompts, and loss is computed only on the assistant prompt tokens. The final training loss is calculated as the average loss over assistant prompt tokens

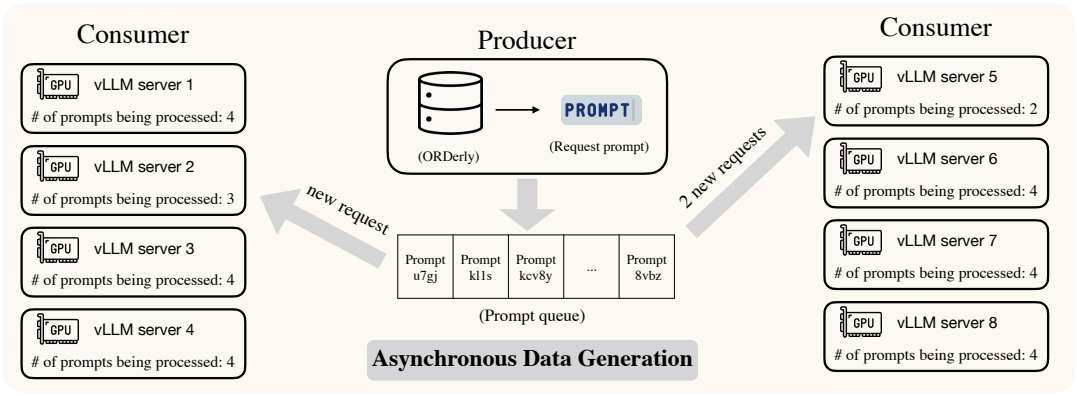


Figure 5: Schematic of the asynchronous data generation system in SyntheticRetro. A producer constructs prompts for linking text generation from ORDerly RXN SMILES and pushes them into a prompt queue, which is then processed asynchronously by multiple vLLM servers. Each vLLM server has a fixed maximum number of concurrent requests, and generation proceeds by filling these slots asynchronously.

Parameter	Value
Model Name	GPT-oss-20B
Temperature	0.8
Max Tokens	500
Presence Penalty	0.0
Frequency Penalty	0.3
Reasoning Effort	Low
Max Producers	20
Max Consumers	31*20
Prompt Queue Size	5000

Table 8: Hyperparameters used in SyntheticRetro data generation.

Experiment Name	Number of Instances	Number of Tokens
Prediction-Only (SFT)	522,630	59,439,861 × 15
RetroReasoner (SFT)	522,630	8,050,108,240
RetroReasoner (SFT, 1 sample)	522,630	536,799,367 × 15
RetroReasoner (SFT, 1.7B)	522,630	8,050,108,240
RetroReasoner (SFT, w/o $\mathcal{L}_{12}, \mathcal{L}_{23}, \mathcal{L}_{34}$)	522,630	319,138,383 × 15
RetroReasoner (SFT, w/ only $\mathcal{R}_1$)	522,630	403,749,140 × 15
Round-trip Forward Model (0.6B)	939,001	107,580,021 × 15
Round-trip Forward Model (8B)	939,001	107,580,021 × 15

Table 9: Number of instances used to train the model used in each experiment in supervised fine-tuning and total number of tokens based on 15 epochs.

in each batch. Reinforcement learning is also performed using eight NVIDIA H100 GPUs with full parameter tuning in BFloat16 precision. Three rollout workers are used, with one worker assigned per GPU, and training is conducted with FSDP using a shard size of four across four GPUs. Reward computation is handled by running a round-trip worker on the remaining GPU. During reward calculation, the caching system described below is used to improve efficiency. The learning curve of reinforcement learning is shown in Figure 7.

C.2 Hyperparameters

All hyperparameters of SFT and RL are shown in Tables 10 and 11.

C.3 Cached Round-trip Verifier

Computing the round-trip reward requires running the round-trip model f_ϕ to predict a product for each sampled reactant set $\hat{\mathbf{y}}^{\text{reactant}}$. Since RL training generates many samples and often revisits the same reactant SMILES strings across workers, repeatedly evaluating f_ϕ on identical inputs becomes a major computational bottleneck. To reduce redundant computation, a shared cache is used for round-trip verification. The cache is a key value store that maps a canonicalized reactant SMILES to the corresponding product prediction from f_ϕ. During training, each rollout worker first queries the cache with $\hat{\mathbf{y}}^{\text{reactant}}$. If a cached entry exists, the stored product prediction is reused to compute $R^{\text{round-trip}}$. Otherwise, $f_\phi(\hat{\mathbf{y}}^{\text{reactant}})$ is computed once, stored in the cache, and then used for reward computation.

D Details of Round-trip Prediction Model

D.1 Training Details

The round-trip model f_ϕ follows the same learning process as the Prediction-Only model, with the only difference being the input and output SMILES. Unlike Prediction-Only, the input consists of reactants and the output is the product. During training, the round-trip model is trained in the same manner as RetroReasoner and Prediction-Only, using cross-entropy for next token prediction. For evaluation,

System Prompt

You are a chemist.

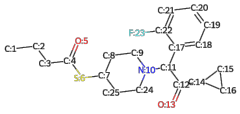
User Prompt

With the given product CCCC(=O)SC1CCN(C(C(=O)C2CC2)c2ccccc2F)CC1, suggest some likely reactants that were used in its synthesis.

Assistant Prompt

$\mathcal{R}_1$

<think>
<PRODUCT_INFO>
{
"Atom mapped SMILES": "[C:1][C:2][C:3][C:4](=[O:5])[S:6][C:7]1[C:8][C:9][N:10]([C:11]([C:12](=[O:13])[C:14]2[C:15][C:16]2)[c:17]2[c:18][c:19][c:20][c:21][c:22]2[F:23])[C:24][C:25]1",
"Functional groups": [
"carbonyl O | C(=O | [4, 5]",
"carbonyl O | C(=O | [12, 13]",
"carbonyl O, excluding COOH | C(=O | [4, 5]",
"carbonyl O, excluding COOH | C(=O | [12, 13]",
"Tertiary amines | N | [10]",
"tert-alicyclic amines (no heteroatoms, not quinine-like bridged N) | N | [10]",
"benzene rings | c2ccccc | [17, 18, 19, 20, 21, 22]",
"halogens | F | [23]",
"ketones | C(C(=O)C | [11, 12, 13, 14]",
"ketones excluding diaryl, a,b-unsat. dienones, heteroatom on Calpha | C(| [12]",
"piperdine rings | C1CCN_CC1 | [7, 8, 9, 10, 24, 25]",
"thioether | C(_SC | [4, 6, 7]"
],
"SMILES statistics": {
"ring": 3,
"carbon": 20
}
}
</PRODUCT_INFO>

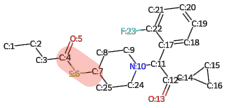


$\mathcal{L}_{12}$

The product contains a sulfur atom linked to a tertiary amine ring that also bears a carbonyl group; this arrangement indicates that the thioether bridge in the molecule must have arisen from the attack of a nucleophilic sulfur on an acyl chloride precursor. Consequently the candidate substructure to focus on is the thioether fragment formed by the S atom connected to the carbonyl-containing ring, as it is the key feature that defines the product's architecture.

$\mathcal{R}_2$

<CANDIDATE_STRUCTURE>
[
"thioether | C(_SC | [4, 6, 7]"
]
</CANDIDATE_STRUCTURE>

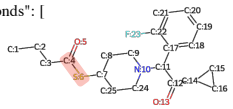


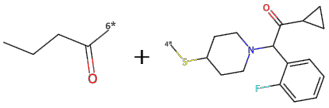
$\mathcal{L}_{23}$

To determine the strategic bond to cut, note that the sulfur atom must have been formed by a nucleophilic attack of a thioether on an acyl chloride; therefore the single S–C bond between atoms 4 and 6 is the logical cleavage point. Consequently, disconnecting this S–C bond yields the two synthons: a thioester fragment and an amino-piperidine ring bearing a ketone that can be assembled from the corresponding acyl chloride and amine precursors.

$\mathcal{R}_3$

<STRATEGIC_BOND_DISCONNECTION>
{
"Disconnect bonds": [
[
4,
6,
"Single"
]
],
"Synthons": [
"[6*]C(=O)CCC",
"[4*]SC1CCN(C(C(=O)C2CC2)c2ccccc2F)CC1"
]
}
</STRATEGIC_BOND_DISCONNECTION>



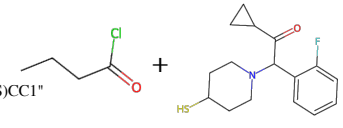


$\mathcal{L}_{34}$

The thioether fragment in the product originates from a nucleophilic sulfur attacking an acyl chloride, so the synthetic equivalent containing the thioester synthon is usually a simple acyl chloride that starts as a linear ketone. The amino-piperidine ring with its ketone group is typically obtained from an amine precursor that begins as a cyclic amide bearing the fluorobenzyl substituent.

$\mathcal{R}_4$

<SYNTHETIC_EQUIVALENT>
[
"CCCC(=O)Cl",
"O=C(C1CC1)C(c1ccccc1F)N1CCC(S)CC1"
]
</SYNTHETIC_EQUIVALENT>
</think>



<ANSWER>CCCC(=O)Cl.O=C(C1CC1)C(c1ccccc1F)N1CCC(S)CC1</ANSWER>

Figure 6: Example of a reasoning process generated by SyntheticRetro.

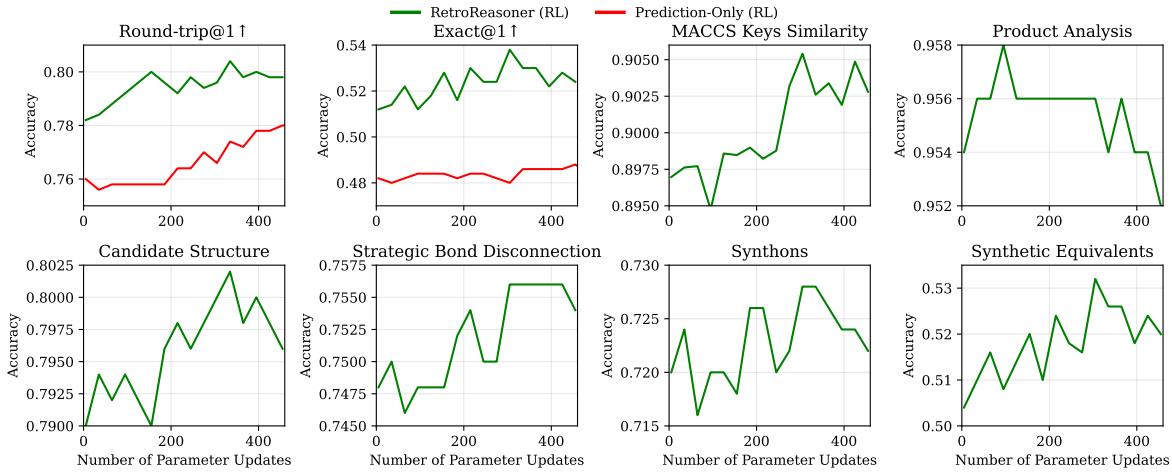


Figure 7: Learning curves for RetroReasoner (RL) and Prediction-Only (RL). The figure shows learning curves for Round-trip@1↑ and Exact@1↑, as well as MACCS Keys fingerprint similarity to the reactants and the accuracy of the structured reasoning steps.

