ChemMA: Multi-Agent Chemical Reasoning with Tools

Anonymous ACL submission

Abstract

Large language models show promise for chemistry tasks but often suffer from hallucination, inefficient retrieval, and invalid reasoning. Retrieval-Augmented Generation improves factual grounding, yet existing approaches rely on static retrieval and lack principled stopping criteria, leading to irrelevant context accumulation and poor multi-step reasoning.

We present **ChemMA**, an agentic retrieval-augmented framework for chemical reasoning. ChemMA integrates a Hybrid Perception module that enforces symbolic grounding against noisy inputs, a Planner that orchestrates dependency-aware actions rooted in structured memory for gap-driven reasoning, and an Executor that interfaces with a heterogeneous toolchain while filtering retrieval noise via embedded knowledge distillation. To ensure scientific rigor, we employ a dual-phase verification protocol where a Judge governs early termination based on information sufficiency, while a Jury performs final semantic consensus checks. This design enables adaptive retrieval and chemically valid reasoning while avoiding unnecessary information acquisition.

Experiments on four chemistry benchmarks totaling 1,932 questions show that **ChemMA** consistently outperforms traditional RAG baselines and significantly narrows the gap with proprietary-model systems using only open-source language models. These results highlight the effectiveness of sufficiency-aware agentic control combined with structured architectural constraints for reliable chemical reasoning.

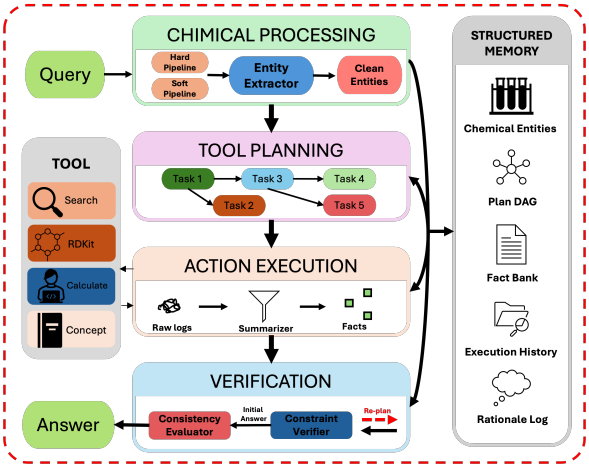


Figure 1: Our chemical multi-agent framework. Our framework grounds noisy inputs into a Structured Memory via hybrid processing, executes non-linear plans using a dependency-aware DAG, and enforces termination through a dual-phase verification mechanism (Constraint Verifier & Consistency Evaluator). This design ensures that reasoning is both chemically valid and logically sufficient.

1 Introduction

Recent years have seen a rapid expansion of efforts to adapt Large Language Models (LLMs) to the chemical domain (Han et al., 2025; Taylor et al., 2022; Zhang et al., 2024), with promising results on foundational tasks such as molecular description and chemical text understanding (Han et al., 2025). Many of these approaches implicitly assume that scaling exposure to chemical literature is sufficient to endow models with genuine chemical reasoning capabilities (Taylor et al., 2022; Zhang et al., 2024). In practice, however, chemistry question answering exposes clear limitations in current methods. Direct LLM inference often performs well on high-level conceptual questions, yet it is prone to hallucination on factual or structural queries—for example, producing syntactically valid but chemically invalid SMILES strings—and remains fundamentally constrained by its training cutoff. To mitigate these issues, Retrieval-Augmented Generation (RAG) has emerged as a widely adopted paradigm (Lewis et al., 2020; Izacard et al., 2022), augmenting LLMs with external knowledge sources by retrieving relevant documents—typically via cosine similarity in vec-

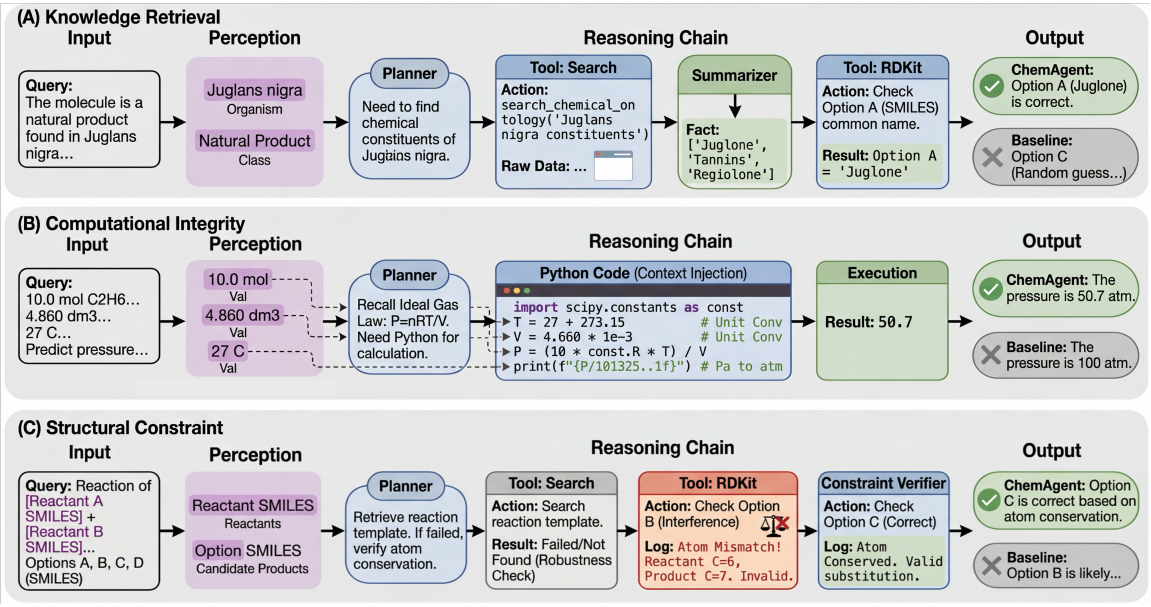


Figure 2: Qualitative Analysis of Reasoning Trajectories. We visualize the step-by-step execution of ChemMA across three distinct task modalities, contrasting it with direct LLM inference (Baseline). **(A) Knowledge Retrieval (Caption2mol)**: The agent employs *Hybrid Chemical Processing* to extract the organism entity, triggers *Adaptive Query Formulation* to retrieve constituents, and uses *Embedded Summarization* to distill the correct SMILES (Juglone) from noisy search results. **(B) Computational Integrity (SciBench)**: For numerical tasks, the agent extracts physical values and injects them into the *Python Computational Engine*, automatically handling unit conversion ($^\circ$C $\rightarrow$ K) and constant invocation (R) to ensure mathematical precision. **(C) Structural Constraint (Product Prediction)**: The *Constraint Verifier* employs RDKit to enforce atom conservation, successfully intercepting and rejecting a hallucinated candidate (Option B) that violates mass balance, triggering a valid selection.

tor databases—prior to generation (Lewis et al., 2020; Lee et al., 2024; Zhong et al., 2025). While RAG improves factual grounding, it does not fully resolve the challenges of reliable and principled chemical reasoning.

