Internship Report

AutoSMILES: An Agentic Framework for Property-Directed Molecular Design Using Large Language Models

Anthony LING

Grenoble INP - Phelma

Master MNIS

Internship conducted at

IBM - Almaden Research Center

California, USA

Supervisors :

Sara CAPPONI

Nathaniel PARK

September 3, 2025

Host Institution Presentation

Research Laboratory Overview

This research was conducted at IBM Almaden Research Center, located in San Jose, California, United States. IBM Almaden is one of IBM's twelve global research laboratories and has been a leading center for scientific discovery since its establishment in 1986.

Research Focus Areas

IBM Almaden Research Center specializes in several key areas of technological innovation:

- Artificial Intelligence and Machine Learning systems
- Hybrid cloud computing and storage solutions
- Quantum computing research and development
- Cybersecurity and data privacy protection
- Sustainable energy and renewable materials
- Healthcare technologies and life sciences
- Semiconductor research and nanotechnology

Scientific Impact and Achievements

IBM Almaden Research Center has produced thousands of scientific publications and patents. The researchers regularly present their work at major international conferences and maintain partnerships with leading universities and industry organizations worldwide. The center's contributions have advanced many fields, from basic scientific understanding to practical applications in technology and medicine.

Supervisor Information

IBM Manager: Sara CAPPONI
IBM Mentor: Nathaniel PARK
School Supervisor: Katell MORIN-ALLORY
Project Duration: March 2025 to September 2025

Abstract

Résumé

La découverte de nouvelles molécules avec des propriétés spécifiques représente un défi majeur en chimie moderne et en développement de médicaments. Les méthodes traditionnelles nécessitent de tester des milliers ou même des millions de molécules différentes, ce qui demande beaucoup de temps et d'argent. Cette thèse présente AutoSMILES, un framework basé sur des agents qui utilise l'intelligence artificielle pour concevoir des molécules avec des caractéristiques désirées.

Le cœur de l'innovation réside dans l'utilisation de deux agents : un agent génératif qui crée des molécules et un agent prompt qui fournit des retours dynamiques et contextuels basés sur l'historique des tentatives précédentes. Cette approche agentic offrent des pistes de solutions sur les problèmes fondamentaux de génération itérative et d'inefficacité qui limitent les systèmes traditionnels.

La recherche révèle des découvertes cruciales en ingénierie de prompt pour la chimie computationnelle. L'agent prompt produit des améliorations drastiques des performances, démontrant que les stratégies d'ingénierie de prompt représentent un facteur plus que critique. L'étude établit également des tendances universelles : les températures plus élevées améliorent systématiquement les taux de succès tout en éliminant les comportements de boucle inefficaces. Ces résultats ouvrent de nouvelles perspectives pour les systèmes de génération moléculaire basés sur des agents.

Abstract

Abstract

The discovery of new molecules with specific properties represents a major challenge in modern chemistry and drug development. Traditional methods involve testing thousands or even millions of different molecules, which is time-consuming and costly. This thesis presents AutoSMILES, an innovative framework based on a multi-agent architecture that revolutionizes AI-driven molecular design.

The core innovation lies in employing two agents: a generative agent that creates molecules and a prompt agent that provides dynamic, contextual feedback based on historical generation attempts. This agentic approach offers paths of solutions to solves fundamental problems of iterative generation and inefficiency that limit traditional systems.

The research reveals crucial discoveries in prompt engineering for computational chemistry. The prompt agent produces drastic performance improvements, demonstrating that prompt engineering strategies represent more than a critical factor. The study also establishes universal trends: higher temperatures systematically improve success rates while eliminating inefficient loop behaviors. Most importantly, the work shows that sophisticated agent-based feedback mechanisms can transform molecular generation effectiveness, often providing greater improvements than scaling model size or complexity. These results open new perspectives for molecular generation systems based on agents.

Keywords: Agents, Artificial Intelligence, Large Language Models, Molecular Generation, Property Optimization

Contents

Introduction . . . 5

1 Background and State of the Art . . . 7
- 1.1 Fundamental Concepts and Technologies . . . 7
  - 1.1.1 SMILES Representation and Molecular Properties . . . 7
  - 1.1.2 Large Language Models and Agent Frameworks . . . 8
  - 1.1.3 Graph-Based Pipeline Architectures . . . 9
- 1.2 Current Approaches . . . 9
  - 1.2.1 Traditional Single and Multi-Property Optimization . . . 9
  - 1.2.2 Existing Agentic Systems and Frameworks . . . 9
- 1.3 Current Limitations . . . 9

2 AutoSMILES Framework and Methodology . . . 11
- 2.1 System Architecture . . . 11
  - 2.1.1 Graph-Based Pipeline Design . . . 11
  - 2.1.2 Agent Specifications . . . 13
- 2.2 Experimental Design and Parameters . . . 13
  - 2.2.1 Parameter Space Definition . . . 13
  - 2.2.2 Workflow Strategies . . . 15
- 2.3 Evaluation Metrics and Methodology . . . 16
  - 2.3.1 Performance Metrics . . . 16
  - 2.3.2 Similarity and Diversity Metrics . . . 16
  - 2.3.3 Experimental Protocol . . . 16

3 Results and Analysis . . . 18
- 3.1 Basic Benchmark Analysis . . . 18
  - 3.1.1 Primary Trends Identification . . . 18
  - 3.1.2 Prompt Template Analysis . . . 18
  - 3.1.3 Range Analysis and Generalization . . . 18
- 3.2 Workflow and Target Value Studies . . . 21
  - 3.2.1 Single vs Multi-Property Analysis . . . 21
  - 3.2.2 Best-of-N Algorithm Impact . . . 21
  - 3.2.3 Target Value Sensitivity . . . 22
- 3.3 Prompt Agent Impact Assessment . . . 22
  - 3.3.1 With vs Without Prompt Agent Comparison . . . 22
  - 3.3.2 Loop Behavior Reduction . . . 22
  - 3.3.3 Range Analysis . . . 25

Conclusion . . . 26

A Environmental and Societal Impact Analysis of the Internship at IBM Almaden Research Center 28

B Soft Skills Appendix . . . 30

List of Figures

1 Examples of SMILES notation and corresponding molecular structures . . . 7
2 AutoSMILES Base Pipeline Architecture . . . 12
3 AutoSMILES Pipeline with Best-of-N Algorithm . . . 12
4 Distribution of molecular properties in the dataset. (a) LogP, (b) MW, (c) MR, and (d) TPSA . . . 17
5 granite3.3 Temperature Effects on Success Rates . . . 19
6 granite3.3 Temperature Effects on Loop Behavior . . . 19
7 Success Rate Range: BASE Workflow . . . 20
8 granite3.3 Target Variations . . . 23
9 granite3.3 Success Rate Without Prompt Agent . . . 24
10 granite3.3 Loop Behavior Without Prompt Agent . . . 24
11 C1 - N1 Radar . . . 30
12 C1 - N2 Radar . . . 30
13 C2 - N1 Radar . . . 31
14 C2 - N2 Radar . . . 31
15 C3 - N1 Radar . . . 32
16 C3 - N2 Radar . . . 32
17 C5 - N1 Radar . . . 33
18 C5 - N2 Radar . . . 33

List of Tables

1 Success Rate Matrix at Temperature 0 and BASE Workflow . . . 18
2 Loop Behavior Matrix at Temperature 0 and BASE Workflow . . . 20
3 Average Success Rate Range Matrix . . . 21
4 Average Loop Behavior Range Matrix . . . 21
5 Average Loop Behavior Range Matrix Without Prompt Agent . . . 25

List of Acronyms and Abbreviations

AI	Artificial Intelligence
BoN	Best-of-N (selection strategy)
LLM	Large Language Model
LogP	Octanol-water partition coefficient
MR	Molecular Refraction
MW	Molecular Weight
NLP	Natural Language Processing
SA	Synthetic Accessibility
SMILES	Simplified Molecular Input Line Entry System
TPSA	Topological Polar Surface Area

Introduction

Context

The field of molecular discovery faces huge challenges that have grown more complex over the past decades. The inverse design of molecules with optimized properties represents a fundamental challenge in computational chemistry and materials science. The vastness of chemical space—estimated to contain 10^{33} drug-like molecules [1]—makes exhaustive exploration computationally impossible. Modern drug development requires an investment of over $2 billion per new drug [2], making it one of the most expensive research processes in any industry. This high cost comes with significant risks, as approximately 90% of potential drugs fail during clinical trials after years of development work[3]. The complete timeline from initial discovery to market approval typically extends 10-15 years, creating urgent pressure to find more efficient approaches to molecular design[4].

Traditional computational workflows, such as quantitative structure-activity relationship (QSAR) modeling, physics-based simulations, and evolutionary algorithms, require experts, large computing resources, and manual supervision. These requirements limit scalability and adaptability to new chemical domains, illustrating the magnitude of this problem where molecular optimization remains a key bottleneck.

The recent AI revolution has begun to transform how scientists approach chemistry and molecular discovery. LLMs have shown remarkable capabilities in understanding and generating complex patterns in text data. This breakthrough has opened new possibilities for molecular design because chemical structures can be represented as text through specialized notation systems. SMILES serves as a crucial bridge between the world of molecular structures and NLP technologies [5]. This text-based representation allows transformer architectures, which were originally designed for language tasks, to operate effectively on chemical structure data.

