Beyond SMILES: Evaluating Agentic Systems for Drug Discovery

Edward Wijaya
StemRIM, Inc.
wijaya@stemrim.com

Abstract

Agentic systems for drug discovery can plan syntheses, mine literature, and design molecules, but how well do they generalize beyond small-molecule workflows? We evaluate six frameworks against 15 task classes spanning peptide therapeutics, in vivo pharmacology, and resource-constrained settings. The result: five critical capability gaps. Current frameworks cannot handle protein language models or peptide-specific prediction, lack bridges between in vivo and in silico data, support only LLM inference rather than ML training or reinforcement learning, assume large-pharma resources, and optimize single objectives while ignoring safety-efficacy-stability trade-offs. A knowledge-probing experiment across four frontier LLMs shows that peptide expertise already exists in foundation models (aggregate gap $= -0.115$, all $p > 0.05$) but remains stranded behind small-molecule-only architectures. We derive design requirements and a capability matrix to guide next-generation frameworks toward computational partners that augment practitioner judgment under realistic constraints.

1 Introduction

1.1 The Current Landscape

Recent agentic AI systems have made tangible progress. Coscientist autonomously plans chemical syntheses [1], ChemCrow orchestrates 18 chemistry tools [2], and ChatInvent completed a deployment at AstraZeneca for molecular design and synthesis planning [3]. PharmAgents integrates knowledge graphs for target identification [4], TxGemma provides therapeutics-focused language understanding [5], while MADD [6] and DiscoVerse [7] promise multi-agent collaboration. The dominant narrative positions agentic systems as the next major advance, moving beyond static models to systems that autonomously navigate literature, design experiments, and propose hypotheses [8, 9].

The architectural pattern is consistent: a large language model orchestrates tool calls, synthesizes results, and generates explanations. ChemCrow routes requests to RDKit, PubChem, and reaction prediction APIs. ChatInvent generates molecular designs informed by literature. Coscientist interfaces with laboratory automation. This LLM-centric design works for text-based reasoning

tasks: literature review, synthesis enumeration, protocol documentation, and safety analysis. However, these demonstrations are narrowly scoped to specific contexts. Most systems are optimized for small-molecule workflows, high-throughput in vitro assays, and organizations with large datasets and extensive compute. When those assumptions break, performance degrades in ways the demos do not reveal.

An important distinction underlies the analysis that follows. Individual tools addressing aspects of each gap are emerging: peptide-aware generative models, multi-objective optimizers, and omics analysis platforms exist as standalone capabilities. The capability gaps we identify are not the absence of individual tools but the absence of agentic *workflow integration* that chains these capabilities into end-to-end pipelines supporting iterative design-test cycles, proprietary data, and human-in-the-loop decision-making. To test whether this bias extends to the foundation models themselves, we probe four frontier LLMs on matched small-molecule and peptide questions as a diagnostic: all four models demonstrate competent peptide reasoning, isolating the bottleneck to agent architecture rather than model capability (§2.5).

1.2 Scope and Motivation

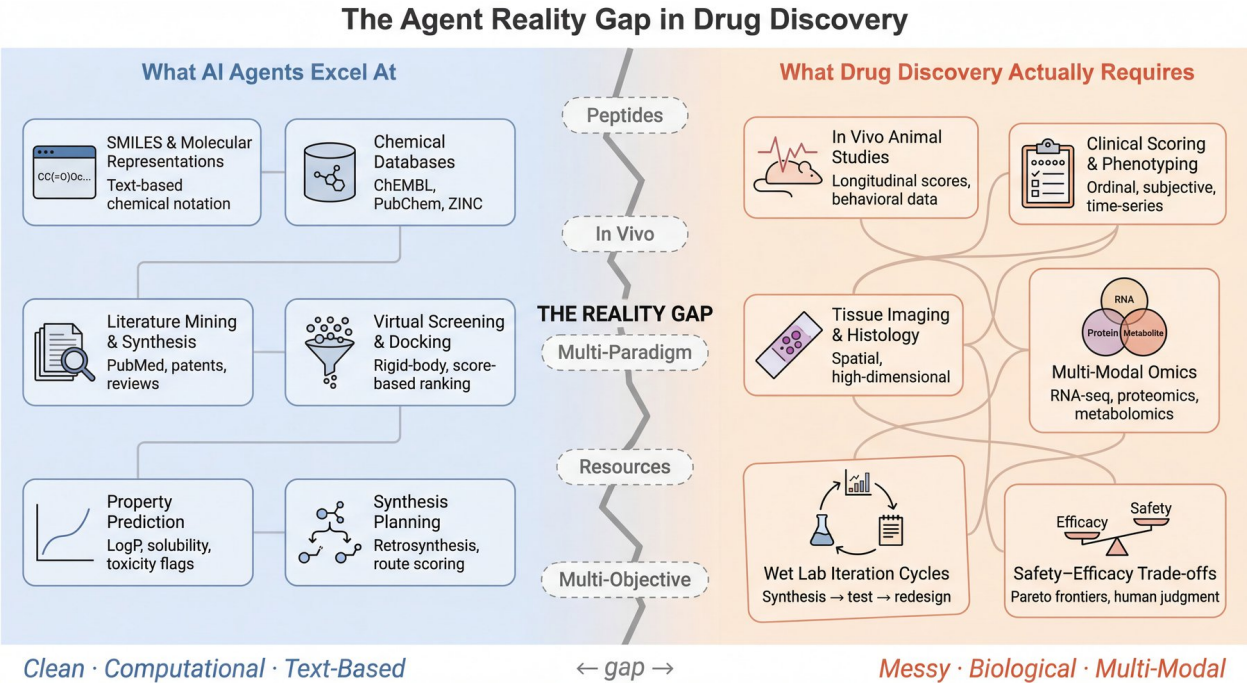

Figure 1: The Agent Reality Gap in Drug Discovery. Left panel shows computational workflows where current agents excel: small molecule representations (SMILES strings), databases, literature mining, and virtual screening. Right panel depicts the messy reality of drug discovery: multi-modal biological data from animal studies, wet lab iteration, and multi-objective trade-offs. The gap between these contexts represents the architectural limitations addressed in this paper.

However, these systems reveal systematic capability gaps outside their design context: small-

molecule discovery at well-resourced pharmaceutical companies. Peptide therapeutics require protein language models like ESM-2 [10] or ProtBERT [11], not molecular fingerprints. Peptides (5 to 50 amino acids) have complex conformational dynamics, aggregation propensities, and protease vulnerabilities absent in small molecules. No current agent supports protein language model fine-tuning, conformational sampling, or aggregation prediction.

In vivo efficacy studies generate longitudinal, multi-modal data: behavioral scores over weeks, tissue histology, RNA sequencing, and clinical notes. In neurological injury models, efficacy manifests through staged recovery endpoints: early motor improvements, subsequent reduction in neuroinflammation, and longer-term neurogenesis. No agent integrates these temporal data streams for outcome prediction. The result is a gap between in vitro promise and in vivo reality, where most development cost and risk actually sit.

Small biotechs face different constraints than AstraZeneca: 50 to 500 proprietary sequences versus millions, single GPU versus clusters, one person handling design, modeling, and analysis. Transfer learning and few-shot adaptation are prerequisites for workflows with 50-500 proprietary sequences. Current agents assume abundant resources and long, interactive cycles that do not match small-team workflows.

Real drug discovery navigates multi-objective trade-offs under uncertainty. A peptide with tenfold higher bioactivity may have narrower safety margins or reduced stability. Current agents optimize single metrics or weighted sums, ignoring Pareto frontiers and uncertainty quantification. Practitioners end up doing this reasoning manually, which slows iteration and increases decision risk.

This paper presents a systematic gap analysis drawing on over a dozen computational projects spanning peptide design, reinforcement learning optimization, in vivo efficacy modeling, behavioral phenotyping via computer vision, RNA-seq analysis, and multi-objective navigation, led by the author at a small biotech serving as both drug designer and computational practitioner. We evaluate six agentic frameworks (Table 1) against 15 task classes derived from practitioner workflows (§2), introducing a capability matrix across five evaluation dimensions. Our analysis reveals five critical capability gaps: small-molecule representation bias (§3.1), absence of in vivo-in silico integration (§3.2), limited computational paradigm support (§3.3), misalignment with small-biotech constraints (§3.4), and single-objective optimization assumptions (§3.5). From these gaps, we derive design requirements for next-generation frameworks (§4). These gaps are analytically distinct but practically intertwined: multi-paradigm orchestration (Gap 3) is a prerequisite for peptide-aware workflows (Gap 1), and multi-objective reasoning (Gap 5) is needed for both in vivo translation (Gap 2) and resource-constrained decision-making (Gap 4). We present them separately to clarify the architectural requirements, while recognizing that solutions must address them jointly.

2 Evaluation Framework

2.1 Agent Framework Selection

We identified candidate frameworks through systematic search of arXiv, PubMed, and Google Scholar (through January 2026) using terms including "agentic drug discovery," "LLM drug design," and "autonomous chemistry agent." We selected six frameworks representing distinct design paradigms in drug discovery automation (Table 1). Selection criteria required published or preprint documentation with sufficient architectural detail, demonstrated application to drug discovery tasks, and representation of distinct paradigms (single-agent, multi-agent, tool-augmented). We excluded systems that are foundation models rather than agent frameworks (TxGemma [5]), evaluation benchmarks rather than deployable systems (BioPlanner [12], ChemToolAgent [13]), and single-paradigm ML automation tools (Agentomics [14], ML-Agent [15]), which we discuss as complementary developments in §3.3.

Table 1: Agentic Frameworks Evaluated

Framework	Year	Organization	Primary Focus
ChemCrow [2]	2023	EPFL/Rochester	Chemistry tool orchestration
Coscientist [1]	2023	CMU	Autonomous synthesis
PharmAgents [4]	2025	Tsinghua	Target-compound interaction
ChatInvent [3]	2026	AstraZeneca	Literature-driven hypothesis
MADD [6]	2025	Multi-institutional	Multi-agent drug design
DiscoVerse [7]	2025	Roche	Discovery workflow automation

2.2 Task Class Definition

We defined 15 task classes derived from real-world drug discovery workflows spanning peptide therapeutics, in vivo pharmacology, and computational biology. These task classes represent the computational requirements encountered across 14 projects at a small biotech specializing in therapeutic peptides:

1. ML bioactivity prediction (multi-endpoint regression)
2. Generative peptide design (PLM fine-tuning)
3. Peptide-receptor binding site analysis and clustering
4. In vivo recovery modeling (longitudinal clinical scores)
5. Peptide-enzyme interaction modeling for stability optimization
6. Protein language model-based receptor type prediction
7. Monte Carlo optimization for peptide landscape exploration
8. RNA sequencing and single-cell transcriptomics analysis
9. Digital image processing for tissue quantification
10. Immune response profiling (pathway analysis)
11. Functional annotation and pathway enrichment
12. Computer vision for behavioral phenotyping

13. Predictive modeling bridging in vivo and in vitro endpoints
14. Reinforcement learning for de novo peptide generation
15. Safety and toxicology modeling (dose-response, multi-objective trade-offs)

These task classes align with established drug discovery workflow taxonomies. Schneider et al. [16] identify generative design, property prediction, and optimization as core computational tasks; Vamathevan et al. [17] catalog ML applications spanning target identification, compound screening, and preclinical development. Our taxonomy extends these to include in vivo modeling, behavioral phenotyping, and multi-objective optimization, which are underrepresented in existing computational frameworks. Some task classes overlap intentionally: T10 (immune profiling) and T11 (functional annotation) share pathway analysis methods but differ in biological focus; T7 (Monte Carlo optimization) and T14 (RL generation) both explore sequence space but use distinct algorithmic approaches. We deliberately exclude task classes where agents already perform well, such as literature synthesis, retrosynthesis planning, and molecular property lookup, focusing instead on capabilities that remain unsupported.

2.3 Evaluation Dimensions

Each framework was evaluated across five dimensions capturing distinct aspects of drug discovery computational requirements:

1. **Molecular representation coverage:** Support for peptides, proteins, and biologics beyond SMILES strings and molecular fingerprints.
2. **Computational paradigm support:** Capacity for ML training, reinforcement learning, simulation, and constrained optimization beyond LLM inference and API calls.
3. **Data modality integration:** Handling of in vivo longitudinal data, imaging, transcriptomics, and behavioral data beyond text and tabular formats.
4. **Resource assumptions:** Alignment with varying data volumes, compute budgets, and team sizes, particularly resource-constrained settings.
5. **Optimization framework:** Support for multi-objective optimization, uncertainty quantification, and constraint satisfaction beyond single-metric objectives.

These five evaluation dimensions and the five gaps identified in §3 are not in one-to-one correspondence. Gaps emerge from clusters of low coverage across multiple dimensions: for example, Gap 1 (small-molecule bias) reflects limitations in both molecular representation (D1) and computational paradigm support (D2), while Gap 4 (small-biotech constraints) spans resource assumptions (D4), computational paradigm support (D2), and data modality integration (D3).

2.4 Analysis Approach

For each framework-task pair, we performed a three-level capability assessment: full support (the framework provides end-to-end workflow coverage for the task class), partial support (the framework provides adjacent or limited capabilities that address aspects of the task class without complete workflow coverage), and not supported (the framework provides no relevant capability). Partial

support received a weight of 0.5 when computing coverage scores. We computed a coverage score as the weighted fraction of 15 task classes addressable per framework. Gaps were identified as task classes with zero or minimal framework coverage. Design requirements were derived from the capabilities needed to close identified gaps, grounded in practitioner experience with the corresponding task classes.

Capability assessments were based on examination of published manuscripts, available source code repositories, official documentation, and publicly accessible demonstrations for each framework. All assessments were performed by a single rater (the author); we acknowledge this as a limitation in §5. Evidence supporting individual assessments is documented in Appendix B.

This approach has limitations: the three-level assessment and the 0.5 weighting for partial support may not capture the full spectrum of capability nuance, and the rapidly evolving nature of the field means new frameworks may address some identified gaps. We discuss these limitations further in §5.

2.5 LLM Knowledge Probing

To determine whether the small-molecule bias identified in the capability matrix (§3.1) reflects upstream knowledge limitations in foundation models, we designed a paired knowledge-probing experiment. We tested four frontier LLMs from independent training pipelines: Kimi K2.5 (Moonshot), DeepSeek V3.2 (DeepSeek), Qwen 3 Next 80B (Alibaba), and Gemini 3 Flash (Google). Each model answered 50 matched question pairs spanning five pharmaceutical knowledge categories: SAR reasoning, ADMET and pharmacokinetic properties, generative design strategies, optimization approaches, and assay interpretation. Each pair tested the same cognitive skill in two modality contexts (one small-molecule, one peptide), controlling for question difficulty and format.