Despite these advances, a central challenge remains unresolved: chemical reasoning requires determining when the available information is sufficient to support a valid conclusion, while simultaneously respecting strict domain-specific constraints. Recent chemical RAG systems such as ChemRAG (Zhong et al., 2025) rely on static retrieval strategies, most commonly retrieving a fixed number of documents based on semantic similarity. However, semantic proximity does not imply causal relevance. Consequently, retrieved contexts often include chemically irrelevant but topically related information, which dilutes model attention and degrades performance under limited context windows. This rigidity further leads to inefficiencies: simple factoid questions incur unnecessary retrieval overhead, while complex, multi-step problems are inadequately supported by a single retrieval round. Recent agent-based frameworks attempt to address these limitations through iterative reasoning and dynamic tool use (Yao et al., 2023; Shen et al., 2024), including chemistry-specific agents like ChemCrow (M. Bran et al., 2024), CACTUS (McNaughton et al., 2024), and MT-Mol (Kim et al., 2025). When applied to chemistry, linear ReAct-style systems remain under-specified. They lack information sufficiency criteria, frequently violate chemical preconditions (e.g., invoking property calculators without validated structures), and fail to compose tools into coherent multi-step workflows (e.g., name normalization, database lookup, and computation). These shortcomings expose the lack of principled control over information acquisition and reasoning termination, causing error propagation and hallucinated tool inputs. Such failures severely degrade performance in structured domains like chemistry.

To address these limitations, we revisit Retrieval-Augmented Generation for chemical reasoning and propose an agentic RAG framework designed specifically for the structural and causal characteristics of chemistry. Rather than following standard execution flows or fixed retrieval schedules, our approach enables adaptive, sufficiency-aware reasoning that overcomes the deficiencies of static

RAG and unconstrained ReAct-style agents. Central to this design is a Judge agent that evaluates the current information state and enforces early termination once task-specific sufficiency criteria are satisfied, preventing unnecessary retrieval and the accumulation of irrelevant context. A Planner agent embeds intrinsic chemical constraints to translate natural language queries into valid operation sequences and enforces domain preconditions such as structure validation prior to property computation, thereby confining reasoning to a chemically admissible strategy space. This constraint-aware planning substantially reduces invalid tool invocations and low-level hallucinations. An Executor agent interfaces with external tools and databases to acquire evidence and provide execution feedback. Once information sufficiency is established, control is transferred to a Synthesiser, which consolidates the validated reasoning trace and retrieved evidence into a coherent final answer in the required format.

We evaluate our agentic RAG framework on four established chemistry benchmarks, namely MMLU-Chem, SciBench, ChemBench4K, and Mol-Instructions, comprising a total of 1,932 questions across diverse task types. Using local open-source models including Llama-3.1-8B, Qwen2.5-7B, and Llama-3.1-70B, our method achieves performance comparable to systems relying on proprietary large language models while consistently outperforming traditional RAG baselines. These results demonstrate that agentic control with sufficiency-aware termination and domain-valid reasoning can substantially improve chemical reasoning without increasing model scale. To facilitate reproducibility and future research on scientific agent systems, we publicly release our code, prompts, and evaluation data.

2 Related Work

2.1 Retrieval and Agent Frameworks for Question Answering

Retrieval-Augmented Generation (RAG) (Lewis et al., 2020) grounds LLM generation in retrieved documents, with dense retrieval models like Contriever (Izacard et al., 2022) enabling semantic matching beyond keyword overlap. ChemRAG (Zhong et al., 2025) extends this to chemistry QA with four benchmarks, showing that Top-5 document retrieval improves GPT-4 accuracy by +4.3% on MMLU-Chem. Recently, general NLP frameworks have explored agentic search (Li et al., 2025) and dynamic DAG-based multi-agent coordination (Liu et al., 2025). However, these systems use *static* retrieval—a single Top-k query—that cannot adapt to question complexity.

Tool-augmented agents introduce dynamic reasoning. ReAct (Yao et al., 2023) interleaves thought-action-observation cycles for iterative tool use, while Reflexion (Shinn et al., 2023) adds self-reflection for error correction. However, these frameworks lack *sufficiency modeling*—they use fixed iteration limits without evaluating whether information answers the question—and do not encode *domain constraints*, causing invalid tool calls (e.g., invoking RDKit without valid SMILES). We address this through explicit sufficiency predicates and chemistry-specific DAG planning constraints.

2.2 Multi-Agent Systems and Chemistry QA

Multi-agent systems decompose tasks across specialized roles. MetaGPT (Hong et al., 2023) assigns software engineering roles (PM, Architect, Engineer), while CAMEL (Li et al., 2023) uses role-playing for problem-solving. These target software/creative tasks; chemistry QA requires distinct capabilities: sufficiency evaluation, constraint checking, and tool composition.

Chemistry-specific systems include Galactica (Taylor et al., 2022) (120B scientific LLM with hallucination issues), MolT5 (Edwards et al., 2022) (molecule-text translation without reasoning), and recent tool-augmented agents like ChemCrow (M. Bran et al., 2024) , CACTUS (McNaughton et al., 2024), and MT-Mol (Kim et al., 2025). While these agents successfully integrate external utilities, they predominantly rely on linear reasoning paths and lack explicit sufficiency modeling. ChemRAG provides benchmarks but uses static retrieval without tool composition or adaptive termination. We contribute a multi-agent architecture with specialized roles (JUDGE, PLANNER, EXECUTOR, SYNTHESIZER) that achieves +4.9% over Llama-3.1 + ChemRAG baseline through sufficiency-driven reasoning and domain-aware constraints.