	Base Model	Optimizer	Learning Rate	Gradient Clip	Weight Decay	Warmup Steps	Batch Size	Backend Framework
Prediction-Only (SFT)	Qwen3-8B	AdamW (Loshchilov and Hutter, 2017)	10^{-5}	0.5	0.1	100	64	PyTorch Lightning (Falcon and The PyTorch Lightning team, 2019), DeepSpeed (ZeRO stage1) (Rasley et al., 2020)
RetroReasoner (SFT)	Qwen3-8B	AdamW	10^{-5}	0.5	0.1	100	64	PyTorch Lightning, DeepSpeed (ZeRO stage1)
RetroReasoner (SFT, 1 sample)	Qwen3-8B	AdamW	10^{-5}	0.5	0.1	100	64	PyTorch Lightning, DeepSpeed (ZeRO stage1)
RetroReasoner (SFT, 1.7B)	Qwen3-1.7B	AdamW	10^{-5}	0.5	0.1	100	64	PyTorch Lightning, DeepSpeed (ZeRO stage1)
RetroReasoner (SFT, w/o $\mathcal{L}_{12}, \mathcal{L}_{23}, \mathcal{L}_{34}$)	Qwen3-8B	AdamW	10^{-5}	0.5	0.1	100	64	PyTorch Lightning, DeepSpeed (ZeRO stage1)
RetroReasoner (SFT, w/ only $\mathcal{R}_1$))	Qwen3-8B	AdamW	10^{-5}	0.5	0.1	100	64	PyTorch Lightning, DeepSpeed (ZeRO stage1)
Round-trip Forward Model (0.6B)	Qwen3-0.6B	AdamW	10^{-5}	0.5	0.1	100	64	PyTorch Lightning, DeepSpeed (ZeRO stage1)
Round-trip Forward Model (8B)	Qwen3-8B	AdamW	10^{-5}	0.5	0.1	100	64	PyTorch Lightning, DeepSpeed (ZeRO stage1)

Table 10: Hyperparameters of the models used in the SFT training experiments.

	Base Model	Optimizer	Learning Rate	Gradient Clip	Weight Decay	Warmup Steps	KL loss coefficient	Entropy Coefficient
Prediction-Only (RL)	Qwen3-8B	AdamW	3×10^{-7}	1.0	0.1	10	0.05	0.05
RetroReasoner (RL)	Qwen3-8B	AdamW	3×10^{-7}	1.0	0.1	10	0.05	0.05
RetroReasoner (RL, w/ R^{exact})	Qwen3-8B	AdamW	3×10^{-7}	1.0	0.1	10	0.05	0.05
	Clip Ratio (low, high)	Group Size	Parameter Sync Step	Batch Size	Backend Framework	Rollout Workers	Trainer Workers	Round-trip Workers
Prediction-Only (RL)	0.2	16	5	48	verl (Sheng et al., 2024), FSDP2 (Zhao et al., 2023)	3	4	1
RetroReasoner (RL)	0.2	16	5	48	verl, FSDP2	3	4	1
RetroReasoner (RL, w/ R^{exact})	0.2	16	5	48	verl, FSDP2	3	4	1

Table 11: Hyperparameters of the models used in the RL training experiments.

the 8B model is trained and used for higher accuracy, while the 0.6B model is trained for reward calculation to ensure faster computation. The instances used for evaluation are excluded from the round-trip model training.

D.2 Hyperparameters

The hyperparameters for the 8B and 0.6B models in round-trip training are shown in Table 10.

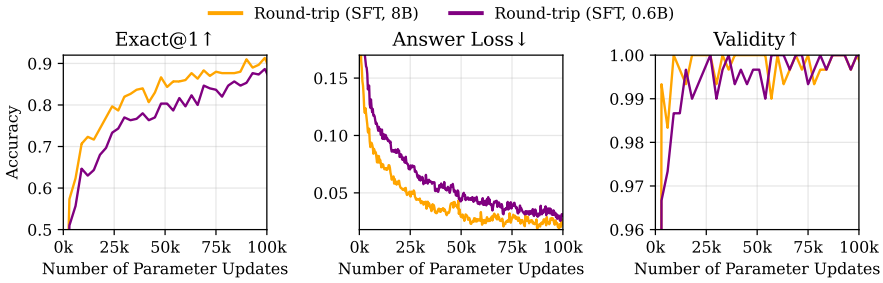


Figure 8: Learning curves for training the round-trip model. The figure shows Exact@1↑, the teacher forcing loss computed on the product SMILES tokens, which is reported as Answer Loss↓, and the validity of the generated product SMILES (Validity↑).

D.3 Learning Curve

The learning curves for the 8B and 0.6B models in round-trip training are shown in Figure 8. In Figure 8, the Exact@1 metric for the round-trip model indicates product prediction accuracy, in contrast to reactant prediction accuracy for RetroReasoner or Prediction-Only. The answer loss represents the average teacher forcing loss computed for the answer tokens, i.e., the product SMILES tokens. Validity indicates whether the predicted product molecule is chemically valid. From the learning curve, it is evident that the 8B model, used for evaluation, shows higher accuracy, achieving a validation accuracy close to 0.91, confirming its suitability for evaluation tasks.

E Details of Monte Carlo Tree Search for Retrosynthesis

E.1 Problem Formulation

Given a target molecule m_{target}, the objective is to find a synthesis route that decomposes m_{target} into commercially available stock molecules in $\mathcal{B}$. At each step, the template-free single-step retrosynthesis proposal model π_θ takes a product molecule m as input and samples k candidate reactant sequences. Each valid sequence is parsed into a reactant set $R_j = \{m_{j,1}, m_{j,2}, \ldots, m_{j,|R_j|}\}$, which corresponds to one retrosynthetic reaction. A synthesis route is considered solved when all of its leaf molecules belong to $\mathcal{B}$.

E.2 Search Tree Structure

The retrosynthesis search tree is represented as an AND-OR tree. A molecule is represented as an OR node, and a retrosynthetic reaction is represented as an AND node. A molecule node is denoted by m, and a reaction node is denoted by r. The function $\text{child}(\cdot)$ denotes the child nodes of a given node. Thus, $\text{child}(m)$ contains reaction nodes, while $\text{child}(r)$ contains molecule nodes. A molecule node is solved if it is already included in the stock set or if at least one of its child reaction nodes is solved. A reaction node is solved only when all of its child molecule nodes are solved. If a non-stock molecule node has no child reaction node, it is treated as unsolved.

E.3 Synthesis Cost Estimation

To estimate the synthesis difficulty of each molecule, the value model proposed in Retro* (Chen et al., 2020) is adopted. To avoid notational conflict with the MCTS node value V, the synthesis cost model is denoted by C. The model takes a 2048-bit Morgan fingerprint with radius 2 as input and outputs a non-negative synthesis cost through a two-layer MLP with a softplus output activation. A larger $C(m)$ indicates that molecule m is more difficult to synthesize, and $C(m) = 0$ is assigned for stock molecules.

E.4 MCTS Algorithm for Retrosynthesis

The AND-OR tree is explored using Monte Carlo Tree Search (MCTS). Each node maintains a visit count N and a cumulative search value V. For a reaction node r, these statistics are denoted by N_r and V_r. Each MCTS iteration consists of four stages: selection, expansion, evaluation, and backpropagation. The procedure is repeated until the root molecule node is solved or the maximum number of iterations T is reached.

Selection Starting from the root molecule node, the tree is traversed until an unexpanded or terminal

molecule node is reached. For the transition from a molecule node to one of its child reaction nodes, the reaction node is selected according to the Upper Confidence Bound (UCB) score. In the WU-UCT setting, the score is computed as

$$\mathrm{UCB}(r)_{\mathrm{WU}} = \frac{V_r}{N_r} + c \cdot p_r \cdot \frac{\sqrt{N_{\text{parent}} + O_{\text{parent}}}}{1 + N_r + O_r}. \tag{5}$$

Here, V_r and N_r denote the cumulative value and visit count of reaction node r, p_r denotes its prior probability, and $c = 1.4$ denotes the exploration constant. The term N_{parent} is the visit count of the parent molecule node. The unobserved counts O_r and O_{parent} track in-flight simulations assigned to the reaction node and its parent molecule node, respectively. For unvisited reaction nodes with $N_r = 0$, the node is selected before applying the UCB score to avoid division by zero. After a reaction node is selected, the transition from the reaction node to a molecule node is made by selecting an unsolved child molecule with the smallest visit count. This encourages balanced exploration among the unsolved reactants of the selected reaction.

Expansion For a leaf molecule node m_ℓ reached during selection, the proposal model π_θ is queried to sample k candidate reactant sequences. Each valid sequence is parsed into a reactant set R_j, and a corresponding reaction node r_j is added as a child of m_ℓ. The reactant molecules in R_j are then added as child molecule nodes of r_j. The prior probability p_{r_j} of each reaction node is computed from the fraction of its reactants that already belong to the stock set, with Laplace smoothing:

$$p_{r_j} = \frac{n_{\mathrm{stock}}(R_j) + 1}{|R_j| + 1}, \tag{6}$$

where $n_{\mathrm{stock}}(R_j)$ is the number of reactants in R_j that belong to $\mathcal{B}$. This prior assigns a higher score to reactions with more immediately purchasable reactants while ensuring that every valid reaction has a positive prior.

Evaluation A value $v(m_\ell) \in [0, 1]$ is estimated for the expanded molecule node m_ℓ. The evaluation combines two signals: the highest reaction prior among the newly expanded reactions and the predicted synthesis cost of the remaining non-stock molecules. The non-stock child molecules generated by the newly expanded reactions are collected, and their average synthesis cost is computed using C. If all generated child molecules are stock molecules, the average synthesis cost is set to zero. The evaluation value is then defined as the larger value between the maximum reaction prior and the inverse-cost score $1/(1 + \bar{C}(m_\ell))$, where $\bar{C}(m_\ell)$ denotes the average synthesis cost of the non-stock child molecules. This optimistic estimate gives a high value when either the immediate stock coverage is high or the predicted synthesis cost of the remaining molecules is low.

Backpropagation After evaluation, all nodes along the selected path are updated in reverse order. For each visited node, the visit count is increased by one. The cumulative value is increased by 1 if the node is solved, and otherwise increased by the evaluation value $v(m_\ell)$. Assigning a value of 1 to solved nodes prioritizes confirmed synthesis routes in subsequent iterations.

E.5 Parallelization via WU-UCT

Calls to the proposal model π_θ are expensive and constitute the main bottleneck of sequential MCTS. To improve efficiency, parallel MCTS based on WU-UCT is used. Each node additionally maintains an unobserved count O, which records the number of in-flight simulations currently assigned to the node but not yet completed. The unobserved count is used during selection to discourage multiple workers from choosing the same path. When a path has already been assigned to a worker, its unobserved count increases, which lowers its exploration priority in subsequent selections. This reduces redundant expansion while preserving the value estimate based only on completed simulations. The master thread performs selection and backpropagation sequentially, while worker processes perform expansion calls to π_θ in parallel. When a selected path is dispatched to a worker, the unobserved count of every node on the path is increased. When the worker returns the expansion result, the corresponding unobserved counts are decreased, and the visit counts and cumulative values are updated using the evaluated value. This enables multiple proposal-model calls to run in parallel without repeatedly expanding the same high-priority path.

F Experiment Details

F.1 Metric Calculation Details

In this section, the calculation methods for the metrics used in the main text are described in detail, along with the corresponding mathematical formulas. Let the input product be denoted as $\mathbf{x}$, the ground-truth reactant as $\mathbf{y}$, the top-1 greedy predicted reactant as $\hat{\mathbf{y}}^{(GR)}$, and the K sampling predicted reactants as $\{\hat{\mathbf{y}}^{(1)}, \hat{\mathbf{y}}^{(2)}, \ldots, \hat{\mathbf{y}}^{(K)}\}$. Throughout this section, $\mathbb{I}(a, b)$ denotes molecular identity based on standard InChI matching, as defined in Equation (3). For an evaluation dataset consisting of N instances $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{N}$, each metric is calculated as follows.