Modern AI systems have evolved beyond single-task models to complex frameworks that can handle multiple steps in a workflow. Graph-based agent systems, built using frameworks like LangGraph and LangChain, enable the creation of pipelines where different AI agents work together to solve complex problems. These systems can manage multiple objectives, provide feedback loops, and adapt their behavior based on previous results.

Problem Statement

Current molecular generation systems face several fundamental limitations that restrict their effectiveness in real-world drug discovery applications.

A significant limitation of many current approaches is their lack of interactivity. Most models operate as single-shot generators, producing molecules from static prompts without iterative learning or adaptation from feedback. This lack of adaptability makes it difficult for these systems to improve their performance over time or to handle complex design challenges that require iterative refinement.

Another significant problem is single-objective optimization, where the system only focusing on improving one molecular property at a time. However, drug design is a multi-dimensional challenge where molecules must simultaneously satisfy multiple criteria. This limitation prevents current systems from capturing the full complexity of molecular design requirements.

Furthermore, there has been limited systematic evaluation of how different technical parameters affect the performance of molecular generation systems. Specifically, the impact of temperature settings (which control the randomness of AI generation) and prompt engineering strategies (how instructions are given to the AI system).

The research presented in this thesis try to address these limitations by developing a comprehensive agentic framework that can handle multiple objectives simultaneously while providing feedback loops. The key research questions that guide this work are:

- How can graph-based argentic architectures improve property-directed molecular generation?
- What are the impacts of temperature variation, prompt template design, and workflow strategies?
- How does feedback from a specialized prompt agent reduce undesirable loop behaviors?

To address these needs, we introduce **AutoSMILES**, an agentic molecular design system. AutoSMILES iteratively modifies input SMILES strings to achieve user-specified property targets using multi-strategy optimization. One of its main innovations is the use of prompt engineering through a dual-agent setup: a generative agent that produces molecules under different prompt conditions, and a prompt agent that dynamically writes prompt templates based on outcomes.

Report Plan

This report is organized as follows:

Chapter 1 reviews the fundamental concepts and technologies that form the foundation of this research. This includes an explanation of SMILES notation and its role in representing molecular structures as text, an overview of LLMs and their applications in chemistry, and an introduction to graph-based agent frameworks.

Chapter 2 presents the AutoSMILES framework architecture and methodology. This chapter explains the graph-based pipeline design that enables AI agents to work together. It describes the experimental methodology used to evaluate the system, including the definition of different workflow strategies and the parameters used during testing.

Chapter 3 analyzes the benchmark results obtained from testing the AutoSMILES framework. The analysis focuses on understanding how parameters like temperature, prompt engineering approaches or workflow strategies affect generation quality.

Conclusion synthesizes the findings from the experimental work and discusses their implications for the future of molecular design using AI systems.

1 Background and State of the Art

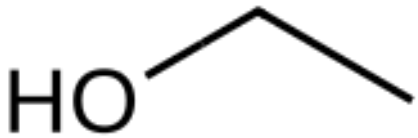
1.1 Fundamental Concepts and Technologies

1.1.1 SMILES Representation and Molecular Properties

SMILES notation represents a breakthrough in molecular representation that makes it possible to describe complex chemical structures using simple text strings [5]. This text-based approach makes SMILES particularly suitable for transformer-based models, which were originally designed to process sequences of words but can easily adapt to process sequences of chemical symbols. This linear ASCII representation creates a bridge between the world of chemistry and NLP technologies.

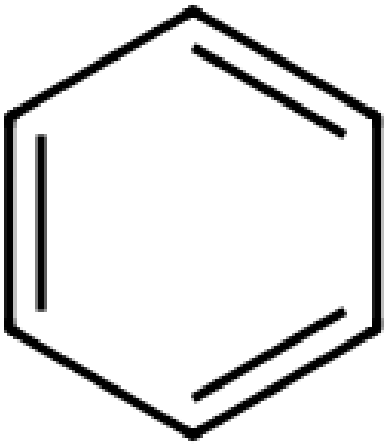
To illustrate the power and simplicity of SMILES notation, Figure 1 shows several examples of molecules represented both as SMILES strings and their corresponding 2D chemical structures.

SMILES: CCO



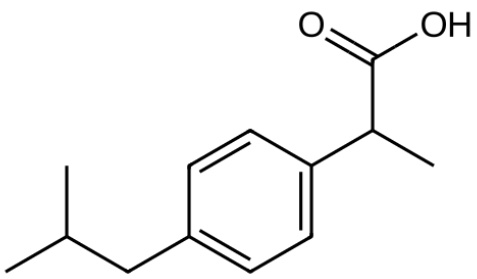
(a) Ethanol

SMILES: c1ccccc1



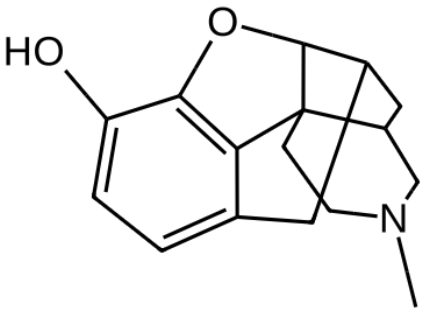
(b) Benzene

SMILES: CC(C)Cc1ccc(cc1)C(C)C(=O)O



(c) Ibuprofen

SMILES: CN1CCC23C4Oc5c3c(ccc5O)CC4CC2C1



(d) Morphine

Figure 1: Examples of SMILES notation and corresponding molecular structures

The examples in Figure 1 illustrate several key aspects of SMILES notation:

- **Atoms and bonds**: Carbon atoms are represented by 'C', oxygen by 'O', nitrogen by 'N'. Single bonds are implicit, while double bonds use '=' and triple bonds use '#'.
- **Branching**: Parentheses indicate side chains, as seen in the isobutyrate structure CC(C)C(=O)OCC.
- **Ring closure**: Numbers mark where rings close, such as the '1' in benzene (c1ccccc1) indicating the ring connection between the first and last carbon.

- **Aromaticity**: Lowercase letters (c, n, o) represent aromatic atoms, distinguishing them from aliphatic atoms (C, N, O).
- **Complexity handling**: Even complex molecules like morphine with multiple ring systems and functional groups can be represented as readable text strings.

This encoding enables language models to process molecular structures using the same attention mechanisms that make them effective for natural language tasks. The SMILES strings allows transformer architectures to learn chemical grammar rules, bond formation patterns, and structure-property relationships through standard sequence approaches.

Several key molecular properties drive most drug design decisions and serve as targets for molecular generation systems. LogP, the octanol-water partition coefficient, measures lipophilicity and determines how well a molecule can cross cell membranes. This property is crucial because drugs need to be absorbed by the body and reach their target locations. Molecular weight plays an important role in drug absorption: oral drugs should typically have molecular weights below 500 Daltons to maintain good absorption properties.

Molecular refraction serves as an indicator of electronic properties and molecular polarizability, helping predict how molecules will interact with biological targets. Topological polar surface area is also important because it predicts membrane permeability.

Finally, synthetic accessibility scores help assess whether generated molecules can actually be made in the laboratory. Computational tools evaluate proposed synthetic routes and assign scores that indicate how difficult or expensive it would be to synthesize a particular molecule.

1.1.2 Large Language Models and Agent Frameworks

Advances in LLMs made a fundamental change in many scientific domains that rely on reasoning and structured data, with chemistry and materials science among the most affected [6, 7]. Although first designed for natural language tasks, LLMs now show strong abilities in handling molecular representations, especially when molecules are written as SMILES strings. This has enabled applications ranging from property prediction and retrosynthetic analysis to inverse design of molecules with user-defined objectives [8, 9].

The combination of NLP and molecular modeling has expanded the methods available for representing and generating molecules. ChemBERTa creates molecular embeddings from SMILES [10], while MolT5 allows translation between natural language and molecular structures [11].

Specialized models have achieved remarkable performance in molecular generation tasks. SmileyLlama can achieve validity rates near 99%, demonstrating that LLMs can learn the complex grammar and constraints of chemical structures [12]. SynLlama extends this capability by generating both molecules and synthetic pathways [13].

However, these advances come with significant challenges. Chemical hallucinations remain a major issue requiring careful prompt optimization, as models can generate chemically implausible or impossible structures despite appearing syntactically correct [14]. This highlights the importance of prompt engineering and validation mechanisms in molecular generation systems.

Temperature parameters play a critical rolse for generation randomness and creativity in language models. Low temperature values (close to 0) produce more predictable outputs, while higher temperature values increase randomness. Understanding how temperature affects molecular generation performance is essential for optimizing system behavior.

In artificial intelligence, an Agent is a computational entity that can interact with its environment, make decisions, and take actions autonomously to achieve specific goals. Agent-based systems represent an evolution beyond simple approaches. Multi-agent architectures decompose complex problems into manageable subtasks that can be handled by specialized components. Each agent can focus on a specific aspect of the overall problem, such as generation, evaluation, or feedback.

Graph-based workflows, implemented through frameworks like LangGraph and LangChain, provide an approache for managing complex state transitions and conditional logic flows. These frameworks enable the creation of dynamic workflows where the next step depends on the results of previous steps like finite state machines.

1.1.3 Graph-Based Pipeline Architectures

Node-based processing workflows offer modular approaches to complex tasks, where each node represents a specific operation or decision point. This design allows individual nodes to be designed, tested, and optimized independently before being integrated into larger systems.

State management and validation systems form critical infrastructure components that ensure process reliability through complex workflows. These systems track the current status of the generation process, validate intermediate results, and provide all necessary information needed for each step of the pipeline.