Responses were scored on a 0–3 rubric (wrong, partially correct, correct, expert-level). Primary scoring used Claude Sonnet 4.5 as an LLM-as-judge [18]; a domain expert independently validated a stratified 20% subset ($N = 80$) under a blind protocol (model identity and domain labels stripped, row order randomized). An additional sensitivity check rescored a 40-response subset using Claude Opus, confirming stability across judge models (quadratic-weighted $\kappa = 0.78$).

Statistical analysis used paired Wilcoxon signed-rank tests per model (Bonferroni-corrected $\alpha = 0.0125$), matched-pairs rank-biserial correlation for effect size, a Friedman test for cross-model consistency, and bootstrap 95% confidence intervals for the aggregate gap. The full question set, raw responses, and scoring data are available in the supplementary materials.

3 Results: Five Capability Gaps

Table 2 presents the core result of our evaluation: a capability matrix mapping 15 task classes against six agentic frameworks. Coverage is sparse. No framework fully supports any of the 15 task classes. Partial support, where a framework provides adjacent functionality that does not address the task-specific requirements, appears in only a few cases. The five gaps identified in the

following subsections emerge directly from this matrix: clusters of zero-coverage task classes that share underlying architectural limitations.

Table 2: Capability Matrix: Six Frameworks Evaluated Against 15 Task Classes. ✓ = supported, ○ = partial (adjacent capability exists but does not meet task requirements), × = not supported. Coverage scores count ✓ as 1 and ○ as 0.5. CC = ChemCrow, CS = Coscientist, PA = PharmAgents, CI = ChatInvent, MA = MADD, DV = DiscoVerse.

#	Task Class	CC	CS	PA	CI	MA	DV	Gap
1	ML bioactivity prediction	×	×	○	×	○	○	3
2	Generative peptide design	×	×	×	×	×	×	1
3	Peptide-receptor binding	○	×	○	×	○	×	1
4	In vivo recovery modeling	×	×	×	×	×	×	2
5	Peptide-enzyme stability	×	×	×	×	×	×	1
6	PLM receptor prediction	×	×	×	×	×	×	1
7	Monte Carlo optimization	×	×	×	×	×	×	1,3
8	RNA-seq / scRNA-seq	×	×	×	×	×	×	2
9	Image-based quantification	×	×	×	×	×	×	2
10	Immune response profiling	×	×	○	○	×	○	2
11	Functional annotation	×	×	○	○	×	○	2
12	Behavioral phenotyping	×	×	×	×	×	×	2
13	In vivo/in vitro bridging	×	×	×	×	×	×	2
14	RL peptide generation	×	×	×	×	×	×	1,3
15	Safety/toxicology (multi-obj)	○	×	○	○	○	○	5
Coverage (%)		**6.7**	**0.0**	**16.7**	**10.0**	**10.0**	**13.3**	

Partial support notes: Task 1: frameworks invoke pre-trained predictors but cannot train, validate, or tune models on proprietary data. Task 3: small-molecule docking tools exist but do not handle peptide conformational flexibility. Tasks 10–11: knowledge graph queries or literature synthesis provide pathway-level information but not computational enrichment pipelines (GSEA, upstream regulator analysis). Task 15: toxicophore flagging or ADMET prediction for small molecules; no multi-objective trade-off reasoning or dose-response modeling.

3.1 Gap 1: Small-Molecule Representation Bias

Current agentic systems are architected around small molecules: SMILES strings, docking scores, synthetic accessibility metrics, and retrosynthesis planning. This works for medicinal chemistry but breaks down for peptide therapeutics requiring fundamentally different computational approaches.

3.1.1 Findings

All six frameworks evaluated assume small-molecule representations (SMILES strings, molecular fingerprints, docking scores). Task classes 2, 3, 5, 6, and 7 (peptide-specific workflows) receive zero coverage across all frameworks. No framework supports protein language models (ProtBERT, ESM-2, ProGen) as first-class components for sequence-based therapeutics design.

3.1.2 Stranded Knowledge: A Diagnostic Experiment

As a diagnostic, we probed four frontier LLMs on matched pharmaceutical knowledge questions to test whether the small-molecule bias extends to the foundation models powering these agents (§2.5). All four models demonstrate competent peptide reasoning across all five categories (Table 3, fig. 2). Across 200 paired observations, peptide questions receive marginally higher scores than small-molecule questions (aggregate gap $= -0.115$, 95% CI: $[-0.255, 0.02]$). No model shows a statistically significant peptide deficit (all $p > 0.6$, Bonferroni-adjusted $p = 1.0$), and the Friedman consistency test confirms this holds uniformly across all four models ($\chi^2 = 0.454$, $p = 0.929$). Four of five categories favor peptides; Optimization Approaches shows the largest pro-peptide gap (+0.225).

Expert validation revealed that models produce directionally correct but quantitatively over-confident reasoning in both domains equally, consistent with a depth limitation rather than a modality-specific knowledge gap. The paired within-model comparisons driving the statistical tests are robust to calibration differences between expert and automated scoring (see §5.1 for details).

The diagnostic isolates the defect: the peptide expertise is present in foundation models but stranded behind small-molecule-only agent architectures. LLMs reason competently about peptide SAR, ADMET properties, generative design, optimization, and assay interpretation. The knowledge exists; no current agent provides a pathway to surface it through peptide-aware tools, protein language model integration, or sequence-native workflows. This strengthens rather than undermines **Gap 1**: the fix requires building the integration pipeline, not retraining the models.

Table 3: Knowledge Probing: Per-Category Score Breakdown. Mean scores (standard deviation) aggregated across four models. Gap = peptide minus small-molecule mean (positive indicates peptide advantage). p-values from one-sided Wilcoxon signed-rank tests (H_1: SM > peptide). Four of five categories show a peptide advantage, confirming that peptide expertise exists across all knowledge categories tested.

Category	**SM Mean (SD)**	**PEP Mean (SD)**	**Gap**	**p-value**
SAR Reasoning	2.60 (0.63)	2.55 (0.55)	−0.05	0.370
ADMET / PK Properties	2.33 (0.66)	2.45 (0.68)	+0.13	0.755
Generative Design	2.10 (0.71)	2.25 (0.74)	+0.15	0.805
Optimization Approaches	2.28 (0.75)	2.50 (0.60)	+0.23	0.938
Assay Interpretation	2.38 (0.74)	2.50 (0.68)	+0.13	0.767

3.1.3 The Peptide-Specific Challenge Space

Therapeutic peptides (2 to 50 amino acids) bridge small molecules and biologics. Unlike rigid small molecules, peptides are conformationally flexible, sampling diverse structural states. They achieve high selectivity through induced-fit binding but face aggregation, protease degradation, and permeability barriers. These challenges extend beyond peptides to biologics broadly: antibodies require CDR loop modeling, nanobodies need single-domain folding, and fusion proteins demand

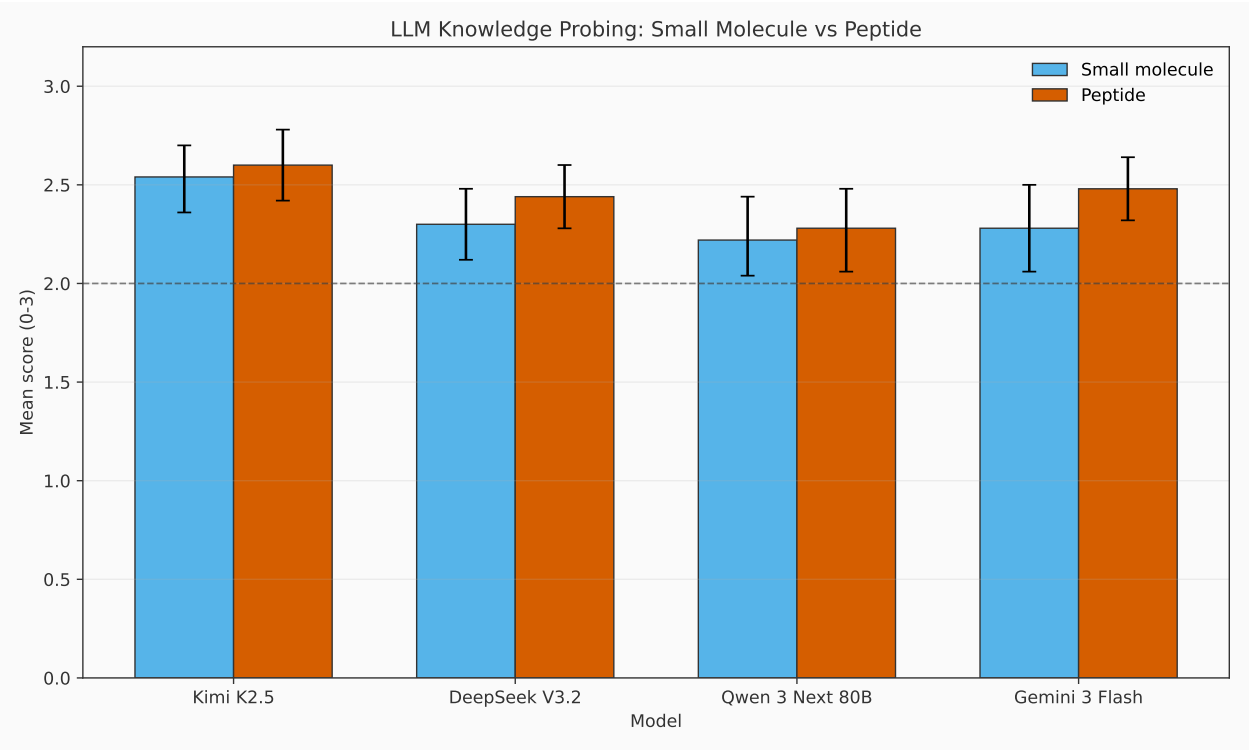

Figure 2: Stranded Knowledge: LLM Peptide Competence vs Agent Capability. Mean scores (0–3 scale) for four frontier LLMs across 50 matched question pairs spanning five pharmaceutical knowledge categories. Blue bars: small-molecule questions; orange bars: peptide questions. Error bars: 95% bootstrap confidence intervals. All four models demonstrate competent peptide reasoning at or above small-molecule levels (aggregate gap $= -0.115$, 95% CI: $[-0.255, 0.02]$, all Bonferroni-adjusted $p = 1.0$). This knowledge is stranded: no current agentic framework surfaces it through peptide-aware tools.

multi-domain interaction prediction. The small-molecule bias is a biologics-wide limitation.

Peptide discovery diverges from small-molecule workflows. Structure-activity relationships do not transfer; conservative substitutions can abolish activity while drastic changes improve potency. Stability dominates. In neurological injury indications, efficacy-stability trade-offs exceeded potency concerns. A bioactive peptide with minute-scale serum half-life has no therapeutic value. Protease resistance requires modeling interactions across dozens of enzyme families. Immunogenicity demands epitope scanning and MHC binding prediction. Aggregation depends on charge distribution and hydrophobic patterning.

Current agents provide no pathway for these requirements. ChemCrow includes RDKit, which does not handle peptide conformational sampling. PharmAgents focuses on small-molecule structure-based design workflows not designed for flexible peptide interactions. ChatInvent mines small-molecule synthesis routes and molecular design, irrelevant to peptide synthesis.

Practitioners encounter immediate friction. SMILES encoding of peptides is error-prone, risking loss of stereochemical and conformational information, particularly for non-natural amino acids. Standard molecular fingerprints (Morgan, MACCS) perform poorly on peptides, which lack equivalent standard representations. Rigid-molecule docking produces unreliable scores. Retrosynthesis metrics are meaningless. Even data storage becomes awkward: sequence variants, post-translational modifications, and assay metadata rarely fit small-molecule databases designed for single canonical structures.

3.1.4 Protein Language Models vs Molecular Fingerprints

Protein language models define peptide design. ProtBERT [11], ESM-2 [10], and ProGen [19] encode evolutionary and structural priors from millions of protein sequences. ESM-2 embeddings predict receptor types from sequence alone. ProtBERT fine-tuning enables transfer learning from limited labeled data, often requiring only hundreds of examples for task-specific classifiers.

Peptide-aware models are emerging. PepTune [20] uses a masked discrete diffusion model with Monte Carlo Tree Guidance for multi-objective peptide generation, optimizing across binding affinity, permeability, and stability simultaneously. PepMLM [21] fine-tunes ESM-2 to design peptide binders conditioned on target protein sequences, with experimental validation on disease-relevant targets. These are real advances in peptide-specific modeling. However, they are standalone tools, not components of agentic workflows. No current system chains peptide generation with proprietary fine-tuning, active learning, iterative design-test cycles, or multi-endpoint optimization in closed loops. The gap is workflow integration, not individual capability.

