Verifiable Chemical Reasoning through Tool-Calling Agentic Workflow

Anonymous Author(s)
Affiliation
Address
email

Abstract

Reasoning models have increasingly been used to perform complex tasks in open ended environments. A challenge facing such efforts is domain specific tuning often requiring large quantities of data and verifiability. We can construct a high-performance reasoning agentic workflow for chemistry that is a) verifiable and b) extensible through the use of tools. We further show that distilling the outputs of the resulting workflow into smaller models results in lighter workflows that are still performant.

1 Introduction

The development of computational chemistry tools over the past decades has enabled significant automation in reaction optimisation and discovery [46, 10]. However, traditional ML methods which rely on hand-crafted features, expert configuration, and rigid input formats do not generalise well across different environments [30]. With their in-context few-shot abilities, Large Language Models (LLMs) emerge as powerful tools that can adapt to diverse or even unseen tasks [4]. LLMs, and in particular, chemistry domain-specialized LLMs (e.g., ChemFormer [21], Generative Chemical Transformer [25]) trained on extensive corpora of chemistry-related natural language data, are showing promising results [14, 33, 44, 12, 16, 61]. But while these models excel at generating coherent and convincing scientific text, LLMs often struggle with complex reasoning problems [33, 45], resulting in occasional widely incorrect answers [54].

To address this we explore the use of tool-calling agentic workflows, to capitalise on the reasoning capabilities of LLMs while maintaining reliability and verifiability of the output. To this end we developed tools for several chemistry tasks from the *ether0* benchmark [35] and evaluated the ability of LLMs to correctly understand queries and subsequently call the correct tools with the appropriate input and present the result.

Our contributions are:

- A chemical-reasoning agentic worflow that produces verifiable results through tool-calling.
- We further finetune smaller models on the reasoning traces of these LLMs and show substantial improvements in their ability to correctly use the provided tools.
- We conduct a preliminary comparison of the workflow perfomance, as well as the underlying model's capability on other chemical reasoning benchmarks such as MMLU-Pro-Chemistry[5, 58] and ChemBench-Mini [34].

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

2 Related Work

Chemistry Reasoning Models. Chain-of-Thought (CoT) prompting, where the model is asked to generate intermediate reasoning steps before producing a final answer [60, 29] was developed in an attempt to elicit deep *system-2* type reasoning [24] in LLMs. Building on this technique, recent "reasoning models" use large-scale reinforcement learning via Group Relative Policy Optimization (GRPO; [50]): e.g., DeepSeek-R1 [11]. Such reasoning models have achieved state-of-the-art performances in a wide range of complex tasks including arithmetics and symbolic reasoning [60]. In chemistry, however, reasoning LLMs are still scarce due to the lack of domain-specific data with explicit reasoning traces needed to effectively elicit CoT [17]. Distilling reasoning traces from stronger models or expert annotations has emerged as a practical solution, enabling the creation of synthetic data that smaller or domain-specific models can learn from through supervised fine-tuning [62]. This technique was notably adopted by FutureHouse in their *ether0* model [35], which is also the first and only current general-purpose chemistry reasoning LLM of its kind. Even more recently, Li et al. [32] introduced a novel distillation strategy called Prior Regulation via In-context Distillation (PRID), which they use to create a high-quality reasoning dataset. They leverage this dataset and build Mol-R1, a specialised reasoning model tailored for text-based molecule discovery, which achieves competitive results on that task. These works highlight the emerging trend of domain-specific reasoning LLMs in chemistry, though the field remains at an early stage.

Agentic Systems for Chemistry. The reasoning capabilities of LLMs are only valuable insofar as the information needed to solve a problem can be learned or inferred from their training data. However, some data constantly change: for example, the CAS (Chemical Abstracts Service) Registry is a seminal chemistry database of over 290 million reported chemical structures that is upadted daily.[1] One solution is to give LLMs access to such external chemistry data sources or software that they can use solve tasks that they could not otherwise perform [38, 2, 47, 3, 7, 23, 40, 6, 41]. This gives rise to what we call "agentic" systems; systems built upon LLMs that can flexibly integrate planning [18, 51, 59, 20], reasoning, retrieval, and computation within one workflow [45]. In chemistry, tools like ChemCrow [3] and Co-scientist [2] can help automate experiment design and execution in chemical synthesis [22, 57]. ProtAgents [15] introduce a multi-agent system to automate protein-related design and analysis. LLaMP [9] propose a retrieval-augmented generation (RAG)-based ReAct agent [63] to simulate inorganic materials by drawing from literature databases and Wikipedia, and interfacing with simulation tools. Most recently, Campbell et al. [7] introduce MDCrow, an agentic LLM assistant capable of automating Molecular dynamics (MD) workflows. For a review of agents in the scientific domain, refer to Ramos et al. [43] and Zheng et al. [66]. We note that, to the best of our knowledge, this is the first piece of work that proposes a multi-agent approach to solving a variety of experimentally-grounded chemistry tasks.

3 Methodology

3.1 Tasks

Narayanan et al. [35] introduced a chemical-reasoning model and its companion benchmark dataset, *ether0*. To evaluate the evolution of large-language-model performance on chemistry questions, the dataset is partitioned into 18 distinct subtasks. Our agent is designed to address 9 of these subtasks without relying on LLMs for core reasoning. Instead, we employ LLMs solely to (1) parse each natural-language question and (2) identify which subtask it belongs to. Each question is specified in natural language; the agent must interpret its semantic content, extract the relevant data, and route the parsed information to pre-designed tool.

Below we describe the five generation-based tasks and include, in parenthesis, the `problem_type` to which they correspond in the *ether0* benchmark.[2]

- **IUPAC name** (`molecule-name`): Given an IUPAC name, convert it to a valid SMILES string.

[1] See https://www.cas.org/cas-data/cas-registry.

[2] The dataset is freely accessible on HuggingFace via https://huggingface.co/datasets/futurehouse/ether0-benchmark.

- **SMILES completion** (`molecule-completion`): Given a truncated SMILES string, return a completed, valid SMILES string that preserves the original prefix.
- **Molecular formula** (`simple-formula`): Given a Hill formula, return a valid SMILES string corresponding to that formula.
- **Functional groups** (`functional-group`): Given a Hill formula and one or more functional groups, return a valid SMILES string that matches the formula and contains those functional groups.
- **Elucidation** (`molecule-formula`): Given a Hill formula, an organism, and some background information on the organism, return a SMILES string for a compound found in that organism whose formula matches the Hill formula.

Similarly, we describe below the four tasks which are framed as multiple-choice questions (MCQ):

- **Safety** (`property-cat-safety`): Given a safety class (either Carcinogenic, Fatally toxic, Fertility damaging, Flammable, or Toxic), and a set of 2-5 molecules given in SMILES notation, select the molecule that is most (or least) expected to possess that safety hazard.
- **LD50** (`property-regression-ld50`): Given an LD50[3] value in mg/kg and a set of 4 molecules given in SMILES notation, select the molecule that is most likely to have that LD50 value for on a population of a given variety of test animal (e.g., mouse) and for a given mode of administration (e.g., intraperitoneal injection).
- **pKa** (`property-regression-pka/pKa1`): Given a target pKa1[4] value and a set of 4 molecules given in SMILES notation, select the molecule that is most likely to have that pka1 value.
- **Aqueous solubility**: Given a target aqueous solubility value given in $\log S$ (where S is a molecule's aqueous solubility in mol/L) and a set of 4 molecules given in SMILES notation, select the molecule that is most likely to have that log solubility value.