Feedback mechanisms are essential against inefficient computational cycles and generation stagnation. Without proper feedback, systems might repeatedly generate the same unsuccessful molecules or get stuck in unproductive patterns. So to detect this behavior, the system has a loop detection metric that helps identify when the system is stuck.

Best-of-N algorithm is a trajectory optimization strategy where it improves output quality by generating multiple candidates and selecting the optimal results based on a defined loss function [15]. Rather than accepting the first generated molecule, these strategies produce several options and choose the best one, leading to higher overall success rates.

1.2 Current Approaches

1.2.1 Traditional Single and Multi-Property Optimization

The field of property-directed molecular generation shows a clear distinction between single-property optimization approaches and the significantly more challenging multi-property scenarios that better reflect real-world drug design requirements. Traditional single-objective methods focus on optimizing one molecular property at a time. While these approaches can achieve good results for their specific targets, they face limitations when drug design requires balancing multiple properties simultaneously.

Multi-objective challenges introduce the complexity of Pareto optimization, where improving one property might require accepting worse performance in another property. Defining appropriate trade-offs between different molecular characteristics becomes a critical challenge, as there is rarely a single "best" solution that optimizes all properties simultaneously.

1.2.2 Existing Agentic Systems and Frameworks

Recently, new agentic frameworks for molecular and materials discovery have emerged, representing a significant evolution in how AI systems approach molecular design. ChemCrow and dZiner use modular agents that are specialized in retrieval, reasoning, scoring, and validation [16, 17]. dZiner introduces an iterative loop for molecular modification, combined with human-in-the-loop evaluation and scoring functions.

CRAG-MoW uses multiple specialized agents for retrieval-augmented generation, hallucination detection, and response refinement, showing that multi-agent systems can outperform single models in molecular design [18].

AutoGen provides infrastructure for multi-agent conversations in scientific applications [19] and systems like ACCELMAT help generate new hypotheses in materials discovery by using goal-conditioned agents, critic and summarization modules [20].

However, most of these systems still focus on broad results—like whole material classes or general reactions—rather than on the precise optimization of molecular properties such as lipophilicity (logP), drug-likeness (QED), or synthetic accessibility (SA).

1.3 Current Limitations

Existing approaches show a lack of feedback mechanisms that will guide iteratively in molecular generation tasks. Many current systems generate molecules in isolation, without learning from previous attempts or adapting their strategies based on what works and what does not work.

Loop behavior and generation redundancy represent persistent problems that waste computational resources and limit practical applicability. When systems repeatedly generate the same unsuccessful

molecules or get stuck in unproductive patterns, they consume time and computational power without making progress toward the desired goals.

Integration challenges add another layer of complexity through model-dependent performance variations that make it difficult to generalize findings across different language model architectures. What works well with one model might not transfer effectively to another model.

There is increasing agreement that next-generation molecular design systems need to include: (i) agents with capabilities for planning, memorizing, and reasoning; and (ii) feedback-driven optimization to improve molecules step by step. This change requires systems that do more than just generate molecules—they must reason about molecular structures, adapt to different feedback, and work independently throughout the entire molecular design process.

2 AutoSMILES Framework and Methodology

2.1 System Architecture

2.1.1 Graph-Based Pipeline Design

The AutoSMILES framework uses a graph-based architecture where individual nodes handle specific tasks within a larger workflow management system. This modular design allows each component to focus on a particular aspect of molecular generation while maintaining coordination with other parts of the system. The basic pipeline algorithm is displayed in the Algorithm 1.

Algorithm 1 AutoSMILES Graph-Based Pipeline

Require: Initial SMILES string s_0, target properties P_{target}, tolerance ϵ, max_attempts N
Initialize: iteration = 0, best_score = ∞, $s_{best} = None$
while iteration $< N$ AND not success **do**
  Validation Node:
  if validate_smiles($s_{current}$) == False **then**
    $s_{current} = s_0$
  end if
  Generative Agent Node
  if using Best-of-N **then**
    candidates = generate_multiple_smiles($s_{current}$, prompt, temperature)
  else
    candidates = [generate_smiles($s_{current}$, prompt, temperature)]
  end if
  Property Computation Node
  for each s_i in candidates **do**
    P_i = compute_properties(s_i)
    score$_i$ = loss_function(P_i, P_{target})
  end for
  Selection Node
  $s_{selected}$ = argmin(score) from candidates
  selected_score = min(score)
  Success Check
  if all properties within ϵ tolerance **then**
    RETURN $s_{selected}$
  end if
  Loop Detection
  if $s_{selected}$ in generation_history **then**
    loop_detected = True
  end if
  Prompt Agent Node (if enabled)
  if using prompt agent **then**
    feedback = generate_feedback(history, P_{target}, current_results)
    prompt = update_prompt(prompt, feedback)
  end if
  if selected_score $<$ best_score **then**
    best_score = selected_score
    $s_{best} = s_{selected}$
  end if
  $s_{current} = s_{selected}$
  iteration += 1
end while
return s_{best}

The input validation node serves as the entry point for the system, performing SMILES syntax verification to ensure that all molecular representations is conform. This validation step prevents errors from propagating through the system and ensures that only properly formatted molecules are processed.

The generative agent node represents the core of the molecular generation process, where LLMs produce new molecular structures. This node takes into account specified property targets.

Property computation nodes calculate new generated molecular properties using established chem-informatics algorithms. These nodes compute properties like LogP, molecular weight, molecular refraction, and topological polar surface area.

The prompt agent node represents a critical add in the system, dynamically generating feedback messages that incorporate information from previous generation attempts. This feedback guides next iterations toward better outcomes by helping the system learn from its previous attempts.

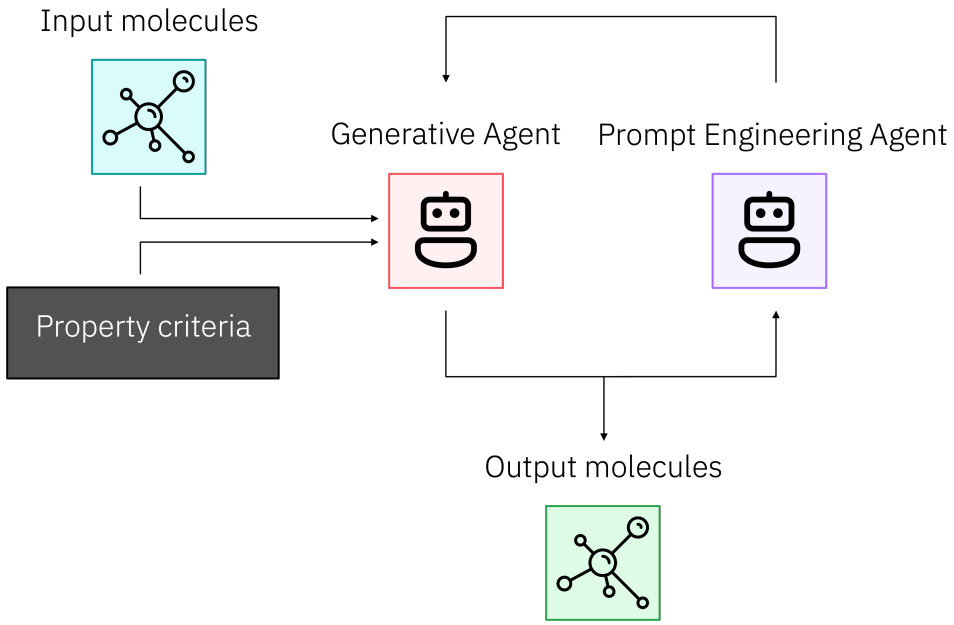


Figure 2: AutoSMILES Base Pipeline Architecture

Selection nodes implement Best-of-N algorithms that evaluate multiple generated candidates and select an optimal molecule based on defined loss functions. These loss functions measure how far generated molecules deviate from target properties.

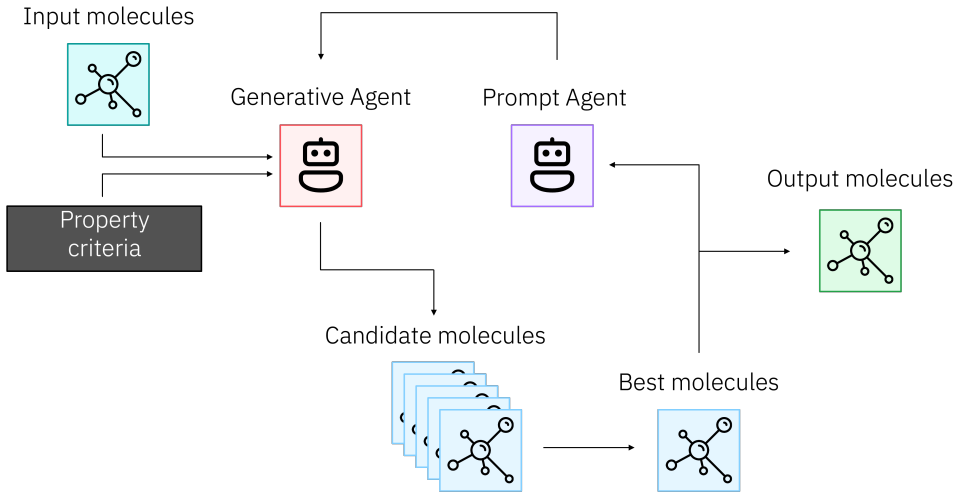


Figure 3: AutoSMILES Pipeline with Best-of-N Algorithm

The pipeline flow logic manages sequential processing with conditional branches that determines next steps based on current generation results. Loop detection mechanisms monitor generation patterns to

identify when the system begins producing redundant outputs, while max_attempts parameters provide definitive termination criteria to prevent infinite cycles.

