

A Systematic Gap Analysis of Agentic AI Frameworks for Drug Discovery Beyond Small Molecules

Edward Wijaya
StemRIM, Inc.
wijaya@stemrim.com

Abstract

Agentic AI systems have advanced drug discovery automation, with frameworks such as ChatInvent, Coscientist, and ChemCrow demonstrating autonomous synthesis planning, literature mining, and molecular design. However, no systematic evaluation of these frameworks against real-world drug discovery requirements beyond small-molecule, target-based workflows has been conducted. We evaluate six agentic AI frameworks against 15 task classes spanning peptide discovery, in vivo modeling, and resource-constrained settings across five evaluation dimensions: molecular representation coverage, computational paradigm support, data modality integration, resource assumptions, and optimization framework. Our analysis reveals five critical capability gaps: small-molecule bias excluding protein language models and peptide-specific prediction; absent in vivo to in silico bridges for longitudinal, multi-modal animal data; limited computational paradigm support excluding ML training, reinforcement learning, and multi-paradigm coordination; resource assumptions mismatched to small biotech realities; and single-objective optimization ignoring multi-objective trade-offs in safety, efficacy, and stability. From these gaps, we derive design requirements for next-generation frameworks: multi-paradigm orchestration, modality-aware architectures, in vivo integration, data-efficient learning, and Pareto-based optimization with uncertainty quantification. We provide a capability matrix, concrete use cases, and infrastructure considerations to guide framework development toward computational partners that augment practitioner judgment under realistic data, modality, and resource constraints.

1 Introduction: The Promise and the Blind Spots

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 frontier, 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. The enthusiasm is warranted, but it is also narrowly scoped. 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. This is the practical gap practitioners encounter in day-to-day work.

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 blind spots 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. It is integration, not individual capability, that defines the gap.

1.2 Scope and Motivation

However, these systems reveal systematic blind spots 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 traumatic brain injury models, efficacy manifests as motor coordination improvements at day 7, reduced neuroinflammation at day 14, and neurogenesis at day 28. 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 essential, not optional. 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.

The Agent Reality Gap in Drug Discovery

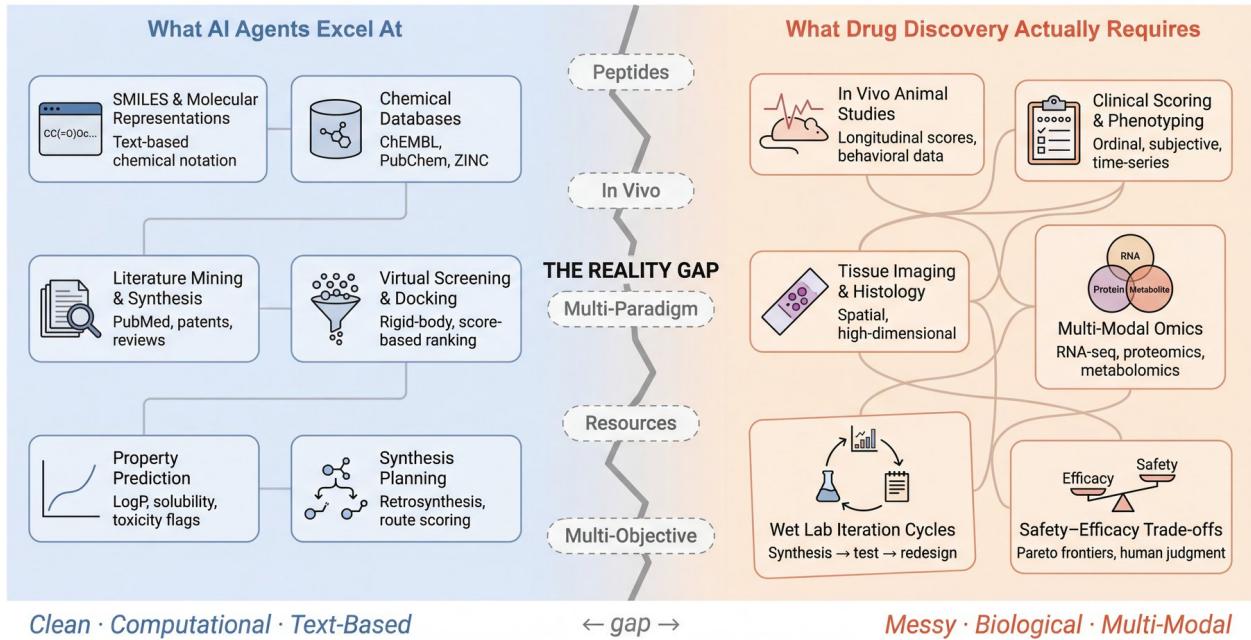


Figure 1: The Agent Reality Gap in Drug Discovery. Left panel shows computational workflows where current agents excel: small molecule representations (SMILES strings), databases, literature mining, and virtual screening. Right panel depicts the messy reality of drug discovery: multi-modal biological 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.