3 Methodology

We introduce **ChemMA**, a domain-adaptive multi-agent framework designed to enhance the chemical reasoning capabilities of large language models. As illustrated in Figure 1, given a chemical query q, the system aims to derive a verifiable an-

swer a^* through iterative interaction with a heterogeneous tool environment $\mathcal{E}$. The framework comprises four hierarchically organized modules: (1) **Chemical Processing** (Sec. 3.1) for grounding noisy inputs into structured entities; (2) **Tool Planning** (Sec. 3.2) for orchestrating non-linear reasoning via a dependency-aware DAG and a heterogeneous toolchain; (3) **Structured Memory** (Sec. 3.3) for maintaining a summarized state of facts and execution history; (4) **Multi-agent Verification** (Sec. 3.4) for enforcing sufficiency and consensus.

3.1 Chemical Processing

Chemical queries often contain multi-modal entities (e.g., SMILES strings, IUPAC names, physical constants) that challenge the tokenization limits of LLMs. To mitigate hallucination at the input stage, we propose a **Hybrid Entity Extraction** mechanism that decouples structural validation from semantic understanding. We define an entity ontology set $\Omega = \{\omega_{\text{struct}}, \omega_{\text{sem}}\}$, where structural entities ω_{struct} (e.g., SMILES, Reaction strings) and semantic entities ω_{sem} (e.g., macroscopic properties, scientific terms) are processed via dual pipelines: **(1) Hard Constraint Pipeline ($\mathcal{P}_{\text{hard}}$):** Uses regular expressions and chemo-informatics tools (e.g., RDKit (Landrum et al., 2013)) to extract and validate molecular structures. Invalid strings (e.g., unbalanced reaction tokens like `»redients`) are intercepted and masked before reaching the LLM to prevent structural hallucination. **(2) Soft Grounding Pipeline ($\mathcal{P}_{\text{soft}}$):** Employs an LLM to extract semantic entities. To ensure fidelity, we implement a *Source Grounding* mechanism where extracted values (e.g., physical constants) are strictly matched against the raw query q, normalizing formats to ensure consistency.

The output is a structured entity set $E_q = \mathcal{P}_{\text{hard}}(q) \cup \mathcal{P}_{\text{soft}}(q)$. Rather than passing this transiently to the planner, E_q is used to initialize the Structured Memory $\mathcal{M}_0$, ensuring that these grounded constraints remain accessible to all agents (Planner, Executor, Judge) throughout the reasoning lifecycle.

3.2 Tool Planning

Standard agents often fail in complex chemical tasks due to linear execution bottlenecks and lack of domain adaptation. We address this through a specialized planning layer that orchestrates a heterogeneous tool environment.

3.2.1 Dependency-Aware DAG Generation

The Planner π functions as the central controller. Conditioned on the query q, entity set E_q, and structured memory $\mathcal{M}_t$, it generates a plan graph $G_t = (V, E)$. Unlike linear chains, this is a Directed Acyclic Graph (DAG) where nodes V represent atomic tool actions and edges E represent data dependencies (e.g., calculating molar mass depends on retrieving the molecular formula). This topology enables greedy parallel execution of independent nodes and dynamic pruning of failed branches.

3.2.2 Tool Environment

To ground reasoning in verifiable reality, the Planner selects actions from a domain-specific environment $\mathcal{E}$ comprising four distinct tool classes: **(1) Hybrid Retrieval (SearXNG & Local KB):** We deploy a dual-source engine. A local vector database indexes standard textbooks for retrieving precise definitions. For empirical data, we utilize SearXNG to aggregate results from scientific sources (PubMed (Kim et al., 2023), Google Scholar), prioritizing causality over keyword matching. **(2) Symbolic Verification (RDKit):** Serves as a rigorous chemo-informatics validator to enforce atom conservation and verify the syntactic validity of generated SMILES strings. **(3) Computational Engine (Python Sandbox):** For numerical reasoning, the Planner generates executable code pre-loaded with physical constant libraries (e.g., `scipy.constants`), ensuring answers are mathematically derived rather than predicted tokens. **(4) Local Knowledge Lookup:** Provides low-latency access to static chemical data (e.g., periodic table properties).

3.2.3 Execution with Adaptive Retrieval and Summarization

The Executor interfaces with the environment to fulfill the plan. To adapt LLMs to complex scientific workflows and manage information density, it employs two key mechanisms:

Adaptive Query Formulation. Raw user queries are rarely optimal for tool execution. The Executor dynamically rewrites the Planner's intent into tool-specific arguments. For instance, when querying reaction conditions, it transforms the SMILES representation `A.B»C` into a semantic search string (e.g., `"synthesis of [Product Name] from [Reactant Name] yield"`). This ensures high-recall retrieval.

Embedded Knowledge Summarization. Directly processing raw web content (e.g., extensive HTML pages) can introduce noise and irrelevant details. To address this, we introduce an **Embedded Knowledge Summarizer**. For every observation o_t^{raw} returned by a tool, the Executor applies a lightweight summarization function $f_{\text{summ}}(o_t^{\text{raw}}) \rightarrow o_t^{\text{fact}}$. This function extracts only the atomic facts relevant to the query node (e.g., extracting a specific boiling point value while discarding navigational text) before writing to the Fact Bank. This ensures that the memory $\mathcal{M}$ remains compact and high-signal, preventing context pollution.