Success and failure state are track at the end of the workflow and report whether property targets are achieved within specified iteration limits or not.

2.1.2 Agent Specifications

The Agents are five distinct LLMs to enable model comparison. The tested models are:

- **granite3.3, context window: 131k tokens**
- **llama4-mvk, context window: 1M tokens**
- **phi4, context window: 32k tokens**
- **qwen2.5, context window: 33k tokens**
- **gpt-oss-120b, context window: 131k tokens**

Depending on the workflows, multiple SMILES generation capabilities enable Best-of-N selection strategies that improve output quality through candidate comparison and optimization. Instead of generating only one molecule per iteration, the system can produce several options and choose the best one.

The prompt agent incorporates feedback integration mechanisms that analyze historical generation attempts and dynamic prompt construction that adapts its output based on current context and generation history.

2.2 Experimental Design and Parameters

2.2.1 Parameter Space Definition

Temperature settings represent a critical experimental dimension with three chosen values:

- **Temperature 0**
- **Temperature 0.3**
- **Temperature 0.7**

Low temperature values (close to 0) produce more predictable outputs, while higher temperature values increase randomness.

The benchmarks tested five different prompt templates for molecular generation, each designed to test how language models respond to different instruction styles:

- **chat_template** follows a structured conversational format with embedded feedback mechanisms that provide feedback as "history" variable within the template structure itself.
- **prompt_base** represents a detailed instruction template that provides extensive context and explicit guidance for molecular generation tasks.
- **Variation_1, variation_2, and variation_3** implement concise template alternatives that test whether reduced instruction complexity can maintain generation effectiveness while improving computational efficiency.

Model comparison across five different language model enables assessment of how different training approaches, model sizes, and architectural decisions affect molecular generation performance. By testing the same tasks across multiple models, the research can identify which approaches work consistently and which are model-dependent.

Target property specification focuses on four key molecular characteristics that represent different aspects of drug-like behavior and synthetic accessibility. LogP targeting examines lipophilicity and Molecular weight, molecular refraction, and topological polar surface area provide additional dimensions for multi-property optimization challenges.

The $\pm 20\%$ tolerance threshold reflects an experimental measurement uncertainties and acceptable variation ranges for practical molecular design applications.

The following code blocks show the five prompt templates used in the experimental evaluation:

Listing 1: chat_template

```
prompt = ChatPromptTemplate.from_messages([
                ("system", "{system_prompt}"),
                MessagesPlaceholder(variable_name="history"),
                ("human", "{human_prompt}"),
        ])

human_prompt = """Given the original molecule represented by the SMILES string: {smiles},
with the following original property values:
{property_list}

Generate a new valid SMILES that satisfies all the following conditions:
1. It is structurally different from the original molecule.
2. It is not present in the list of previously generated SMILES.
3. Its predicted properties are as close as possible to the target values:
{target_list}

Return {candidates_per_iteration} different SMILES strings in the required JSON format.
"""

system_prompt = """You are a molecular design assistant specialized in organic chemistry
and polymer science.
You generate valid SMILES strings that represent chemically valid molecules.
You must generate {candidates_per_iteration} different valid SMILES strings that each
satisfy:
- Syntactically valid according to SMILES conventions.
- Chemically plausible (no valence errors, no unstable radicals unless specified).
- Structurally different from each other and from the original molecule.
- Each targeting the specified property values as closely as possible.
"""
```

Listing 2: prompt_base

```
prompt_base = """### Tools
- **SMILES Generator**
  - Purpose: Generate chemically valid SMILES strings.
  - Arguments:
    - {smiles}: Input molecule in SMILES format.
    - {property_list}: Original property list of the molecule.
    - {target_list}: Property target list for the new molecule.
    - {candidates_per_iteration}: Number of different SMILES strings to generate.
  - Example usage: Not explicitly provided, but expected to ensure syntactical validity,
  chemical plausibility, and uniqueness.

### Plan of Action
1. **Analysis**:
   - Input the original molecule SMILES and the history of generated SMILES.
   - Identify structural features of the original molecule.

2. **Generation**:
   - Use the SMILES generator to create a new SMILES string.
   - Ensure syntactical validity per SMILES conventions and chemical plausibility (e.g.,
   correct valence, stability).

3. **Validation**:
   - Check for structural difference from the original molecule.
   - Verify that the generated SMILES is not already in the history list.

4. **Optimization**:
   - Adjust the new SMILES structure to approach the target property values as closely
   as possible.

5. **Output**:
   - Return the new SMILES string.

### Retry or Recovery Logic
- If a generated SMILES is not valid, attempt generation again with slight modifications.
- Ensure no infinite loops occur by limiting retries and keeping track of attempts.

### Additional Heuristics and Examples
- Emphasize avoiding valence errors and ensuring chemical stability unless radical states
```

```
are explicitly desired.
- Implicitly adhere to SMILES conventions in all string generations.
- Maintain data integrity by consistently checking against history to avoid repetitions.

Return {candidates_per_iteration} different SMILES strings in the required JSON format.
"""
```

Listing 3: variation_1

```
variation_1 = """As a chemist with expertise in optimization, analyze the
starting molecule {smiles} which has {property_list}.
Your goal is to design a molecule with {property_list} closest to {target_list}.
Consider:
1. Structure-activity relationships for {property_list} modification,
2. Metabolic stability implications,
3. Synthetic feasibility of proposed changes.
Return {candidates_per_iteration} different SMILES strings in the required JSON format.
"""
```

Listing 4: variation_2

```
variation_2 = """Given molecule: {smiles} (current {property_list})
Target: {target_list}

Step 1: Analyze current molecular features affecting {property_list}
Step 2: Identify specific substructures contributing to current {property_list}
Step 3: Plan strategic modifications to reach {target_list}
Step 4: Consider ADMET trade-offs and synthetic accessibility
Step 5: Generate optimized molecule maintaining drug-like properties

Return {candidates_per_iteration} different SMILES strings in the required JSON format.
"""
```

Listing 5: variation_3

```
variation_3 = """MOLECULAR OPTIMIZATION TASK
Initial: {smiles} | Property: {property_list} | Target: {target_list}
Constraints: Preserve scaffold topology, maintain drug-like properties (Lipinski
compliance), ensure synthetic feasibility
Optimization Strategy: If target > current, increase lipophilic substituents; if target
< current, add polar/ionizable groups
Success Metric: Minimize |predicted_{property_list} - {target_list}|
Return {candidates_per_iteration} different SMILES strings in the required JSON format.
"""
```

2.2.2 Workflow Strategies

Single property workflows provide focused optimization that enable detailed analysis of how different parameters affect generation performance when complexity is minimized through single-objective targeting. These workflows are easier to analyze because they only need to optimize one property at a time.

The BASE workflow implements straightforward single property targeting with LogP = 4 ± 20%, providing fundamental baseline performance measurement across all parameter combinations.

SINGLE_BON extends single property targeting with Best-of-N selection algorithms that generate multiple candidates per iteration and select optimal molecules based on property achievement metrics.

Multi-property workflows represent significantly more challenging scenarios that better reflect real-world molecular design requirements where multiple properties must be simultaneously satisfied. In drug discovery, molecules typically need to meet several criteria at once.

MULTI_PROPERTY workflow targets four simultaneous properties with LogP = 4 ± 20%, MR = 90 ± 20%, MW = 400 ± 20% Da, and TPSA = 80 ± 20% Å^2. This configuration requires balancing of competing molecular characteristics, as improving one property might make it harder to achieve another.

MULTI_BON combines multi-property targeting with Best-of-N selection strategies.

2.3 Evaluation Metrics and Methodology

2.3.1 Performance Metrics

Success rate measurement provides the primary quantitative metric of the system effectiveness, calculated as the proportion of generation tasks that achieve all specified property targets within defined tolerance ranges.

Loop behavior quantification measures the percentage of generation attempts that produce duplicate SMILES strings across different iterations within the same task. This metric specifically tracks when the system generates identical molecules that have already been produced in previous iterations, indicating ineffective feedback mechanisms or insufficient exploration strategies.

The measurement focuses on the best-selected molecule from each iteration when Best-of-N strategies are employed, meaning that loop detection reflects the most relevant generation outputs rather than all intermediate candidates. High loop behavior indicates that the system is not learning effectively from previous attempts and may be wasting computational resources.

2.3.2 Similarity and Diversity Metrics

Tanimoto similarity calculations provide standardized molecular fingerprint comparison that enables quantitative measurement of structural relationships between generated molecules and reference compounds.

Tanimoto Similarity:

$$T(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{1}$$

where A and B are molecular fingerprint bit vectors.

Cosine distance measurements compute vector-embedded similarity that captures different aspects of molecular similarity beyond structural fingerprints.

Euclidean distance provide direct measurement of how closely generated molecules approach target property combinations, enabling optimization tracking and convergence analysis.

Synthetic accessibility score evaluates whether generated molecules represent synthetically feasible targets or computationally interesting but practically inaccessible structures.

Loss Function for Property Targeting:

$$L_{\text{total}} = \sum_{p \in \mathcal{P}} w_p \left(\frac{P_{gen} - P_{target}}{P_{target}} \right)^2 \tag{2}$$

where w_i are property weights.