Building peptide workflows requires capabilities current agents lack. Developing a receptor binding classifier involves curating training sets, extracting ESM-2 embeddings, training supervised classifiers, validating performance, and iterating hyperparameters. This is gradient-based ML, not API calls. The work also depends on small, noisy datasets where careful cross-validation and calibration matter more than single headline metrics. Exposing these uncertainties clearly is a prerequisite for biologists to prioritize synthesis and testing. Without that, the workflow reverts

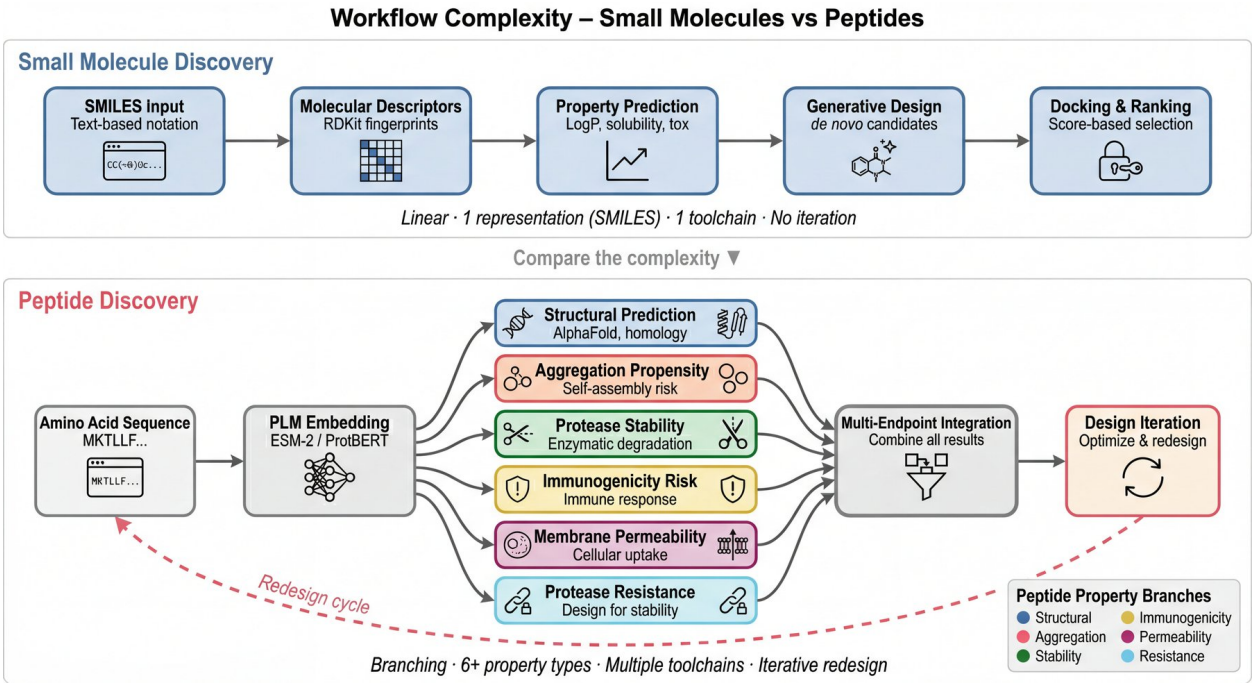

Figure 3: Workflow Complexity: Small Molecules vs Peptides. Top: Small molecule workflow follows a linear path from SMILES representation through RDKit property calculation to docking. Bottom: Peptide workflow branches into multiple parallel analysis streams including structural prediction, aggregation propensity, stability, immunogenicity, membrane permeability, and protease resistance, requiring integration of diverse computational tools and protein language models.

to manual triage and ad hoc heuristics. No current system supports this autonomously. LLM orchestrators treat models as inference-only APIs (see §3.3 for detailed analysis of this architectural limitation). Agents retrieve embeddings but cannot train task-specific classifiers on proprietary data.

Generative modeling extends this gap. ProtGPT2 [22] fine-tunes on therapeutic sequences for de novo generation. Reinforcement learning optimizes multi-objective reward functions combining bioactivity and stability, requiring reward models, policy networks, gradients, and KL regularization. These ML workflows requiring end-to-end training control exceed current agent architectures.

The gap reflects a missing paradigm, not a missing tool. Protein language models are the foundation of peptide discovery, demanding first-class support for training and fine-tuning. Without this, agents cannot support the core workflows practitioners use to move from sequence space exploration to experimentally validated leads.

3.1.5 Capability Requirements Implied by Gap

A peptide-aware agent meeting this requirement would accept a FASTA file of 200 labeled sequences and return a classifier with per-class AUC-ROC and calibration curves, or accept a generative model specification and return novel sequences with predicted property distributions and diversity metrics.

Peptide-aware architectures require protein language models as core components, not external APIs. First, fine-tuning pipelines: dataset curation, train-validation-test splits, learning rate scheduling, early stopping, checkpointing. Effective frameworks would fine-tune ProtBERT on 200 sequences and return calibrated classifiers with uncertainty estimates.

Second, structural biology integration. AlphaFold [23] structure prediction is central to peptide design. Flexible docking requires conformational sampling. Molecular dynamics provides stability and kinetics insights. These tools would need to integrate into multi-step workflows and feed back into sequence optimization, rather than remaining isolated analyses.

Third, multi-objective optimization. Peptide design balances bioactivity, stability, selectivity, and immunogenicity. We used curriculum learning for in vivo efficacy: initially rewarding bioactivity improvements, then progressively adding stability and toxicity constraints. This prevented local optima and maintained diversity. Current agents provide no framework for multi-stage optimization.

Fourth, diversity-aware generation. Generative models suffer mode collapse, producing reward-maximizing sequences with little variety. Practitioners use diversity penalties and max-min rewards, requiring state maintenance and dynamic sampling. LLM agents treat tool calls as stateless.

Finally, active learning loops. Limited budgets require careful peptide selection via acquisition functions that prioritize uncertain predictions across multiple rounds (see §3.4).

Peptide discovery is a distinct paradigm requiring ML training, structural biology, and multi-objective optimization as core capabilities. It also requires sequence-aware data management: tracking modifications, synthesis constraints, and assay provenance across iterative cycles. The small-molecule bias reflects architectural assumptions that limit current frameworks.

3.2 Gap 2: Absence of In Vivo-In Silico Integration

The absence of in vivo modeling is a fundamental gap. Current agents excel at in vitro automation (Coscientist, ChemCrow, ChatInvent), but critical validation happens in vivo, where candidates confront pharmacokinetics, biodistribution, metabolism, toxicology, and long-term efficacy unpredictable from binding affinity.

Animal studies generate a distinct class of data: longitudinal (days to months), multi-modal (behavioral scores, imaging, molecular profiling), noisy (biological variability dwarfs plate assay precision), low-throughput (tens of compounds, not thousands), and expensive. These characteristics make in vivo the bottleneck, yet agents provide no pathway to incorporate this data.

3.2.1 Findings

Task classes 4, 9, 12, and 13 (in vivo and imaging tasks) receive zero coverage across all six frameworks. All frameworks terminate at in vitro automation or literature-based hypothesis generation. No framework supports longitudinal data modeling, multi-modal fusion, or causal inference from animal study data.

3.2.2 The Lab Automation Ceiling

Lab automation reaches a hard ceiling at in vivo studies. Synthesis platforms and high-throughput screening test thousands of compounds daily. But animal experiments cannot be scaled or automated, requiring specialized facilities, personnel, ethical oversight, and weeks of time.

An agent might design a peptide and predict binding, but cannot integrate a multi-week neurological injury study measuring behavioral recovery, histological regeneration, and transcriptomic neuroprotection. Data formats, temporal structure, and statistical requirements exceed current capabilities.

In vivo studies yield heterogeneous streams. Neurological injury evaluation includes behavioral assessments (motor coordination tests generating ordinal scores), tissue histology (cell proliferation requiring computer vision), RNA sequencing (high-dimensional gene expression needing differential expression and pathway enrichment), and clinical notes (semi-structured weight and adverse events). This multi-modal integration remains outside current agent scope.

Developing composite efficacy metrics exposed this gap. A peptide increasing cell proliferation threefold in vitro shows temporal in vivo dynamics: inflammatory response (days 1-3), progenitor proliferation (days 7-10), functional recovery (day 28). Predicting sustained benefit requires temporal modeling, dataset curation, feature engineering, and validation outside LLM tool-calling scope.

3.2.3 Multi-Modal, Longitudinal Data Integration

The core challenge is that in vivo data does not fit the tidy CSV format that machine learning pipelines expect (Table 4). Behavioral scores are ordinal and subject to inter-rater variability.

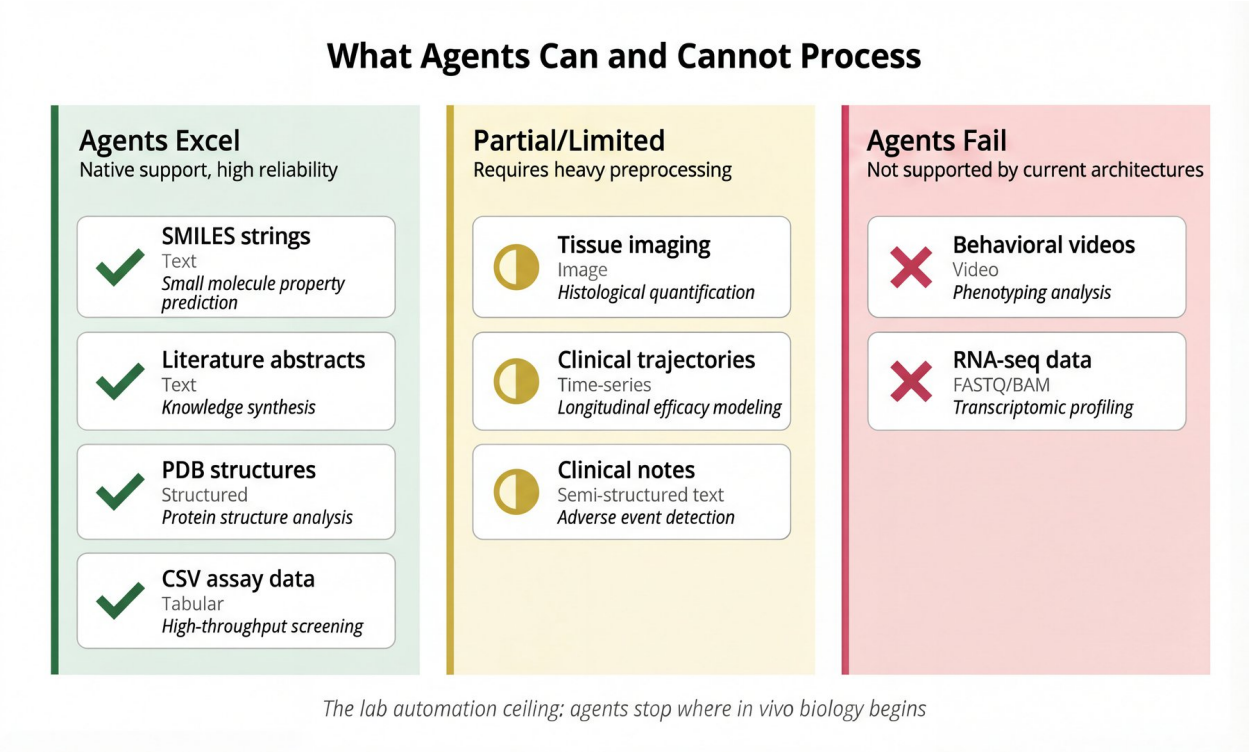

Figure 4: What Agents Can and Cannot Process. Data types grouped by agent accessibility into three tiers. Green checkmarks indicate natively supported formats (SMILES strings, literature abstracts, PDB structures, CSV assay data). Yellow half-circles denote partial support requiring heavy preprocessing (tissue imaging, clinical trajectories, clinical notes). Red X marks denote data types with no current agent support (behavioral videos, RNA-seq data). Most in vivo data modalities fall in the partial or inaccessible tiers, revealing the systematic exclusion of biological validation data from current agent architectures.

Table 4: Data Type Accessibility for Current Agent Systems

Data Type	**Format**	**Agent-Readable**	**Example Use Case**
SMILES strings	Text	Yes	Small molecule property prediction
Literature abstracts	Text	Yes	Knowledge synthesis
PDB structures	Structured	Yes	Protein structure analysis
CSV assay data	Tabular	Yes	High-throughput screening
Behavioral videos	Video	No	Phenotyping analysis
Clinical trajectories	Time-series	Partial	Longitudinal efficacy modeling
Tissue imaging	Image	Partial	Histological quantification
RNA-seq data	FASTQ/BAM	No	Transcriptomic profiling
Clinical notes	Semi-structured text	Partial	Adverse event detection

Behavioral phenotyping via DeepLabCut [24] tracks animal poses in videos, generating time-series keypoint coordinates. Computing behavioral metrics from tracked keypoints requires training pose estimation, validating tracking, computing features, and statistical testing. Practitioners must handle this workflow manually, spanning video processing, supervised learning, time-series engineering, and hypothesis testing.

RNA-seq requires quality control, alignment, and quantification into expression matrices. Differential expression identifies treatment effects. Pathway enrichment maps genes to biological processes via KEGG [25] or Gene Ontology. Upstream regulator analysis infers transcription factors driving changes. The FASTQ-to-hypothesis pipeline needs bioinformatics tools (STAR, HISAT2, DESeq2, edgeR, GSEA [26]) that agents do not integrate. Recent systems like Medea [27] have begun addressing multi-omics analysis agenically, handling transcriptomics, protein networks, and pathway analysis. However, these process static datasets rather than longitudinal in vivo time-series, and do not integrate behavioral phenotyping, imaging, or temporal efficacy modeling.

Integrating heterogeneous sources for predictive biomarkers is valuable yet absent. Correlating in vitro bioactivity with in vivo efficacy required extracting features from multiple assays, normalizing across scales, aligning with temporal data (days 3, 7, 14, 28), and training regression models predicting long-term outcomes. This workflow involved feature engineering, imputation, stratified cross-validation, and model selection; these are ML workflows, not LLM reasoning.

3.2.4 Safety, Efficacy, and Translation

In vivo models surface safety-efficacy trade-offs that in silico screens miss. Tenfold bioactivity increases may trigger immune activation or hepatotoxicity. Stability modifications may introduce affinity or aggregation trade-offs (§3.1).

Toxicology requires dose-response analysis via generalized linear mixed models accounting for repeated measures and time-dependent effects. Identifying therapeutic windows where efficacy plateaus but toxicity remains acceptable is a standard requirement in preclinical development. Species translation compounds this: mouse pharmacokinetics extrapolate to humans via allometric scaling, but peptide stability varies by species protease expression and rodent-tolerated peptides may provoke primate antibody responses. Agents cannot construct dose-response curves, compute LD50 confidence intervals, or model cross-species translation uncertainties. We return to the broader multi-objective trade-offs these challenges imply in §3.5.

3.2.5 Capability Requirements Implied by Gap

An agent meeting this requirement would ingest longitudinal in vivo data (behavioral scores, imaging features, transcriptomic profiles across multiple timepoints), produce predictive models for late-stage outcomes with confidence intervals, and generate mechanistic hypotheses linking early biomarkers to treatment response.

Closing the in vivo-in silico gap requires three capabilities absent from current frameworks. First, temporal state-space models for longitudinal in vivo trajectories that capture treatment dy-

namics across days to weeks. Second, causal inference tools (do-calculus, counterfactual reasoning) to separate correlation from mechanism in complex biological systems. Third, multi-modal data fusion integrating clinical scores, imaging, transcriptomics, and behavioral data into unified predictive models. Until agents ingest longitudinal scores, integrate transcriptomics, quantify dose-response uncertainty, and navigate multi-objective trade-offs under biological variability, utility remains confined to early hit identification. Most development cost and risk lies in translating in vitro activity to in vivo efficacy and safety [28].