Practitioners end up doing this reasoning manually, which slows iteration and increases decision risk.

This paper draws 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 AI practitioner. Each revealed architectural assumptions that do not generalize and where pragmatic workarounds were required to get results on real timelines. For each gap, we characterize the specific architectural limitation, provide evidence from practitioner workflows, and propose concrete design requirements for next-generation systems. This paper presents a systematic gap analysis. We evaluate six agentic AI 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

This section describes the systematic approach used to evaluate agentic AI frameworks against drug discovery task requirements.

2.1 Agent Framework Selection

We selected six agentic AI 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).

Table 1: Agentic AI 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 for regenerative medicine:

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 span the full discovery pipeline from target identification through preclinical validation, representing computational requirements that extend well beyond the small-molecule, target-based workflows that current frameworks primarily support.

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.

2.4 Analysis Approach

For each framework-task pair, we performed a binary capability assessment (supported or not supported) based on published documentation, source code availability, and demonstrated use cases. We computed a coverage score as the 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.

This approach has limitations: binary assessment may oversimplify nuanced partial support, and the rapidly evolving nature of the field means new frameworks may address some identified gaps. We discuss these limitations further in §5.

3 Results: Five Capability Gaps

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 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 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. Developing peptides for traumatic brain injury, 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.

3.1.3 Protein Language Models vs Molecular Fingerprints

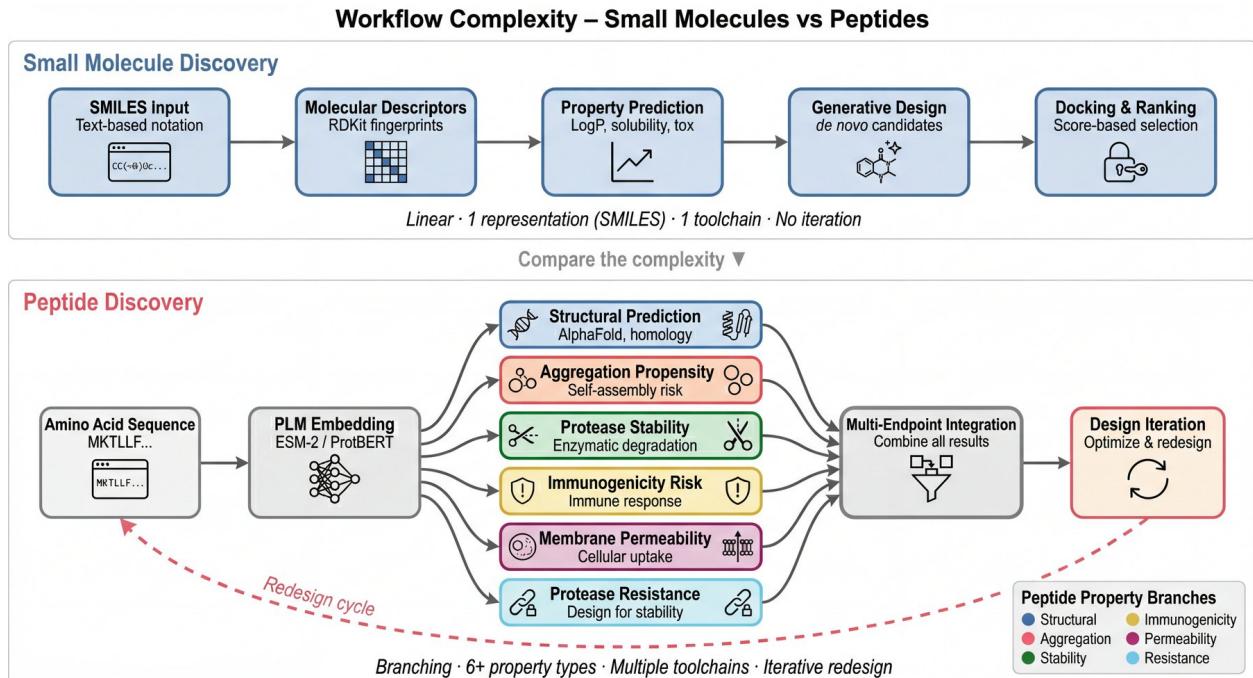


Figure 2: Workflow Complexity: Small Molecules vs Peptides. Top: Small molecule workflow follows a linear path from SMILES representation through RDKit property calculation to docking. Bottom: Peptide workflow branches into multiple parallel analysis streams including structural prediction, aggregation propensity, stability, immunogenicity, membrane permeability, and protease resistance, requiring integration of diverse computational tools and protein language models.