2.3.3 Experimental Protocol

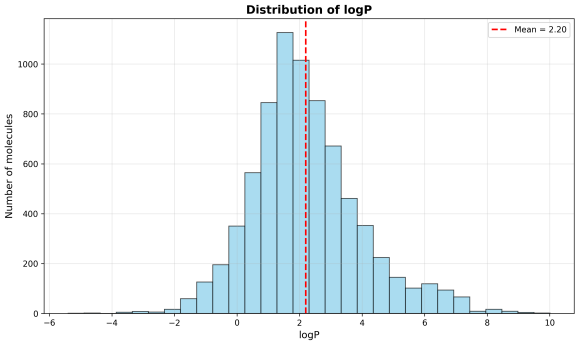
Dataset selection needs to be done carefully with a wide range of the tested properties, ensuring robust baseline measurements for comparative analysis. The property histograms in Figure 4 provide visual and statistical characterization of dataset distributions.

Controlled parameter variation methodology ensures systematic exploration of the experimental space while maintaining statistical validity through appropriate replication and randomization strategies.

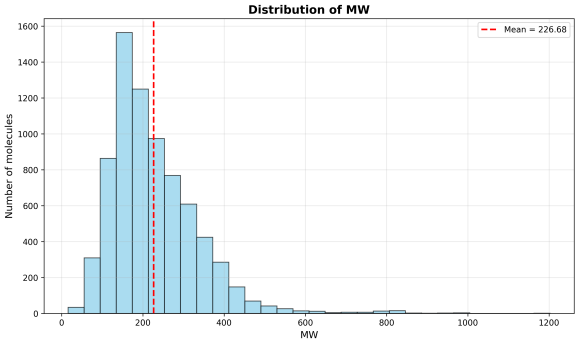
Statistical analysis focus on trend identification across parameter combinations, with particular focus on temperature effect and prompt template impact.

Range analysis enables quantitative measurement of temperature impact magnitude by calculating variation ranges across different temperature settings for fixed combinations of other parameters. This approach provides generalizable information about parameter sensitivity.

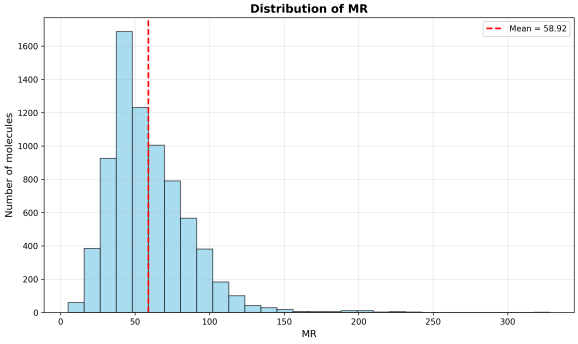
The AutoSMILES framework implements a graph-based architecture that enables systematic evaluation of multiple parameters affecting molecular generation performance, with a particular focus on



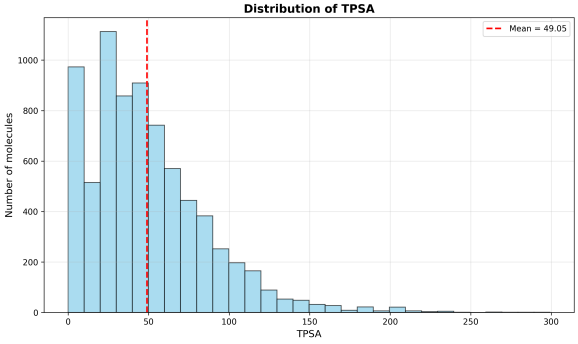
(a) LogP Distribution



(b) MW Distribution



(c) MR Distribution



(d) TPSA Distribution

Figure 4: Distribution of molecular properties in the dataset. (a) LogP, (b) MW, (c) MR, and (d) TPSA

quantifying temperature effects and prompt engineering impacts through many different experimental set ups.

3 Results and Analysis

3.1 Basic Benchmark Analysis

3.1.1 Primary Trends Identification

The experimental results reveal two universal temperature effects that appear consistently across all tested parameter combinations. These patterns are so reliable that they can be considered fundamental characteristics of how language models behave in molecular generation tasks.

The first universal trend shows that higher temperature settings consistently produce higher success rates in molecular generation tasks (Figure 5). This means that when the AI system generates molecules with more randomness, it is more likely to create molecules that meet the specified property targets. This pattern indicates that increased randomness enables more effective exploration of chemical space, helping the system find molecules that satisfy the required characteristics.

The second universal trend demonstrates that higher temperature settings consistently reduce loop behavior percentages (Figure 6). Loop behavior occurs when the system generates the same molecule multiple times, wasting computational resources without making progress. Higher temperature settings help prevent this problem by introducing more variability in generation, making it less likely that the system will get stuck repeating the same unsuccessful attempts.

These trends demonstrate model behavior, appearing consistently across all five tested language model architectures: granite3.3, llama4-mvk, phi4, qwen2.5, and gpt-oss-120b. Despite their different training approaches, parameter counts, and model sizes, all models show the same temperature-related patterns.

Template-independent trend validation confirms that these temperature effects persist regardless of prompt template structure. Whether using the chat_template with embedded feedback or the variation templates with minimal instruction content, the same temperature patterns appear.

These two effects appearing in both single and multi-property workflows confirms that they represent fundamental characteristics of language model behavior in molecular generation contexts rather than artifacts of specific experimental setups. The same patterns appear whether the system is trying to optimize one property or multiple properties simultaneously.

3.1.2 Prompt Template Analysis

While temperature effects remain consistent across different prompt templates, the templates themselves produce significant variations in absolute performance. This means that while temperature always has the same directional effect, the starting performance level depends heavily on which template is used with which model (Table 1 & Table 2).

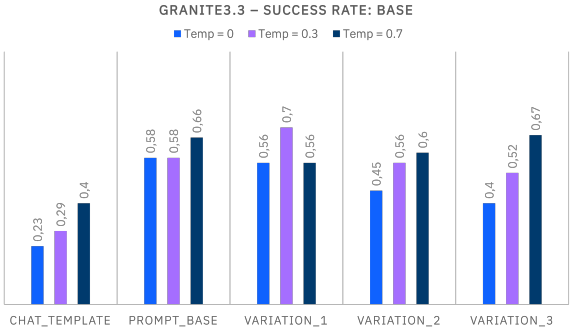
Model-template interaction effects demonstrate that different combinations produce substantially different absolute performance values. This means that template optimization should consider target model, as what works well with one model might not be optimal for another.

Table 1: Success Rate Matrix at Temperature 0 and BASE Workflow

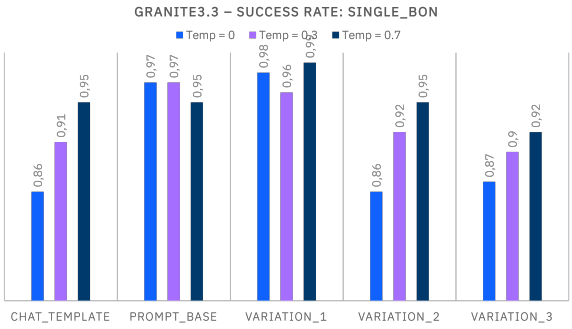
Model	chat_template	prompt_base	variation_1	variation_2	variation_3
granite3.3	0.23	0.59	0.56	0.45	0.40
llama4-mvk	0.92	0.86	0.94	0.94	0.88
phi4	0.75	0.82	0.91	0.84	0.81
qwen2.5	0.78	0.71	0.92	0.94	0.88
gpt-oss-120b	0.99	0.86	0.93	0.92	0.84

3.1.3 Range Analysis and Generalization

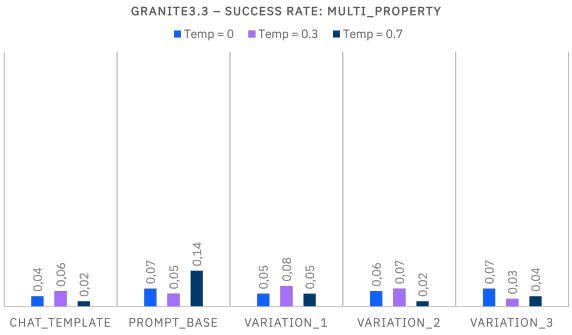
Success rate range analysis provides measurement of temperature impact magnitude by calculating the difference between maximum and minimum success rates observed across the three temperature settings for each fixed combination of model and template parameters (Figure 7).