F.1.1 Greedy Metrics

The Exact@1↑ metric is computed by averaging the indicator function across all instances to check if the top-1 greedy predicted reactant exactly matches the ground-truth reactant:

$$\text{Exact@1}\uparrow = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}\left(\hat{\mathbf{y}}_i^{(GR)}, \mathbf{y}_i\right). \quad (7)$$

The Round-trip@1↑ metric is calculated by averaging the indicator function to check if the top-1 greedy predicted reactant, when passed through the round-trip model f_ϕ, reconstructs the original product:

$$\text{Round-trip@1}\uparrow = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}\left(\mathbf{x}_i, f_\phi(\hat{\mathbf{y}}_i^{(GR)})\right). \quad (8)$$

F.1.2 Sampling Metrics

The Exact@100↑ metric is calculated by averaging across all instances, where at least one of the K sampled predictions matches the ground-truth reactant:

$$\text{Exact@100}\uparrow = \frac{1}{N}\sum_{i=1}^{N}\max_{1\le k\le K}\mathbb{I}\left(\hat{\mathbf{y}}_i^{(k)}, \mathbf{y}_i\right). \quad (9)$$

The Round-trip@100↑ metric is computed by averaging across all instances, where at least one of the K sampled predictions, when passed through the round-trip model f_ϕ, successfully reconstructs the original product:

$$\text{Round-trip@100}\uparrow = \frac{1}{N}\sum_{i=1}^{N}\max_{1\le k\le K}\mathbb{I}\left(\mathbf{x}_i, f_\phi(\hat{\mathbf{y}}_i^{(k)})\right). \quad (10)$$

The Feasible Ratio↑ metric measures the proportion of the K sampled predictions that pass the round-trip validation, averaged across all instances:

$$\text{Feasible Ratio}\uparrow = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{K}\sum_{k=1}^{K}\mathbb{I}\left(\mathbf{x}_i, f_\phi(\hat{\mathbf{y}}_i^{(k)})\right)\right). \quad (11)$$

The Template Diversity↑ metric is calculated by counting the number of distinct reaction templates for the K sampled predictions that successfully pass the round-trip validation, and then averaging this count across all instances:

$$\text{Template Diversity}\uparrow = \frac{1}{N}\sum_{i=1}^{N}|S_i|, \quad (12)$$

where

$$S_i = \left\{T\left(\hat{\mathbf{y}}_i^{(k)}, \mathbf{x}_i\right) : \mathbb{I}\left(\mathbf{x}_i, f_\phi(\hat{\mathbf{y}}_i^{(k)})\right) = 1, \; 1 \le k \le K\right\}, \quad (13)$$

Here, T extracts a canonical reaction template from a predicted-reactant–target-product pair.

F.1.3 Multi-step Retrosynthesis Metrics

Three complementary metrics introduced in PaRoutes (Genheden and Bjerrum, 2022) are adopted to evaluate the performance of retrosynthesis search. Let $\mathcal{B}$ denote the purchasable building-block stock, and let $\mathcal{R}_i$ denote the set of complete synthesis routes extracted for the i-th target molecule $m_{\text{target}}^{(i)}$ after search. A route is considered complete if all of its leaf molecules belong to $\mathcal{B}$. The Solved Targets↑ metric is calculated by counting the number of target molecules for which the search finds at least one complete synthesis route:

$$\text{Solved Targets}\uparrow = \sum_{i=1}^{N}\mathbb{I}\left(\text{solved}(m_{\text{target}}^{(i)}), \text{True}\right), \quad (14)$$

where $\text{solved}(m_{\text{target}}^{(i)})$ follows the recursive solved-status definition in Appendix E, and evaluates to true when the target molecule can be recursively decomposed into purchasable molecules in $\mathcal{B}$.

The Routes Extracted↑ metric measures the total number of complete synthesis routes discovered by the search across all targets. For each

target, complete routes are enumerated from the constructed AND-OR tree after search termination:

$$\text{Routes Extracted}\uparrow = \sum_{i=1}^{N} |\mathcal{R}_i|, \qquad (15)$$

where $\mathcal{R}_i$ is the set of complete routes extracted for the i-th target molecule. If no complete route is found for a target, then $|\mathcal{R}_i| = 0$. This metric reflects the overall number of complete synthesis routes discovered by the search.

The Number of Clusters↑ metric measures the total number of structurally distinct route clusters discovered across all targets. For each target molecule with at least three extracted routes ($N_i \geq 3$), routes are clustered based on their pairwise structural distances, which are computed using the LSTM-based route distance model from PaRoutes (Genheden and Bjerrum, 2022). Targets with fewer than three routes are excluded from the cluster-count summation.

The clustering procedure proceeds as follows. First, the maximum number of candidate clusters is set to $\lceil N_i/d_{\min} \rceil$, where $d_{\min} = 2$ is the minimum cluster density hyperparameter. Then, the optimal number of clusters k_i^* is selected from the range $[2, \min(K_i^{\max}, N_i - 1)]$ by maximizing the mean Silhouette score over all routes of the target. The Silhouette coefficient of each route is defined in the standard way as $(b - a)/\max(a, b)$, where a is the mean distance from the route to other routes in the same cluster, and b is the smallest mean distance from the route to routes in any other cluster. Once the optimal partition is obtained, the number of clusters n_i for the i-th target is given by the number of distinct cluster labels.

The final Number of Clusters↑ metric is computed by summing the number of clusters over all targets for which clustering is performed:

$$\text{Number of Clusters}\uparrow = \sum_{i:N_i \geq 3} n_i. \qquad (16)$$

A higher value indicates that the search discovers a larger number of structurally distinct synthesis strategies across the benchmark.

F.2 Evaluation Dataset Generation

F.2.1 In-distribution Evaluation Dataset

The in-distribution dataset consists of 500 instances, which are specifically derived from the ORDerly retrosynthesis benchmark test dataset[1]. Instances that overlap with the training datasets from LlaSMol and Mol-Instructions are excluded from the ORDerly benchmark test dataset. Overlap is determined by checking if both the reactant and product molecules are the same. Instances are also excluded if LocalMapper fails to make a prediction or if there are multiple reactants for a single product in the entire ORDerly training dataset. After exclusions, 500 instances are randomly selected from the remaining ones to be used as the in-distribution evaluation dataset.

F.2.2 Rare Template and Rare Atom/Token Evaluation Dataset

Hard instances are created by constructing a dataset that includes rare reaction templates and rare tokens. First, canonical reaction templates are extracted from the entire ORDerly train and test sets, and the count of each reaction template is recorded. Additionally, 2-grams and 3-grams are extracted from the tokenized product SMILES, and the count of each 2-gram and 3-gram is stored. The distribution of canonical reaction templates is shown in Figure 9, while the distributions of 2-grams and 3-grams in product SMILES are shown in Figures 10 and 11, respectively. Each distribution displays the top 15 and bottom 15 most frequent entries. The most common reaction template is `[N:2].O-[C:1]=O»O=[C:1]-[N:2]`, which represents the condensation reaction between amines and carboxylic acids, a common reaction in organic chemistry. On the other hand, examining the bottom reaction templates reveals longer and more complex reactions. When examining the top distributions of 2-grams and 3-grams, it is observed that frequently occurring atoms such as carbon (C) and oxygen (O) in organic molecules are common, while the bottom distributions include elements that are relatively rare in organic molecules, such as selenium (Se) and titanium (Ti).

Rare Reaction Template Instances The rare reaction template evaluation dataset is based on this distribution. From the ORDerly test benchmark, 50 instances are randomly selected whose reaction-template frequency is between 1 and 3, and 50 instances are randomly selected whose reaction-template frequency is between 4 and 6, resulting in a total of 100 instances.

[1] https://figshare.com/articles/dataset/ORDerly_chemical_reactions_condition_benchmarks/23298467

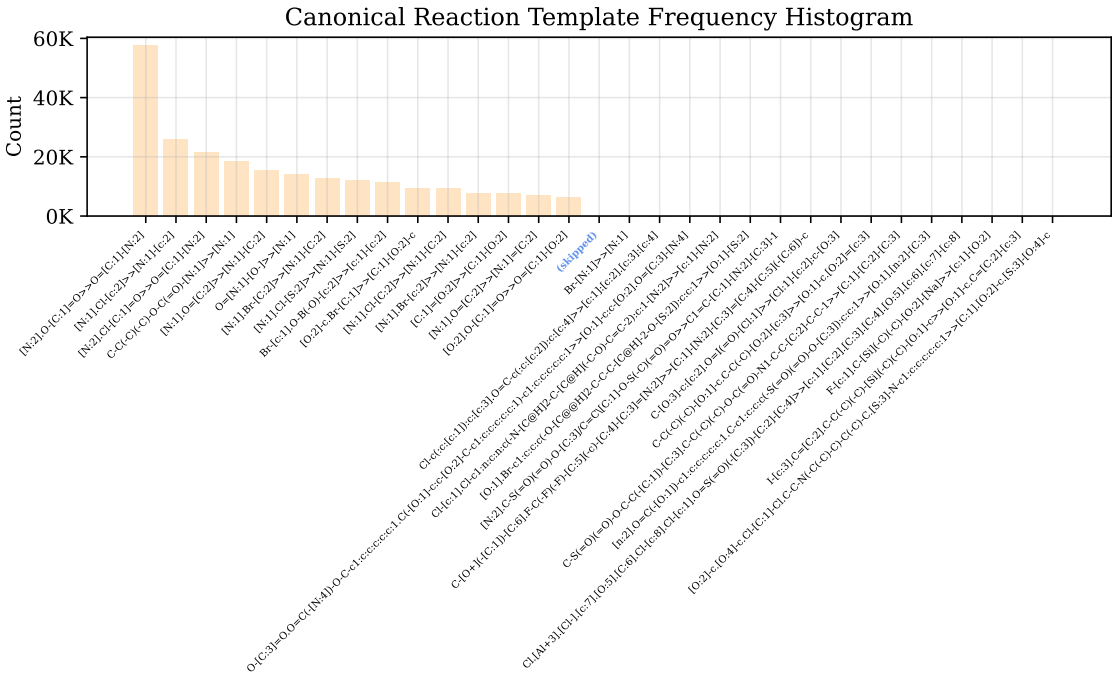


Figure 9: Histogram of the canonical reaction template distribution extracted from all instances in ORDerly. The x-axis lists canonical reaction templates. The top 15 and bottom 15 templates are shown, and templates with intermediate frequencies are skipped.

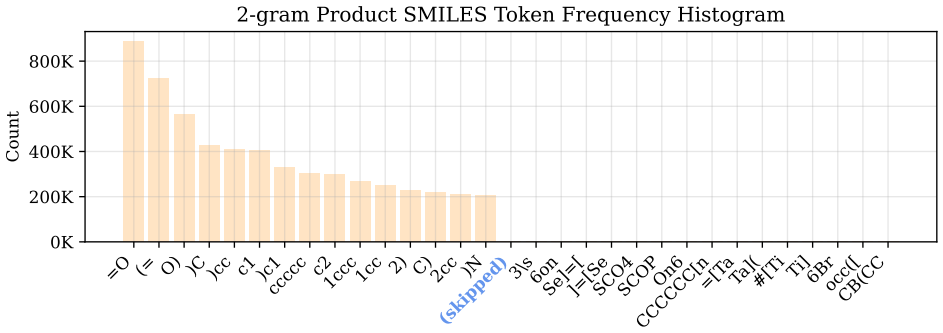


Figure 10: Histogram of the distribution of 2-gram tokens in product SMILES. The top 15 and bottom 15 2-gram tokens are shown, and tokens with intermediate frequencies are skipped.