3.3 Gap 3: Limited Computational Paradigm Support

The in vivo data integration challenge from the previous section illustrates a deeper architectural limitation: agents are not missing specific tools so much as they lack the capacity to orchestrate diverse computational paradigms. "Multi-agent" systems in drug discovery have multiple LLM-based agents collaborating [9]. PharmAgents deploys specialized agents for target identification and synthesis. MADD coordinates molecular design and docking [6]. BioPlanner benchmarks LLM-driven protocol planning [12], and ChemToolAgent evaluates chemistry tool integration [13]. The pattern: an orchestrator LLM decomposes queries, delegates to modules, and synthesizes outputs.

This distinction matters. "Multi-agent" is multiple LLM instances with different tools. Practitioners need multi-paradigm orchestration: coordinating fundamentally different computational approaches (supervised learning, generative modeling, RL, simulation, optimization) within workflows. Current architectures support the former, not the latter, which is why most real pipelines still require manual glue code.

3.3.1 Findings

All six frameworks use LLM-as-orchestrator architecture: LLM reasoning combined with tool API calls. Task classes requiring model training (1, 2, 6, 14), reinforcement learning (7, 14), or simulation receive no support. No framework supports gradient-based optimization, hyperparameter search, or curriculum learning as first-class primitives.

3.3.2 The LLM-as-Orchestrator Limitation

LLM-centric design treats the language model as central coordinator, invoking external tools via APIs. This works for text-based reasoning and stateless tools.

The paradigm breaks for tasks like training a multi-task neural network predicting peptide bioactivity across four endpoints using 300 sequences. The workflow: dataset preparation (stratified train-validation-test splits), feature extraction (ESM-2 embeddings), architecture selection, hyperparameter tuning, training with early stopping, validation with confidence intervals.

LLMs cannot orchestrate this via API calls. It requires gradient-based ML with end-to-end control over data loading, loss computation, parameter updates, and checkpointing. Agents do

not support supervised learning as a first-class primitive they can configure, execute, monitor, and iterate.

The limitation extends beyond supervised learning. Generative modeling, reinforcement learning, Monte Carlo sampling, molecular dynamics, and Bayesian optimization all require iterative optimization with intermediate states, convergence monitoring, branching logic, resource management (GPU allocation, parallelization, checkpointing), and artifact versioning (models, hyperparameters, trajectories). None fit stateless API calls.

3.3.3 The Missing Paradigms

Absent paradigms define modern drug discovery [16]: supervised learning (bioactivity prediction [29], toxicity modeling), unsupervised learning (chemical space clustering), generative modeling (de novo design [30]), reinforcement learning (multi-objective optimization [31]), simulation (binding kinetics), and optimization (experimental design).

Table 5: Computational Task Types: Agent Support vs Practitioner Need

Task Type	Capability	Needs Human Review	Typical Runtime
Literature review	✓	No	Minutes
SMILES generation	✓	Yes	Seconds
Docking (small molecules)	✓	Yes	Hours
Aggregation prediction	×	Yes	Days
In vivo analysis	×	Yes	Days to weeks*
Multi-objective optimization	×	Yes	Hours to days

*Runtime includes experimental turnaround, not just computation.

Across our projects, most computational work involved these paradigms, not LLM reasoning. Peptide-receptor classifiers required ESM-2 embeddings, logistic regression, gradient-boosted trees, cross-validation, and AUC-ROC comparison. RNA-seq needed alignment, normalization, differential expression, clustering, and pathway enrichment. Bone formation quantification trained semantic segmentation models. Peptide optimization via RL required reward models, proximal policy optimization, and diversity penalties.

Projects spanned paradigms in integrated pipelines: generative models produced sequences, supervised models predicted bioactivity, Bayesian optimization selected synthesis batches based on uncertainty, experimental results updated training sets, cycles repeated.

Agents cannot express these workflows. Orchestrators call models for inference but cannot train models, incorporate new data, retrain with hyperparameters, validate on test sets, or coordinate generative, predictive, and experimental design algorithms in closed loops. They assume pre-trained models and inference-only tasks, which is the opposite of how discovery actually proceeds.

Recent systems have begun addressing single-paradigm ML automation. Agentomics [14] achieves autonomous ML experimentation, surpassing human-engineered state-of-the-art on 11 of 20 biomedical benchmark datasets across protein engineering, drug discovery, and regulatory genomics. ML-Agent [15] uses reinforcement learning to train LLM agents for autonomous ML engineering.

These demonstrate that agents can automate individual ML workflows. However, neither orchestrates multi-paradigm pipelines: concurrent RL-based generation, supervised bioactivity prediction, Bayesian batch selection, and structural simulation in closed loops. The gap is cross-paradigm integration, not single-paradigm automation.

3.3.4 Capability Requirements Implied by Gap

An agent meeting this requirement would accept a declarative workflow specification (e.g., "train ensemble bioactivity predictors with 5-fold cross-validation") and produce an executable workflow graph with resource estimates, provenance tracking, and human decision points.

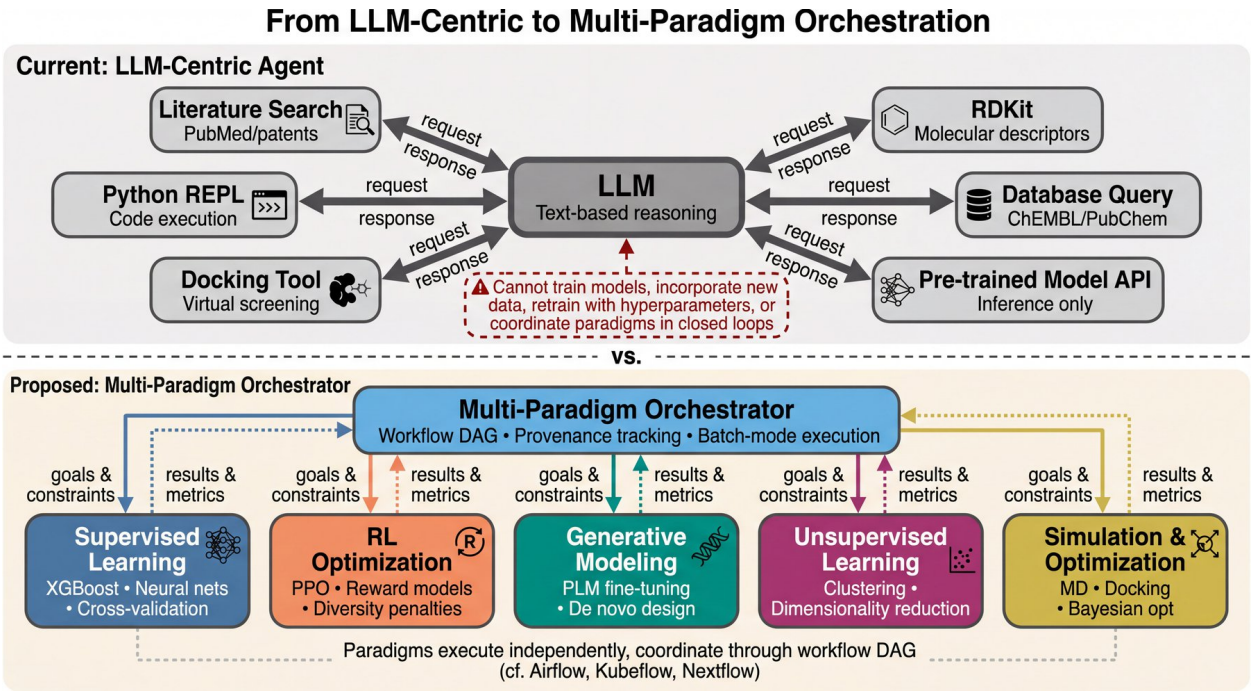

Figure 5: From LLM-Centric to Multi-Paradigm Orchestration. Top: Current LLM-centric architecture where a central language model orchestrates all tools through API calls. Bottom: Multi-paradigm architecture addressing the identified orchestration gap, where a coordinator manages fundamentally different computational paradigms (ML training pipelines, RL optimization loops, PLM fine-tuning, CV analysis, physics simulations) that execute independently with results aggregated for decision-making.

Multi-paradigm architectures treat ML training, RL, simulation, and optimization as core primitives. Practitioners specify workflows declaratively: "Train ensemble bioactivity predictors (XGBoost, random forests, neural nets) with 5-fold cross-validation and hyperparameter tuning. Return Pareto frontier trading AUC-ROC versus calibration error." Agents translate specifications into executable workflow graphs, allocate resources, monitor convergence, and track provenance.

Workflow graphs have nodes (data loading, feature extraction, training, evaluation) and edges (data dependencies). They support parallelization (simultaneous hyperparameter configs), checkpointing (cached intermediate results), and branching (automatic hyperparameter search if valida-

tion fails), with clear provenance for every artifact. Workflow orchestration exists: Apache Airflow, Kubeflow, Nextflow provide task graphs, dependency resolution, resource allocation, and checkpointing. Missing is agent reasoning integration: inspecting results, diagnosing failures, proposing modifications, learning from executions (deprioritizing architectures that overfit).

Effective interaction would be batch-mode, not chat. Bottlenecks are orchestrating end-to-end analyses, not formulating queries. Small biotechs managing dozens of projects require batch-mode automation with agents intervening only for human decisions (see §3.4).

Human-in-the-loop decision points are explicit and actionable. Agents present Pareto frontiers (accuracy-interpretability trade-offs), candidates with uncertainty estimates. Practitioners select based on context; agents structure decision spaces.

The gap reflects a mismatch with computational drug discovery practice. Current practice relies on systems coordinating paradigms in integrated workflows, batch-mode execution, explicit decision points, and version control for reproducibility, which is absent from current LLM-to-LLM delegation architectures.

3.4 Gap 4: Misalignment with Small-Biotech Constraints

ChatInvent's deployment at AstraZeneca [3] accessed institutional databases, HPC clusters, and proprietary libraries built over decades, with specialized teams (medicinal chemists, computational chemists, biologists, data scientists). This large pharma context defines current agent design assumptions but is not representative.

Small biotechs face different constraints: 50-100 employees, single wet labs, limited computational infrastructure, modest funding. Proprietary datasets have hundreds of compounds, not millions. One person designs experiments, analyzes results, and manages projects. Resource profiles are 10-100 times leaner, yet agents assume large pharma contexts.

3.4.1 Findings

All evaluated frameworks assume large-pharma resource levels: abundant proprietary data, cluster-scale compute, and specialized teams. No framework supports few-shot adaptation, active learning, or transfer learning for data-scarce settings. Interactive chat interfaces assume dedicated operator time, impractical for small teams managing multiple projects simultaneously.

3.4.2 Resource Assumptions vs Reality

Data scarcity is the first constraint. Large pharma accumulates millions of tested molecules enabling high-capacity models. Small biotechs have 50-500 proprietary sequences. Every assay is precious.

Agents assume abundant data, recommending deep neural networks with millions of parameters, hundreds of hyperparameter configs, and dozens of ensemble models. On 100 sequences, deep networks overfit catastrophically. 5-fold cross-validation leaves only 20 examples for evaluation. Ensembles offer no benefit on small datasets.

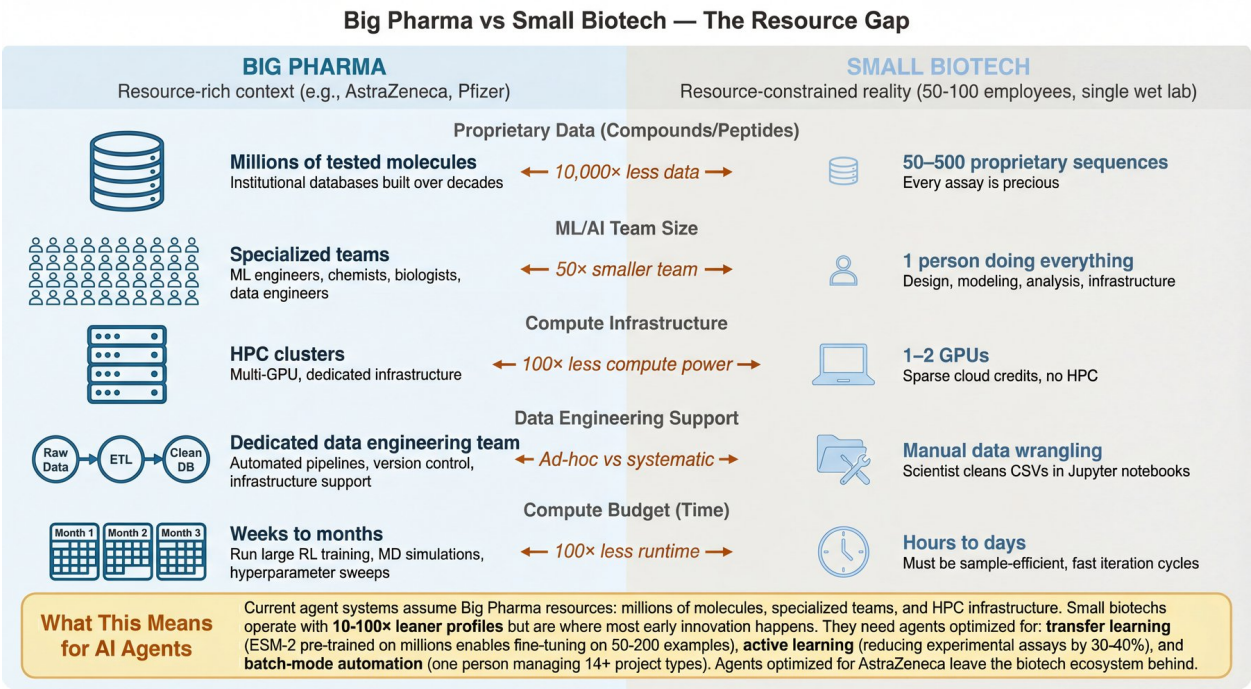

Figure 6: Big Pharma vs Small Biotech: The Resource Gap. Comparative visualization of computational resources, team size, and data availability. Left: Large pharmaceutical companies with 10,000+ compounds, 50-person ML teams, multi-GPU clusters, dedicated data engineers, and months of compute budget. Right: Small biotechnology companies with 50-200 compounds, 1-person computational teams, single GPU workstations, scientist-developers, and hours-to-days compute budgets. Current agent architectures assume the left context but significant innovation happens on the right.

Small biotechs need data efficiency: models generalizing from few examples via transfer learning, quantifying uncertainty for experimental design. ESM-2 pre-trained on millions of protein sequences enables fine-tuning lightweight classifiers on 50-200 examples. Active learning can substantially reduce experimental burden by prioritizing uncertain predictions [32]; in our peptide projects, this reduced required assays by approximately one-third. Both require workflows agents do not support.