3.2 Tools

To solve the tasks presented in the previous section, we built a series of tools intended to be use by the respective task agents. Below we present the logic of each of these:

- `iupac_to_smiles`: This tool accepts an IUPAC name and submits it to the PubChem PUG REST API [36], which returns the corresponding SMILES string.
- `smiles_completion`: This tool acceptes a truncated SMILES string and returns a completed SMILES string. There is two stages to this tool. The first stage applies balancing rules: trim trailing punctuation, close unmatched brackets, pair unbalanced ring digits, balance parentheses, convert dangling bond symbols (=, #, /, \) into socket atoms (*), and append a trailing socket. The second stage performs a breadth-first search over those sockets, replacing each * in turn with chemically plausible elements (C, N, O, S, F, Cl, Br, I) within valence limits, sanitising each candidate, and continuing until all sockets are filled and a valid SMILES is produced.
- `formula_to_smiles`: This tool accepts a HILL-format formula and submits it to the PubChem PUG REST API [36], which returns the corresponding SMILES strings. We use the first SMILES string returned as the answer.
- `functional_groups`: This tool accepts a HILL-format formula and a list of functional groups. It retrieves all plain and isomeric SMILES strings for the formula via the PubChem PUG REST API [36]. It then iterates over each SMILES string, converts it to an RDKit molecule [31], and uses ExMol [56] to identify the functional groups. The first molecule containing all specified functional groups is returned as a SMILES string.

[3] An LD50 value represents the lethal dose of a molecule needed to kill 50% of a test population, typically animals, upon exposure. A lower LD50 value indicates higher toxicity.

[4] pKa is a measure of the acidity of a molecule. It is calculated as the negative logarithm of the acid dissociation constant (Ka). The lower the pKa value, the stronger the acid. Specifically, pKa1 refers to the pKa value associated with the first ionization of a polyprotic acid (an acid with multiple ionizable protons).

- `safety_mcq`: This tool accepts a multiple-choice chemistry problem containing SMILES strings and a target safety class. It uses an LLM to extract the SMILES candidates and safety class in JSON format. It then queries the Globally Harmonized System (GHS) of Classification and Labeling of Chemicals[5] of each molecule. The retrieved GHS Classifications are compared against the target safety class, and the SMILES that matches most closely is returned as the answer. For reference, we include a mapping between the safety classes and their respective GHS hazard codes in Table 4.
- `ld50_mcq`: This tool accepts a multiple-choice chemistry problem that contains SMILES strings and a target LD50 value, along with the taxon (laboratory animal) and administration route. It first uses an LLM to extract the SMILES candidates, LD50 value, taxon, and route in JSON format. It then searches the LD50 dataset for matching entries. For each candidate SMILES, it compares the dataset's LD50 values to the target and selects the molecule whose value is closest under the specified conditions. The selected SMILES string is returned as the answer.
- `pka_mcq`: This tool accepts a multiple-choice chemistry problem containing SMILES strings and a target pKa value. It uses an LLM to extract the SMILES candidates and the target pKa value in JSON format. It then queries the pKa dataset to retrieve candidate values and compares them against the target. The SMILES whose dataset pKa most closely matches the specified value is returned as the answer.
- `solubility_mcq`: This tool accepts a multiple-choice chemistry problem containing SMILES strings and a target solubility value (log solubility in µg/mL). It extracts the candidate SMILES strings and the target solubility from the problem text, then predicts solubility for each candidate using a ML model. The SMILES whose predicted solubility is closest to the target value is returned as the answer. To predict solubility we reimplement the *DNN* model from [42].

Task name	**Tool name**
IUPAC name	`iupac_to_smiles`
SMILES completion	`smiles_completion`
Molecular formula	`formula_to_smiles`
Functional groups	`functional_groups`
Elucidation	`formula_to_smiles`
Safety	`safety_mcq`
LD50	`ld50_mcq`
pKa	`pka_mcq`
Aqueous solubility	`solubility_mcq`

Table 1: Mapping between the task names and the name of their corresponding tool.

3.3 Agentic Workflow

The agentic workflow of our system is structured as a multi-agent architecture in which a central supervisor agent coordinates a set of specialized sub-agents, each aligned with one of the chemistry tasks described in Table 1. The supervisor agent is responsible for interpreting the user's input query, reasoning over its intent, and delegating the problem to the appropriate sub-agent. This delegation is implemented through a handoff tool, which transfers control and the complete message history to the selected sub-agent. Each sub-agent is equipped with task-specific prompts and one or more tools to carry out its designated task. Upon receiving control of the conversation history, the sub-agent extracts the arguments required by the necessary tools and invokes them to generate an answer. It then performs the chemical reasoning needed to ensure the output is valid against the task specification before returning the solution to the supervisor. The supervisor consolidates the workflow and delivers the final answer to the user. This modular design ensures that the large language model is used only

[5]At the time of writing, the latest GHS Classification (Rev.10, 2023) can be accessed from https://pubchem.ncbi.nlm.nih.gov/ghs/.

for parsing and task routing, while domain reasoning is handled by deterministic tools and agents purpose-built for each chemistry task.

3.4 Reasoning Data Generation

As mentioned in Section 2, there is lack of datasets which include explicit reasoning traces on chemistry problems [17]. Such data is however required for refining the CoT reasoning of an LLM-based system. In this section, we describe how we re-built part of the *ether0* training set and used it to generate a dataset of reasoning traces.

3.4.1 Building the *ether0* training set

Following Narayanan et al. [35], we re-created a subset of the data used to train the *ether0* model focusing on the tasks introduced in Section 3.1.

COCONUT. The COlleCtion of Open Natural prodUcTs (COCONUT)[6] is the largest open collection of natural products, small molecules produced by living organisms with significant potential in pharmacology and various industries due to their bioactivity [53, 8]. This dataset was used in four tasks: namely IUPAC name, SMILES completion, Molecular formula and Elucidation. We detail below how we re-constructed the data for these task:

- **IUPAC name**: Drawing from the **Molecules** table of the dataset, we take the `iupac_name` field as input and the `canonical_smiles` field as the ground-truth.
- **SMILES completion**: As above, we take the `canonical_smiles` field as ground-truth. From there, we artificially create an incomplete SMILES by randomly truncating the ground-truth somewhere between 25% and 75% of its full lengths (to avoid overly short/long fragments) and check that the obtained partial SMILES is indeed no longer a valid molecule using RDKit's `MolFromSmiles()` method [31]. An invalid fragment is then used as input.
- **Molecular formula**: Again, we take the `canonical_smiles` field as ground-truth. The input, however, is obtained by joining the **Molecules** and **Properties** tables on the respective `id` and `molecule_id` columns and drawing from the `molecular_formula`.
- **Elucidation**: The ground-truth for this task is also taken from the `canonical_smiles` column of the **Molecules** table. The input is in part the `molecular_formula` as in the previous task and on the other the organism `name` that can be found in the **Organisms** table after joining it with **Molecules** on `molecule_id` and `id` respectively.