3.3 Structured Memory

Effective planning requires maintaining a comprehensive state across non-linear execution paths. Instead of a raw linear chat history, CHEMMA maintains a **Structured Memory State** $\mathcal{M}_t$, formally defined as a tuple $(\mathcal{E}_q, \mathcal{G}_t^{(k)}, \mathcal{F}_t, \mathcal{H}_t, \mathcal{R}_t)$: **(1) Grounded Entities ($\mathcal{E}_q$):** A read-only registry of structured entities (e.g., validated SMILES, physical constants) extracted during the Chemical Processing phase. This serves as the source of truth for tool arguments, ensuring that planners always reference verified inputs rather than hallucinated tokens. **(2) Iterative Plan Graph ($\mathcal{G}_t^{(k)}$):** The topological state of the reasoning DAG for the current iteration cycle k. It acts as a versioned blueprint, tracking the execution status of nodes and their dependencies. By indexing plans by cycle k, the system maintains a lineage of reasoning strategies. **(3) Fact Bank ($\mathcal{F}_t$):** A dynamic, key-value knowledge store containing summarized atomic facts (e.g., `Density(Ethanol) = 0.789 g/mL`) acquired from external tools. This isolates high-signal evidence from noise. **(4) Execution History ($\mathcal{H}_t$):** A chronological log of tool invocations, raw outputs, and error messages. This enables the detection of reasoning loops or repeated failures across iterations. **(5) Cognitive Rationale Log ($\mathcal{R}_t$):** A discursive storage acting as the system's **"Inner Monologue."** unlike the Execution History which records *what* happened (tool outputs), $\mathcal{R}_t$ records *why* decisions were made. It captures the **Planner's Strategic Intent** (e.g., "Strategy: Decompose yield calculation into theoretical vs. experimental mass lookup") and the **Judge's Evaluative Reasoning** (e.g., "Rejection: Found density but missing volume; insufficient for mass calculation"). This persistent dialogue space allows the Jury to reconstruct the complete logical arc behind the derived facts.

This structured state serves as the shared context for all agents, enabling the Judge to evaluate sufficiency explicitly against the query requirements ($\mathcal{E}_q$) using the accumulated evidence ($\mathcal{F}_t$).

3.4 Multi-agent Verification

To ensure both physical validity and semantic coherence, we employ a two-stage verification mechanism that explicitly separates *constraint satisfaction* from *semantic consistency*. This design avoids reliance on implicit model confidence and enforces correctness through explicit checks over accumulated evidence.

Constraint Verifier. The Constraint Verifier serves as a deterministic gatekeeper for the reasoning process. Its primary role is to assess **information sufficiency**, i.e., whether the current Fact Bank $\mathcal{F}_t$ contains adequate and valid evidence to support a conclusive answer to the query q. Rather than relying on confidence-based heuristics, the verifier applies an explicit checklist to evaluate answerability: **(1) Sufficiency Check:** Does $\mathcal{F}_t$ cover all required reasoning steps? For example, in retrieval tasks, has the target value been extracted from a reliable source; in calculation tasks, are all necessary variables defined and numerically resolved? **(2) Constraint Check:** Do any candidate facts violate domain invariants? This includes rejecting chemically invalid structures (e.g., molecules failing RDKit valency checks) and flagging numerical results that are inferred rather than explicitly computed. If missing evidence is detected ($\Delta \neq \emptyset$) or any constraint is violated, the Constraint Verifier rejects the current trajectory and records a structured feedback signal δ_{err} in memory, triggering a corrective replanning step.

Consistency Evaluator. Once constraint verification is satisfied, the Consistency Evaluator performs a final semantic validation. This stage examines whether the verified facts jointly support a coherent and logically sound response, free of internal contradictions or unsupported extrapolations. The evaluator operates through two complementary functions: **(1) Answer Synthesis:** A candidate answer a' is generated conditioned on the full memory state $\mathcal{M}_t$, integrating distilled evidence from the Fact Bank $\mathcal{F}_t$ with the recorded reasoning rationale $\mathcal{R}_t$. **(2) Consistency Assessment:** The synthesized answer is examined for logical gaps, factual inconsistencies, and claims not grounded in

Algorithm 1 ChemMA Reasoning Procedure

Require: Query q, Tools $\mathcal{T}$, Max iterations K
Init: $E_q \leftarrow \text{ExtractEntities}(q)$; $\mathcal{M}_0 \leftarrow \{\text{Facts} : \emptyset, \text{Traj} : \emptyset, \text{Graph} : \emptyset\}$
for $k \leftarrow 1$ **to** K **do**
  $G_k \leftarrow \textsc{Planner}(q, E_q, \mathcal{M}_{k-1})$
  if G_k is empty $\vee$ sufficient **then**
    break
  end if
  for node $v \in G_k$ **in parallel do**
    if Dependencies (v) are met **then**
      $o^{\text{fact}} \leftarrow \textsc{Summarize}(\textsc{ToolCall}(v, \mathcal{T}))$
      Update $\mathcal{M}_k$ with o^{fact}
    else
      Prune dependent branch
    end if
  end for
  status, $\delta_{err} \leftarrow \textsc{ConstraintVerifier}(\mathcal{M}_k, q)$
  if status is REJECT **then**
    Update $\mathcal{M}_k$ with δ_{err}; **continue**
  end if
end for
return $\textsc{ConsistencyEvaluator}(\mathcal{M}_K, q)$

verified evidence. The final answer a^* is produced only when the Consistency Evaluator confirms that the response is both semantically coherent and fully supported by the validated facts.

The complete deliberation process is formalized in Algorithm 1.

4 Experimental Setup

Datasets. We evaluate on CHEMRAG-BENCH, comprising 1,932 questions across four benchmarks: MMLU-Chem (Hendrycks et al., 2021) (303) College/high school chemistry MCQs, SciBench-Chem (Wang et al., 2023) (229) Open-ended numerical calculations (thermodynamics, quantum, physical chemistry), ChemBench4K (Zhang et al., 2024) (800) MCQs covering 8 task types (Name Conversion, Mol2Caption, Caption2Mol, Reaction Prediction, etc.), and Mol-Instructions (Fang et al., 2024) (600) comprising molecular generation (400), text generation (100), and numerical prediction (100).

Baselines. We compare against two categories of baselines: (1) *Direct Inference*, where different LLMs generate answers without external tools or retrieval; and (2) *ChemRAG* (Zhong et al., 2025), a retrieval-augmented generation baseline that uses a Contriever-based retriever with the top-$k = 5$ documents provided as context.

ChemAgent Variants (Ablation Settings). To deconstruct the architectural benefits of our framework, we evaluate four internal variants (E3–E6) against the baselines: (1) *Internal Reasoner (E3)*: Enables Perception and Reflection but disables external tools to isolate System 2 reasoning capabilities; (2) *Linear Agent (E4)*: A standard linear tool-use agent based on the ReAct paradigm (Yao et al., 2023). This variant serves as a structural proxy for existing chemistry agents like ChemCrow (M. Bran et al., 2024), which rely on linear reasoning paths and external tools but lack our explicit dual-phase verification loop; (3) *Cyclic Agent (E5)*: Introduces the CONSTRAINTVERIFIER to enforce hard domain constraints within a basic iterative loop; (4) *Full Framework (E6)*: The complete architecture upgrading the planner to a Dependency-Aware DAG and including the CONSISTENCYEVALUATOR for semantic consensus.