Computational constraints compound scarcity. Small biotechs have 1-2 GPUs, sparse cloud credits, no HPC. RL, molecular dynamics, and docking are expensive. Agents recommend intensive methods without considering infrastructure, ignoring efficiency optimizations (parallelization, caching, cheaper approximations). Resource-aware agents would propose: "Given one GPU and 24 hours: 100 high-precision docking runs or 1,000 fast approximations?"

Team structure is the third constraint. Large pharma has specialized roles (ML engineers, chemists, biologists, data engineers). Small biotechs have one person doing everything: peptide design, ML modeling, experimental analysis. No dedicated infrastructure support.

This context demands tools for generalists handling infrastructure complexity (dependencies, environments, memory, debugging), where practitioners specify what, not how. Current agents assume infrastructure exists; generated code assumes installed libraries, formatted data, available resources. For small biotechs, these assumptions fail.

3.4.3 Transfer Learning and Few-Shot Adaptation

Transfer learning leverages public data for small proprietary tasks. Protein language models trained on UniProt's millions of sequences achieve performance with 50-200 examples that would require tens of thousands if trained from scratch.

Effective transfer learning requires selecting models (ESM-2, ProtBERT, ProGen), choosing fine-tuning layers (freezing early layers is data-efficient), setting learning rates (avoiding catastrophic forgetting or slow convergence), and implementing regularization. These are experimental decisions requiring domain knowledge.

Few-shot learning extends transfer to extreme scarcity. In our experience, prototypical networks achieved reasonable accuracy on peptide-receptor binding classification with tens of examples per type, sufficient for initial screening prioritization though dependent on the number of receptor classes and baseline rates. Agents provide no few-shot pathways, meta-learning support, or confidence interval quantification. A 70

Active learning selects which data to acquire next, prioritizing experiments reducing uncertainty. Acquisition functions (expected improvement, upper confidence bound) balance exploitation and exploration. Developing peptide bioactivity predictors, active learning reduced assays by one-third. Round one: 20 diverse peptides. Round two: 15 targeting high uncertainty. Round three: exploiting predicted best candidates. This iterative loop between models, acquisition functions, and feedback exceeds agent capabilities.

3.4.4 Batch-Mode Efficiency for Small Teams

Interactive chat assumes time for conversational interaction. This works for specific queries, not managing multiple projects simultaneously.

Small biotechs need batch-mode automation: "Analyze eight RNA-seq samples, identify differentially expressed genes, perform pathway enrichment, generate report." Agents execute autonomously overnight, intervening only for human decisions (which mechanism aligns with biological knowledge?).

Batch-mode requires robustness. If RNA-seq alignment fails (memory limits), effective frameworks would adjust parameters (reduce threads) or flag issues without losing progress. Checkpointing, fault tolerance, and version control enable reproducibility.

Parallelization is a standard requirement for computational drug discovery. Given 100 peptide docking jobs, effective frameworks would automatically parallelize across available resources (eight CPU cores, GPU acceleration), optimizing throughput without manual scheduling.

Current agents support minimal batch capabilities, designed for interactive queries not unsupervised analyses. They lack checkpointing, parallelization, and error handling, assuming interactive debugging impractical for overnight jobs.

3.4.5 Capability Requirements Implied by Gap

An agent meeting this requirement would, given 50–200 labeled sequences and a single GPU, recommend an appropriate modeling strategy, execute transfer learning with uncertainty quantification, and return predictions with calibrated confidence intervals within a 24-hour compute budget.

Small biotech is not large pharma with fewer resources. It is a different operating mode requiring few-shot learning modules for rapid adaptation to new assays, active learning loops with acquisition functions balancing exploration and exploitation, transfer learning pipelines composing public pretraining with private fine-tuning, and batch-mode orchestration enabling unsupervised overnight analyses. Resource-aware frameworks would propose strategies matched to available infrastructure rather than assuming cluster-scale compute. The biotech sector, which accounts for a growing share of therapeutic innovation, remains underserved by current agent architectures designed for large pharma contexts.

3.5 Gap 5: Single-Objective Optimization Assumptions

3.5.1 Findings

All frameworks optimize single objectives or use fixed weighted sums. No framework supports Pareto optimization, constraint satisfaction, or uncertainty-aware candidate selection. Task class 15 (safety/toxicology modeling with multi-objective trade-offs) receives only partial coverage: five of six frameworks provide adjacent capabilities (toxicophore flagging, ADMET prediction) but none supports multi-objective trade-off reasoning, Pareto optimization, or dose-response modeling.

3.5.2 Analysis

Resource constraints amplify the cost of poor decisions. When every experiment is precious, navigating multi-objective trade-offs becomes critical. Consider three peptide candidates from a development campaign. One shows tenfold higher proliferation bioactivity but triggers hepatotoxicity at effective doses. Another has half the bioactivity but higher tolerated dosing, yielding comparable efficacy with better safety. A third has intermediate bioactivity and safety but superior stability enabling less frequent dosing. "Optimal" depends on patient population, dosing regimen, and regulatory risk tolerance. No single metric captures this.

Drug discovery is multi-objective optimization: candidates must satisfy bioactivity, selectivity, safety, stability, manufacturability, and cost. Objectives conflict: potency improvements reduce selectivity, stability enhancements increase immunogenicity, high-purity synthesis is expensive. Navigating requires understanding Pareto frontiers (candidates where improving one objective degrades another) and decisions based on risk tolerance, development stage, and context.

Current agents optimize single objectives. ChemCrow optimizes binding affinity or synthetic accessibility [2]. Coscientist targets synthesis yield [1]. Multiple objectives collapse to weighted sums: "Maximize 0.6×bioactivity + 0.4×drug-likeness" [33]. This discards information: which candidates are Pareto-optimal, how sensitive are rankings to weights, what trade-offs exist. Agents present single "optimal" solutions, obscuring decision spaces.

3.5.3 The Single-Metric Trap

Single-metric optimization mirrors ML training objectives: minimize loss, maximize accuracy. This works for unidimensional goals, but drug discovery is multidimensional and context-dependent. Optimality depends on indication, development stage, competitive landscape, and risk tolerance.

Peptide stability illustrates the trap. The stability-activity trade-offs described in §3.1 create a multidimensional decision space where the balance depends on route of administration, therapeutic window, and development timeline.

Agents cannot represent these trade-offs. They predict A > B but cannot articulate: "A is twice as potent but has a threefold narrower safety margin; choose A if dosing can be tightly controlled, choose B for robustness." They do not visualize Pareto frontiers or sensitivity to weight changes.

3.5.4 Pareto Frontiers and Constraint Satisfaction

Pareto optimization is the appropriate framework. A candidate is Pareto-optimal if no other improves one objective without degrading another. The Pareto frontier is a curve (two objectives) or surface (three+). Practitioners navigate this frontier based on context.

Frontier visualization reveals trade-off structure. Steep regions require large sacrifices for modest gains; flat regions allow improvements with minimal cost. Clusters suggest distinct strategies (high-potency narrow-margin versus moderate-potency wide-margin). Gaps reveal unexplored regions.

Constructing the frontier requires multi-objective optimization. NSGA-II maintains candidate

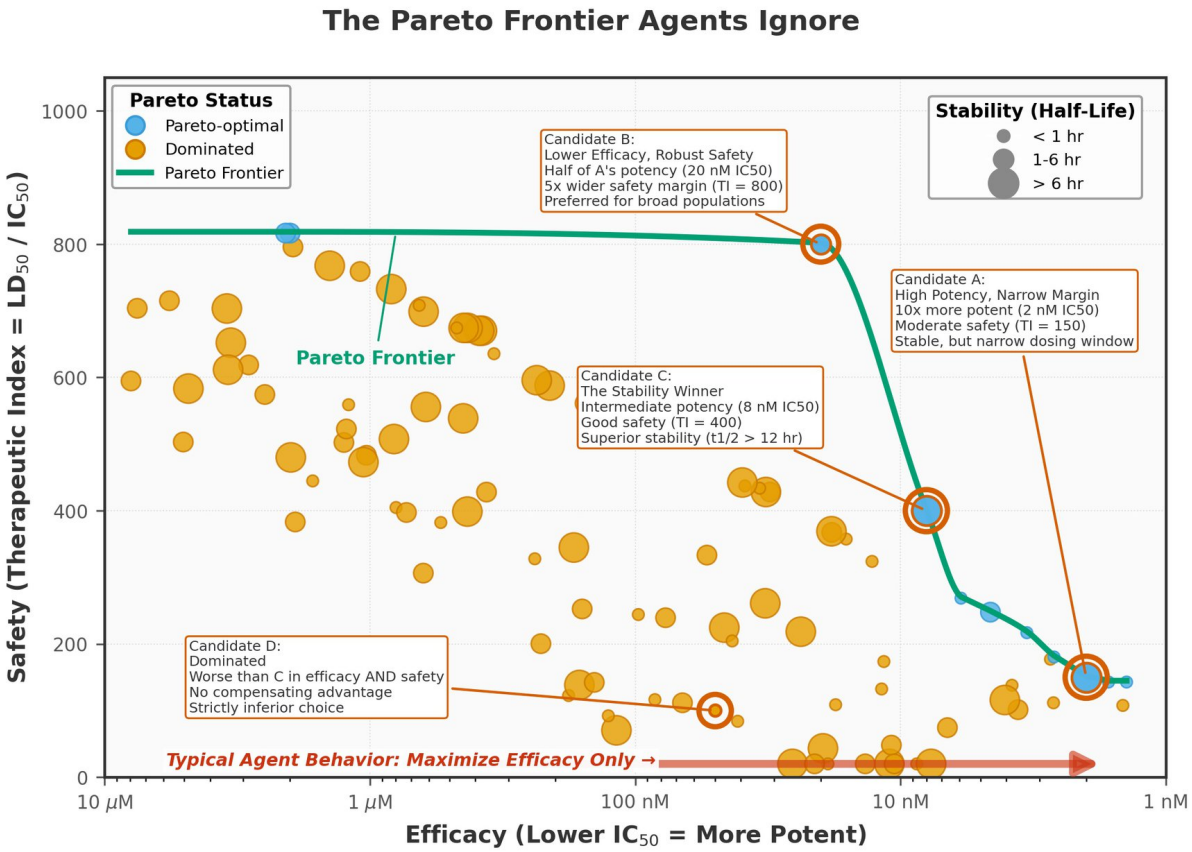

Figure 7: The Pareto Frontier Agents Ignore. Two-dimensional scatter plot of candidate compounds across efficacy (IC50) and safety (LD50 ratio), with stability (half-life) encoded by point size. The Pareto frontier curve identifies candidates where improving one objective requires degrading another. Annotations show real decision trade-offs: lower efficacy but much safer, highly effective but stability concerns. Current single-objective optimization (red arrow pointing to maximum efficacy) misses this complexity.

populations and selects non-dominated solutions. Multi-objective Bayesian optimization models objectives, selects candidates via acquisition functions balancing exploration and Pareto improvement, and updates with experimental results. These require tight integration between generative models, predictive models, and optimizers, which agents do not support.

Standalone multi-objective tools are advancing. PMMG [34] uses Pareto-guided Monte Carlo Tree Search to generate molecules satisfying seven simultaneous objectives with a 51.65% success rate, outperforming baselines by 2.5 times. CheapVS [35] enables human-guided preferential multi-objective Bayesian optimization, allowing chemists to express trade-off preferences via pairwise comparisons. These are effective optimization algorithms, but they are not integrated into agentic workflows providing uncertainty quantification, sensitivity analysis, stage-adaptive recommendations, and interactive decision support across iterative design cycles.

Constraints add complexity: synthesizability, solubility, permeability, absence of toxicophores. Constrained optimization identifies the Pareto frontier within feasible regions, which may be disjoint or conflicting. In peptide design, synthesis feasibility is often binding. Sequences with non-natural residues may be predicted optimal but are unavailable or prohibitively expensive. Regioselective cyclization can be unfeasible. Practitioners must balance optimization with synthetic pragmatism.

3.5.5 Incorporating Uncertainty and Risk

Predictions include uncertainty. IC50 equals 10 nM might have a 95

Bayesian optimization handles uncertainty via Gaussian processes providing means and variances. Acquisition functions balance exploitation and exploration (see §3.4 for detailed discussion). Visualizing uncertainty to guide exploration versus exploitation is a key capability gap.

Risk profiles vary by stage. Early discovery tolerates high-risk, high-reward candidates. Late-stage demands well-characterized properties and high confidence. Effective frameworks would adapt recommendations accordingly.

In peptide pipelines, early rounds prioritized diversity, middle rounds balanced exploration and exploitation, final rounds focused on de-risking. This requires dynamic acquisition functions and explicit risk management absent from agents.

Sensitivity analysis is missing. How robust are rankings to model error? If bioactivity predictions have 20

3.5.6 Capability Requirements Implied by Gap

An agent meeting this requirement would, given multi-endpoint assay data, construct a Pareto frontier with confidence intervals, provide interactive filtering by feasibility constraints, and generate sensitivity analyses quantifying ranking robustness to model error.

Closing this gap requires multi-objective Bayesian optimization with constraint handling, Pareto frontier visualization with interactive filtering, uncertainty quantification providing epistemic uncertainty in bioactivity prediction and dose-response confidence bands, and risk-aware candidate selection that adapts recommendations to development stage. The gap analysis identifies Pareto

frontier representation, uncertainty quantification, risk-aware decision support, and structured decision spaces as design requirements, replacing the current pattern of outputting single optima.

4 Design Requirements for Next-Generation Frameworks

The gaps identified in the preceding analysis reflect architectural assumptions that limit current frameworks. This section synthesizes five design requirements derived from the gap analysis and illustrates them with concrete use cases. The knowledge probing experiment (§2.5) confirms that foundation models already possess peptide expertise that current agents strand behind small-molecule-only tool chains. Closing the gaps requires building the integration pipeline, not retraining the models.

4.1 Requirements Derived from Gap Analysis

Table 6 summarizes the five design requirements derived from the gap analysis. Each requirement addresses the architectural limitations identified in the corresponding gap section, where detailed evidence and practitioner context are provided.