ChEMBL. The ChEMBL Database[7] is a manually curated dataset of bioactive molecules with drug-like properties [64]. This dataset was used for the Functional group task only; we describe this process below:

- **Functional groups**: We retrieve all the molecule `canonical_smiles` in the **compound_structures** table of the dataset and use these as ground-truths. For the inputs, we fetch their related molecular formulas given by `full_molformula` in the the **compound_properties** table (joining on the `molregno` column), and use ExMol's `get_functional_groups()` method [56] to find the functional groups of each molecule.

PubChem. PubChem [28, 27, 26] is the largest public database of chemical molecules which is maintained by the National Center for Biotechnology Information (NCBI) at the National Library of Medicine (NLM). This dataset was used for two MCQ-based tasks:

- **Safety**: The candidate SMILES are restricted to the records retrieved from PubChem records that contain GHS Hazard statements. We manually prepare a mapping between the target safety classes (Carcinogenic, Fatally toxic, Fertility damaging, Flammable, and Toxic) and its corresponding GHS Hazard code (H-code) as shown in Table4. The SMILES are then divided into those that possess at least one H-code belonging to the target class and those

[6]The dataset can be downloaded from https://coconut.naturalproducts.net/download.

[7]The dataset can be downloaded from https://chembl.gitbook.io/chembl-interface-documentation/downloads.

that do not, to form the correct and incorrect sets. To ensure that the distractor candidates are as similar as the correct ones, MCQs are constructed using only SMILES pairs whose Tanimoto similarity exceeds a certain threshold.

- **LD50**: For the PubChem Toxicity data, each entry in the dataset contains a `SMILES`, an associated species (Taxon), an administration route (Route), and the corresponding LD50 value in mg/kg. To avoid duplicates, we filter to unique combinations of (SMILES, Taxon, Route). Distractor choices are selected from molecules with the same species and route whose LD50 values differ from the ground-truth within predefined thresholds in mg/kg ($1 \leq \Delta \leq 100$). Similarity between candidate molecules is computed using RDKit fingerprints and Tanimoto similarity, and the closest candidates are chosen as distractors. If not enough valid candidates are found, additional distractors are sampled at random from the dataset. The final question prompt presents the Taxon, Route, and LD50 value, with several SMILES as options, one of which is correct.

IUPAC. The IUPAC Digitized pKa Dataset [65] is an ongoing digitisation of pKa data from reference works of Serjeant and Dempsey [48] and Perrin [39] published by the International Union of Pure and Applied Chemistry (IUPAC). The original dataset can be downloaded from here.

- **pKa**: Starting from 6,678 unique rows for pKa1 (based on temperature and pressure), we narrowed it down by keeping temperatures between 17–27°C and only rows with numerical answers, which gave 4,026 rows. Then, we removed entries with non-atmospheric pressure when that info was available, leaving 3,946 rows. To handle duplicates with different temperatures, we kept just one row per case, ending up with 3,596 rows. For each `SMILES` molecule, candidate distractors are identified by: (i) computing Tanimoto similarity scores between the query molecule and all others in the dataset using the RDKit fingerprint, and (ii) filtering based on the absolute difference in pKa1 values falling within predefined thresholds ($0.2 \leq \Delta \leq 1.0$). The filtered candidates are then ranked by similarity, and the top three are selected as distractors. If fewer than three valid candidates remain, random molecules (excluding the correct answer and already selected distractors) are added. Finally, the distractors and the correct answer are shuffled to produce the answer options.

AqSolDB. The AqSolDB Dataset [52] contains a curated reference set of aqueous solubility values, comprising 9,982 unique compounds collected from nine publicly available solubility datasets. The original dataset can be downloaded here.

- **Aqueous solubility**: Similar to the pKa MCQ dataset, for each `SMILES` molecule, candidate distractors are identified by: (i) computing Tanimoto similarity scores between the query molecule and all others in the dataset using the RDKit fingerprint, and (ii) filtering based on the absolute difference in solubility values falling within predefined thresholds ($0.2 \leq \Delta \leq 1.0$). The filtered candidates are then ranked by similarity, and the top three are selected as distractors. If fewer than three valid candidates remain, random molecules (excluding the correct answer and already selected distractors) are added. Finally, the distractors and the correct answer are shuffled to produce the answer options.

From there, we randomly selected 500 input/ground-truth pairs from each task to form our own training set, making sure that there was no overlap with those present in the *ether0* benchmark dataset. Finally, we used the selected inputs to create natural language problems using a variety of prompts templates[8] used by Narayanan et al. [35].

3.4.2 Generating the reasoning traces

Our next step is to generate reasoning traces for each task using the training set presented in the previous section to build a CoT chemistry-specific dataset. For this, we use three different base LLMs: namely `gpt-4o-mini` [37], `gpt-oss-20b` [13] and `qwen3-8b` [55]. In each setting, we create different instances of the same model as the basis for the supervisor and the agents introduced in Section 3.3. Then, we prompt the supervisor to solve all the problems in the training set (with 500 problems per task) and record the generated workflow trace to form our reasoning trace dataset.

[8] The prompts can be found via https://github.com/Future-House/ether0/blob/main/src/ether0/problem_prompts.py. For reference, we include a mapping between the task names and the prompt variable names in Table 5.

Since we are building a dataset to elicit CoT reasoning in subsequent models, we want to maximise the correctness of the reasoning traces rather than test the performance of the tools. To this end, we modify the tools such that given an input present in the training set, the tool will always return the associated ground truth. This, however, is of course dependent on the supervisor model calling the right agent for a given problem and on the agent successfully recognising and parsing the inputs within the problem statement. We report in Table 6 the tool calling and overall accuracies of the workflows for each of the three base models.

3.5 Reasoning Supervised Fine-Tuning

Finally, we distill the generated reasoning traces into smaller LLMs through supervised fine-tuning (SFT). Given a query and a reasoning trace, the LLM is required to generate the *"assistant"* output, once in the role of *"supervisor"* and once again as the task-specific subagent for that query. Additionally, we finetune a second version of the model that includes the `<think>` tag outputs to determine the impact on overall performance.

For our initial investigation we focus on fine-tuning small models such as `Qwen3-0.6B` and `Qwen3-1.7B`. Additionally, we limit SFT to the reasoning traces generated by the larger `Qwen3-8B`, as these models were both distilled from the same original flagship models [55], and there has been a documented tendency for LLMs to prefer their own outputs [1], making this configuration most likely to succeed. We show the results of this experiment in Tables **??**, **??**, and 2.

3.6 Evaluation

To evaluate our approach, we report the performance of the supervised fine-tuned models: MMLU Pro Chemistry [5, 58], ChemBench-Mini [34], and *ether0* [35].

MMLU-Pro-Chemistry. The MMLU-Pro-Chemistry benchmark is a subset of the larger MMLU-Pro dataset [58] which is itself an extension of the seminal MMLU benchmark [19]. The subset contains 1,132 multiple-choice questions (MCQs) which assess graduate-level knowledge across areas such as organic, inorganic, physical, and analytical chemistry. Overall, the benchmark primarily tests factual recall, conceptual understanding, and problem-solving ability, and thus serves as a measure of general chemistry competence.