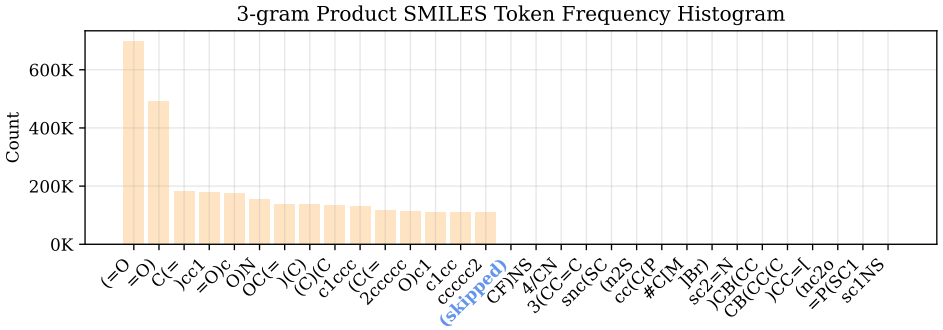


Figure 11: Histogram of the distribution of 3-gram tokens in product SMILES. The top 15 and bottom 15 3-gram tokens are shown, and tokens with intermediate frequencies are skipped.

Rare Atom/Token Instances Rare atom/token instances are selected by scoring each instance based on the presence of rare tokens in the product SMILES. The top 50 instances based on 2-grams and the top 50 instances based on 3-grams are combined to form a total of 100 instances. Specifically, let the entire n-gram token set be V, and let the count set of each token be $\{c(t)\}_{t\in V}$. The total token count is $N = \sum_{v\in V} c(v)$, and the occurrence ratio for each token t is $p_a(t) = \frac{c(t)}{N}$. The score for each token is defined based on the occurrence ratio as $s(t) = -\log p_a(t)$. If a product SMILES is composed of $\mathbf{x} = (t_1, \ldots, t_L)$, the instance score $S(x)$ is calculated as $S(x) = \frac{1}{L}\sum_{i=1}^{L} s(t_i)$, and instances are selected based on this score.

F.3 Sampling Hyperparameters

The hyperparameters used for reactant sampling in the experiment are summarized in Table 12.

F.4 Training Details of Expert Models

RetroSynFlow Training RetroSynFlow consists of two stages: reaction center prediction and synthon completion. For the reaction center prediction model, the publicly available G2G (Shi et al., 2020) architecture of RetroSynFlow is used without modification and trained with AdamW using a learning rate of 1×10^{-4}. The model is trained for 100 epochs with a total batch size of 128, and the resulting checkpoint is used for inference. For the synthon completion model, the flow matching model is trained with distributed data parallelism on 8 NVIDIA H100 GPUs. The base model follows the 5-layer Graph Transformer architecture of RetroSynFlow. AdamW is used with a constant learning rate schedule, where the learning rate is 8×10^{-4}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, weight decay is 10^{-12}, and AMSGrad is enabled. The model is trained for 500 epochs with a total batch size of 256 across 8 GPUs.

During inference, retrosynthesis is performed through a two-stage pipeline. First, the trained reaction center identification model predicts the bonds to be disconnected in the product molecular graph and generates the top-2 synthon candidates. Then, for each synthon candidate, the synthon completion model restores the leaving groups by performing 100-step Euler integration with a linear time schedule. A total of 100 reactant candidates are generated for each product, with 70 candidates sampled from the top-1 synthon and 30 candidates sampled from the top-2 synthon. At each integration step, the model takes the current noisy graph, the synthon, and the product context as input and predicts the final reactant. The vector field computed from this prediction is used to update the probability distribution, and the discrete graph state is transitioned through categorical sampling. Atoms that are already fixed in the synthon are kept unchanged throughout the entire generation process.

G2S-HCVAE Training G2S-HCVAE is performed end-to-end as a single Hierarchical Conditional VAE plugged into a Graph2SMILES (Tu and Coley, 2022) backbone, with a 256-dimensional continuous latent variable z and a discrete latent variable c over $K = 10$ reaction classes. The decoder is trained with teacher forcing to reconstruct reactant SMILES, and the loss is the ELBO combining a token-level cross-entropy reconstruction term (label smoothing 0.1) and the KL terms for z and c, with a 10-epoch KL warmup. Optimization uses AdamW with the original Transformer inverse-square-root learning rate schedule (peak $\mathrm{lr} = 1.0$, 8000 warmup steps) and token-level batching (batch size 4096, token limit $80{,}000$, gradient accumulation of 2). Checkpointing maintains a queue of the last 10 saved models, and weight-averaging over the top 5 is applied automatically every 10 saves. Training was run for 545 epochs on 8 NVIDIA B200 GPUs under a fixed random seed of 17.

During inference, only the product is available, so the Recognition Network is discarded and the Prior Network alone seeds the decoder through a latent-seeded beam search. For each beam, an independent continuous latent is sampled and mapped to a discrete reaction-class token, yielding distinct class tokens that are injected as the initial token of each beam. Decoding then proceeds as standard autoregressive Transformer beam search, with candidates ranked by accumulated log-probability. This seeding fragments the beam across different reaction modes before decoding begins, mitigating the mode-collapse of deterministic Transformer decoding. The beam size is 10 for single-step retrosynthesis, 5 for Retro*-based multistep search, and 50 for DirectMultiStep on PaRoutes returning the top-5 routes.

	Temperature	Top-p	Max new tokens	Reasoning Effort	Enable Thinking
Molecular Prediction LLMs					
LlaSMol	1.0	1.0	256	-	-
Mol-Instructions	1.0	1.0	256	-	-
BioT5+	1.0	1.0	-	-	-
PRESTO	1.0	1.0	-	-	-
Mol-LLM	1.0	1.0	-	-	-
Molecular Reasoning LLMs					
Chem-R	1.0	1.0	-	-	-
ChemDFM	1.0	1.0	-	-	-
ether0	1.0	1.0	-	-	-
General Purpose LLMs					
OpenAI-o3	-	1.0	2048	low	-
GPT-5-mini	-	1.0	2048	medium	-
GPT-oss-120B	1.0	0.95	-	high	-
Qwen3-8B	1.2	0.95	-	-	O
Qwen3-235B-A22B	1.2	0.95	-	-	O
Expert Models					
RetroSynFlow	N/A	N/A	N/A	N/A	N/A
G2S-HCVAE	1.0	N/A	N/A	N/A	N/A
Prediction-Only (SFT)	1.2	1.0	500	-	-
Prediction-Only (RL)	1.2	1.0	500	-	-
RetroReasoner (SFT)	1.2	1.0	3000	-	-
RetroReasoner (SFT, w/o $\mathcal{L}_{12}, \mathcal{L}_{23}, \mathcal{L}_{34}$)	1.2	1.0	3000	-	-
RetroReasoner (SFT, w/ only $\mathcal{R}_3, \mathcal{R}_4$)	1.2	1.0	3000	-	-
RetroReasoner (SFT, w/ only $\mathcal{R}_1$)	1.2	1.0	3000	-	-
RetroReasoner (RL)	1.2	1.0	3000	-	-
RetroReasoner (RL, w/ R^{exact})	1.2	1.0	3000	-	-

Table 12: Summary table of the sampling parameters used to compute the sampling metrics.

	Exact@1↑	Round-trip@1↑	Exact@100↑	Round-trip@100↑	Feasible Ratio↑	Template Diversity↑
RetroReasoner (SFT)	**0.512**	**0.812**	**0.734**	0.944	0.765	**3.898**
RetroReasoner (SFT, w/o $\mathcal{L}_{12}, \mathcal{L}_{23}, \mathcal{L}_{34}$)	0.494	0.806	0.712	0.944	0.779	2.656
RetroReasoner (SFT, w/ only $\mathcal{R}_3, \mathcal{R}_4$)	0.498	0.810	0.706	0.926	0.784	2.402
RetroReasoner (SFT, w/ only $\mathcal{R}_1$)	0.498	0.806	0.708	**0.948**	**0.777**	2.862

Table 13: Performance comparison of SFT training when linking texts $\mathcal{L}_{12}, \mathcal{L}_{23}, \mathcal{L}_{34}$ are not used and when only the product analysis step $\mathcal{R}_1$ is used as structured reasoning. The best performance is highlighted in **bold**.

G Additional Experimental Results

G.1 Ablation Studies

G.1.1 Reasoning Strategy

Retrosynthesis prediction needs strategic reasoning. Table 13 reports SFT training results for RetroReasoner and an ablation model trained using only $\mathcal{R}_1$, which performs product analysis as its only reasoning step. Using only $\mathcal{R}_1$ can be regarded as representative of molecular reasoning LLMs. Comparing the first and fourth rows, Round-trip@1↑, Round-trip@100↑, and Feasible Ratio↑ are similar, whereas the full RetroReasoner achieves clearly higher Exact@1↑, Exact@100↑, and Template Diversity↑. This suggests that complex tasks such as retrosynthesis prediction require strategic reasoning similar to that used by chemists.

G.1.2 Effect of Linking Text

Linking text helps improve prediction accuracy and produce more diverse reactant suggestions. Table 13 also compares the SFT training performance of RetroReasoner when only structured reasoning steps $\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \mathcal{R}_4$ are used, excluding linking texts $\mathcal{L}_{12}, \mathcal{L}_{23}, \mathcal{L}_{34}$. The results show similar Round-trip@1↑ and Round-trip@100↑, while the full model achieves higher Exact@1↑, Exact@100↑, and Template Diversity↑ than the variant without linking texts. This suggests that linking texts that connect the structured reasoning steps is essential and enables the model to handle a broader feasible reactant space through more diverse reasoning paths.

G.1.3 Diversity of Linking Text

Training diverse reasoning paths improves train-test generalization. Figure 12 presents the learn-

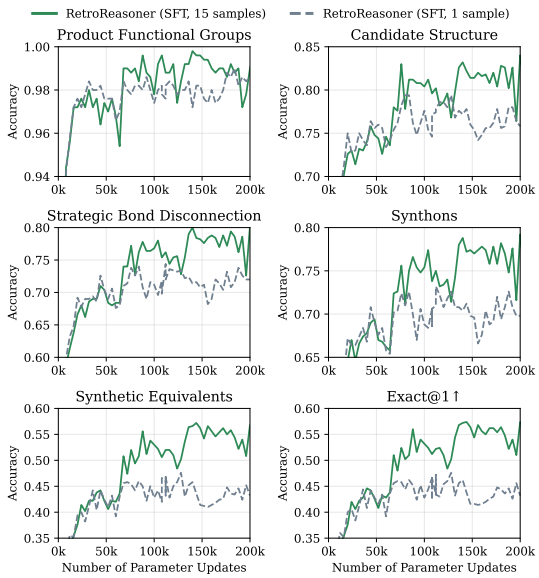


Figure 12: Learning curve comparing the effect of the number of linking texts used during SFT. In the default setting, 15 linking texts are generated per instance, with a different linking text used in each epoch. This is compared to the case where only one linking text ($n = 1$) is used consistently across all epochs. The accuracy of each structured reasoning step and the Exact@1↑ metric for reactant prediction are shown. The x-axis represents the number of parameter updates.

ing curves comparing the use of $n = 1$ linking text per instance, where the same linking text is used in each epoch during SFT training, with $n = 15$ different linking texts per instance, where a different linking text is used in each epoch. The changes in Exact@1↑ and accuracy for structured reasoning steps are reflected in the corresponding learning curve. The results clearly show a performance difference in the middle of the learning process. This indicates that increasing the diversity of reasoning paths, by using different linking texts each epoch, is essential and improves the generalization from train to test.

G.1.4 Model Size

Larger models are essential for learning the chemist's reasoning strategy. Figure 13 compares the learning curves of two different model sizes: Qwen3-8B, the base model for RetroReasoner, and Qwen3-1.7B. While both models show improvement during training, a clear performance gap emerges toward the end, indicating that a sufficiently large model size is crucial for effectively learning a reasoning model that mimics the chemist's strategy.