Protein language models define peptide design. ProtBERT [11], ESM-2 [10], and ProGen [12] 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 [13] 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 [14] 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. Agents must expose these uncertainties clearly so biologists can prioritize synthesis and testing. Without that, the workflow reverts to manual triage and ad hoc heuristics. No current system supports this autonomously. LLM orchestrators assume models are black-box inference APIs. There is no fine-tuning support, dataset version control, or hyperparameter search. Agents retrieve embeddings but cannot adjust attention heads or train task-specific classifiers on proprietary data.

Generative modeling extends this gap. ProtGPT2 [15] 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.

3.1.4 Derived Requirements

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. Agents should fine-tune ProtBERT on 200 sequences and return calibrated classifiers with uncertainty estimates.

Second, structural biology integration. AlphaFold [16] structure prediction is central to peptide design. Flexible docking requires conformational sampling. Molecular dynamics provides stability and kinetics insights. These tools must integrate into multi-step workflows and feed back into sequence optimization, not sit as 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. Active learning maximizes information gain by prioritizing uncertain predictions. This requires uncertainty quantification, acquisition functions, and feedback loops updating models across multiple rounds.

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 must be revisited.

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 plans syntheses [1], ChemCrow screens compound libraries, ChatInvent mines literature. 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 28-day traumatic brain 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 via beam walking, 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

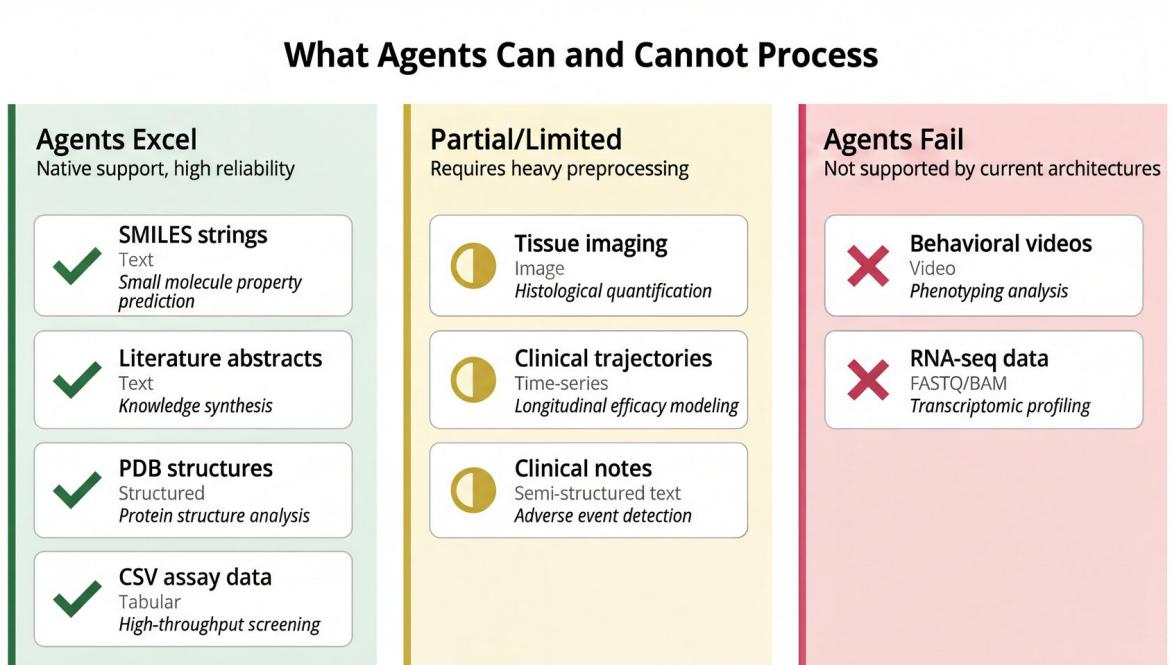


Figure 3: What Agents Can and Cannot Process. Matrix showing data types on the vertical axis and agent accessibility on the horizontal axis. Green checkmarks indicate data types current agents handle well (text, SMILES, PDB files, CSV data, literature). Red X marks denote data types agents struggle with (behavioral videos, clinical score trajectories, tissue imaging, multi-modal transcriptomics, longitudinal measurements with dropout). This visualization reveals the systematic exclusion of in vivo data modalities from current agent architectures.