Table 6: Design Requirements Derived from Gap Analysis

Requirement (Gap)	**Core Capabilities**	**Key Primitives**
R1: Multi-paradigm orchestration (Gap 3)	ML training, RL, simulation, and optimization as first-class primitives	Declarative workflow graphs, checkpointing, human-in-the-loop decision points
R2: Modality-aware representations (Gap 1)	PLM fine-tuning, structural biology integration, peptide-specific prediction	ESM-2/ProtBERT pipelines, AlphaFold integration, aggregation prediction
R3: In vivo–in silico data fusion (Gap 2)	Temporal modeling, multi-modal fusion, causal inference	State-space models, bioinformatics pipelines, pathway enrichment
R4: Data-efficient learning (Gap 4)	Few-shot adaptation, active learning, transfer learning	Meta-learning, Bayesian uncertainty, public-to-private fine-tuning
R5: Risk-aware optimization (Gap 5)	Pareto optimization, uncertainty quantification, sensitivity analysis	Constraint handling, frontier visualization, stage-adaptive recommendations

4.2 Illustrative Use Cases

These requirements imply a shift from chat-first tooling to workflow-first systems. Agents would maintain state across iterations, log decisions, and make assumptions explicit so practitioners can

audit results and reproduce outcomes. This is a standard requirement in regulated environments and for teams that revisit decisions months later, often under new personnel or budget constraints.

To make these requirements concrete, we describe three representative use cases that current agents cannot handle but next-generation systems should support.

4.2.1 Use Case 1: Peptide Lead Optimization

Input: Fifty peptides with four assay endpoints, receptor structure, synthesis constraints.

Workflow: Fine-tune ESM-2, train multi-task regressor, use it as RL reward, filter by synthesis feasibility and stability, dock top candidates, cluster binding modes, present activity versus safety Pareto frontier with uncertainty.

Output: Ten synthesis candidates with rationales and confidence intervals.

Human decisions: Select among Pareto-optimal candidates based on strategic priorities and budget.

4.2.2 Use Case 2: In Vivo Efficacy Prediction

Input: In vitro assays and early in vivo markers for 20 peptides; predict day 28 outcomes.

Workflow: Normalize features, align temporal data with missing values, train regression models with stratified validation, identify early predictors and mechanistic drivers.

Output: Day 28 predictions with uncertainty and mechanistic hypotheses.

Human decisions: Choose peptides for full validation and whether to collect additional early markers.

4.2.3 Use Case 3: Multi-Endpoint Assay Analysis

Input: Four-endpoint data for 100 peptides.

Workflow: Normalize, cluster, validate stability, extract enriched sequence motifs, run pathway enrichment, visualize clusters.

Output: Distinct activity profile clusters with mechanistic hypotheses and selection recommendations.

Human decisions: Validate hypotheses and decide whether to focus on a single cluster or maintain diversity.

4.3 Infrastructure Considerations

Realizing these capabilities requires infrastructure beyond current agents. API standards would need to cover PLMs (embedding extraction, fine-tuning, uncertainty), structural biology tools (AlphaFold, docking, molecular dynamics), and bioinformatics pipelines (alignment, differential expression, pathway enrichment). Version-controlled datasets, model registries, and provenance tracking are standard requirements for reproducibility and auditability.

Table 7: Gap-to-Requirement Mapping: Priority Matrix for Next-Generation Agent Features

Feature	Impact	Difficulty	Who Needs It	Priority
Multi-paradigm orchestration	High	High	All	Critical
PLM fine-tuning pipelines	High	Medium	Biologics	Critical
Active learning loops	High	Medium	Small biotech	Critical
Pareto frontier visualization	High	Medium	All	High
Uncertainty quantification	High	Medium	All	High
In vivo data integration	High	High	All	High
Batch-mode workflows	Medium	Low	Small biotech	Medium
Transfer learning support	High	Medium	Small biotech	High
Constraint satisfaction	Medium	Medium	All	Medium
Workflow checkpointing	Low	Low	All	Medium

Organizational alignment matters. Systems targeting small biotech contexts would need to fit modest compute budgets, minimal storage, and lightweight deployment. Documentation targeting non-ML-expert biologists would lower adoption barriers. Integration with existing tools (GraphPad, FlowJo, ImageJ, R/Bioconductor) avoids workflow disruption.

5 Discussion

5.1 Scope and Limitations

The 15 task classes evaluated in this analysis are derived from peptide therapeutics development at a small biotech. While we expect many findings to generalize to other biologics (antibodies, nucleic acid therapeutics) and resource-constrained settings, applicability to these modalities requires further evaluation. The specific challenges of antibody CDR optimization, mRNA design, or gene therapy vector engineering may surface additional gaps not captured here. All capability assessments were performed by a single rater. While we provide detailed evidence for each assessment in Appendix B to support reproducibility, independent validation by additional raters would strengthen the findings.

Our three-level capability assessment (full support, partial support, not supported) captures more nuance than a binary scheme but may still oversimplify. The 0.5 weighting assigned to partial support is a modeling choice; alternative weightings could shift coverage scores. Some frameworks provide functionality that falls short of full workflow integration but represents meaningful progress. Where we identified partial support, we documented the specific capabilities and limitations in Appendix B, but a more granular scoring framework could reveal additional nuance.

The knowledge probing experiment has limitations. The low expert-automated scorer agreement (quadratic-weighted $\kappa = 0.22$) reflects systematic calibration differences between expert and LLM evaluation rather than random noise: the automated scorer assigned higher scores in 81% of disagreements, concentrated at the score 2/3 boundary. The Opus sensitivity check ($\kappa = 0.78$) demonstrates stability across judge models. However, the 50-pair design provides limited statistical power for detecting small effect sizes, and the null finding should be interpreted as absence of

evidence for a large gap rather than evidence of exact parity.

The field is rapidly evolving. Since beginning this analysis, systems like Agentomics [14] and ML-Agent [15] have demonstrated single-paradigm ML automation, and tools like PepTune [20] and PepMLM [21] have advanced peptide-specific generation. These developments address individual capabilities but do not yet close the integration gaps we identify.

5.2 Relation to Existing Work

This analysis complements two distinct contributions in the field. Seal et al. [9] provide a comprehensive architectural survey cataloging agent designs, tool integrations, and benchmarks. Their work maps what exists; ours evaluates what is missing when these systems confront diverse real-world requirements. The gaps we characterize are precisely the ones their survey catalogs but does not critique.

He et al. [3] demonstrate a deep single-organization deployment at AstraZeneca, establishing that agentic systems can deliver value in practice. However, their evaluation reflects one organizational context with extensive resources. Our analysis extends this by assessing generalizability across resource levels, data modalities, and therapeutic modalities.

Lakhan [8] advocates for broader agentic adoption in biopharma. The adoption they advocate requires the design requirements identified in this analysis; without addressing the identified gaps, adoption will remain limited to settings that match current framework assumptions.

5.3 Future Directions

Three directions would advance the field. First, empirical benchmarks reflecting diverse drug discovery contexts beyond molecular generation. Current benchmarks (MoleculeNet, GuacaMol, Therapeutic Data Commons) focus on small-molecule property prediction and generation. Benchmarks incorporating peptide design, in vivo modeling, multi-modal integration, and resource-constrained settings would enable more representative framework evaluation.

Second, open-source multi-paradigm orchestration frameworks. Integrating agent reasoning with existing workflow orchestration systems (§3.3), enabling inspection, diagnosis, and iterative improvement, would close the gap between workflow automation and intelligent orchestration.

Third, community-driven task class definitions spanning therapeutic modalities. Our 15 task classes reflect one practitioner's experience. Broader community input would extend coverage to antibodies, cell therapies, gene therapies, and other modalities, creating a shared evaluation framework for the field.

The appropriate design target is systems that augment practitioner judgment, not replace it. Drug discovery is too complex and context-dependent for full automation. Computational partners that handle preprocessing, training, tuning, and visualization while practitioners focus on hypotheses, mechanistic interpretation, and strategic trade-offs represent the appropriate design target. Partnership requires bidirectional communication: agents explain reasoning, expose assumptions, and quantify uncertainty; practitioners provide feedback and correct errors.

6 Conclusion

We evaluated six agentic frameworks against 15 drug discovery task classes and identified five critical capability gaps: small-molecule representation bias, absence of in vivo-in silico integration, limited computational paradigm support, misalignment with small-biotech constraints, and single-objective optimization assumptions. These gaps reflect architectural constraints that individual feature additions do not address: closing them requires changes to core framework design. Our knowledge probing experiment reinforces this conclusion: four frontier LLMs demonstrate competent peptide reasoning across all categories tested, yet this expertise remains stranded behind agent architectures that provide no peptide-aware tools or sequence-native workflows.

From the identified gaps, we derived five design requirements for next-generation frameworks: multi-paradigm orchestration supporting ML training, RL, and simulation as first-class primitives; modality-aware representations for peptides, proteins, and biologics; in vivo data integration with temporal modeling and multi-modal fusion; data-efficient learning through few-shot adaptation, active learning, and transfer learning; and multi-objective, risk-aware optimization with Pareto frontiers and uncertainty quantification.

The capability matrix and derived requirements provide a roadmap for framework developers and a benchmarking tool for practitioners evaluating agentic systems against their specific discovery contexts. Progress since 2023 has established what agentic systems can achieve. The next phase will determine whether these systems generalize beyond small-molecule workflows at well-resourced organizations to become core infrastructure for drug discovery broadly.

References

[1] Daniil A. Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. Autonomous chemical research with large language models. *Nature*, 624:570–578, 2023. doi: 10.1038/s41586-023-06792-0. Coscientist system for autonomous synthesis planning and execution.

[2] Andres M. Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D. White, and Philippe Schwaller. ChemCrow: Augmenting large-language models with chemistry tools. *Nature Machine Intelligence*, 6:525–535, 2024. doi: 10.1038/s42256-024-00832-8. arXiv:2304.05376 (preprint April 2023). GPT-4 orchestrating 18 chemistry tools.

[3] Jiazhen He et al. Democratising real-world drug discovery through agentic AI. *Drug Discovery Today*, 31(2):104605, January 2026. doi: 10.1016/j.drudis.2026.104605. PMID: 41548711. ChatInvent system deployed at AstraZeneca.

[4] Bowen Gao, Yanwen Huang, Yiqiao Liu, Wenxuan Xie, Wei-Ying Ma, Ya-Qin Zhang, and Yanyan Lan. PharmAgents: Building a virtual pharma with large language model agents. *arXiv preprint arXiv:2503.22164*, March 2025. Multi-agent system for target identification and compound selection.

[5] Eric Wang, Samuel Schmidgall, Paul F. Jaeger, Fan Zhang, Rory Pilgrim, Yossi Matias, Joelle Barral, David Fleet, and Shekoofeh Azizi. TxGemma: Efficient and agentic LLMs for therapeutics. *arXiv preprint arXiv:2504.06196*, April 2025. Google Research therapeutics-focused language model.

[6] Gleb V. Solovev, Alina B. Zhidkovskaya, Anastasia Orlova, et al. MADD: Multi-agent drug discovery orchestra. *arXiv preprint arXiv:2511.08217*, November 2025. Accepted to EMNLP 2025 Findings. Multi-agent collaboration for molecular design.

[7] Xiaochen Zheng, Alvaro Serra, Ilya Schneider Chernov, Maddalena Marchesi, Eunice Musvasva, and Tatyana Y. Doktorova. DiscoVerse: Multi-agent pharmaceutical co-scientist for traceable drug discovery and reverse translation. *arXiv preprint arXiv:2511.18259*, November 2025. Roche-affiliated researchers.

[8] Shaheen E. Lakhan. The agentic era: Why biopharma must embrace artificial intelligence that acts, not just informs. *Cureus*, 17(5), May 2025. doi: 10.7759/cureus.83390. PMID: 40322603. Editorial on agentic AI in pharmaceutical industry.

[9] Srijit Seal et al. AI agents in drug discovery. *arXiv preprint arXiv:2510.27130*, October 2025. Comprehensive 45-page survey of agentic AI systems.

[10] Zeming Lin, Halil Akin, et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023. doi: 10.1126/science.ade2574. PMID: 36927031.

[11] Ahmed Elnaggar et al. ProtTrans: Toward understanding the language of life through self-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10): 7112–7127, 2022. doi: 10.1109/TPAMI.2021.3095381. Online July 2021, print October 2022.

[12] Odhran O'Donoghue, Aleksandar Shtedritski, John Ginger, Ralph Abboud, Ali Ghareeb, and Samuel G. Rodriques. BioPlanner: Automatic evaluation of LLMs on protocol planning in biology. In *Proceedings of EMNLP 2023*, pages 2676–2694, 2023. doi: 10.18653/v1/2023.emnlp-main.162. arXiv:2310.10632.

[13] Botao Yu, Frazier N. Baker, Ziru Chen, Garrett Herb, Boyu Gou, Daniel Adu-Ampratwum, Xia Ning, and Huan Sun. ChemToolAgent: The impact of tools on language agents for chemistry problem solving. *arXiv preprint arXiv:2411.07228*, November 2024. Accepted to NAACL 2025 Findings.

[14] Vlastimil Martinek, Andrea Gariboldi, Dimosthenis Tzimotoudis, Mark Galea, Elissavet Zacharopoulou, Aitor Alberdi Escudero, Edward Blake, David Čechák, Luke Cassar, Alessandro Balestrucci, and Panagiotis Alexiou. Agentomics: An agentic system that autonomously develops novel state-of-the-art solutions for biomedical machine learning tasks. *bioRxiv*, 2026.

doi: 10.64898/2026.01.27.702049. Autonomous ML experimentation achieving SOTA on 11/20 biomedical benchmarks.

[15] Zexi Liu, Jingyi Chai, Xinyu Zhu, Shuo Tang, Rui Ye, Bo Zhang, Lei Bai, and Siheng Chen. ML-Agent: Reinforcing LLM agents for autonomous machine learning engineering. *arXiv preprint arXiv:2505.23723*, 2025. doi: 10.48550/arXiv.2505.23723. Online RL training of LLM agents for autonomous ML engineering.

[16] Petra Schneider et al. Rethinking drug design in the artificial intelligence era. *Nature Reviews Drug Discovery*, 19:353–364, 2020. doi: 10.1038/s41573-019-0050-3. PMID: 31801986.

[17] Jessica Vamathevan, Dominic Clark, Paul Czodrowski, Ian Dunham, Edgardo Ferran, George Lee, Bin Li, Anant Madabhushi, Parantu Shah, Michaela Spitzer, and Shanrong Zhao. Applications of machine learning in drug discovery and development. *Nature Reviews Drug Discovery*, 18(6):463–477, 2019. doi: 10.1038/s41573-019-0024-5. PMID: 30976107.

[18] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

[19] Ali Madani et al. Large language models generate functional protein sequences across diverse families. *Nature Biotechnology*, 41:1099–1106, 2023. doi: 10.1038/s41587-022-01618-2.