ChemBench-Mini. ChemBench [34] is a recent benchmark created to systematically assess the capabilities of LLMs in chemistry. Unlike MMLU-Pro-Chemistry which only includes MCQs, ChemBench also samples open-ended questions including interpretation of molecular structures and reactivity. As a result, it provides a direct measure of a model's chemistry domain reasoning patterns beyond surface-level memorisation. ChemBench-Mini was curated to be a light-weight, diverse and representative subset of the full corpus, and contains only 236 questions.

Together, these two benchmarks provide a comprehensive evaluation landscape: MMLU-Pro Chemistry tests general chemistry knowledge, while ChemBench assesses chemical reasoning.

***ether0*.** The *ether0* benchmark[9] comprises 18 tasks, spanning both open-ended and multiple-choice formats, that evaluate a model's ability to manipulate chemical structures and perform sophisticated reasoning tasks, similarly to ChemBench. Among existing benchmarks, *ether0* is most closely aligned with our work, as we selected our tasks directly from the set of 18 defined by Narayanan et al. [35] (Section 3.1). While the full benchmark includes 325 questions, we restricted evaluation to the 181 questions corresponding to our nine chosen tasks. This choice both avoids overlap with ChemBench-Mini, which already covers similar capabilities and question formats, and allows for a more targeted assessment of our agentic workflow on the chosen tasks.

Unlike MMLU-Pro-Chemistry and ChemBench, where models are evaluated in a direct prompting setup, we evaluate the models on the *ether0* benchmark by instantiating each model within our agentic workflow. Accordingly, we report two complementary metrics: (1) tool-calling accuracy, measuring whether the supervisor correctly delegates to the appropriate sub-agent (aggregated across the nine tasks using the Macro F1 score), and (2) final-answer accuracy, measuring whether the complete workflow produces the correct solution.

[9]The dataset can be accessed from HuggingFace via https://huggingface.co/futurehouse/ether0.

Base models. For comparison, we evaluate our fine-tuned models against a series of base models:

- their respective base models `Qwen/Qwen3-0.6B` and `Qwen/Qwen3-1.7B`,
- `Qwen/Qwen3-8B` which we used to generate the reasoning training data (Section 3.4.2),
- OpenAi's `gpt-4o-mini` [37] a highly performant, small, closed-source model for reference,
- and `futurehouse/ether0`[10] model [35].

Note that we do not evaluate `futurehouse/ether0` on the *ether0* benchmark as it was not trained for tool-calling in an agentic setting, and so perfomance would not be directly comparable.

4 Result

The results of our evaluation (Section 3.6) are summarised in Tables 2 and 3.

***ether0*.** Table 2 reports tool-calling and final-answer accuracy on the nine *ether0* tasks within our agentic workflow. We also plot the fine-grained Macro F1 results (for each task) in Figures 1 and 2 As expected, larger base models such as `Qwen/Qwen3-8B` and `gpt-4o-mini` achieve the strongest performance overall, with macro F1 scores above 0.85. The smaller `Qwen/Qwen3-0.6B` and `Qwen/Qwen3-1.7B` struggle in this setting, reflecting their limited capacity for complex multi-step reasoning. Supervised fine-tuning leads to clear gains for some models. Notably, `sft-Qwen/Qwen3-0.6B` improves tool-calling accuracy from 0.254 to 0.707 and final-answer accuracy from 0.088 to 0.376. Similarly, `sft-Qwen/Qwen3-1.7B` yields a large jump in final-answer accuracy (+0.420). These improvements demonstrate that distilling reasoning traces into small models enables them to recover part of the reasoning ability of larger bases. However, gains are not uniform: `sft-think-Qwen/Qwen3-0.6B` underperforms compared to its base counterparts, suggesting that not all fine-tuning strategies are equally effective.

Table 2: Performance of base and fine-tuned models on the *ether0* benchmark, rounded to three decimal places.

Model	Tool calling Macro F1	Accuracy
`Qwen/Qwen3-0.6B`	0.254	0.088
`Qwen/Qwen3-1.7B`	0.530	0.033
`Qwen/Qwen3-8B`	0.856	0.812
`gpt-4o-mini`	0.867	0.779
`sft-Qwen/Qwen3-0.6B`	0.707 +0.453	0.376 +0.288
`sft-Qwen/Qwen3-1.7B`	0.707 +0.177	0.044 +0.011
`sft-think-Qwen/Qwen3-0.6B`	0.177 −0.077	0.055 −0.033
`sft-think-Qwen/Qwen3-1.7B`	0.669 +0.139	0.453 +0.420

ChemBench-Mini. Table 3 shows results on ChemBench-Mini. Here, base models show a strong scaling trend, with `Qwen/Qwen3-8B` reaching 0.610 accuracy, competitive with `gpt-4o-mini` (0.585). The `futurehouse/ether0` model performs poorly (0.008), consistent with the fact that it is not instruction-tuned for general chemistry question-answering. Fine-tuning has mixed impact: `sft-Qwen/Qwen3-1.7B` improves substantially over its base (0.424 vs. 0.263), while other variants underperform relative to their starting points. This suggests that reasoning traces are helpful for models in the mid-size regime, but may not transfer straightforwardly to very small models.

MMLU-Pro Chemistry. Results on MMLU-Pro Chemistry (Table 3) reflect a similar picture. Larger base models perform well (`Qwen/Qwen3-8B` at 0.784), though `gpt-4o-mini` lags behind. SFT models consistently fall short of their base counterparts, with drops of 0.12–0.24 in accuracy. This highlights that reasoning distillation is less effective on benchmarks emphasising factual recall and broad conceptual coverage, as opposed to structured tool-augmented reasoning.

[10]The model is open-source and available on HuggingFace via https://huggingface.co/futurehouse/ether0.

Table 3: Performance (in terms of overall accuracy) of base and fine-tuned models on ChemBench-Mini and MMLU-Pro Chemistry, rounded to three decimal places.

Model	ChemBench-Mini	MMLU-Pro-Chemistry
`Qwen/Qwen3-0.6B`	0.127	0.349
`Qwen/Qwen3-1.7B`	0.263	0.652
`Qwen/Qwen3-8B`	0.610	0.784
`gpt-4o-mini`	0.585	0.092
`futurehouse/ether0`	0.008	0.113
`sft-Qwen/Qwen3-0.6B`	0.034 -0.093	0.109 -0.240
`sft-Qwen/Qwen3-1.7B`	0.424 $+0.161$	0.521 -0.131
`sft-think-Qwen/Qwen3-0.6B`	0.042 -0.085	0.159 -0.190
`sft-think-Qwen/Qwen3-1.7B`	0.178 -0.085	0.531 -0.121

Taken together, these results show that supervised fine-tuning on reasoning traces can substantially improve performance of small models in structured, tool-mediated workflows (as evaluated on the *ether0* benchmark). However, the benefits are do not seem to translate to benchmarks like ChemBench-Mini and MMLU-Pro Chemistry that emphasise direct question-answering and factual recall. This divergence underscores the importance of aligning training data with the target evaluation setting: reasoning-focused distillation primarily enhances performance when models are embedded in agentic workflows rather than when they are directly prompted.

5 Conclusion

In this work, we evaluated the ability of tool-calling agentic workflows to reason in the chemistry domain. We show that LLMs are capable of utilising task-specific tools to great effect (Table 2) even when the LLMs themselves are not fully capable in the chemistry domain by themselves (Table 3).