Evaluation Metrics. (1) **Reasoning Accuracy:** For multiple-choice MCQs (MMLU, ChemBench), we report exact match accuracy. For numerical tasks (SciBench, Mol-Instruct properties), we report accuracy with a strict tolerance threshold ($\pm 5\%$); (2) **Molecular Generation:** Validity (RDKit parsability) and Fingerprint Similarity (Tanimoto coefficient averaged over MACCS, RDKit, and Morgan); (3) **Text Generation:** BLEU-4 and ROUGE-L to assess semantic overlap.

5 Results and Analysis

5.1 Main Results

Table 1 presents the performance of various models across reasoning-intensive benchmarks. CHEMMA achieves substantial improvements over both Direct Inference and Static RAG baselines across diverse model backends.

Democratizing Scientific AI via Agentic Scaffolding. A central finding is that agentic scaffolding effectively compensates for raw parameter scale. Notably, when equipped with CHEMMA, the 7B-parameter Qwen-2.5 achieves an average score of 50.44%, not only outperforming peer open-weight models but also surpassing the zero-shot baseline performance of the proprietary GPT-4o (46.35%). Similarly, GPT-3.5 augmented with our framework reaches 52.19%, approaching the performance of OpenAI's o1 model (56.09%). While frontier models augmented with retrieval (e.g., o1 + ChemRAG) still represent the upper bound, this demonstrates that rigorous tool-use and dual-phase verification provide a viable pathway for democratizing expert-level scientific reasoning on resource-constrained hardware.

Model	MMLU	SciBench	ChemB.	Avg.
Baseline (No Retrieval)				
Llama-3.1-8B	42.90	3.30	27.25	24.48
Llama-3.1-70B	62.38	5.99	24.25	30.87
ChemLLM	37.62	8.72	23.50	23.28
DeepSeek-R1-8B	55.44	3.09	35.38	31.30
GPT-3.5	49.17	9.66	30.50	29.78
GPT-4o	74.59	4.97	59.50	46.35
o1	85.81	40.82	41.63	56.09
ChemRAG (Static Retrieval)				
Llama-3.1-8B	52.15	3.56	25.88	27.20
Llama-3.1-70B	61.05	13.63	26.25	33.64
ChemLLM	42.57	0.00	11.13	17.90
DeepSeek-R1-8B	43.23	2.03	16.75	20.67
GPT-3.5	57.43	3.87	29.13	30.14
GPT-4o	73.92	8.59	67.25	49.92
o1	85.48	43.61	58.38	62.49
CHEMMA *(Ours)*				
Llama-3.1-8B	62.38	18.78	56.88	46.01
Qwen-2.5-7B	67.66	13.54	70.13	**50.44**
ChemLLM	51.81	9.17	48.13	36.37
DeepSeek-R1-8B	66.67	11.35	57.88	45.30
GPT-3.5	65.36	20.09	71.13	**52.19**

Table 1: Accuracy (%) on reasoning-intensive benchmarks. CHEMMA fits efficient reasoning into a compact framework. *ChemB. denotes ChemBench4k.

The Retrieval Paradox. Our results explicitly highlight the limitations of static retrieval. For models like ChemLLM and DeepSeek-R1-8B, naive static retrieval (ChemRAG) severely degrades performance compared to the baseline (e.g., ChemLLM drops from 23.28% to 17.90%). This confirms the *Retrieval Paradox*: indiscriminately injecting top-k documents introduces semantic noise that distracts the model's reasoning process. CHEMMA resolves this by utilizing the Constraint Verifier to filter out irrelevant contexts and enforce domain rules.

The "Alignment Tax" in Agentic Workflows. An intriguing and counter-intuitive finding emerges from the performance of DeepSeek-R1-8B. While it significantly outperforms Llama-3.1-8B in direct inference (31.30% vs. 24.48%) due to its intrinsic Chain-of-Thought (CoT) capabilities, it underperforms Llama-3.1-8B when integrated into CHEMMA (45.30% vs. 46.01%). We hypothesize this is due to an *Alignment Tax* in agentic workflows: reasoning models optimized for free-form `<think>` generation frequently conflict with the strict JSON formatting and rigid structural constraints required by our Planner and Verifier agents. This suggests that current reasoning models struggle to balance internal probabilistic CoT with external deterministic tool execution, highlighting a critical area for future agent alignment research.

Evaluation Scope. Due to the multi-turn nature of the CHEMMA framework (Perception-Cognition-Action-Reflection cycle), evaluating proprietary models like GPT-4o or o1 across 1,932 questions incurs prohibitive API costs and strict rate limits. Therefore, our primary evaluation focuses on open-weight models ($\leq$8B) and highly accessible models (GPT-3.5) to demonstrate the efficacy of agentic scaffolding. Future work will explore the scaling behavior of CHEMMA on frontier models.

5.2 Generative Capabilities

Table 3 details performance on Mol-Instructions. This benchmark highlights the impact of **Hard Constraints** on generation quality.

Enforcing Structural Validity. The most striking improvement is in molecular validity. The baseline Llama-3.1-8B generates chemically invalid strings in nearly half of the cases (e.g., Forward Reaction Prediction validity: 0.53).CHEMMA achieves near-perfect validity (**0.98-0.99**) across all structure tasks. This provides direct evidence for the efficacy of the **Constraint Verifier**.

Retrieval-Augmented Fidelity. For *Description Guided Molecule Design*, the Fingerprint Similarity improves from 0.24 (Baseline) to **0.68** (Ours). While the baseline attempts to hallucinate structures from latent space, CHEMMA effectively retrieves ground-truth or highly similar analogues from the chemical ontology, bridging the gap between semantic description and chemical structure.

Agent Variant (Llama-3.1-8B)	SciBench	ChemB.	Avg.
E3: Internal Reasoner (No Tools)	3.30	27.25	15.27
E4: Linear Agent (ReAct-style)	10.48	38.38	24.43
E5: Cyclic Agent (+ Constraint Verifier)	16.16	52.00	34.08
E6: Full Framework (+ Consistency)	**18.78**	**56.88**	**37.83**

Table 2: Ablation study isolating the impact of tools, hard constraints, and semantic consensus.