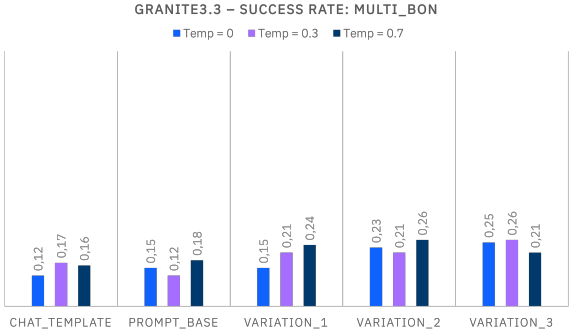
(a) BASE



(b) SINGLE_BON

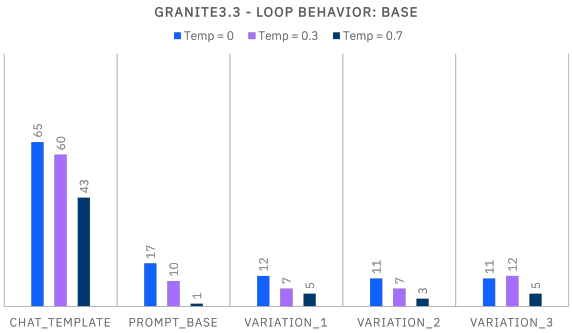


(c) MULTI_PROPERTY

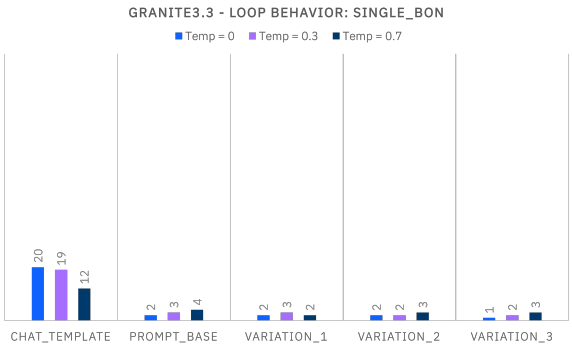


(d) MULTI_BON

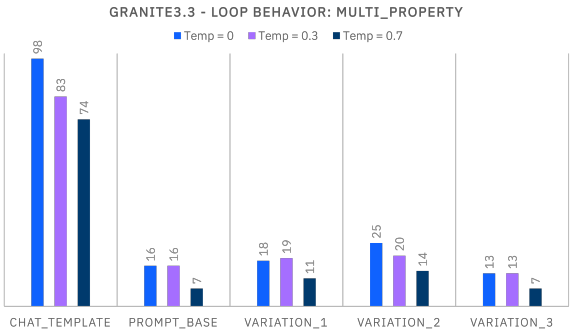
Figure 5: granite3.3 Temperature Effects on Success Rates



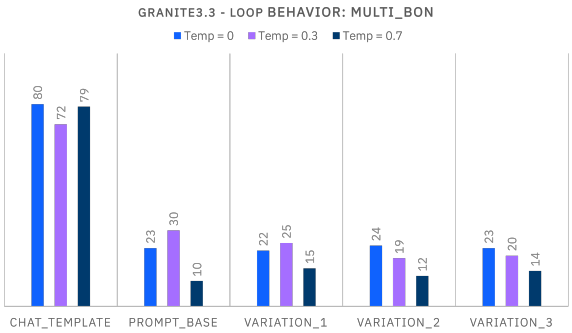
(a) BASE



(b) SINGLE_BON



(c) MULTI_PROPERTY



(d) MULTI_BON

Figure 6: granite3.3 Temperature Effects on Loop Behavior

Table 2: Loop Behavior Matrix at Temperature 0 and BASE Workflow

Model	chat_template	prompt_base	variation_1	variation_2	variation_3
granite3.3	65	17	12	11	11
llama4-mvk	2	10	3	0	0
phi4	24	3	1	2	2
qwen2.5	18	8	2	1	1
gpt-oss-120b	0	1	0	0	0

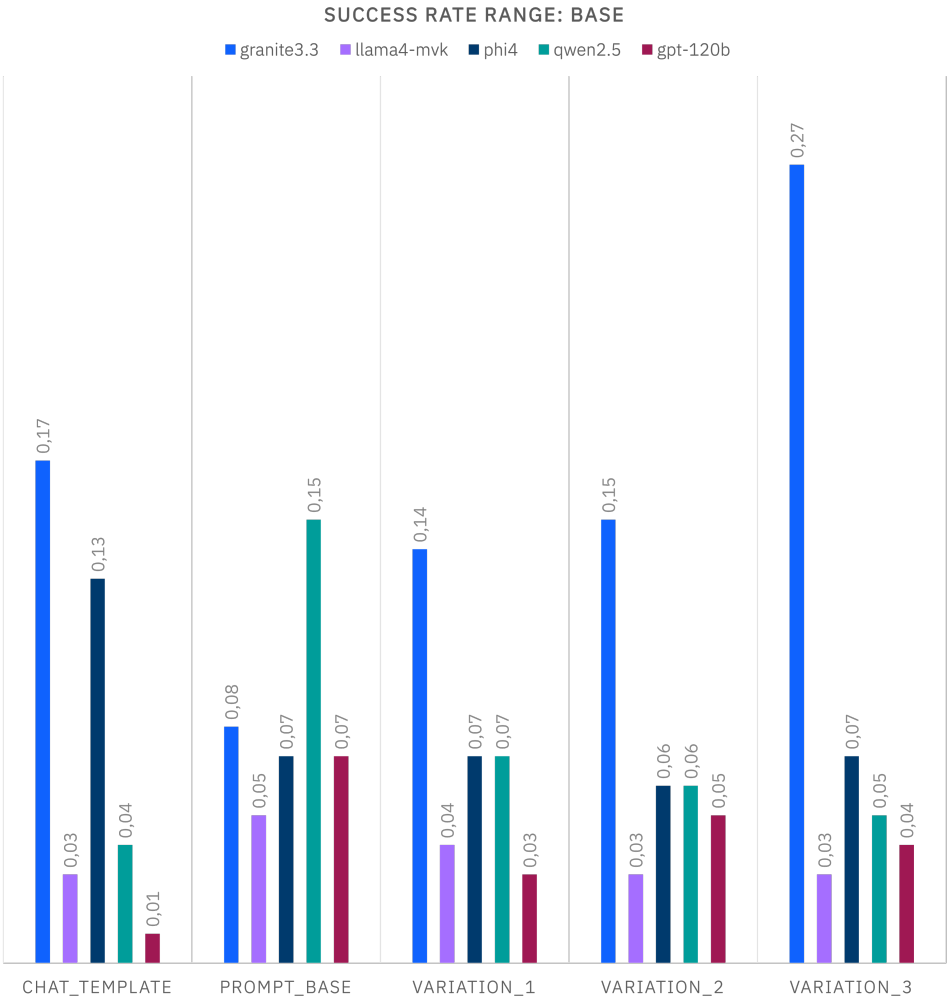


Figure 7: Success Rate Range: BASE Workflow

Average range values across different templates demonstrate consistent temperature sensitivity (Table 3). This means that changing temperature settings typically improves success rates by fixed amount, regardless of which template or model is used.

Looking at the other workflows, it confirms that this patters persist across BASE, SINGLE_BON, MULTI_PROPERTY, and MULTI_BON configurations. Whether the system is working on simple single-property tasks or complex multi-property challenges, temperature has the same magnitude of effect.

Table 3: Average Success Rate Range Matrix

Workflows	**chat_template**	**prompt_base**	**variation_1**	**variation_2**	**variation_3**
BASE	0.08	0.08	0.07	0.07	0.09
SINGLE_BON	0.03	0.03	0.04	0.05	0.07
MULTI_PROP	0.08	0.11	0.09	0.05	0.08
MULTI_BON	0.07	0.10	0.11	0.08	0.06

Model temperature sensitivity demonstrates that range patterns remain consistent across all five tested language model. This provides strong evidence for generalizable temperature optimization guidelines that should work across current and future language models.

Loop behavior range analysis reveals dramatically different patterns, with one notable exception to the general consistency. The chat_template consistently shows exceptional sensitivity to temperature variation with average ranges substantially higher than other templates that cluster (Table 4).

Table 4: Average Loop Behavior Range Matrix

Workflows	**chat_template**	**prompt_base**	**variation_1**	**variation_2**	**variation_3**
BASE	8.40	5.80	2.20	2.40	2.80
SINGLE_BON	2.25	1.75	1.00	0.25	1.75
MULTI_PROP	19.50	8.75	4.50	3.75	4.25
MULTI_BON	11.00	8.75	7.75	6.25	6.25

This exceptional sensitivity reflects the interaction between the chat_template's built-in feedback mechanisms and external temperature control. The conversational format creates amplified sensitivity to randomness parameters that affects loop behavior more dramatically than other template designs. This finding suggests that templates with built-in feedback structures may require different temperature optimization strategies.

3.2 Workflow and Target Value Studies

3.2.1 Single vs Multi-Property Analysis

Performance degradation patterns between single and multi-property workflows reveal the expected complexity-performance trade-offs as additional property constraints significantly reduce success rates while increasing computational requirements for achieving satisfactory results. When the system must satisfy multiple properties simultaneously, the task becomes much more challenging.

Multi-property scenarios require balancing of competing molecular characteristics, often involving trade-offs where improving one property may negatively affect another. This creates optimization challenges that extend beyond simple parameter adjustment, as the system must find molecules that represent acceptable compromises across multiple objectives (Figure 5).

3.2.2 Best-of-N Algorithm Impact

Best-of-N algorithm demonstrates consistent performance improvement across all tested parameter combinations, providing clear evidence for the effectiveness of multi-candidate generation with optimized selection strategies (Figure 5).

Success rate improvements appear across all workflow configurations, indicating that generating multiple candidates per iteration and selecting optimal results based on property achievement metrics

provides significant advantages over single-generation approaches. The improvement is consistent whether working on simple or complex molecular design tasks.

Loop behavior reduction in multi-generation scenarios confirms that Best-of-N selection helps prevent repetitive generation cycles by providing multiple exploration paths at each iteration. When the system generates several options instead of just one, it is less likely to become trapped in limited regions of chemical space that produce the same molecules repeatedly.