The core challenge is that in vivo data does not fit the tidy CSV format that machine learning pipelines expect (Table 2). Behavioral scores are ordinal and subject to inter-rater variability.

Behavioral phenotyping via DeepLabCut [17] tracks animal poses in videos, generating time-

Table 2: 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

series keypoint coordinates. Computing behavioral metrics (inter-animal distance, contact time, grooming) 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 [18] or Gene Ontology. Upstream regulator analysis infers transcription factors driving changes. The FASTQ-to-hypothesis pipeline needs bioinformatics tools (STAR, HISAT2, DESeq2, edgeR, GSEA [19]) that agents do not integrate. Recent systems like Medea [20] 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 (D-amino acids, cyclization) may reduce affinity or increase aggregation.

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 essential. 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 Derived Requirements

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 dynamics 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 [21].

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 [22], and ChemToolAgent evaluates chemistry tool integration [23]. 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. ChemCrow calls RDKit and PubChem [2]. ChatInvent supports molecular design and synthesis planning [3]. 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 [24]: supervised learning (bioactivity prediction [25], toxicity modeling), unsupervised learning (chemical space clustering), generative modeling (de novo design [26]), reinforcement learning (multi-objective optimization [27]), simulation (binding kinetics), and optimization (experimental design).

Table 3: Agent Capability Matrix

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 [28] 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 [29] 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 Derived Requirements

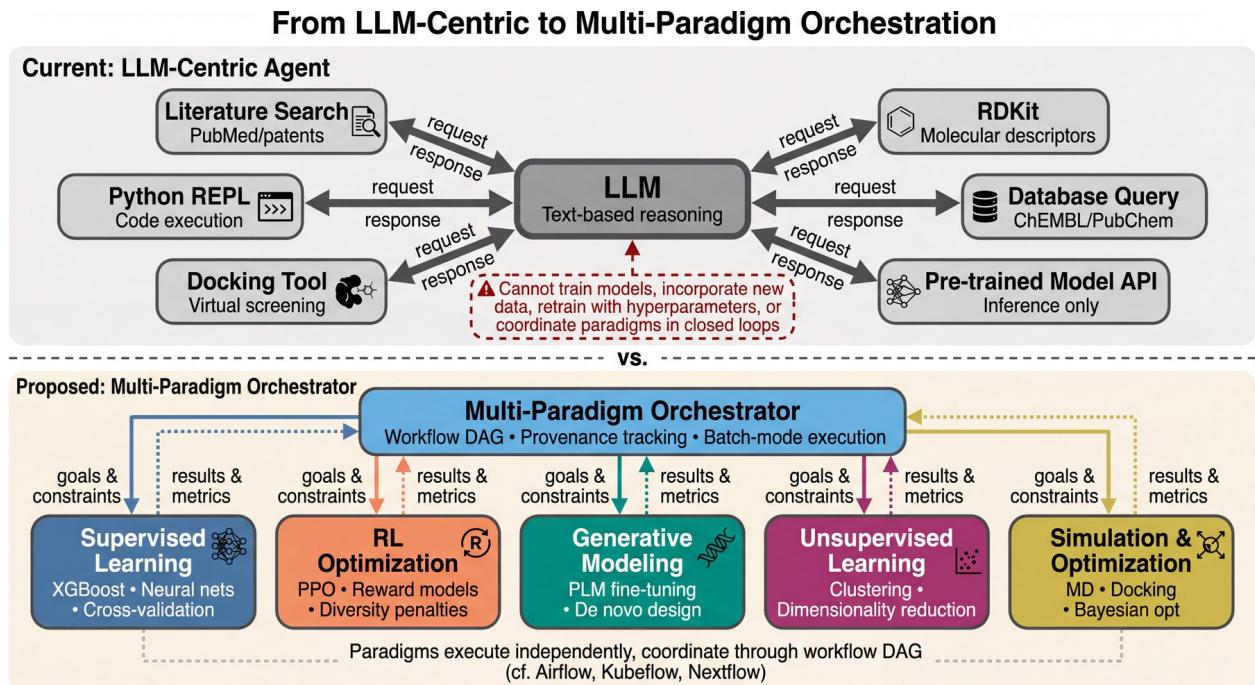


Figure 4: From LLM-Centric to Multi-Paradigm Orchestration. Top: Current LLM-centric architecture where a central language model orchestrates all tools through API calls. Bottom: Proposed multi-paradigm architecture where an orchestrator coordinates fundamentally different computational paradigms (ML 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 (XG-

Boost, 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), check-pointing (cached intermediate results), and branching (automatic hyperparameter search if validation fails), with clear provenance for every artifact. Workflow orchestration exists: Apache Airflow, Kubeflow, Nextflow provide task graphs, dependency resolution, resource allocation, and check-pointing. Missing is agent reasoning integration: inspecting results, diagnosing failures, proposing modifications, learning from executions (deprioritizing architectures that overfit).