5.3 Ablation Study

Table 2 ablates our architectural components using Llama-3.1-8B (variants defined in Sec. 4; de-

Subtask	Val. ↑		Fing. ↑		B-4 ↑		Acc. ↑	
	Base	**Ours**	Base	**Ours**	Base	**Ours**	Base	**Ours**
Desc. Guided Design	0.57	**0.99**	0.24	**0.68**	0.09	**0.25**	–	–
Forward Reaction	0.53	**0.98**	0.39	**0.75**	0.17	**0.35**	–	–
Reagent Prediction	0.47	**0.95**	0.07	**0.55**	0.02	**0.20**	–	–
Retrosynthesis	0.65	**0.98**	0.30	**0.62**	0.21	**0.41**	–	–
Mol. Desc. Gen.	–	–	–	–	0.05	**0.18**	–	–
Property Prediction	–	–	–	–	–	–	0.00	**0.09**

Table 3: Detailed performance on Mol-Instructions. "Base" denotes Direct Inference (Llama-3.1-8B). **Val.**: RDKit parsability; **Fing.**: Avg Tanimoto similarity; **B-4**: BLEU-4; **Acc.**: Accuracy (± 0.05).

tails in App. B.2). Comparing the tool-free reasoner (E3) to a standard ReAct baseline equipped with our toolset (E4) confirms that chemical calculations require external tools. However, E4 frequently fails due to hallucinated SMILES or invalid code. Adding the **Constraint Verifier** (E5) yields the largest accuracy gain (+13.7% on ChemBench4K) by acting as a deterministic gatekeeper that rejects invalid intermediate steps and triggers replanning. The full framework (E6) introduces the **Dependency-Aware DAG Planner** and the **Consistency Evaluator**. Unlike E5's linear cyclic loop, E6 employs a DAG topology to manage complex tool dependencies and resolve execution bottlenecks. Finally, the evaluator enforces semantic consensus between retrieved facts and the synthesized answer, securing the remaining performance improvements.

6 Discussion

Symbolic Execution vs. Probabilistic Estimation. The qualitative leap in SciBench performance (from 3.30% to 18.78%) validates a fundamental hypothesis: LLMs are inherently probabilistic engines ill-suited for deterministic arithmetic. Direct inference fails not due to a lack of chemical knowledge, but due to *computational hallucination* (e.g., unit inconsistencies). By forcing the Planner to map physical values into a **Python Computational Engine**, CHEMMA enforces **Computational Integrity**.

The Retrieval Paradox: Quality over Quantity. A counter-intuitive finding on ChemBench is that standard ChemRAG underperforms the baseline (25.88% vs. 27.25%). This highlights the *Retrieval Paradox*: indiscriminately injecting top-k documents introduces semantic noise that distracts smaller models. CHEMMA reverses this trend (achieving 56.88%) through two mechanisms: (1) **Hybrid Perception** cleanses input noise, and (2) the **Constraint Verifier** acts as an information bottleneck, rejecting irrelevant retrieval results. This confirms that for domain-specific reasoning, the *validity* of context is far more critical than the *volume* of retrieval.

Architecture as a Scaling Multiplier. Our results with Llama-3.1-8B and Qwen-2.5-7B demonstrate that agentic scaffolding can effectively compensate for model scale. By externalizing memory (Search) and verification (RDKit), CHEMMA enables 7B/8B models to achieve reasoning capabilities previously reserved for proprietary large models. This offers a viable path for **Democratizing Scientific AI**, allowing sophisticated chemical reasoning to run locally on consumer-grade hardware with high reliability.

7 Conclusion

We presented CHEMMA, a domain-adaptive framework enabling open-weight Language Models to perform expert-level chemical reasoning. To overcome the error propagation and rigidity of static RAG and linear ReAct-style agents, CHEMMA employs two architectural mechanisms: (1) a **Dependency-Aware DAG Planner** for non-linear, parallel tool execution, and (2) a **Dual-Phase Verification** protocol that enforces chemo-informatics constraints and semantic consensus prior to termination.

Evaluations across 1,932 questions demonstrate that CHEMMA outperforms standard baselines (+21% average gain over direct inference) and achieves near-perfect structural validity (98-99%) on generation tasks. These results advocate shifting scientific agents from unconstrained retrieve-then-generate"pipelines to averify-then-conclude" approach, grounding AI-driven discovery in physical laws and verifiable evidence.

Limitations

Inference Latency and Token Overhead. Although CHEMMA improves reasoning accuracy, transitioning from direct inference to a multi-agent DAG topology introduces computational overhead. The Perception-Cognition-Action-Reflection cycle—specifically the parallel execution of DAG tool nodes—requires multiple concurrent LLM calls. This increases token consumption and peak memory usage compared to linear baselines, making the framework unsuitable for real-time appli-

cations requiring millisecond-level latency. Future work will explore *trajectory distillation*—fine-tuning a monolithic model on the agent's non-linear reasoning traces to amortize inference costs.

Dependency on Tool Availability. CHEMMA enforces **Computational Integrity** by offloading reasoning to external tools (SearXNG, RDKit). Consequently, performance remains bounded by tool coverage and API stability. For example, the agent cannot answer if a reaction yield is absent from public databases or if a property requires density functional theory (DFT) calculations beyond standard RDKit descriptors. Unlike parametric hallucinations, this yields a "refusal to answer"; while safer for scientific domains, it restricts the system's scope to retrievable or computable knowledge

Ethics Statement

This work focuses on enhancing the chemical reasoning capabilities of LLMs for scientific discovery. However, we acknowledge the potential dual-use risks associated with chemistry agents, such as the automated planning of toxic or hazardous substances. To mitigate these risks, our framework relies on open-source models that have undergone safety alignment. Furthermore, our Constraint Verifier is designed to enforce physical and chemical validity, but it does not currently screen for malicious intent. Future deployments of such agentic systems in real-world laboratories must incorporate strict safety guardrails and toxicity screening APIs to prevent misuse.

References

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 (EMNLP)*, pages 375–413.