Additionally, we evaluated the effectiveness of supervised fine-tuning on reasoning traces, generated from a larger model (here `Qwen3-8B`) in an agentic workflow setting, for improving the performance of small and mid-sized language models on chemistry-reasoning tasks. Our results show that fine-tuning on this type of reasoning traces can substantially boost both tool-calling and final-answer accuracy for smaller models, enabling them to recover part of the reasoning ability of larger base model. However, these gains do not consistently transfer when evaluated in non-agentic settings on benchmarks such as ChemBench-Mini and MMLU-Pro Chemistry. This contrast highlights the importance of aligning training data with the target evaluation setting: distilling from a chemistry reasoning agentic system is most effective when downstream models are also embedded in said agentic workflows to perform the same tasks.

Future work. We can see several avenues could further enhance model performance in structured reasoning tasks. These include combining supervised fine-tuning with reinforcement learning techniques such as Group Relative Policy Optimization (GRPO; [11]), developing interleaved reasoning architectures that dynamically alternate between reasoning and tool usage, distilling more knowledge from larger, stronger models, and incorporating executable tool code (e.g., Python) into the reasoning traces [49]. Exploring these directions may yield more robust small models capable of both complex reasoning and general question-answering.

References

[1] Derek T. Ahneman, Jesús G. Estrada, Shishi Lin, Spencer D. Dreher, and Abigail G. Doyle. Predicting reaction performance in c–n cross-coupling using machine learning. *Science*, 360 (6385):186–190, 2018. doi: 10.1126/science.aar5169. URL `https://www.science.org/doi/abs/10.1126/science.aar5169`.

[2] Daniil Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. Autonomous chemical research with large language models. *Nature*, 624:570–578, 12 2023. doi: 10.1038/s41586-023-06792-0.

[3] Andres M Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D White, and Philippe Schwaller. Chemcrow: Augmenting large-language models with chemistry tools, 2023. URL https://arxiv.org/abs/2304.05376.

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

[5] Hengxing Cai, Xiaochen Cai, Junhan Chang, Sihang Li, Lin Yao, Wang Changxin, Zhifeng Gao, Hongshuai Wang, Li Yongge, Mujie Lin, Shuwen Yang, Jiankun Wang, Mingjun Xu, Jin Huang, Xi Fang, Jiaxi Zhuang, Yuqi Yin, Yaqi Li, Changhong Chen, Zheng Cheng, Zifeng Zhao, Linfeng Zhang, and Guolin Ke. SciAssess: Benchmarking LLM proficiency in scientific literature analysis. In Luis Chiruzzo, Alan Ritter, and Lu Wang, editors, *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 2335–2357, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-195-7. doi: 10.18653/v1/2025.findings-naacl.125. URL https://aclanthology.org/2025.findings-naacl.125/.

[6] Tianle Cai, Xuezhi Wang, Tengyu Ma, Xinyun Chen, and Denny Zhou. Large language models as tool makers, 2024. URL https://arxiv.org/abs/2305.17126.

[7] Quintina Campbell, Sam Cox, Jorge Medina, Brittany Watterson, and Andrew D. White. Mdcrow: Automating molecular dynamics workflows with large language models, 2025. URL https://arxiv.org/abs/2502.09565.

[8] Venkata Chandrasekhar, Kohulan Rajan, Sri Ram Sagar Kanakam, Nisha Sharma, Viktor Weißenborn, Jonas Schaub, and Christoph Steinbeck. COCONUT 2.0: a comprehensive overhaul and curation of the collection of open natural products database. *Nucleic Acids Res*, 53(D1):D634–D643, January 2025.

[9] Yuan Chiang, Elvis Hsieh, Chia-Hong Chou, and Janosh Riebesell. Llamp: Large language model made powerful for high-fidelity materials knowledge retrieval and distillation, 2024. URL https://arxiv.org/abs/2401.17244.

[10] Karl D Collins, Tobias Gensch, and Frank Glorius. Contemporary screening approaches to reaction discovery and development. *Nat Chem*, 6(10):859–871, October 2014.

[11] DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

[12] Long Phan et al. Humanity's last exam, 2025. URL https://arxiv.org/abs/2501.14249.

[13] OpenAI et al. gpt-oss-120b & gpt-oss-20b model card, 2025. URL https://arxiv.org/abs/2508.10925.

[14] Yin Fang, Xiaozhuan Liang, Ningyu Zhang, Kangwei Liu, Rui Huang, Zhuo Chen, Xiaohui Fan, and Huajun Chen. Mol-instructions: A large-scale biomolecular instruction dataset for large language models, 2024. URL https://arxiv.org/abs/2306.08018.

[15] A. Ghafarollahi and M. J. Buehler. Protagents: Protein discovery via large language model multi-agent collaborations combining physics and machine learning, 2024. URL https://arxiv.org/abs/2402.04268.

[16] Taicheng Guo, Kehan Guo, Bozhao Nan, Zhenwen Liang, Zhichun Guo, Nitesh V. Chawla, Olaf Wiest, and Xiangliang Zhang. What can large language models do in chemistry? a comprehensive benchmark on eight tasks, 2023. URL https://arxiv.org/abs/2305.18365.

[17] Yang Han, Ziping Wan, Lu Chen, Kai Yu, and Xin Chen. From generalist to specialist: A survey of large language models for chemistry, 2024. URL https://arxiv.org/abs/2412.19994.

[18] Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. Reasoning with language model is planning with world model, 2023. URL https://arxiv.org/abs/2305.14992.

[19] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021. URL https://arxiv.org/abs/2009.03300.

[20] Xu Huang, Weiwen Liu, Xiaolong Chen, Xingmei Wang, Hao Wang, Defu Lian, Yasheng Wang, Ruiming Tang, and Enhong Chen. Understanding the planning of llm agents: A survey, 2024. URL https://arxiv.org/abs/2402.02716.

[21] Ross Irwin, Spyridon Dimitriadis, Jiazhen He, and Esben Jannik Bjerrum. Chemformer: a pre-trained transformer for computational chemistry. *Machine Learning: Science and Technology*, 3(1):015022, jan 2022. doi: 10.1088/2632-2153/ac3ffb. URL https://dx.doi.org/10.1088/2632-2153/ac3ffb.

[22] Kevin Maik Jablonka, Qianxiang Ai, Alexander Al-Feghali, Shruti Badhwar, Joshua D. Bocarsly, Andres M. Bran, Stefan Bringuier, L. Catherine Brinson, Kamal Choudhary, Defne Circi, Sam Cox, Wibe A. de Jong, Matthew L. Evans, Nicolas Gastellu, Jerome Genzling, María Victoria Gil, Ankur K. Gupta, Zhi Hong, Alishba Imran, Sabine Kruschwitz, Anne Labarre, Jakub Lála, Tao Liu, Steven Ma, Sauradeep Majumdar, Garrett W. Merz, Nicolas Moitessier, Elias Moubarak, Beatriz Mouriño, Brenden Pelkie, Michael Pieler, Mayk Caldas Ramos, Bojana Ranković, Samuel G. Rodriques, Jacob N. Sanders, Philippe Schwaller, Marcus Schwarting, Jiale Shi, Berend Smit, Ben E. Smith, Joren Van Herck, Christoph Völker, Logan Ward, Sean Warren, Benjamin Weiser, Sylvester Zhang, Xiaoqi Zhang, Ghezal Ahmad Zia, Aristana Scourtas, K. J. Schmidt, Ian Foster, Andrew D. White, and Ben Blaiszik. 14 examples of how llms can transform materials science and chemistry: a reflection on a large language model hackathon. *Digital Discovery*, 2:1233–1250, 2023. doi: 10.1039/D3DD00113J. URL http://dx.doi.org/10.1039/D3DD00113J.