[20] Sophia Tang, Yinuo Zhang, and Pranam Chatterjee. PepTune: De novo generation of therapeutic peptides with multi-objective-guided discrete diffusion. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*. PMLR, 2025. doi: 10.48550/arXiv.2412.17780. Masked discrete diffusion model with Monte Carlo Tree Guidance for multi-objective peptide generation.

[21] Leo Tianlai Chen, Zachary Quinn, Madeleine Dumas, Christina Peng, Lauren Hong, Moises Lopez-Gonzalez, Alexander Mestre, Rio Watson, Sophia Vincoff, Lin Zhao, Jianli Wu, Audrey Stavrand, Mayumi Schaepers-Cheu, Tian Zi Wang, Divya Srijay, Connor Monticello, Pranay Vure, Rishab Pulugurta, Sarah Pertsemlidis, Kseniia Kholina, Shrey Goel, Matthew P. DeLisa, Jen-Tsan Ashley Chi, Ray Truant, Hector C. Aguilar, and Pranam Chatterjee. Target sequence-conditioned design of peptide binders using masked language modeling. *Nature Biotechnology*, 2025. doi: 10.1038/s41587-025-02761-2. PMID: 40804173. ESM-2-based peptide binder design conditioned on target protein sequences.

[22] Noelia Ferruz, Steffen Schmidt, and Birte Höcker. ProtGPT2 is a deep unsupervised language model for protein design. *Nature Communications*, 13:4348, 2022. doi: 10.1038/s41467-022-32007-7.

[23] John Jumper et al. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596: 583–589, 2021. doi: 10.1038/s41586-021-03819-2. PMID: 34265844.

[24] Alexander Mathis et al. DeepLabCut: markerless pose estimation of user-defined body parts with deep learning. *Nature Neuroscience*, 21:1281–1289, 2018. doi: 10.1038/s41593-018-0209-y. PMID: 30127430.

[25] Minoru Kanehisa, Miho Furumichi, Yoko Sato, Masayuki Kawashima, and Mari Ishiguro-Watanabe. KEGG for taxonomy-based analysis of pathways and genomes. *Nucleic Acids Research*, 51(D1):D587–D592, 2023. doi: 10.1093/nar/gkac963. PMID: 36300620.

[26] Aravind Subramanian et al. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences*, 102(43):15545–15550, 2005. doi: 10.1073/pnas.0506580102. PMID: 16199517.

[27] Pengwei Sui, Michelle M. Li, Shanghua Gao, Wanxiang Shen, Valentina Giunchiglia, Andrew Shen, Yepeng Huang, Zhenglun Kong, and Marinka Zitnik. Medea: An omics AI agent for therapeutic discovery. *bioRxiv*, 2026. doi: 10.64898/2026.01.16.696667. Omics AI agent with 20 tools for transcriptomics, protein networks, and pathway analysis.

[28] Joseph A. DiMasi, Henry G. Grabowski, and Ronald W. Hansen. Innovation in the pharmaceutical industry: new estimates of R&D costs. *Journal of Health Economics*, 47:20–33, 2016. doi: 10.1016/j.jhealeco.2016.01.012. PMID: 26928437.

[29] Jonathan M. Stokes et al. A deep learning approach to antibiotic discovery. *Cell*, 180(4): 688–702.e13, 2020. doi: 10.1016/j.cell.2020.01.021. PMID: 32084340.

[30] Francesca Grisoni et al. Combining generative artificial intelligence and on-chip synthesis for de novo drug design. *Science Advances*, 7(24), 2021. doi: 10.1126/sciadv.abg3338. PMID: 34117066.

[31] Ryan K. Tan, Yang Liu, and Lei Xie. Reinforcement learning for systems pharmacology-oriented and personalized drug design. *Expert Opinion on Drug Discovery*, 17(8):849–863, 2022. doi: 10.1080/17460441.2022.2072288. PMID: 35510835.

[32] Daniel Reker, Petra Schneider, Gisbert Schneider, and J. B. Brown. Active learning for computational chemogenomics. *Future Medicinal Chemistry*, 9(4):381–402, 2017. doi: 10.4155/fmc-2016-0197. PMID: 28263088.

[33] G. Richard Bickerton, Gaia V. Paolini, Jérémy Besnard, Sorel Muresan, and Andrew L. Hopkins. Quantifying the chemical beauty of drugs. *Nature Chemistry*, 4:90–98, 2012. doi: 10.1038/nchem.1243. PMID: 22270643.

[34] Yifei Liu, Yiheng Zhu, Jike Wang, Renling Hu, Chao Shen, Wanglin Qu, Gaoang Wang, Qun Su, Yuchen Zhu, Yu Kang, Peichen Pan, Chang-Yu Hsieh, and Tingjun Hou. A multi-objective

molecular generation method based on Pareto algorithm and Monte Carlo tree search. *Advanced Science*, 12(20):2410640, 2025. doi: 10.1002/advs.202410640. Pareto MCTS achieving 51.65% success on 7 simultaneous objectives.

[35] Tai Dang, Long-Hung Pham, Sang T. Truong, Ari Glenn, Wendy Nguyen, Edward A. Pham, Jeffrey S. Glenn, Sanmi Koyejo, and Thang Luong. Preferential multi-objective Bayesian optimization for drug discovery. *arXiv preprint arXiv:2503.16841*, 2025. doi: 10.48550/arXiv.2503.16841. Human-guided preferential multi-objective Bayesian optimization for virtual screening.

Appendices

A Task Class Descriptions

This appendix provides full descriptions of the 15 task classes used in our evaluation, including representative inputs, outputs, computational requirements, and evaluation criteria. Task classes are derived from real-world drug discovery workflows spanning peptide therapeutics, in vivo pharmacology, and computational biology.

T1. ML bioactivity prediction (multi-endpoint regression). Train supervised models predicting multiple biological endpoints (e.g., proliferation, migration, secretion, toxicity) from peptide features.

Inputs: Peptide sequences or descriptors (50–500 compounds), multi-endpoint assay measurements.

Outputs: Predictive models with per-endpoint performance metrics and uncertainty estimates.

Requirements: Feature extraction (sequence-based or PLM embeddings), stratified cross-validation, hyperparameter tuning, multi-task or multi-output regression, calibration.

Evaluation: Per-endpoint R^2, RMSE, calibration error, cross-validated confidence intervals.

T2. Generative peptide design (PLM fine-tuning). Fine-tune protein language models on therapeutic sequences for conditional de novo peptide generation.

Inputs: Training set of therapeutic peptide sequences (50–500), target property constraints.

Outputs: Novel peptide sequences satisfying property constraints with diversity metrics.

Requirements: PLM fine-tuning (ProtGPT2, ProtBERT), transfer learning, sampling with temperature control, property filtering, novelty and diversity assessment.

Evaluation: Validity, novelty, diversity, predicted property distributions, KL divergence from training set.

T3. Peptide-receptor binding site analysis and clustering. Analyze peptide-receptor interactions through flexible docking, cluster by binding region, and correlate with bioactivity.

Inputs: Peptide sequences, receptor structure (experimental or AlphaFold-predicted), bioactivity data.

Outputs: Binding mode clusters, per-cluster activity profiles, key interacting residues.

Requirements: Conformational sampling, flexible docking, interaction fingerprinting, clustering, statistical correlation with bioactivity.

Evaluation: Cluster separation (silhouette score), activity-cluster correlation (ANOVA), docking score convergence.

T4. In vivo recovery modeling (longitudinal clinical scores). Model longitudinal clinical outcomes from animal studies to identify temporal efficacy signatures and classify treatment responses.

Inputs: Time-series clinical scores (e.g., motor coordination assessments at days 1, 3, 7, 14,

28), treatment groups, covariates.

Outputs: Temporal efficacy profiles, responder/non-responder classification, predictive early markers.

Requirements: Mixed-effects models, time-series classification, missing data imputation, temporal feature engineering, bootstrap confidence intervals.

Evaluation: Classification accuracy (responder vs non-responder), prediction of late outcomes from early markers (R^2, AUC).

T5. Peptide-enzyme interaction modeling for stability optimization. Predict protease cleavage sites and design modifications that improve serum stability while preserving bioactivity.

Inputs: Peptide sequences, known half-life measurements, protease specificity data.

Outputs: Predicted cleavage sites, suggested stabilizing modifications, stability-activity trade-off analysis.

Requirements: Sequence-based cleavage prediction across protease families, molecular modeling of modifications (D-amino acids, cyclization, PEGylation), multi-objective balancing of stability and affinity.

Evaluation: Cleavage site prediction accuracy, correlation of predicted vs measured half-life, activity retention after modification.

T6. Protein language model-based receptor type prediction. Classify peptide sequences by receptor type using PLM embeddings and supervised learning.

Inputs: Peptide sequences with receptor type labels (50–200 labeled examples), pre-trained PLM (ESM-2).

Outputs: Multi-class classifier with per-class probabilities and uncertainty estimates.

Requirements: PLM embedding extraction, layer selection, classifier training (logistic regression, gradient-boosted trees), cross-validation, calibration.

Evaluation: Multi-class AUC-ROC, precision-recall per class, calibration curves, confusion matrix.

T7. Monte Carlo optimization for peptide landscape exploration. Explore peptide sequence space using stochastic optimization to balance exploitation of known active regions with exploration of novel regions.

Inputs: Initial peptide set, fitness function (bioactivity predictor), sequence constraints.

Outputs: Optimized peptide set with diversity metrics, landscape topology characterization.

Requirements: Metropolis-Hastings sampling, acceptance criteria tuning, fitness landscape estimation, convergence diagnostics, diversity maintenance.

Evaluation: Best fitness achieved, sequence diversity (edit distance distribution), convergence rate, landscape coverage.

T8. RNA sequencing and single-cell transcriptomics analysis. Process bulk or single-cell RNA-seq data to identify differentially expressed genes, cell-type compositions, and treatment-responsive pathways.

Inputs: FASTQ files or count matrices, experimental design (treatment vs control, time points).

Outputs: Differentially expressed gene lists, pathway enrichment results, cell-type annotations (scRNA-seq), pseudotime trajectories.

Requirements: Read alignment (STAR, HISAT2), quantification, normalization (DESeq2, edgeR), dimensionality reduction (UMAP, t-SNE), clustering, trajectory inference, pathway enrichment (GSEA, KEGG, GO).

Evaluation: Alignment rates, library complexity, differential expression FDR control, biological coherence of enriched pathways.

T9. Digital image processing for tissue quantification. Extract quantitative morphometric features from tissue imaging (histology, radiographic imaging) for treatment efficacy assessment.

Inputs: Tissue images, region-of-interest annotations, treatment group labels.

Outputs: Quantified tissue features (area, density, gap measurements), statistical comparisons across treatment groups.

Requirements: Image preprocessing (contrast normalization, artifact removal), segmentation (thresholding or learned), feature extraction, automated ROI detection, statistical testing.

Evaluation: Segmentation accuracy (Dice coefficient), inter-rater agreement, effect size and statistical significance of treatment differences.

T10. Immune response profiling (pathway analysis). Characterize immune modulation by analyzing pathway activation, upstream regulators, and inflammatory scoring from transcriptomic or proteomic data.

Inputs: Expression data (genes or proteins), treatment conditions, reference pathway databases.

Outputs: Activated/suppressed pathways, upstream regulator predictions, inflammatory response scores.

Requirements: Pathway enrichment (GSEA, IPA-style analysis), upstream regulator inference, network construction, scoring metrics for inflammatory vs anti-inflammatory balance.

Evaluation: Pathway enrichment FDR, consistency across methods, biological plausibility of upstream regulators.

T11. Functional annotation and pathway enrichment. Annotate gene or protein lists with functional categories (GO terms, KEGG pathways) and identify statistically enriched biological processes.

Inputs: Gene/protein lists from differential expression or clustering, background gene set, annotation databases.

Outputs: Enriched GO terms and KEGG pathways with p-values, gene network visualizations.

Requirements: GO/KEGG annotation retrieval, hypergeometric or Fisher enrichment testing, multiple testing correction, network construction (STRING, Cytoscape).

Evaluation: Enrichment significance after correction, semantic coherence of enriched terms, overlap with known biology.

T12. Computer vision for behavioral phenotyping. Apply pose estimation and tracking to animal behavioral videos to quantify treatment effects on social behavior, motor function, or anxiety-like phenotypes.

Inputs: Behavioral videos (multiple animals per session), body part definitions, treatment group assignments.

Outputs: Tracked keypoint trajectories, derived behavioral metrics, statistical comparisons.

Requirements: Pose estimation model training (DeepLabCut), tracking validation, time-series feature engineering, behavioral metric computation, statistical testing with repeated measures.

Evaluation: Tracking accuracy (pixel error), behavioral metric reliability (test-retest), effect size and significance of treatment differences.

T13. Predictive modeling bridging in vivo and in vitro endpoints. Develop composite efficacy metrics that correlate in vitro bioactivity with long-term in vivo outcomes to enable early candidate prioritization.

Inputs: In vitro assay data (multiple endpoints), early and late in vivo measurements, compound identifiers.

Outputs: Predictive efficacy model, feature importance rankings, predicted long-term outcomes with confidence intervals.

Requirements: Multi-source feature extraction, temporal alignment, imputation, regression or classification modeling, stratified cross-validation on temporally ordered splits.

Evaluation: Prediction accuracy (R^2, AUC for classification), early marker predictive power, model stability across cross-validation folds.

T14. Reinforcement learning for de novo peptide generation. Optimize peptide sequences using RL with multi-objective reward functions combining bioactivity, stability, and diversity.

Inputs: Pre-trained generative model (ProtGPT2), reward model(s) for bioactivity and stability, diversity constraints.

Outputs: Optimized peptide sequences with reward decomposition, diversity statistics, training curves.

Requirements: Policy optimization (PPO, GRPO variants), reward model training, KL regularization against reference policy, curriculum learning (staged reward introduction), diversity penalties.

Evaluation: Mean reward, reward component distributions, sequence diversity, KL divergence from reference, mode collapse indicators.

T15. Safety and toxicology modeling (dose-response, multi-objective trade-offs). Model dose-response relationships accounting for repeated measures and time-dependent effects, and navigate safety-efficacy trade-offs for candidate selection.

Inputs: Dose-response data (multiple doses, time points, endpoints), efficacy measurements, compound properties.

Outputs: Dose-response curves with confidence bands, therapeutic window estimates, Pareto frontier of safety vs efficacy.

Requirements: Generalized linear mixed models, repeated measures analysis, LD50/ED50 estimation with confidence intervals, multi-objective optimization, Pareto frontier construction, species translation modeling.
Evaluation: Model fit (AIC/BIC), confidence interval coverage, Pareto frontier hypervolume, robustness of therapeutic window estimates.