Reasoning step	RetroReasoner (SFT)	RetroReasoner (RL)
$\mathcal{R}_1$ - product info	<0.001	<0.001
$\mathcal{R}_1$ - atom-mapped smiles	<0.001	<0.001
$\mathcal{R}_1$ - functional groups	<0.001	<0.001
$\mathcal{L}_{12}$	0.822	0.857
$\mathcal{R}_2$ - candidate structure	0.016	0.009
$\mathcal{L}_{23}$	0.619	0.672
$\mathcal{R}_3$ - bond disconnection	0.001	<0.001
$\mathcal{R}_3$ - synthons	0.001	0.001
$\mathcal{L}_{34}$	0.714	0.759
$\mathcal{R}_4$ - synthetic equivalents	0.010	0.005

Table 14: Average token entropy of each reasoning component for RetroReasoner before and after RL. Structured reasoning components exhibit substantially lower entropy than the free-form linking texts.

G.1.5 Effect of Sampling Hyperparameters

Sampling parameters control the diversity and feasibility of reactant proposals. Increasing the sampling temperature allows lower-probability tokens to be selected more frequently, which can increase output diversity but also destabilize generation. Figure 14 shows that the Invalid Ratio↓ increases sharply at high temperatures, accompanied by a decrease in Feasible Ratio↑. Based on this trade-off, a temperature of 1.2 is used as the default setting for the sampling-based evaluation.

The diversity reduction after RL is associated with increased uncertainty in the linking texts. Table 14 reports the average token entropy of each reasoning component for RetroReasoner before and after RL. The structured reasoning components remain highly concentrated, with near-zero entropy for most fields. In contrast, the free-form linking texts $\mathcal{L}_{12}$, $\mathcal{L}_{23}$, and $\mathcal{L}_{34}$ exhibit substantially higher entropy, which further increases after RL from 0.822 to 0.857, 0.619 to 0.672, and 0.714 to 0.759, respectively. At higher temperatures, this high-entropy tail increases the probability of sampling low-probability out-of-distribution tokens, including unrelated foreign-language characters and Unicode symbols. Such tokens can disrupt subsequent reasoning and sharply reduce the feasibility of the generated reactants.

To control this decoding instability, min-p decoding with $p = 0.05$ is applied. At each decoding step, tokens with probabilities below 0.05 times the maximum token probability are filtered before sampling. Sampling temperature is then varied from 1.2 to 3.0 in increments of 0.1 to explore different feasibility–diversity operating points. As shown in Figure 3, RetroReasoner (RL) consistently achieves higher Template Diversity↑ than RetroReasoner (SFT) at comparable Feasible Ra-

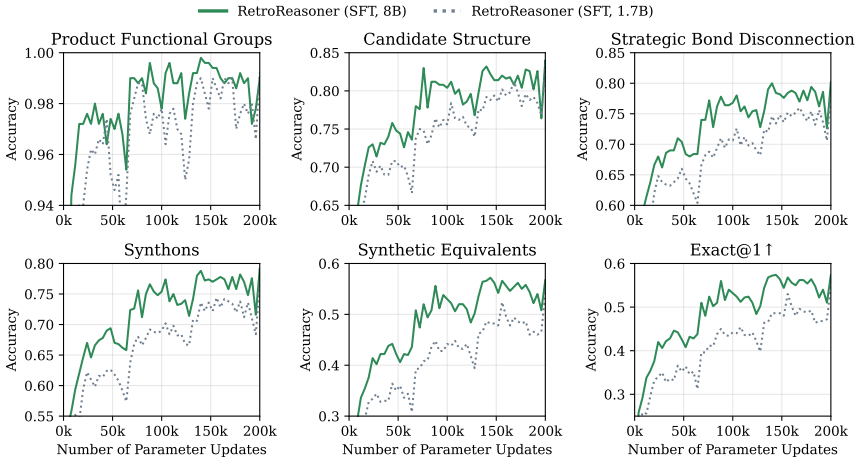


Figure 13: Learning curve comparing different model sizes (1.7B, 8B) of the start model, Qwen3. The accuracy of each structured reasoning step and the Exact@1 metric for reactant prediction are shown.

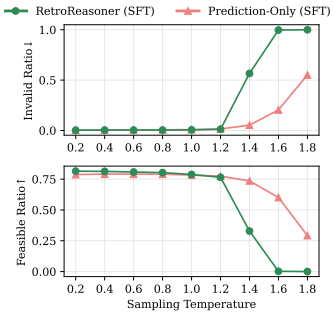


Figure 14: Changes in Feasible Ratio↑ and Invalid Ratio↓ across sampling temperatures. Invalid Ratio is the average, over all instances, of the fraction of sampled reactants that do not satisfy SMILES format.

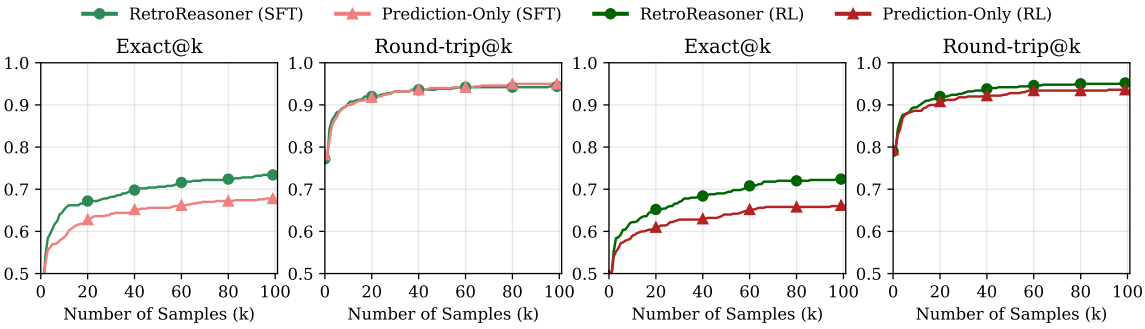


Figure 15: Changes in Exact@k↑ and Round-trip@k↑ performance of the SFT and RL trained models for RetroReasoner and Prediction-Only across different numbers of samples k.

tio↑ levels, forming a more favorable empirical Pareto frontier. This result indicates that the lower Template Diversity↑ observed under the default decoding configuration is sensitive to the decoding operating point rather than representing a uniform reduction in diversity after RL.

Finally, Figure 15 reports Exact@k↑ and Round-trip@k↑ as the number of sampled candidates increases. Both metrics improve with additional samples and gradually saturate, showing the benefit of sampling multiple reactant candidates for covering alternative retrosynthetic solutions.

	Exact@1↑	Round-trip@1↑
Original - RetroReasoner (RL)	0.526	0.826
$\mathcal{R}_1$-corrupt	0.018	0.028
$\mathcal{R}_2$-corrupt	0.390	0.660
$\mathcal{R}_3$-corrupt	0.014	0.074
$\mathcal{R}_4$-corrupt	0.000	0.000
$\mathcal{L}_{12}$-corrupt	0.212	0.532
$\mathcal{L}_{23}$-corrupt	0.464	0.794
$\mathcal{L}_{34}$-corrupt	0.468	0.774

Table 15: Comparison of performance changes when each component of RetroReasoner's reasoning is corrupted.

G.2 Effect of Reasoning Components

All components that make up RetroReasoner's reasoning play an important role in reactant prediction. To analyze whether each component of the structured reasoning process contributes to the final reactant prediction, a step-corruption experiment is conducted. RetroReasoner follows the reasoning structure $\mathcal{R}_1 \rightarrow \mathcal{L}_{12} \rightarrow \mathcal{R}_2 \rightarrow \mathcal{L}_{23} \rightarrow \mathcal{R}_3 \rightarrow \mathcal{L}_{34} \rightarrow \mathcal{R}_4 \rightarrow$ reactant prediction. In each

corruption setting, one intermediate reasoning component is replaced with the corresponding component generated for another instance. Starting from the corrupted component, RetroReasoner then generates the remaining reasoning sequence and the final reactant prediction. Table 15 reports the results. Original denotes the standard inference procedure without any perturbation, i.e., RetroReasoner (RL), whereas {X}-corrupt denotes the setting in which component {X} is randomly replaced by the corresponding component from another RetroReasoner-generated instance. As shown in Table 15, corrupting any reasoning component consistently leads to a substantial decrease in final prediction performance compared with the original inference procedure. This result indicates that each component in the structured reasoning chain plays a functional role in deriving the final reactant prediction. In other words, the generated reasoning is not merely an explanatory text produced after the fact; rather, it serves as an intermediate computational process that guides subsequent reasoning steps and ultimately affects the predicted reactants.

G.3 Effect of Forward-model Error on RL

Although the error of the forward model affects performance after RL training to some extent, it is not significant. To analyze how forward-model accuracy affects RL training, RetroReasoner is trained with forward models of different scales. The main experiments use a 0.6B forward model during RL training for inference efficiency, while evaluation is performed with a more accurate 8B forward model. The 0.6B and 8B forward models achieve approximately 87% and 91% accuracy on the forward-model test set, respectively. Table 16 reports the results when the 8B forward model is also used during RL training. As shown in Table 16, replacing the 0.6B forward model with the more accurate 8B model during RL training produces only slight changes in the final performance. This suggests that forward-model errors introduce some noise into the RL reward signal, but their effect on the learned policy is limited. Therefore, the 0.6B forward model offers a practical trade-off between reward reliability and training efficiency.

G.4 Effect of Removing Early Reasoning Steps

All reasoning components must be used together to enable more feasible and diverse reactant predictions. To analyze the contribution of the full structured reasoning process, a variant using only $\mathcal{R}_3$ and $\mathcal{R}_4$ is evaluated. This setting is the closest among the variants to conventional semi-template retrosynthesis pipelines, as it primarily relies on reaction-center-level reasoning and reactant prediction without the earlier reasoning components or linking texts. Table 13 reports the results. As shown in Table 13, the variant using only $\mathcal{R}_3$ and $\mathcal{R}_4$ performs worse than the full model, particularly in sampling performance and diversity-related metrics. This indicates that the earlier reasoning components, $\mathcal{R}_1$ and $\mathcal{R}_2$, as well as the linking texts between reasoning steps, provide meaningful guidance for the overall prediction process. The results suggest that RetroReasoner benefits from the complete structured reasoning trajectory, rather than relying only on the later reaction-centered reasoning steps.

G.5 Effect of Batch Inference

Throughput can be significantly increased when performing predictive processing on a batch basis. Inference cost is measured using 100 test examples under both sequential serving and batched serving with vLLM. Table 17 reports the inference cost comparison. The results show that $\mathcal{R}_1$ and $\mathcal{R}_2$ account for a large portion of the generated output length, indicating that the early reasoning stages contribute substantially to the overall inference cost. As shown in Table 17, LLM-based structured reasoning introduces a nontrivial computational cost compared with more compact prediction-only inference. However, batched serving with vLLM substantially improves throughput, reducing the practical inference overhead. These results indicate that although RetroReasoner requires longer generation due to its explicit reasoning process, efficient serving strategies can mitigate the computational cost in practical deployment.

G.6 Confidence Interval Measurement

RetroReasoner demonstrates statistically significantly more feasible and diverse predictive performance than Prediction-Only. To assess whether the observed performance gains are robust to the choice of evaluation samples, additional experiments are conducted with repeatedly resampled test sets. Specifically, ten main in-distribution test sets are constructed with 500 instances each, along with ten rare-template test sets and ten rare atom/token test sets with 100 instances each. Table 18 reports the mean and standard deviation

	Exact@1↑	Round-trip@1↑	Exact@100↑	Round-trip@100↑	Feasible Ratio↑	Template Diversity↑
RetroReasoner (RL w/ 87% round-trip)	0.526	0.826	0.724	0.952	0.786	3.186
RetroReasoner (RL w/ 91% round-trip)	0.538	0.834	0.728	0.942	0.793	3.172

Table 16: Comparison of RL training performance based on the accuracy of the round-trip model used for RetroReasoner training. The 87% round-trip model is 0.6B, and the 91% round-trip model is 8B.

			Batch 1		**Batch 100**	
	Avg input tokens	Avg output tokens	Total wall time (s)	Throughput (samples/s)	Total wall time (s)	Throughput (samples/s)
RetroReasoner (SFT)	67.4	953.8	733	0.14	17	5.91
RetroReasoner (RL)	67.4	955.0	735	0.14	17	5.94

Table 17: Comparison of inference speed when serving RetroReasoner as vLLM and predicting one instance 100 times consecutively versus processing 100 requests simultaneously.