Selection node demonstrates that the implemented loss function approaches successfully identify optimal candidates from multi-molecule generation sets. The system can reliably choose the best option from several alternatives, contributing to overall system performance improvements that justify the additional computational overhead of multiple generation per iteration.

3.2.3 Target Value Sensitivity

LogP comparison studies between target values of 4 and 6 provide data about how target difficulty affects generation performance.

Trend preservation confirms that the fundamental temperature patterns (higher temperature leading to higher success rates and lower loop behavior) remain consistent regardless of target property values. This indicates that these represent intrinsic characteristics of the generation process rather than artifacts of specific target selection.

Difficulty scaling effects produce predictable changes in absolute performance values, with higher target values (LogP = 6) generally producing lower success rates due to increased constraint difficulty. However, relative performance patterns and temperature sensitivities remain consistent across target variations. This means that while harder targets are more difficult to achieve, the same optimization strategies still work (Figure 8).

3.3 Prompt Agent Impact Assessment

3.3.1 With vs Without Prompt Agent Comparison

Prompt agent activation produces dramatic improvements in success rates across all parameter combinations tested, providing evidence for the critical importance of feedback mechanisms in iterative molecular generation systems. The difference between systems with and without prompt agents is substantial and consistent (Figure 9).

Success rate shows improvements in already high-performing configurations and challenging multi-property scenarios. This demonstrates that feedback-guided generation represents a fundamental requirement for effective molecular design.

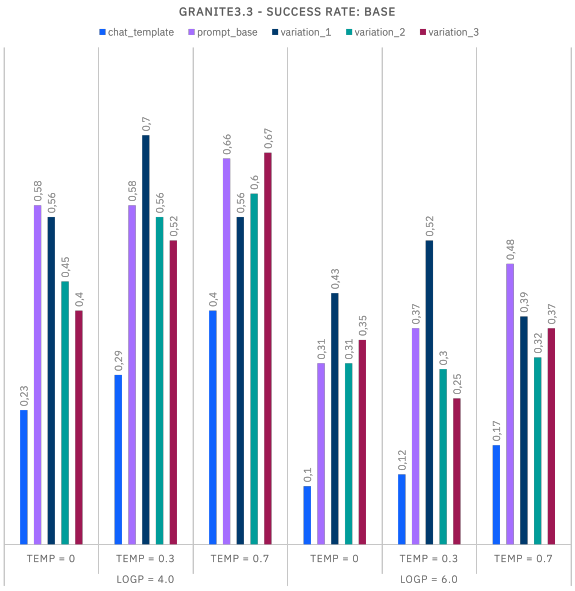
Whether using simple or complex templates, single or multiple properties, or different language models, the prompt agent always provides significant benefits.

3.3.2 Loop Behavior Reduction

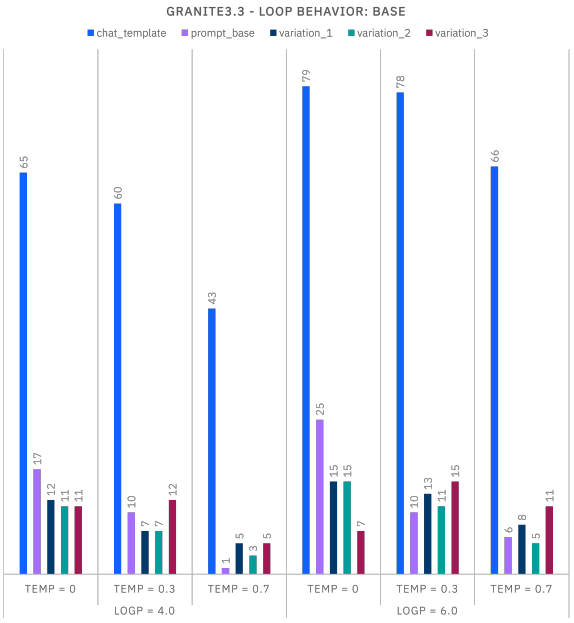
Feedback mechanism effectiveness in reducing loop behavior provides one of the most drastic performance improvements observed in the experimental results. Systems lacking prompt agents show extremely high loop behavior percentages that indicate severe efficiency problems, with the system repeatedly generating the same unsuccessful molecules (Figure 10).

Significant loop behavior reduction with prompt agent activation demonstrates that historical generation tracking and dynamic feedback generation effectively prevent repetitive generation cycles. The prompt agent provides increasingly specific guidance as iterations progress, helping the system avoid previously explored areas that did not produce successful results.

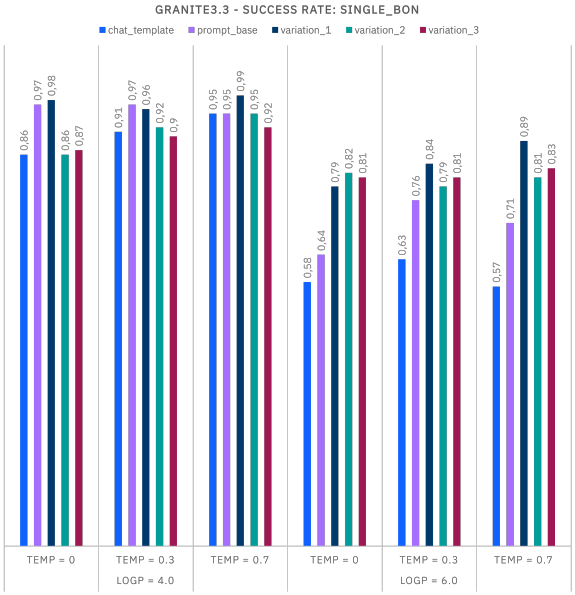
Without prompt agent, trend reveals that temperature effects become more pronounced when feedback mechanisms are absent. This suggests that randomness parameters provide partial compensation for missing feedback but cannot fully compensate. Temperature becomes more important when other guidance mechanisms are not available.