B Detailed Capability Matrix

This appendix extends the capability matrix (Table 2) with per-dimension assessments for each framework. Each framework is evaluated across the five dimensions defined in §2: molecular representation (D1), computational paradigm support (D2), data modality integration (D3), resource assumptions (D4), and optimization framework (D5). Assessments are based on published documentation, available source code, and demonstrated use cases as of early 2026.

B.1 ChemCrow

ChemCrow [2] orchestrates 18 chemistry tools via GPT-4, including RDKit for molecular property calculation, PubChem for compound lookup, and reaction prediction APIs for retrosynthesis.

Table 8: ChemCrow: Per-Dimension Assessment

Dimension	Rating	Evidence and Notes
D1: Molecular repr.	○	Supports SMILES and molecular fingerprints via RDKit. No peptide sequence representations, PLM embeddings, or conformational sampling.
D2: Comp. paradigm	×	Tool invocation only (stateless API calls). No ML training, RL, or simulation orchestration.
D3: Data modality	×	Text and SMILES inputs only. No imaging, time-series, transcriptomics, or behavioral data support.
D4: Resource assumptions	○	Lightweight tool calls; does not require HPC. However, no few-shot, active learning, or transfer learning support.
D5: Optimization	×	Single-objective property optimization. No Pareto frontiers, constraint satisfaction, or uncertainty quantification.

Partial support details. Task 3 (peptide-receptor binding): ChemCrow includes docking tools designed for small-molecule rigid-body screening; these do not handle peptide conformational flexibility or induced-fit binding. Task 15 (safety/toxicology): includes toxicophore detection and basic ADMET prediction for small molecules but lacks dose-response modeling, mixed-effects analysis, or multi-objective trade-off reasoning.

B.2 Coscientist

Coscientist [1] autonomously plans and executes chemical syntheses by interfacing with laboratory automation, web search, and code execution.

Coverage: 0/15 task classes. Coscientist's strength lies in closed-loop synthesis execution, a capability orthogonal to the 15 task classes evaluated here, which focus on computational modeling rather than laboratory automation.

Table 9: Coscientist: Per-Dimension Assessment

Dimension	Rating	Evidence and Notes
D1: Molecular repr.	×	Designed for small-molecule organic synthesis. No peptide or protein representations.
D2: Comp. paradigm	×	Code execution capability exists but is used for synthesis protocol generation, not ML training or RL.
D3: Data modality	×	Text and structured chemical data only. No in vivo, imaging, or transcriptomic data handling.
D4: Resource assumptions	○	Operates with standard compute but assumes access to automated laboratory equipment (Emerald Cloud Lab).
D5: Optimization	×	Optimizes synthesis yield as a single objective. No multi-objective framework.

B.3 PharmAgents

PharmAgents [4] deploys specialized LLM agents for target identification, compound screening, and interaction prediction, coordinated through a multi-agent architecture with knowledge graph integration.

Table 10: PharmAgents: Per-Dimension Assessment

Dimension	Rating	Evidence and Notes
D1: Molecular repr.	○	Supports SMILES and molecular graphs. Knowledge graph includes protein targets but no PLM-based peptide representations.
D2: Comp. paradigm	○	Invokes pre-trained predictors for property estimation. No model training, fine-tuning, or RL support.
D3: Data modality	○	Structured databases and knowledge graphs. Pathway-level information available but not computational enrichment pipelines.
D4: Resource assumptions	×	Assumes large compound libraries and knowledge graph infrastructure typical of large pharma.
D5: Optimization	×	Rank-based compound selection. No Pareto optimization or uncertainty quantification.

Partial support details. Task 1 (ML bioactivity): can invoke pre-trained models for single-endpoint prediction but cannot train multi-endpoint regressors on proprietary data. Tasks 10–11 (immune profiling, functional annotation): knowledge graph queries return pathway-level information but do not perform computational enrichment analysis (GSEA, hypergeometric testing). Task 15 (safety): includes ADMET prediction modules for small molecules.

B.4 ChatInvent

ChatInvent [3] was deployed at AstraZeneca over 13 months for literature-driven molecular design and synthesis planning, with access to institutional databases and computational infrastructure.

Table 11: ChatInvent: Per-Dimension Assessment

Dimension	Rating	Evidence and Notes
D1: Molecular repr.	○	Supports SMILES-based molecular design and literature-based structure analysis. No peptide-specific representations or PLMs.
D2: Comp. paradigm	×	LLM reasoning over literature and molecular design tools. No ML training, RL, or simulation.
D3: Data modality	○	Literature text, molecular structures, and internal databases. No in vivo data, imaging, or transcriptomics.
D4: Resource assumptions	×	Designed for AstraZeneca-scale resources: institutional literature access, HPC, large proprietary databases, specialized teams.
D5: Optimization	×	Literature-guided hypothesis generation. No formal multi-objective optimization or uncertainty quantification.

Partial support details. Tasks 10–11 (immune profiling, functional annotation): literature synthesis can surface pathway-level knowledge but does not perform computational analysis. Task 15 (safety): can retrieve safety-related literature and flag known liabilities from text; no quantitative dose-response modeling.

B.5 MADD

MADD [6] coordinates multiple LLM agents for molecular design, property prediction, and docking in a multi-agent collaboration framework.

Table 12: MADD: Per-Dimension Assessment

Dimension	Rating	Evidence and Notes
D1: Molecular repr.	○	Supports SMILES-based molecular generation and property prediction. No peptide or protein representations.
D2: Comp. paradigm	○	Multi-agent coordination for design-predict-dock cycles. Agents invoke tools but do not train models or run RL.
D3: Data modality	×	Molecular structures and predicted properties only. No in vivo, imaging, or transcriptomic data.
D4: Resource assumptions	○	Moderate compute requirements for docking. Does not assume pharma-scale databases but lacks few-shot or active learning.
D5: Optimization	×	Iterative design refinement toward single objectives. No Pareto optimization or constraint satisfaction.

Partial support details. Task 1 (ML bioactivity): prediction agents can estimate properties but from pre-trained models, not trained on proprietary data. Task 3 (peptide-receptor binding): includes docking workflows designed for small molecules. Task 15 (safety): property prediction

agents can flag liabilities but without multi-objective reasoning.

B.6 DiscoVerse

DiscoVerse [7] provides multi-agent pharmaceutical workflow automation with traceable reasoning, developed by Roche-affiliated researchers.

Table 13: DiscoVerse: Per-Dimension Assessment

Dimension	Rating	Evidence and Notes
D1: Molecular repr.	○	Supports small-molecule representations and reverse translation workflows. Limited protein or peptide support.
D2: Comp. paradigm	○	Workflow automation with traceability. Orchestrates existing tools but does not support model training or RL.
D3: Data modality	○	Integrates multiple data sources including clinical and preclinical information. Limited in vivo temporal modeling or imaging.
D4: Resource assumptions	×	Designed for large pharma infrastructure and data availability.
D5: Optimization	×	Workflow-driven decision support. No formal multi-objective optimization framework.

Partial support details. Task 1 (ML bioactivity): workflow automation may include prediction steps using pre-trained models. Tasks 10–11 (immune profiling, functional annotation): reverse translation capability includes pathway-level reasoning from clinical observations. Task 15 (safety): clinical-preclinical integration provides safety context but lacks quantitative multi-objective optimization.

B.7 Cross-Framework Summary

Table 14 summarizes per-dimension coverage across all six frameworks. The sparsest dimensions, computational paradigm support (D2) and optimization framework (D5), reflect architectural limitations inherent to LLM-as-orchestrator designs: cannot support gradient-based training or multi-objective reasoning without fundamental changes.

Table 14: Cross-Framework Dimension Coverage Summary. Fraction of frameworks providing full (✓) or partial (○) support per evaluation dimension.

Dimension	Full	Partial	Primary Limitation
D1: Molecular representation	0/6	5/6	No PLM or peptide-native support
D2: Computational paradigm	0/6	3/6	No ML training, RL, or simulation
D3: Data modality	0/6	3/6	No in vivo, imaging, or omics pipelines
D4: Resource assumptions	0/6	3/6	No few-shot, active learning, or transfer
D5: Optimization framework	0/6	0/6	No multi-objective or uncertainty support

C Knowledge Probing Experiment: Supplementary Details

This appendix provides additional detail for the LLM knowledge probing experiment described in §2.5.

C.1 Question Design

We constructed 50 matched question pairs (100 questions total) spanning five pharmaceutical knowledge categories: SAR reasoning, ADMET and pharmacokinetic properties, generative design strategies, optimization approaches, and assay interpretation (10 pairs per category). Each pair tested the same cognitive skill in two modality contexts: one small-molecule question and one peptide question. Questions were balanced for difficulty and format across domains. The full question set is available in the supplementary materials.

C.2 Scoring Rubric

Responses were scored on a four-point ordinal scale:

Table 15: Knowledge Probing Scoring Rubric

Score	Criteria
0	Wrong or hallucinated: factually incorrect, fabricated data, or nonsensical
1	Partially correct: some correct elements but missing key nuance or significant errors
2	Correct: accurate and reasonably complete
3	Expert-level: correct with domain-expert nuance and specific quantitative details

C.3 Per-Model Results

Table 16 presents the full per-model statistical breakdown.

Table 16: Knowledge Probing: Per-Model Results. SM and PEP columns show mean scores on a 0–3 scale. Gap = SM minus PEP (negative indicates peptide advantage). Wilcoxon signed-rank tests are one-sided (H_1: SM > PEP), Bonferroni-corrected ($\alpha = 0.0125$). Rank-biserial r is the matched-pairs effect size.

Model	SM Mean	PEP Mean	Gap	p	p (Bonf.)	r
Kimi K2.5	2.54	2.60	−0.06	0.680	1.0	−0.095
DeepSeek V3.2	2.30	2.44	−0.14	0.880	1.0	−0.232
Qwen 3 Next 80B	2.22	2.28	−0.06	0.619	1.0	−0.057
Gemini 3 Flash	2.28	2.48	−0.20	0.901	1.0	−0.265

C.4 Expert Validation Protocol

A domain expert independently scored a stratified 20% subset ($N = 80$) under a blind protocol. The subset was drawn by stratified random sampling (2 responses per cell in a 4 models × 5

categories × 2 domains grid; 40 cells total; random seed = 42). Model identity and domain labels were stripped, and row order was randomized before presentation to the expert.

C.5 Inter-Rater Agreement

Table 17 shows the confusion matrix between expert and automated (Claude Sonnet 4.5) scores.

Table 17: Confusion Matrix: Expert vs Automated Scoring. Rows are expert scores, columns are Claude Sonnet 4.5 scores. The dominant disagreement pattern is expert score 2 vs Claude score 3 (34 of 80 cases, 42.5%), indicating systematic calibration differences at the correct/expert-level boundary.

	Claude 0	**Claude 1**	**Claude 2**	**Claude 3**
Expert 0	0	0	0	0
Expert 1	1	2	8	4
Expert 2	2	7	13	34
Expert 3	0	0	1	8

Quadratic-weighted Cohen's $\kappa = 0.22$, reflecting systematic calibration differences rather than random noise. Of 57 disagreements (71.25%), the automated scorer assigned a higher score in 46 cases (80.7%), concentrated at the score 2/3 boundary (expert gives 2, Claude gives 3 in 34 of 56 expert-score-2 cases, 60.7%). Disagreements were balanced across domains (peptide: 27, small-molecule: 30).

An additional sensitivity check rescored a 40-response stratified subset using Claude Opus. Opus-Sonnet agreement was substantially higher: quadratic-weighted $\kappa = 0.78$, exact agreement 80%, within-one agreement 100%, mean signed shift +0.15.

C.6 Category Heatmap

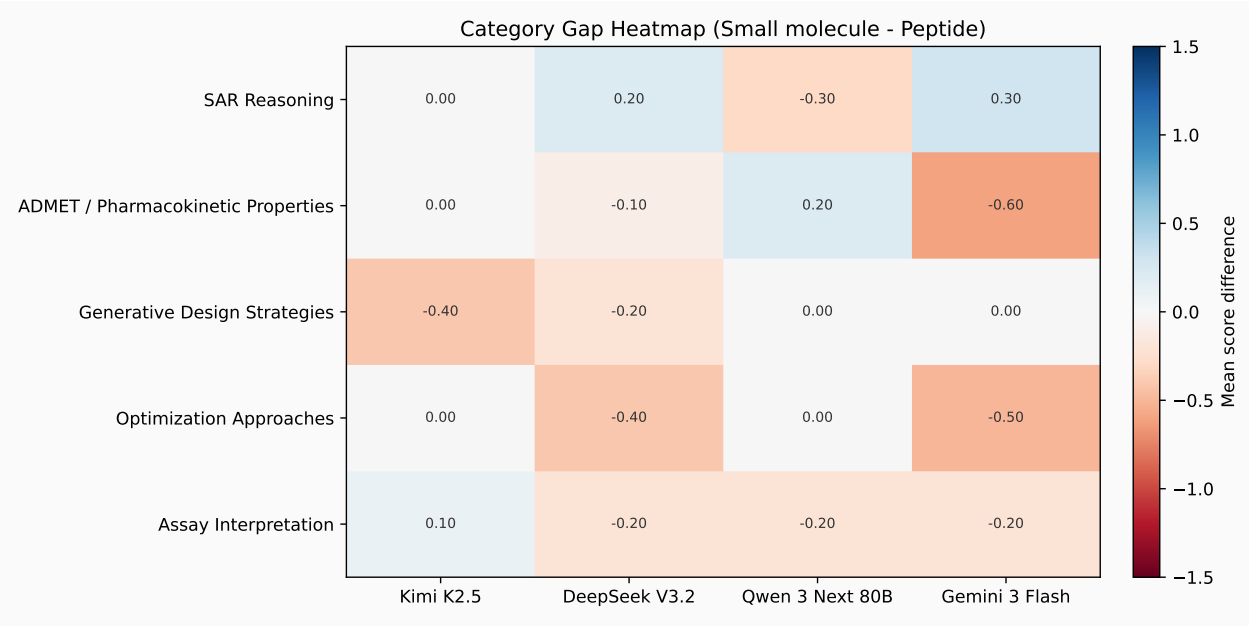

Figure 8: Knowledge Probing: Per-Category Score Heatmap. Mean scores for each model across five pharmaceutical knowledge categories, split by domain (small-molecule vs peptide). Color intensity reflects mean score on a 0–3 scale.