[23] Yunhui Jang, Jaehyung Kim, and Sungsoo Ahn. Chain-of-thoughts for molecular understanding. In *NeurIPS 2024 Workshop on AI for New Drug Modalities*, 2024. URL https://openreview.net/forum?id=cFHjEo2oCd.

[24] Daniel Kahneman. *Thinking, fast and slow*. Farrar, Straus and Giroux, New York, 2011. ISBN 9780374275631 0374275637. URL https://www.amazon.de/Thinking-Fast-Slow-Daniel-Kahneman/dp/0374275637/ref=wl_it_dp_o_pdT1_nS_nC?ie=UTF8&colid=151193SNGKJT9&coliid=I3OCESLZCVDFL7.

[25] Hyunseung Kim, Jonggeol Na, and Won Bo Lee. Generative chemical transformer: Neural machine learning of molecular geometric structures from chemical language via attention. *Journal of Chemical Information and Modeling*, 61(12):5804–5814, December 2021. ISSN 1549-960X. doi: 10.1021/acs.jcim.1c01289. URL http://dx.doi.org/10.1021/acs.jcim.1c01289.

[26] Sunghwan Kim. Exploring chemical information in pubchem. *Current protocols*, 1(8):e217, 2021.

[27] Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, Bo Yu, et al. Pubchem 2023 update. *Nucleic acids research*, 51(D1):D1373–D1380, 2023.

[28] Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, Bo Yu, Leonid Zaslavsky, Jian Zhang, and Evan E Bolton. Pubchem 2025 update. *Nucleic Acids Research*, 53(D1):D1516–D1525, 11 2024. ISSN 1362-4962. doi: 10.1093/nar/gkae1059. URL https://doi.org/10.1093/nar/gkae1059.

[29] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners, 2023. URL https://arxiv.org/abs/2205.11916.

[30] Brenden Lake, Tomer Ullman, Joshua Tenenbaum, and Samuel Gershman. Building machines that learn and think like people. *Center for Brains, Minds & Machines (CBMM) Memo No. 046*, arXiv, 04 2016. doi: 10.1017/S0140525X16001837.

[31] Greg Landrum. RDKit: Open-source cheminformatics. http://www.rdkit.org, 2006.

[32] Jiatong Li, Weida Wang, Qinggang Zhang, Junxian Li, Di Zhang, Changmeng Zheng, Shufei Zhang, Xiaoyong Wei, and Qing Li. Mol-r1: Towards explicit long-cot reasoning in molecule discovery, 2025. URL https://arxiv.org/abs/2508.08401.

[33] Adrian Mirza, Nawaf Alampara, Sreekanth Kunchapu, Martiño Ríos-García, Benedict Emoekabu, Aswanth Krishnan, Tanya Gupta, Mara Schilling-Wilhelmi, Macjonathan Okereke, Anagha Aneesh, Amir Mohammad Elahi, Mehrdad Asgari, Juliane Eberhardt, Hani M. Elbeheiry, María Victoria Gil, Maximilian Greiner, Caroline T. Holick, Christina Glaubitz, Tim Hoffmann, Abdelrahman Ibrahim, Lea C. Klepsch, Yannik Köster, Fabian Alexander Kreth, Jakob Meyer, Santiago Miret, Jan Matthias Peschel, Michael Ringleb, Nicole Roesner, Johanna Schreiber, Ulrich S. Schubert, Leanne M. Stafast, Dinga Wonanke, Michael Pieler, Philippe Schwaller, and Kevin Maik Jablonka. Are large language models superhuman chemists?, 2024. URL https://arxiv.org/abs/2404.01475.

[34] Adrian Mirza, Nawaf Alampara, Sreekanth Kunchapu, Martiño Ríos-García, Benedict Emoekabu, Aswanth Krishnan, Tanya Gupta, Mara Schilling-Wilhelmi, Macjonathan Okereke, Anagha Aneesh, Mehrdad Asgari, Juliane Eberhardt, Amir Mohammad Elahi, Hani M. Elbeheiry, María Victoria Gil, Christina Glaubitz, Maximilian Greiner, Caroline T. Holick, Tim Hoffmann, Abdelrahman Ibrahim, Lea C. Klepsch, Yannik K"oster, Fabian Alexander Kreth, Jakob Meyer, Santiago Miret, Jan Matthias Peschel, Michael Ringleb, Nicole C. Roesner, Johanna Schreiber, Ulrich S. Schubert, Leanne M. Stafast, A. D. Dinga Wonanke, Michael Pieler, Philippe Schwaller, and Kevin Maik Jablonka. A framework for evaluating the chemical knowledge and reasoning abilities of large language models against the expertise of chemists. *Nature Chemistry*, May 2025. ISSN 1755-4349. doi: 10.1038/s41557-025-01815-x. URL http://dx.doi.org/10.1038/s41557-025-01815-x.

[35] 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. Training a scientific reasoning model for chemistry, 2025. URL https://arxiv.org/abs/2506.17238.

[36] National Center for Biotechnology Information (NCBI). PubChem pug-rest api. https://pubchem.ncbi.nlm.nih.gov/rest/pug, 2025. Accessed: 2025-08-05.

[37] OpenAI. Gpt-4o mini: advancing cost-efficient intelligence, 2024. URL https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/. Accessed: 2025-08-20.

[38] Aaron Parisi, Yao Zhao, and Noah Fiedel. Talm: Tool augmented language models, 2022. URL https://arxiv.org/abs/2205.12255.

[39] Douglas Dalzell Perrin. *Dissociation constants of organic bases in aqueous solution: supplement 1972*, volume 1. Pergamon, 1972.

[40] Yujia Qin, Shengding Hu, Yankai Lin, Weize Chen, Ning Ding, Ganqu Cui, Zheni Zeng, Xuanhe Zhou, Yufei Huang, Chaojun Xiao, Chi Han, Yi Ren Fung, Yusheng Su, Huadong Wang, Cheng Qian, Runchu Tian, Kunlun Zhu, Shihao Liang, Xingyu Shen, Bokai Xu, Zhen Zhang, Yining Ye, Bowen Li, Ziwei Tang, Jing Yi, Yuzhang Zhu, Zhenning Dai, Lan Yan, Xin Cong, Yaxi Lu, Weilin Zhao, Yuxiang Huang, Junxi Yan, Xu Han, Xian Sun, Dahai Li, Jason Phang, Cheng Yang, Tongshuang Wu, Heng Ji, Guoliang Li, Zhiyuan Liu, and Maosong Sun. Tool learning with foundation models. *ACM Comput. Surv.*, 57(4), December 2024. ISSN 0360-0300. doi: 10.1145/3704435. URL https://doi.org/10.1145/3704435.

[41] Changle Qu, Sunhao Dai, Xiaochi Wei, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Jun Xu, and Ji-rong Wen. Tool learning with large language models: a survey. *Frontiers of Computer Science*, 19(8), January 2025. ISSN 2095-2236. doi: 10.1007/s11704-024-40678-2. URL http://dx.doi.org/10.1007/s11704-024-40678-2.