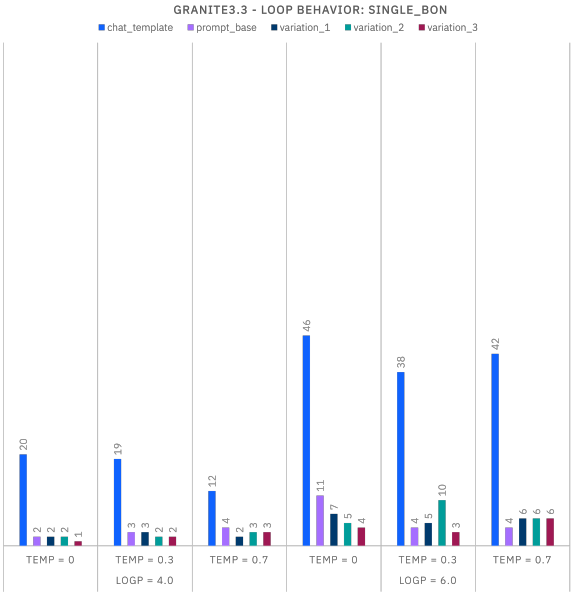
(a) Success Rate: BASE



(b) Loop Behavior: BASE

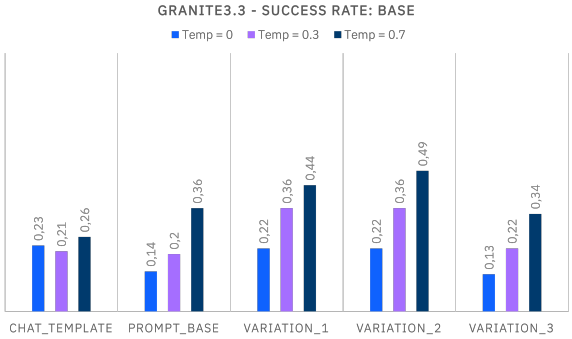


(c) Success Rate: SINGLE_BON

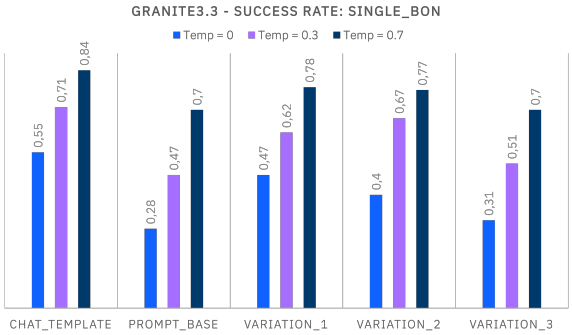


(d) Loop Behavior: SINGLE_BON

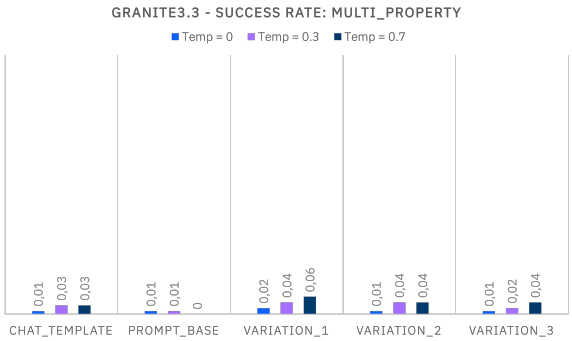
Figure 8: granite3.3 Target Variations



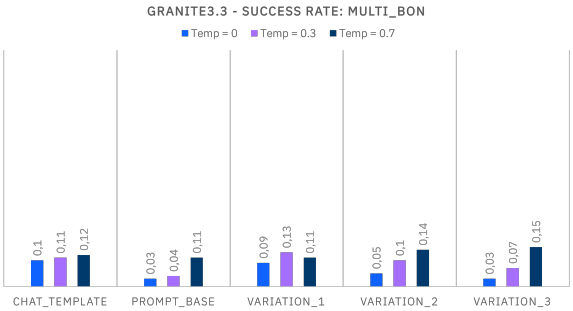
(a) BASE



(b) SINGLE_BON

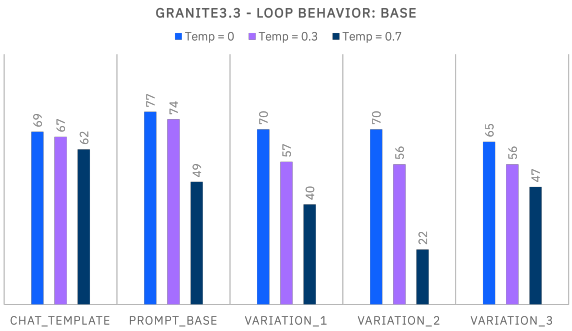


(c) MULTI_PROPERTY

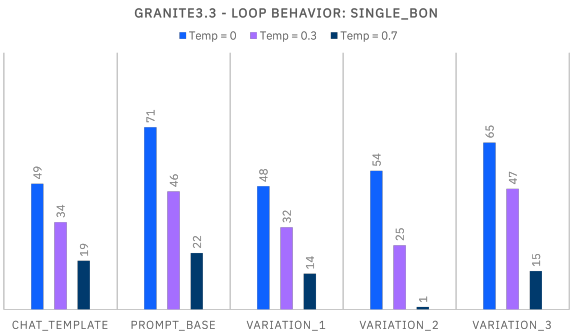


(d) MULTI_BON

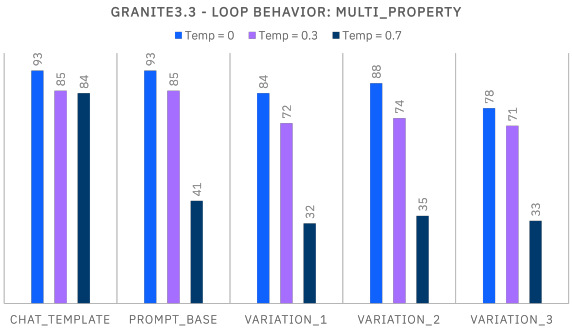
Figure 9: granite3.3 Success Rate Without Prompt Agent



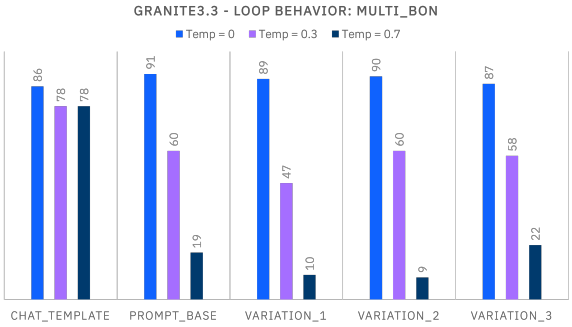
(a) BASE



(b) SINGLE_BON



(c) MULTI_PROPERTY



(d) MULTI_BON

Figure 10: granite3.3 Loop Behavior Without Prompt Agent

3.3.3 Range Analysis

The chat_template exhibits unique behavior patterns due to its built-in feedback structure that creates interaction effects when combined with external prompt agent systems (Table 5). This template already includes some feedback mechanisms within its conversational format.

Built-in feedback structure create dual feedback mechanisms when chat_template is combined with prompt agent. The template itself provides some guidance, while the prompt agent provides additional feedback, explaining the exceptional sensitivity patterns observed in range analysis.

Table 5: Average Loop Behavior Range Matrix Without Prompt Agent

Workflows	**chat_template**	**prompt_base**	**variation_1**	**variation_2**	**variation_3**
BASE	9.00	23.00	22.25	25.25	20.00
SINGLE_BON	9.75	27.50	19.25	21.50	20.50
MULTI_PROP	17.00	35.50	37.75	36.75	33.00
MULTI_BON	13.50	44.00	48.00	46.00	41.75

Conclusion

The key contributions of this research include several important findings that advance the field of AI-driven molecular design. Universal temperature trend identification provides a first evidence for consistent optimization strategies across different language model in molecular generation applications. The discovery that higher temperatures consistently improve success rates while reducing loop behavior gives a reliable optimization strategy that works regardless of other system choices.

Systematic prompt agent impact demonstrates that feedback mechanisms represent essential rather than optional components for effective molecular generation systems. The drastic improvements in both success rates and computational efficiency show that implementing feedback should be a priority.

Best-of-N algorithm validation in chemical contexts confirms that multi-candidate generation with optimized selection provides consistent benefits that justify additional computational overhead. This finding gives system designers confidence that implementing multi-candidate approaches will improve performance.

The practical impact of this research extends to several areas that can immediately benefit existing systems. The parameter optimization guidelines can improve current molecular generation systems without requiring major architectural changes. Template design can increase performance by choosing appropriate instruction formats for their specific models and applications.

This work provides concrete evidence that prompt engineering strategies and feedback system design often provide greater performance improvements than computational scaling or model modifications. This suggests that research efforts might be better directed toward improving feedback and prompt design.

While this study demonstrates interesting and promising results, several limitations should be acknowledged. The findings are inherently study-dependent and may be specific to the AutoSMILES system architecture and experimental conditions used here. The template analysis, in particular, represents a limited exploration of the prompt engineering space, having benchmarked only five template variations. This constraint suggests that much more extensive work is needed to fully understand the relationship between prompt design and molecular generation performance.

Future research should explore a broader range of template structures, instruction formats, and prompt engineering strategies to validate whether the observed patterns generalize across different system architectures and molecular design tasks. Additionally, the temperature and feedback mechanism findings, while consistent across the tested configurations, would benefit from validation on different agent frameworks and molecular property targets to establish their broader applicability in the field of AI-driven molecular design.

References

[1] P. G. Polishchuk, T. I. Madzhidov, and A. Varnek. "Estimation of the size of drug-like chemical space based on GDB-17 data". In: *Journal of Computer-Aided Molecular Design* 27.8 (2013), pp. 675–679.

[2] J. A. DiMasi, H. G. Grabowski, and R. W. Hansen. "Innovation in the pharmaceutical industry: new estimates of R&D costs". In: *Journal of Health Economics* 47 (2016), pp. 20–33.

[3] D. Sun et al. "Why 90% of clinical drug development fails and how to improve it?" In: *Acta Pharmaceutica Sinica B* 12.7 (2022), pp. 3049–3062.

[4] Pharmaceutical Research and Manufacturers of America (PhRMA). *Research & Development*. https://phrma.org/policy-issues/research-development.

[5] D. Weininger. "SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules". In: *Journal of Chemical Information and Computer Sciences* 28.1 (1988), pp. 31–36.

[6] K. M. Jablonka et al. "Leveraging large language models for predictive chemistry". In: *Nature Machine Intelligence* 6.2 (2024), pp. 161–169.

[7] M. C. Ramos, C. J. Collison, and A. D. White. "A Review of Large Language Models and Autonomous Agents in Chemistry". In: *arXiv preprint arXiv:2407.01603v3* (2024). cs.LG.

[8] D. A. Boiko et al. "Autonomous chemical research with large language models". In: *Nature* 624.7992 (2024), pp. 570–578.

[9] J. Ross et al. "Large-scale chemical language representations capture molecular structure and properties". In: *Nature Machine Intelligence* 4 (2022), pp. 1256–1264.

[10] S. Chithrananda, G. Grand, and B. Ramsundar. "ChemBERTa: Large-scale self-supervised pretraining for molecular property prediction". In: *arXiv preprint arXiv:2010.09885* (2020).

[11] C. Edwards et al. "Translation between molecules and natural language". In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. 2022, pp. 375–413.

[12] J. M. Cavanagh et al. "SmileyLlama: Modifying Large Language Models for Directed Chemical Space Exploration". In: *arXiv preprint arXiv:2409.02231* (2024). physics.chem-ph.

[13] K. Sun et al. "SynLlama: Generating Synthesizable Molecules and Their Analogs with Large Language Models". In: *arXiv preprint arXiv:2503.12602v1* (2025). cs.LG.

[14] S. Reed. "Augmented and Programmatically Optimized LLM Prompts Reduce Chemical Hallucinations". In: *ChemRxiv* (2025).

[15] C. Yang et al. "Large Language Models as Optimizers". In: *arXiv preprint arXiv:2309.03409* (2024). cs.LG.

[16] A. M. Bran et al. "ChemCrow: Augmenting large-language models with chemistry tools". In: *arXiv preprint arXiv:2304.05376v5* (2023). physics.chem-ph.

[17] M. Ansari et al. "dZiner: Rational Inverse Design of Materials with AI Agents". In: *arXiv preprint arXiv:2410.03963* (2024). physics.chem-ph.

[18] T. J. Callahan, N. H. Park, and S. Capponi. "Agentic Mixture-of-Workflows for Multi-Modal Chemical Search". In: *arXiv preprint arXiv:2502.19629* (2025). cs.AI.

[19] Q. Wu et al. "AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation". In: *arXiv preprint arXiv:2308.08155v2* (2023). cs.AI.

[20] S. Kumbhar et al. "Hypothesis Generation for Materials Discovery and Design Using Goal-Driven and Constraint-Guided LLM Agents". In: *arXiv preprint arXiv:2501.13299v2* (2025). cs.CL.