	Exact@1↑	Round-trip@1↑	Exact@100↑	Round-trip@100↑	Feasible Ratio↑	Template Diversity↑
Main Evaluation						
Prediction-Only (SFT)	0.475±0.008	0.790±0.007	0.669±0.007	0.940±0.007	0.774±0.005	2.477±0.044
Prediction-Only (RL)	0.480±0.008	0.805±0.008	0.655±0.008	0.935±0.006	0.788±0.006	2.213±0.033
RetroReasoner (SFT)	0.510±0.009	0.819±0.009	**0.746±0.008**	**0.957±0.007**	0.770±0.006	**3.986±0.090**
RetroReasoner (RL)	**0.519±0.008**	**0.834±0.008**	0.725±0.009	0.951±0.006	**0.792±0.006**	3.153±0.069
Rare Template						
Prediction-Only (SFT)	0.494±0.059	0.766±0.034	0.696±0.056	0.926±0.022	0.738±0.029	2.282±0.189
Prediction-Only (RL)	0.494±0.055	0.774±0.025	0.687±0.059	0.927±0.021	0.751±0.028	2.096±0.176
RetroReasoner (SFT)	0.527±0.068	0.808±0.031	**0.783±0.039**	**0.962±0.021**	0.742±0.028	**4.010±0.355**
RetroReasoner (RL)	**0.529±0.071**	**0.813±0.025**	0.764±0.048	0.955±0.021	**0.766±0.029**	3.375±0.320
Rare Atom/Token						
Prediction-Only (SFT)	0.147±0.030	0.713±0.050	0.310±0.030	0.912±0.042	0.639±0.041	3.111±0.298
Prediction-Only (RL)	0.147±0.030	**0.714±0.049**	0.279±0.041	0.910±0.027	**0.652±0.040**	2.803±0.276
RetroReasoner (SFT)	**0.155±0.021**	0.682±0.035	**0.432±0.030**	**0.934±0.021**	0.621±0.032	**5.400±0.577**
RetroReasoner (RL)	0.137±0.022	0.694±0.043	0.403±0.040	0.931±0.022	0.646±0.029	4.470±0.364

Table 18: Performance comparison including confidence intervals for each evaluation dataset of RetroReasoner and Prediction-Only. The mean and standard deviation of each metric are shown.

across these resampled test sets. As shown in Table 18, the performance gains remain consistent across different resampled evaluation sets, particularly in the main in-distribution setting. The relatively small standard deviations indicate that the improvements are not driven by a specific test-set sample, but are stable across multiple subsets drawn from the eligible test data. These results further support the reliability of the reported performance improvements.

G.7 Effects of Using an External Forward Model

Evaluations using an external round-trip model are similar, and the use of a round-trip model provides a learning signal based on actual reactions. To assess whether the observed gains are specific to the internally trained round-trip model, Molecular Transformer (MT) (Schwaller et al., 2019) is used as an independent validator that is not trained on ORDerly. Table 5 reports Round-trip@1↑ and Round-trip@100↑ under both validators. A similar trend is observed when using either the internal round-trip model or MT, indicating that the improvements are not specific to a single validator. In addition to round-trip validation, the chemical feasibility of the predicted reactions is examined using reaction templates extracted from an external USPTO reaction dataset. For each prediction, a reaction of the form {*predicted reactants*} → {*target product*} is constructed, and its reaction template is compared against a template set extracted from USPTO. This provides an additional criterion for evaluating whether the predicted reactants correspond to known chemically feasible forward reaction patterns. The analysis is conducted on the full 500-instance main in-distribution test set. For each target product, 10 predicted reactant candidates are sampled, and duplicate predictions within each instance are removed. Among these candidates, only the predictions that pass round-trip validation are retained, i.e., cases where the predicted reactants reconstruct the original target product through the round-trip model. For the resulting reactions, Table 19 re-

	N (data count)	In USPTO↑	USPTO Ratio↑	Unique USPTO Template↑
Our round-trip model				
Prediction-Only (RL)	690	458	66.4%	150
RetroReasoner (RL)	805	**542**	**67.3%**	**176**
MolecularTransformer				
Prediction-Only (RL)	454	313	68.9%	112
RetroReasoner (RL)	550	**387**	**70.4%**	**143**

Table 19: Comparison of the degree to which reaction templates extracted from reactant-product pairs match the reaction template set in the actual reaction dataset, the USPTO dataset, when predicting the product using a round-trip model.

ports three template-based feasibility metrics: the number of predicted reaction templates included in the USPTO template set (In USPTO↑), the ratio of such templates among round-trip-valid predictions (USPTO Ratio↑), and the number of distinct USPTO-covered templates (Unique USPTO Template↑). The comparison is performed between RetroReasoner (RL) and Prediction-Only (RL), using both the internal round-trip model and MT as validators. As shown in Table 19, RetroReasoner achieves higher In USPTO↑ and USPTO Ratio↑ than Prediction-Only, indicating that its round-trip-valid predictions more frequently correspond to reaction patterns observed in the external USPTO-derived template set. RetroReasoner also obtains a higher Unique USPTO Template↑, suggesting that it covers a broader range of chemically feasible reaction templates. Notably, among the predicted reactants judged as valid by round-trip validation, nearly 70% match USPTO-covered reaction patterns capable of producing the target product. The same overall trend is observed when MT is used as the independent round-trip validator. These results suggest that the RL signal provided by round-trip validation is meaningfully aligned with chemically feasible reaction patterns, rather than merely exploiting specificity of the internally trained forward model. The agreement between the internal validator, MT, and the USPTO template-based analysis provides additional evidence that RetroReasoner improves both predictive performance and chemical feasibility.

G.8 Performance on Unfiltered Multi-label Test Set

The main evaluation excludes instances with multiple valid reactant sets from the ORDerly test set to reduce potential bias in exact-match metrics such as Exact@1↑ and Exact@100↑. Since this filtering choice can affect the interpretation of exact-match performance, an additional evaluation is conducted on an unfiltered 500-instance test set that includes such cases. Table 20 reports the comparison between the filtered and unfiltered settings. As shown in Table 20, the relative performance gap between Prediction-Only and RetroReasoner remains substantial under both evaluation settings. This indicates that the improvements of RetroReasoner are not an artifact of excluding instances with multiple valid reactant sets. The main conclusion therefore remains unchanged: RetroReasoner consistently improves retrosynthesis prediction performance over the prediction-only baseline.

G.9 Failure Analysis

Most exact-match errors arise from alternative retrosynthetic decisions rather than internally inconsistent reasoning. Complete reasoning trajectories from RetroReasoner (RL) are analyzed on the 500-instance test set under greedy decoding. Internal consistency checks verify the selected disconnection, resulting synthons, synthetic equivalents, and final answer, while comparison with the SyntheticRetro reference identifies the first deviating step. As shown in Table 21, among 237 Exact@1 errors, 97.9% pass all internal consistency checks, with most deviations occurring at alternative bond disconnections (49.4%) or synthetic-equivalent choices (38.8%). Moreover, 69.2% (164/237) are round-trip consistent. In contrast, only 0.4% of Exact@1↑-correct predictions fail an internal consistency check, and 97.0% match the reference reasoning throughout. These results indicate that many exact-match failures reflect alternative, internally consistent retrosynthetic decisions rather than malformed reasoning, highlighting the limitation of single-reference evaluation in retrosynthesis.

	Exact@1↑	Round-trip@1↑	Exact@100↑	Round-trip@100↑	Feasible Ratio↑	Template Diversity↑
Filtered						
Prediction-Only (SFT)	0.482	0.784	0.678	0.950	0.774	2.562
Prediction-Only (RL)	0.486	0.802	0.662	0.936	0.785	2.324
RetroReasoner (SFT)	0.512	0.812	0.734	0.944	0.765	3.898
RetroReasoner (RL)	0.526	0.826	0.724	0.952	0.786	3.186
Unfiltered						
Prediction-Only (SFT)	0.462	0.772	0.666	0.922	0.761	2.454
Prediction-Only (RL)	0.464	0.790	0.652	0.918	0.773	2.166
RetroReasoner (SFT)	0.482	0.786	0.716	0.946	0.744	4.004
RetroReasoner (RL)	0.506	0.814	0.700	0.942	0.763	3.192

Table 20: Performance comparison when duplicate reactant data instances are not filtered from the main evaluation dataset.

First deviation point	**Fraction of Exact@1↑ errors**
Product perception error (PRODUCT_INFO)	7.6%
Different (but existing) bond chosen for disconnection	49.4%
Incorrect intermolecular versus intramolecular grouping (synthon grouping)	4.2%
Different synthetic equivalent choice	38.8%

Table 21: First deviation point of RetroReasoner (RL) Exact@1↑ errors (n=237).

	Exact@1↑	Round-trip@1↑	Exact@100↑	Round-trip@100↑	Feasible Ratio↑	Template Diversity↑
MHNreact (Seidl et al., 2021)						
Prediction-Only (SFT)	0.131	0.845	0.345	0.952	0.810	2.988
Prediction-Only (RL)	0.119	0.845	0.333	0.964	**0.825**	2.667
RetroReasoner (SFT)	0.143	0.845	**0.417**	0.976	0.764	**5.310**
RetroReasoner (RL)	**0.179**	**0.857**	0.405	**0.988**	0.793	4.405
TempRe (Xuan-Vu et al., 2025)						
Prediction-Only (SFT)	0.083	0.719	0.146	0.905	0.676	2.542
Prediction-Only (RL)	0.087	0.731	**0.166**	0.909	0.702	2.245
RetroReasoner (SFT)	**0.103**	**0.751**	0.154	**0.964**	0.694	**3.708**
RetroReasoner (RL)	**0.103**	**0.751**	0.158	0.949	**0.726**	3.067

Table 22: Performance comparison of RetroReasoner and Prediction-Only in MHNreact and TempRe, external rare reaction template benchmarks. For both datasets, the remaining datasets are used for evaluation after excluding the training instances used in RetroReasoner and Prediction-Only. (84 MHNreact instances, 253 TempRe instances)

G.10 Performance on Existing Benchmarks

In this study, the robustness of RetroReasoner is evaluated by constructing relatively challenging evaluation datasets from ORDerly, including rare template and rare atom/token subsets. However, there are also publicly available benchmark datasets designed for such challenging evaluation, with representative examples including MHNreact (Seidl et al., 2021) and TempRe (Xuan-Vu et al., 2025). These benchmarks consist of evaluation instances based on rare reaction templates. However, they may partially overlap with ORDerly, which is used to train RetroReasoner. Therefore, before evaluation, instances with overlapping reactants and products are removed through molecular comparison. The results are shown in Table 22. Overall, RetroReasoner outperforms Prediction-Only on these benchmarks. For RetroReasoner with RL, Template Diversity↑ slightly decreases compared to the SFT model, while Feasible Ratio↑ improves. This trend is consistent with the results on the rare template and rare atom/token datasets constructed in this study, as well as with the main in-distribution results.

G.11 Qualitative Comparison

Figure 16 shows a qualitative comparison of the reasoning processes of Chem-R, ether0, OpenAI-o3, and RetroReasoner (RL). First, Chem-R follows a reasoning sequence of product analysis → functional group identification → reaction type analysis → reactant prediction. In the product analysis step, general reasoning is followed by identification of the relevant functional groups. During this process, the corresponding SMILES of the functional groups are also provided in parenthe-

ses, which often differ from the product SMILES. For example, the ester COC(C)=O in the reasoning is represented as COC(=O) in the actual product SMILES. RetroReasoner, however, accurately represents functional groups' SMILES along with the position index numbers for each SMILES in <PRODUCT_INFO>. Next, after analyzing the reaction type, reactant prediction is made. In contrast, RetroReasoner follows a chemist's step-by-step reasoning strategy: it finds candidate substructures, performs strategic bond disconnections, and then derives synthetic equivalents from the synthons. ether0 performs functional group recognition and predicts the sulfonamide group as a key substructure but does not carry out strategic bond disconnections and proceeds directly to reactant prediction. OpenAI-o3 mainly analyzes the product and describes the process of forming a single bond, but fails to generate accurate SMILES. On the other hand, RetroReasoner follows a detailed strategy that proceeds from product analysis to identifying key substructures, performing strategic bond disconnections, and mapping synthetic equivalents, thereby gradually predicting the reactants in line with a chemist's approach.