[42] Mayk Caldas Ramos and Andrew D. White. Predicting small molecules solubility on endpoint devices using deep ensemble neural networks. *Digital Discovery*, 3:786–795, 2024. doi: 10.1039/D3DD00217A. URL http://dx.doi.org/10.1039/D3DD00217A.

[43] Mayk Caldas Ramos, Christopher J. Collison, and Andrew D. White. A review of large language models and autonomous agents in chemistry. *Chem. Sci.*, 16:2514–2572, 2025. doi: 10.1039/D4SC03921A. URL http://dx.doi.org/10.1039/D4SC03921A.

[44] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. Gpqa: A graduate-level google-proof q&a benchmark, 2023. URL https://arxiv.org/abs/2311.12022.

[45] Nicholas T. Runcie, Charlotte M. Deane, and Fergus Imrie. Assessing the chemical intelligence of large language models, 2025. URL https://arxiv.org/abs/2505.07735.

[46] Alexander Buitrago Santanilla, Erik L. Regalado, Tony Pereira, Michael Shevlin, Kevin Bateman, Louis-Charles Campeau, Jonathan Schneeweis, Simon Berritt, Zhi-Cai Shi, Philippe Nantermet, Yong Liu, Roy Helmy, Christopher J. Welch, Petr Vachal, Ian W. Davies, Tim Cernak, and Spencer D. Dreher. Nanomole-scale high-throughput chemistry for the synthesis of complex molecules. *Science*, 347(6217):49–53, 2015. doi: 10.1126/science.1259203. URL https://www.science.org/doi/abs/10.1126/science.1259203.

[47] Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 68539–68551. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/d842425e4bf79ba039352da0f658a906-Paper-Conference.pdf.

[48] Eldon Percy Serjeant and Boyd Dempsey. *Ionisation constants of organic acids in aqueous solution*. Number 23 in Chemical data series. Pergamon, 1979.

[49] Rulin Shao, Shuyue Stella Li, Rui Xin, Scott Geng, Yiping Wang, Sewoong Oh, Simon Shaolei Du, Nathan Lambert, Sewon Min, Ranjay Krishna, Yulia Tsvetkov, Hannaneh Hajishirzi, Pang Wei Koh, and Luke Zettlemoyer. Spurious rewards: Rethinking training signals in rlvr, 2025. URL https://arxiv.org/abs/2506.10947.

[50] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

[51] Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M. Sadler, Wei-Lun Chao, and Yu Su. Llm-planner: Few-shot grounded planning for embodied agents with large language models, 2023. URL https://arxiv.org/abs/2212.04088.

[52] Murat Cihan Sorkun, Abhishek Khetan, and Süleyman Er. Aqsoldb, a curated reference set of aqueous solubility and 2d descriptors for a diverse set of compounds. *Scientific data*, 6(1):143, 2019.

[53] Maria Sorokina, Peter Merseburger, Kohulan Rajan, Mehmet Yirik, and Christoph Steinbeck. Coconut online: Collection of open natural products database. *Journal of Cheminformatics*, 13, 01 2021. doi: 10.1186/s13321-020-00478-9.

[54] Alex Tamkin, Miles Brundage, Jack Clark, and Deep Ganguli. Understanding the capabilities, limitations, and societal impact of large language models, 2021. URL https://arxiv.org/abs/2102.02503.

[55] Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

[56] ur-whitelab. exmol: Explainable molecular functional–group extraction. https://github.com/ur-whitelab/exmol, 2024.

[57] Juan Luis Villarreal-Haro, Remy Gardier, Erick J. Canales-Rodríguez, Elda Fischi-Gomez, Gabriel Girard, Jean-Philippe Thiran, and Jonathan Rafael-Patiño. Cactus: a computational framework for generating realistic white matter microstructure substrates. *Frontiers*

in Neuroinformatics, Volume 17 - 2023, 2023. ISSN 1662-5196. doi: 10.3389/fninf.2023.1208073. URL https://www.frontiersin.org/journals/neuroinformatics/articles/10.3389/fninf.2023.1208073.

[58] Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark, 2024. URL https://arxiv.org/abs/2406.01574.

[59] Zihao Wang, Shaofei Cai, Guanzhou Chen, Anji Liu, Xiaojian Ma, and Yitao Liang. Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents, 2024. URL https://arxiv.org/abs/2302.01560.

[60] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023. URL https://arxiv.org/abs/2201.11903.

[61] Andrew D. White, Glen M. Hocky, Heta A. Gandhi, Mehrad Ansari, Sam Cox, Geemi P. Wellawatte, Subarna Sasmal, Ziyue Yang, Kangxin Liu, Yuvraj Singh, and Willmor J. Peña Ccoa. Assessment of chemistry knowledge in large language models that generate code. *Digital Discovery*, 2:368–376, 2023. doi: 10.1039/D2DD00087C. URL http://dx.doi.org/10.1039/D2DD00087C.

[62] Xiaohan Xu, Ming Li, Chongyang Tao, Tao Shen, Reynold Cheng, Jinyang Li, Can Xu, Dacheng Tao, and Tianyi Zhou. A survey on knowledge distillation of large language models, 2024. URL https://arxiv.org/abs/2402.13116.

[63] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023. URL https://arxiv.org/abs/2210.03629.

[64] Barbara Zdrazil, Eloy Felix, Fiona Hunter, Emma J Manners, James Blackshaw, Sybilla Corbett, Marleen de Veij, Harris Ioannidis, David Mendez Lopez, Juan F Mosquera, Maria Paula Magarinos, Nicolas Bosc, Ricardo Arcila, Tevfik Kizilören, Anna Gaulton, A Patrícia Bento, Melissa F Adasme, Peter Monecke, Gregory A Landrum, and Andrew R Leach. The ChEMBL database in 2023: a drug discovery platform spanning multiple bioactivity data types and time periods. *Nucleic Acids Res*, 52(D1):D1180–D1192, January 2024.

[65] Jonathan Zheng and Olivier Lafontant-Joseph. IUPAC Digitized pKa Dataset, 2025. URL https://doi.org/10.5281/zenodo.15375522.

[66] Tianshi Zheng, Zheye Deng, Hong Ting Tsang, Weiqi Wang, Jiaxin Bai, Zihao Wang, and Yangqiu Song. From automation to autonomy: A survey on large language models in scientific discovery, 2025. URL https://arxiv.org/abs/2505.13259.

A Methodology Details

A.1 Safety classes

A.2 *ether0* prompts to task names

A.3 Agentic data generation results

A.4 Fine-grained *ether0* Macro F1 results

Safety Class	GHS Hazard code
Carcinogenic	H300, H350i, H351
Fatally toxic	H300, H304, H310, H330
Fertility damaging	H360, H360F, H360D, H360FD, H360Fd, H360Df, H361, H361f, H361d, H361fd
Flammable	H205, H206, H207, H208, H220, H221, H222, H223, H224, H225, H226, H227, H228, H229, H230, H231, H232, H241, H242, H250, H251, H252, H260, H261, H270, H271, H272, H282, H283
Toxic	H300, H301, H302, H303, H310, H311, H312, H313, H330, H331, H332, H333, H335, H336, H370, H371, H372, H373

Table 4: Safety task classes and their GHS Hazard codes.