Yin Fang, Xiaozhuan Liang, Ningyu Zhang, Kangwei Liu, Rui Huang, Zhuo Chen, Xiaohui Fan, and Huajun Chen. 2024. Mol-instructions: A large-scale biomolecular instruction dataset for large language models. In *The Twelfth International Conference on Learning Representations*.

Yang Han, Ziping Wan, Lu Chen, Kai Yu, and Xin Chen. 2025. From generalist to specialist: A survey of large language models for chemistry. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 1106–1123.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. 2023. Metagpt: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*.

Hyomin Kim, Yunhui Jang, and Sungsoo Ahn. 2025. Mt-mol: Multi agent system with tool-based reasoning for molecular optimization. *Artificial Intelligence Repository*.

Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, and Bo Yu. 2023. PubChem 2023 update. *Nucleic Acids Research*, 51(D1):D1373–D1380.

Greg Landrum, Paolo Tosco, Brian Kelley, and RDKit Core Team. 2013. RDKit: Open-source cheminformatics. http://www.rdkit.org. Accessed: 2024-11-17.

Seul Lee, Karsten Kreis, Srimukh Veccham, Meng Liu, Danny Reidenbach, Saee Paliwal, Arash Vahdat, and Weili Nie. 2024. Molecule generation with fragment retrieval augmentation. *Advances in Neural Information Processing Systems*, 37:132463–132490.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. CAMEL: Communicative agents for "mind" exploration of large language model society. *Advances in Neural Information Processing Systems*, 36.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025. Search-o1: Agentic search-enhanced large reasoning models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 5420–5438.

Yuhan Liu, Cong Xu, Lu Liu, Yihua Wang, Feiyu Chen, Qi Jia, Yaqian Zhao, Zhichun Wang, and Xiang Li.

2025. DeMAC: Enhancing multi-agent coordination with dynamic DAG and manager-player feedback. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 14072–14098, Suzhou, China. Association for Computational Linguistics.

Andres M. Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D. White, and Philippe Schwaller. 2024. Augmenting large language models with chemistry tools. *Nature Machine Intelligence*, 6(5):525–535.

Andrew D McNaughton, Gautham Krishna Sankar Ramalaxmi, Agustin Kruel, Carter R Knutson, Rohith A Varikoti, and Neeraj Kumar. 2024. Cactus: Chemistry agent connecting tool usage to science. *ACS omega*, 9(46):46563–46573.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2024. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36.

Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. 2022. Galactica: A large language model for science. *arXiv preprint arXiv:2211.09085*.

Xiaoxuan Wang, Ziniu Hu, Pan Lu, Yanqiao Zhu, Jieyu Zhang, Satyen Subramaniam, Arjun Loomba, Shichang Zhang, Yizhou Sun, and Wei Wang. 2023. SCIBENCH: Evaluating college-level scientific problem-solving abilities of large language models. In *The 3rd Workshop on Mathematical Reasoning and AI at NeurIPS'23*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Di Zhang, Wei Liu, Qian Tan, Jingdan Chen, Hang Yan, Yuliang Yan, Jiatong Li, Weiran Huang, Xin Jiang, and Qun Liu. 2024. Chemllm: A chemical large language model. *arXiv preprint arXiv:2402.06852*.

Xianrui Zhong, Bowen Jin, Siru Ouyang, Yanzhen Shen, Qiao Jin, Yin Fang, Zhiyong Lu, and Jiawei Han. 2025. Benchmarking retrieval-augmented generation for chemistry. *arXiv preprint arXiv:2505.07671*.

A Implementation Details

A.1 System Prompts Blueprint

To ensure reproducibility and enforce the architectural constraints described in Section 3, we utilize structured system prompts for the three core reasoning agents. Unlike open-ended chat instructions, our prompts impose strict JSON output schemas and explicit checklist constraints (e.g., the Verifier's hard rules) to stabilize the behavior of 8B-parameter models.

Figure A1 visualizes the blueprint of these prompts.

A.2 Tool Interface Specifications

Table A1 details the input/output schemas for the 13 core tools integrated into the ChemAgent environment. These tools are categorized by their underlying mechanism: Hybrid Retrieval (SearXNG/VectorDB), Chemo-Informatics (RDKit), Computational Engine (Python), and Local Lookup.

A.3 Experimental Configuration

To demonstrate the accessibility and efficiency of our framework (Democratizing Scientific AI), all experiments were conducted on local consumer-grade hardware rather than high-end server clusters. To ensure strict reproducibility of our evaluation, all SearXNG retrieval results during the benchmark testing were cached.

- **Hardware:** Apple M4 Pro Chip (48GB Unified Memory).
- **Model Serving:** Ollama framework with `OLLAMA_NUM_PARALLEL=4` to enable concurrent inference for the DAG Executor.
- **Local Model Suite:** To demonstrate architectural generalization, we evaluated a diverse set of open-source models (all deployed locally with 4-bit quantization to fit consumer hardware):
  - **Primary:** `Meta-Llama-3.1-8B-Instruct` (Used for all ablation studies).
  - **Generalist:** `Qwen-2.5-7B-Instruct`, `DeepSeek-R1-Distill-Llama-8B`.
  - **Specialist:** `ChemLLM-7B`.
  - **Large Scale:** `Meta-Llama-3.1-70B-Instruct` (Running sequentially due to memory constraints).
- **Hyperparameters:** Temperature is set to 0.1 for the Planner and Constraint Verifier to ensure deterministic behavior, and 0.3 for the Consistency Evaluator to allow for fluent synthesis. The maximum planning depth is set to $K = 3$.

B Additional Analysis

B.1 Error Analysis

We manually inspected 100 failed cases from the ChemBench4K dataset to categorize the remaining failure modes of ChemMA.