G.12 Structured Reasoning Step Accuracy

RetroReasoner uses structured reasoning steps that contain predefined elements such as functional groups in the product, candidate substructures, and synthons. Table 23 reports results that measure whether each structured reasoning step produces these elements correctly. The components in $\mathcal{R}_1$, including atom mapping, functional groups, and SMILES statistics, can be predicted almost perfectly after training. However, as the steps progress, errors from earlier structured reasoning steps accumulate, and accuracy gradually decreases. Interestingly, the highest accuracy in these later steps is achieved after RL with round-trip rewards. This suggests that optimizing round-trip feasibility is partially aligned with improving the correctness of these intermediate reasoning paths.

G.13 LLM Judge Evaluation

A comparison of reasoning strategies among *Molecular Reasoning LLMs* is conducted using an LLM-based evaluation. The evaluation uses the GPT-oss-120B model as the judge and compares RetroReasoner (SFT), Chem-R, ChemDFM, and ether0. The criterion is how well the reasoning supports inference of the correct reactants. The evaluated reasoning texts are those generated during in-distribution evaluation. For each comparison, two reasoning outputs are provided as A and B, and the judge determines a winner or a tie. To reduce bias caused by the ordering of A and B, the same comparison is evaluated again with the order swapped. The result is accepted only when the two judgments are the same. The judge prompt is shown in Figure 20, and the results are shown in Figure 17. RetroReasoner shows a clear advantage over all other models, indicating stronger reasoning based on chemists' strategies, followed by Chem-R, ether0, and then ChemDFM.

G.14 Details of Human Expert Evaluation

We provide additional details of the human expert evaluation reported in Section 3.5.5. The evaluation compares RetroReasoner (RL) with Chem-R, the strongest Molecular Reasoning LLM baseline in our main comparison. As described in Section 3.5.5, the evaluation set consists of 50 instances: 33 instances for which both models predict the correct reactants, 9 instances for which only RetroReasoner predicts the correct reactants, and 8 instances for which only Chem-R predicts the correct reactants.

Evaluation Procedure. For each instance, evaluators are shown the complete reasoning process and predicted reactants from the two models. The model identities are hidden, and the two responses are presented in randomized order as Response A and Response B. Five chemistry experts independently evaluate the responses and select the one that demonstrates better chemical reasoning for retrosynthesis, or indicate a tie when neither response is clearly preferable.

Evaluation Criteria. The evaluators are instructed to assess the responses according to the following criteria:

- **Functional Group Analysis:** Does the response correctly identify key functional groups and their reactivity?
- **Reaction Mechanism Understanding:** Does the reasoning reflect an accurate understanding of the underlying reaction mechanisms?
- **Overall Logical Flow:** Is the retrosynthetic analysis coherent, well-structured, and chemically sound?

Structured Reasoning Step			$\mathcal{R}_1$	$\mathcal{R}_1$	$\mathcal{R}_1$
	Exact@1↑	Round-trip@1↑	Atom Mapping↑	Functional Groups↑	Product SMILES Statistics↑
RetroReasoner (SFT)	0.512	0.812	**0.980**	0.974	0.992
RetroReasoner (SFT, w/o $\mathcal{L}_{12}, \mathcal{L}_{23}, \mathcal{L}_{34}$)	0.494	0.806	0.978	**0.984**	**0.996**
RetroReasoner (SFT, w/ only $\mathcal{R}_1$)	0.498	0.806	0.978	0.978	0.988
RetroReasoner (RL)	**0.526**	**0.826**	**0.980**	0.972	0.990
RetroReasoner (RL, w/ R^{exact})	0.524	0.824	**0.980**	0.976	0.992
Structured Reasoning Step	$\mathcal{R}_2$	$\mathcal{R}_3$	$\mathcal{R}_3$	$\mathcal{R}_4$	
	Candidate Structure↑	Strategic Bond Disconnection↑	Synthon↑	Synthetic Equivalent↑	
RetroReasoner (SFT)	0.788	0.746	0.716	0.512	
RetroReasoner (SFT, w/o $\mathcal{L}_{12}, \mathcal{L}_{23}, \mathcal{L}_{34}$)	0.760	0.712	0.676	0.494	
RetroReasoner (SFT, w/ only $\mathcal{R}_1$)	N/A	N/A	N/A	N/A	
RetroReasoner (RL)	**0.802**	**0.752**	**0.722**	**0.526**	
RetroReasoner (RL, w/ R^{exact})	0.788	0.752	0.716	0.524	

Table 23: Table comparing the accuracy of structured reasoning steps. Exact@1↑ and Round-trip@1↑ are also reported for reference, and N/A indicates cases where a step is not generated by design, making the metric unavailable.

In addition, evaluators are asked to consider which response would be more helpful for planning an actual synthesis. The complete instruction presented to the evaluators is as follows:

> *Which response provides better chemical reasoning for retrosynthesis? As an expert chemist, please evaluate based on the following criteria:*
>
> *Functional Group Analysis: Does the response correctly identify key functional groups and their reactivity?*
>
> *Reaction Mechanism Understanding: Does the reasoning reflect accurate understanding of reaction mechanisms?*
>
> *Overall Logical Flow: Is the retrosynthetic analysis coherent, well-structured, and chemically sound?*
>
> *Select the response that would be more helpful for planning an actual synthesis.*

As reported in Table 6, RetroReasoner is preferred in 55.2% of the comparisons, while Chem-R is preferred in 26.8%, with the remaining 18.0% judged as ties. These results show that experts more frequently prefer the overall single-step reasoning and reactant predictions generated by RetroReasoner.

Qualitative Expert Feedback. We additionally inspect representative evaluator comments to illustrate the considerations underlying their preferences. Representative comments from cases in which RetroReasoner is preferred include:

> "The reasoning of A does not appear relevant to a Suzuki–Miyaura reaction."
>
> "Both responses assume a reaction involving a tertiary amine, but the product contains two tertiary amines. A explicitly identifies the position bearing the trifluoromethyl group and explains the corresponding reactant."
>
> "The term 'tetraroylated' used by B does not appear to exist."

The expert evaluation also reveals cases in which Chem-R is preferred. Representative comments include:

> "A contains an invalid SMILES."
>
> "Without additional constraints on which reactant is preferable, this would be a tie; otherwise, A appears more reasonable."
>
> "Both are similar, but judging from the final products, B appears more reasonable."

These examples illustrate that the evaluators consider not only the high-level logical flow of the reasoning, but also reaction-specific chemical interpretation and the plausibility of the resulting reactants. They also highlight remaining failure modes of RetroReasoner, such as invalid molecular representations or ambiguous precursor choices.

H Prompts

H.1 SyntheticRetro Generation Prompt

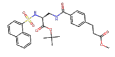
SyntheticRetro generates each linking text $\mathcal{L}_{12}$, $\mathcal{L}_{23}$, and $\mathcal{L}_{34}$ in order, and the prompts for each stage are shown in Figure 18.

H.2 Evaluation

Molecular Prediction LLMs and *Molecular Reasoning LLMs* are evaluated using their original prompt formats when such prompts are provided in prior work. *General Purpose LLMs* that are not directly trained for the retrosynthesis task are evaluated using the prompt format in Figure 19, which instructs the model to generate SMILES.

H.3 LLM Judge Win Rate Evaluation

Reasoning text evaluation for *Molecular Reasoning LLMs* that use an LLM as a judge is conducted using the prompt shown in Figure 20.

Input product SMILES: COC(=O)CCc1ccc(C(=O)NC[C@H](NS(=O)(=O)c2ccc3ccccc23)C(=O)OC(C)(C)C)cc1 

Labeled reactant SMILES: COC(=O)CCc1ccc(C(=O)NC[C@H](N)C(=O)OC(C)(C)C)cc1.O=S(=O)(Cl)c1ccc2ccccc12