Task name	Prompt variable name
IUPAC name	`NAME_IUPAC_PROMPTS`
SMILES completion	`COMPLETE_MOL_PROMPTS`
Molecular formula	`SMILES_FROM_FORMULA_PROMPTS`
Functional groups	`FUNCTIONAL_GROUP_PROMPTS`
Elucidation	`MOL_FORMULA_PROMPTS`
Safety	`PROPERTY_PROMPTS`
LD50	`PROPERTY_PROMPTS`
pKa	`PROPERTY_PROMPTS`
Aqueous solubility	`PROPERTY_PROMPTS`

Table 5: Mapping between the task names and the prompt variable names in *ether0*'s `problem_prompts.py`.[8]

	`gpt-4o-mini`		`gpt-oss-20b`		`qwen3-8b`	
Task name	**Tool calling**	**Acc.**	**Tool calling**	**Acc.**	**Tool calling**	**Acc.**
IUPAC name	0.992	**0.866**	0.992	0.858	**1.000**	0.856
SMILES completion	**1.000**	**0.894**	0.992	0.774	**1.000**	0.740
Molecular formula	**1.000**	**0.920**	0.996	0.910	**1.000**	0.894
Functional groups	**1.000**	**0.890**	0.978	0.838	0.988	0.886
Elucidation	0.912	0.570	0.740	0.458	**1.000**	**0.602**
Safety	0.730	0.580	**0.894**	**0.704**	0.746	0.620
LD50	0.988	0.908	0.922	0.828	**1.000**	**0.920**
pKa	0.984	0.824	0.990	0.916	**1.000**	**0.928**
Aqueous solubility	0.984	0.760	0.990	0.868	**1.000**	**0.886**
Average	0.954	0.801	0.944	0.795	**0.972**	**0.815**

Table 6: **Tool calling** accuracy (did the supervisor delegate to the right agent) and overall accuracy (**Acc.**; is the final answer correct) of the workflows for three base LLMs on the training set, rounded to three decimal places.

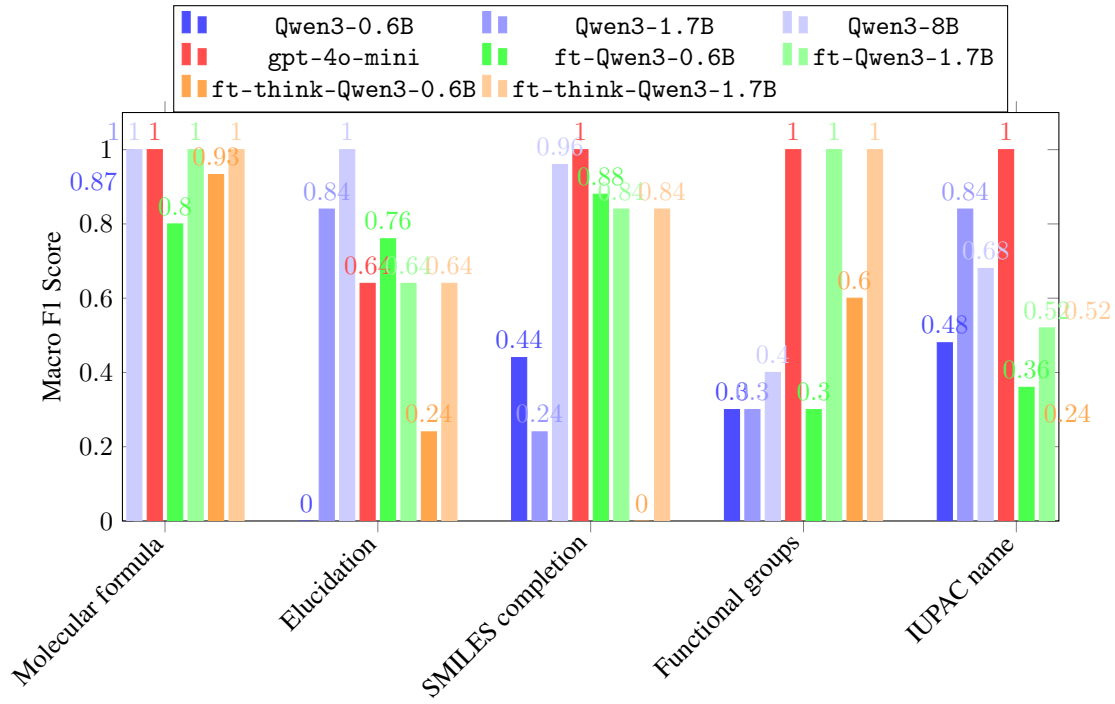

Figure 1: Macro F1 scores for non-MCQ tasks (Molecular formula, Elucidation, SMILES completion, Functional groups, and IUPAC name).

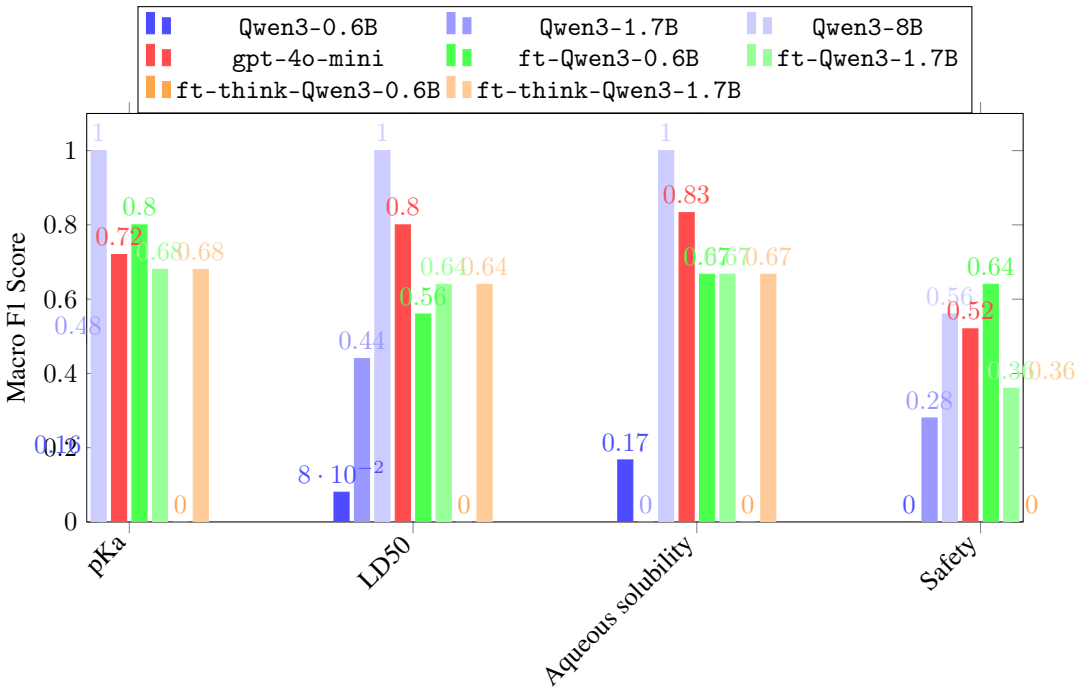

Figure 2: Macro F1 scores for MCQ tasks (pKa, LD50, Aqueous solubility, and Safety).