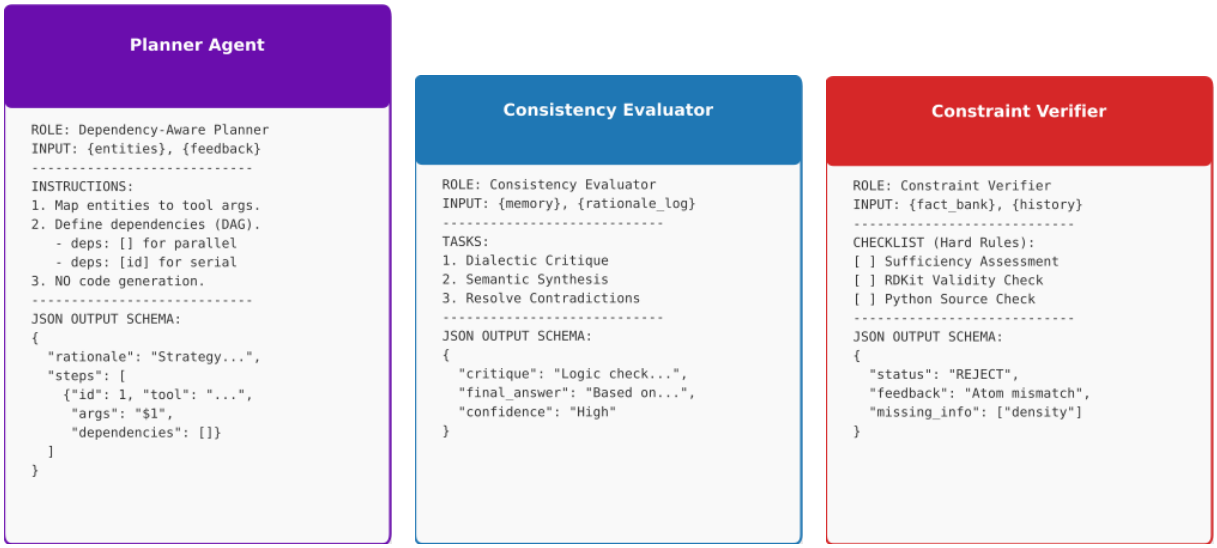
- **Retrieval Void (49%):** The specific reaction yield or property does not exist in the indexed web corpus (e.g., obscure unpublished compounds).
- **Extraction Failure (21%):** The Hybrid Perception module failed to parse highly non-standard input formats (e.g., ASCII art structures).
- **Planner Loop (8%):** The Planner entered a loop of trying alternative search terms without success, eventually hitting the step limit.
- **Other (22%):** Miscellaneous errors including timeouts or ambiguous ground truth.

B.2 Detailed ChemBench4K Ablation

Tables B2 and B3 present the detailed accuracy breakdown for each subtask within the ChemBench4K benchmark for Llama-3.1-8B and Qwen-2.5-7B, respectively. This granular analysis demonstrates how ChemAgent's multi-layered architecture addresses specific challenges (e.g., knowledge retrieval, constraint validation) in different chemical reasoning tasks.

B.3 Detailed ChemBench4K Breakdown

We provide a granular performance analysis on the 8 subtasks of ChemBench4K. Tables B2 and B3 compare the Direct Inference (Baseline) against our fully augmented ChemMA (E6) framework.



(a) **Planner Agent** (b) **Constraint Verifier** (c) **Consistency Evaluator**

Figure A1: Structural Blueprint of System Prompts. **(a)** The Planner focuses on dependency-aware DAG generation and entity grounding. **(b)** The Verifier implements hard constraints via a checklist mechanism. **(c)** The Evaluator focuses on dialectic synthesis and consensus. All agents act via strict JSON interfaces.

Category	Tool Name	Core Functionality	Underlying Mechanism
Retrieval	`QueryTextbookKB`	Retrieve definitions/laws	Local ChromaDB (OpenStax Chemistry)
	`SearchChemicalOntology`	Search species/structures	SearXNG + PubChem/Wikipedia Parsing
	`SearchReactionDetails`	Search reaction conditions	SearXNG + Adaptive Query Expansion
	`SearchStructureProperty`	Search physical properties	SearXNG + Site-specific Filtering
Chemo-Info	`SMILESProcessor`	Validate & Canonicalize	RDKit `MolFromSmiles` + `SaltRemover`
	`MoleculeAnalyzer`	Extract Scaffolds/Groups	RDKit `MurckoScaffold` + Descriptors
	`ReactionValidator`	Check Atom Conservation	RDKit `AtomMapping` (Judge Tool)
	`IonicBuilder`	Balance Charge	Rule-based Assembly for Inorganic Salts
Compute	`PythonCalculator`	Numerical Calculation	Python Sandbox (pre-loaded `scipy.constants`)
	`EquationBalancer`	Balance Equations	Linear Algebra Matrix Solver
	`SymbolicSolver`	Calculus/Series	`sympy` Symbolic Engine
Lookup	`LocalConstLookup`	Get Physical Constants	JSON Key-Value Store (NIST Data)
	`NameConverter`	Name $\leftrightarrow$ SMILES	PubChem API (Exact Match Priority)

Table A1: Complete specifications of the heterogeneous toolchain in CHEMAGENT. The toolset is designed to offload specific cognitive deficiencies (e.g., calculation, memorization) to deterministic engines.

Subtask	Base	Ours (E6)	Gain
Caption2mol	38.0%	**55.0%**	+17.0%
Mol2caption	86.0%	**88.0%**	+2.0%
Name Conversion	52.0%	**94.0%**	+42.0%
Product Prediction	32.0%	**51.0%**	+19.0%
Retrosynthesis	35.0%	**97.0%**	+62.0%
Solvent Prediction	30.0%	**50.0%**	+20.0%
Temperature Pred.	45.0%	**53.0%**	+8.0%
Yield Prediction	39.0%	**61.0%**	+22.0%
Average	44.6%	**68.6%**	**+24.0%**

Table B2: Detailed breakdown on **Llama-3.1-8B**. CHEMMA achieves massive gains in complex structural tasks like Name Conversion and Retrosynthesis.

Subtask	Base	Ours (E6)	Gain
Caption2mol	50.0%	**61.0%**	+11.0%
Mol2caption	93.0%	**96.0%**	+3.0%
Name Conversion	68.0%	**92.0%**	+24.0%
Product Prediction	42.0%	**65.0%**	+23.0%
Retrosynthesis	26.0%	**100.0%**	+74.0%
Solvent Prediction	26.0%	**44.0%**	+18.0%
Temperature Pred.	18.0%	**49.0%**	+31.0%
Yield Prediction	28.0%	**54.0%**	+26.0%
Average	43.9%	**70.1%**	**+26.2%**

Table B3: Detailed breakdown on **Qwen-2.5-7B**. The framework effectively compensates for weaker parametric knowledge in empirical tasks (e.g., Temperature).