Chem-R	<think> 1. **Product analysis**: The product SMILES string COC(=O)CCc1ccc(C(=O)NC[C@H](NS(=O)(=O)c2ccc3ccccc23)C(=O)OC(C)(C)C)cc1 represents a complex organic molecule .It contains multiple rings, including a benzene ring, and several functional groups. The overall structure suggests a molecule with diverse chemical properties, including polar and non-polar regions. 2. **Functional group identification**: Upon examining the product, several key functional groups can be identified: an ether (COC), an ester (CO(C)(C)=O), an amide (CC(N)=O), a secondary amine (C[NH]C), and a sulfonamide (NS(=O)(=O)). These functional groups are crucial for determining the disconnection sites in the retrosynthesis process. The presence of these groups indicates potential sites for bond cleavage, as they are often formed through specific types of reactions. 3. **Reaction type analysis**: Considering the functional groups present, the formation of the amide and sulfonamide groups likely involves nucleophilic acyl substitution reactions. The ether and ester groups could be formed through various reactions, including Williamson ether synthesis for the ether and esterification or transesterification for the ester. The sulfonamide group suggests a reaction involving a sulfonyl chloride .Analyzing the product's structure and the identified functional groups helps in deducing the possible reactants and the types of reactions that could lead to the product. 4. **Reactant prediction**: Based on the analysis of the functional groups and the likely reactions involved, the reactants can be predicted. For the amide group, a carboxylic acid or an acyl chloride and an amine would be required. The sulfonamide group could be formed from a sulfonyl chloride and an amine. The ether and ester groups suggest the involvement of alcohols and alkyl halides or carboxylic acids. Combining these considerations, the predicted reactants would include a compound containing the sulfonyl chloride group (to form the sulfonamide) and another with the appropriate functional groups to form the amide, ether, and ester linkages. Given the complexity of the molecule and the need to form multiple bonds, the reactants are likely to be COC(=O)CCc1ccc(C(=O)O)cc1 and CC(C)(C)OC(=O)[C@H](CNS(=O)(=O)c1ccc2ccccc12, which would undergo a series of reactions to form the desired product. </think> <answer>COC(C)(C)OC(=O)[C@H](CNS(=O)(=O)c1ccc2ccccc12.COC(=O)CCc1ccc(C(=O)O)cc1</answer>
Chem-DFM	CC(C)(C)OC(=O)[C@H](CNC(=O)c1ccc(CCC(=O)Oxc1)NS(=O)(=O)c1ccc2ccccc12.O=S(=O)(Cl)c1ccc2ccccc12.CO
ether0	<think_start>Okay, let's see. The user is asking about a chemical molecule and wants a single-step synthesis. The molecule given is COC(=O)CCc1ccc(C(=O)NC[C@H](NS(=O)(=O)c2ccc3ccccc23)C(=O)OC(C)(C)C)cc1 . This is a complex structure, so let's break it down. The molecule has a methyl ester (COC=O), a chain with a benzene ring substituted with a carbonyl group, and an amide linkage. The part with NS(=O)(=O) suggests a sulfonamide group. The structure includes a bicyclic system with a benzene ring fused to another ring. The key part is the amide linkage: C(=O)NC[C@H](NS(=O)(=O)c2ccc3ccccc23)C(=O)OC(C)(C)C. The [@H] indicates a chiral center. The NS(=O)(=O) part is a sulfonamide group attached to a bicyclic aromatic ring . The most straightforward way to synthesize this molecule is via an amide coupling reaction. The most likely reaction is an amide bond formation between an acid derivative and an amine. The existing amide is part of a peptide-like linkage, suggesting a peptide coupling reaction .</think_end> <answer_start>COC(=O)CCc1ccc(C(=O)O)cc1.NC[C@H](NS(=O)(=O)c2ccc3ccccc23)C(=O)OC(C)(C)C>>[C-]=C(OC(=O)CCc1ccc(C(=O)NC[C@H](NS(=O)(=O)c2ccc3ccccc23)C(=O)OC(C)(C)C)cc1-<answer_end>
OpenAI-o3	1. The product contains an amide bond between a 4-(3-methoxycarbonylpropyl)benzoyl group and a β-amino-tert-butyl ester bearing an N-naphthylsulfonamide. 2. To forge the amide, the carboxylic component would most conveniently be used as the corresponding acid chloride (or another activated derivative). 3. The nitrogen component must be a primary amine that already carries the N-sulfonyl substituent and the tert-butyl ester. 4. The N-sulfonyl substituent itself would be introduced by treating the free amino-ester with 1-naphthalenesulfonyl chloride. 5. Therefore the three key building blocks are: a) 4-(3-methoxycarbonylpropyl)benzoyl chloride – provides the acyl fragment. b) tert-butyl (S)-2-amino-3-(1-naphthylsulfonamido)propionate – provides the nucleophilic nitrogen and chiral center. c) 1-naphthalenesulfonyl chloride – reagent for installing the sulfonamide on the amino-ester. <ANSWER> COC(=O)CCCc1ccc(C(=O)O[C]cc1 N[C@@H](CNS(=O)(=O)c2ccc3ccccc23)C(=O)OC(C)(C)C O=S(=O)(Cl)c1ccc2ccccc12 </ANSWER>
RetroReasoner (SFT)	 <PRODUCT_INFO> { "Atom mapped SMILES": "[C:1]([O:2][C:3](=[O:4])[C:5][C:6][c:7]1[c:8][c:9][c:10]([C:11](=[O:12])[N:13][C:14]([C@H:15])[N:16][S:17](=[O:18])(=[O:19])[c:20]2[c:21][c:22][c:23][c:24]2)[c:25][c:26][c:27][c:28][c:29]1)[C:30](=[O:31])[O:32][C:33]([C:34])([C:35][C:36])[c:37][c:38]1", "Functional groups": ["carboxyl O (C)=O	[3, 4]", "carboxyl O (C)=O	[11, 12]", "carboxyl O (C)=O	[30, 31]", "carboxyl O, excluding COOH	C(=O	[3, 4]", "carboxyl O, excluding COOH	C(=O	[11, 12]", "carboxyl O, excluding COOH	C(=O	[30, 31]", "Secondary amines	N	[13]", "Secondary amines	N	[16]", "amides	C(=O)N	[11, 12, 13]", "aryl methyl sites for hydroxylation	c	[7]", "benzene rings	c1ccc2ccccc2c1	[7, 8, 9, 10, 37, 38]", "benzene rings	c2ccc(-c2)[20, 21, 22, 23, 24, 29]", "benzene rings	c3ccccc23	[24, 25, 26, 27, 28, 29]", "Bicyclic	c, c23	[24, 29]", "esters	COC(=O)Cl	[1, 2, 3, 4, 5]", "esters	C@H](_C(=O)OC)	[15, 30, 31, 32, 33]", "ether oxygens (including phenoxy)	COC	[1, 2, 3]", "ether oxygens (including phenoxy)	C(_OC)	[30, 32, 33]", "methoxy groups	COC	[1, 2, 3]", "sulfonamides	NS(=O)(=O)c	[16, 17, 18, 19, 20]"], "SMILES statistics": { "ring": 3, "carbon": 28, "#": 1 } } </PRODUCT_INFO> The product contains a sulfonamide linkage where a nitrogen is bonded to a trisubstituted sulfur that is further attached to an aromatic ring, indicating that the nitrogen–sulfur bond must have been formed during the synthesis. Consequently, the reaction step that generates this substructure is a nucleophilic substitution of a chlorosulfonyl group by the amine, yielding the observed sulfonamide fragment. <CANDIDATE_STRUCTURE> ["sulfonamides	NS(=O)(=O)c	[16, 17, 18, 19, 20]"] </CANDIDATE_STRUCTURE> Since the product bears a sulfonamide where the nitrogen is bonded to a trisubstituted sulfur that is further attached to an aromatic ring, the key bond that must have been formed is the N–S linkage. Therefore, a strategic disconnection of the single bond between the nitrogen and sulfur will isolate a sulfonyl chloride fragment and an amine-containing synthon for the subsequent coupling. <STRATEGIC_BOND_DISCONNECTION> { "Disconnect bonds": [[16, 17, "Single"]], "Synthons": ["[17*]N[C@@H](CNC(=O)c1ccc(CCC(=O)OC)cc1)C(=O)OC(C)(C)C", "[16*]S(=O)(=O)c1ccc2ccccc12"] } </STRATEGIC_BOND_DISCONNECTION> The amine-containing synthon is typically available as a free amine or a protected amide that can be deprotected to expose the nucleophilic nitrogen, while the sulfonyl fragment is usually introduced as a chlorosulfonyl derivative that reacts readily with the amine to form the sulfonamide linkage. Thus, the synthetic equivalents are chosen as a methoxy-ester-protected amide and a chlorosulfonyl aromatic compound, which together will couple to give the desired product. <SYNTHETIC_EQUIVALENT> ["COC(=O)CCc1ccc(C(=O)N[C@@H](CNC(=O)OC(C)(C)C)cc1", "O=S(=O)(Cl)c1ccc2ccccc12"] </SYNTHETIC_EQUIVALENT>

Figure 16: Qualitative comparison of the reasoning processes of *Molecular Reasoning LLMs* and OpenAI-o3.

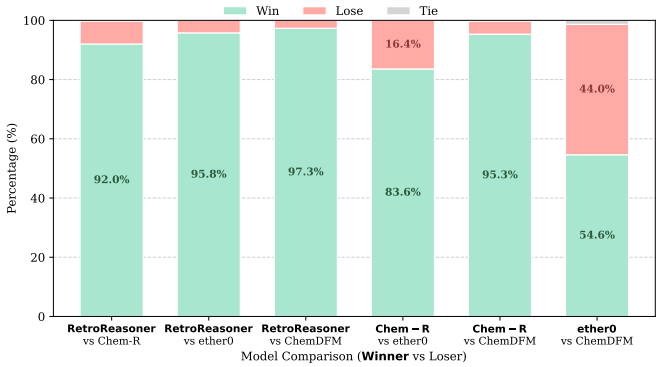


Figure 17: LLM judge evaluation results. Reasoning process generated by *Molecular Reasoning LLMs* on the in-distribution evaluation set are compared in pairs. The judge model is GPT-oss-120B. Each pair is evaluated twice with the order of the two reasonings swapped, and the outcome is accepted only when the two evaluations agree. On the x-axis, the winner model name is shown in **bold**.

Target Linking Text	$\mathcal{L}_{12}$	$\mathcal{L}_{23}$	$\mathcal{L}_{34}$
System Prompt	You are a chemistry expert and a generator tasked with filling in the gaps between two pieces of text contents related to chemical reaction prediction. The user provides you with three texts: 1. Previous Content, 2. Next Content, 3. Supporting Information. You must create a text that logically connects the Previous Content to the Next Content. In other words, your reasoning process should flow naturally from {Previous Content} → {your reasoning process} → {Next Content}. # Rules for generating the reasoning process: 1. The inference process should be one that gradually infers the content of the next content when only the content of the previous content is given. 2. Your reasoning requires a process of justifying why the Next Content should be based on the Previous Content. 3. There is Supporting Information you can refer to, but you must speak as if you are inferring it and not say where it is mentioned or described. 4. SMILES expressions must not be directly included in the inference process and should be mentioned in natural language. 5. It should be no longer than two sentences.		
User Prompt	Please generate the reasoning process that infers the Next Content from the Previous Content without directly using the content in the Supporting Information and Next Content. Your reasoning should focus on reasoning the substructure formed by the chemical reaction and its position without mentioning strategic disconnection and synthons and reactant SMILES directly. For example, your response should be like: Looking into the product information above, it is highly likely that the XXX reaction occurred because it is ... If this reaction occurs, XXX structure is created consists of atoms A, B, C... in product." And your response must not be like: "... this matches the candidate structure" (Pretend not to be proposed content.) ## Previous Content {input_content} ## Next Content {output_content} ## Supporting Information {supporting_info}	lease generate the reasoning process that infers the Next Content from the Previous Content without directly using the content in the Supporting Information and Next Content. Your reasoning should focus on how to determine the strategic bond disconnection, but do not mention synthon SMILES or reactant SMILES directly. For example, your response should be like: To be a little more specific, since the reaction is ..., bond ... must have been formed. Therefore, ... must undergo strategic cleavage." And your response should not be like: "... this matches the proposed ..." (Pretend not to be proposed content.) ## Previous Content {input_content} ## Next Content {output_content} ## Supporting Information {supporting_info}	Please generate the reasoning process that infers the Next Content from the Previous Content without directly using the content in the Supporting Information and Next Content. Your reasoning should focus on how to predict the synthetic equivalent for each synthon, but do not mention synthetic equivalent(reactant) SMILES directly. It should be mentioned in what state the synthetic equivalent containing each synthon usually exists or starts. For example, your response should be like: "Synthon X has ..., which is usually obtained from ..., because it does ... And Synthon Y has ..., which is usually obtained from ..., because it does ... " And your response should not be like: "... this matches the proposed ..." (Pretend not to be proposed content.) ## Previous Content {input_content} ## Next Content {output_content} ## Supporting Information {supporting_info}

Figure 18: Prompt used by SyntheticRetro for linking text generation.

System Prompt	You are a chemist. Please MUST start with your reasoning steps first and finally your answer inside <ANSWER></ANSWER> tags. Be sure to respond exclusively with SMILES, the chemical structure format.
User Prompt	Given the product [SMILES], what are some likely reactants that could have been used in its synthesis?"

Figure 19: Prompt used to evaluate *General Purpose LLMs*. It includes instructions intended to elicit SMILES generation.

System Prompt	You are a chemistry expert tasked with evaluating which of two AI assistants provides a better rationale for predicting the correct reactant of a chemical reaction. You will be provided with: - A chemistry question about a chemical reaction. - The correct reactant in SMILES format (ground-truth answer). - Reasoning-based answers from Assistant A and Assistant B enclosed in <ANSWER>...</ANSWER> tags. Your evaluation should focus only on the quality of the reasoning in terms of how effectively it leads toward the correct SMILES reactant. You are judging which assistant better guided the reasoning toward the correct reactant, regardless of the actual final SMILES they may have output. ### Evaluation Guidelines: - You are not scoring based on style, length, factual accuracy, or logical structure unless it impacts alignment with the correct reactant. - Ignore surface-level eloquence or complexity in explanation. - Focus on how chemically sound and reactant-oriented the reasoning is. That is, whose steps better indicate understanding of how to reach the correct SMILES. - Partial reasoning that sets up the correct mechanism or intermediate—even if incomplete—should be valued over incorrect but confident reasoning. Your final judgment must be strictly based on which assistant's reasoning better matches the process required to arrive at the correct SMILES. After analyzing both rationales: - Write a brief explanation comparing the two. - Then conclude with a final verdict using one of the following formats only: - '[[A]]' → Assistant A gave the better reasoning. - '[[B]]' → Assistant B gave the better reasoning. - '[[C]]' → Both were equally good or flawed.
User Prompt	# Input: - Question: {question} - Ground-Truth SMILES Answer: {ground_truth} - Assistant A Reasoning: {answer_a} - Assistant B Reasoning: {answer_b}

Figure 20: Prompt used for the LLM judge, GPT-oss-120B, in the evaluation of reasoning processes produced by *Molecular Reasoning LLMs*.