Interaction should be batch-mode, not chat. Bottlenecks are orchestrating end-to-end analyses, not formulating queries. Small biotechs managing dozens of projects need automation: ”Analyze eight RNA-seq samples, perform differential expression and pathway enrichment, generate report.” Agents intervene only for human decisions (conflicting pathway interpretations).

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. The field needs systems coordinating paradigms in integrated workflows, batch-mode execution, explicit decision points, and version control for reproducibility, not LLM-to-LLM delegation.

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.

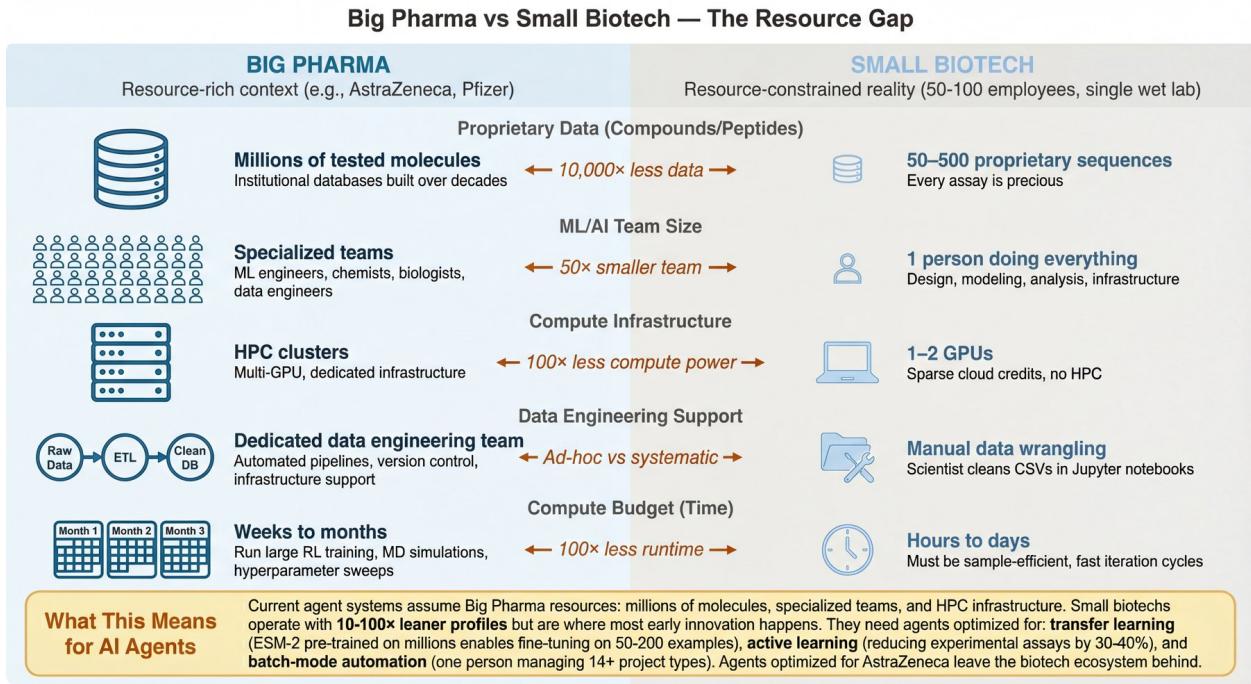


Figure 5: Big Pharma vs Small Biotech: The Resource Gap. Comparative visualization of computational resources, team size, and data availability. Left: Large pharmaceutical companies with 10,000+ compounds, 50-person ML teams, multi-GPU clusters, dedicated data engineers, and months of compute budget. 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.

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.

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 [30]; 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 demands tools for generalists handling infrastructure complexity (dependencies, environments, memory, debugging). 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 60-70% accuracy on peptide-receptor binding classification with only 20 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), agents should adjust parameters (reduce threads) or flag issues without losing progress. Checkpointing, fault tolerance, and version control enable reproducibility.

Parallelization is essential. Given 100 peptide docking jobs, agents should 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 Derived Requirements

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 pre-training with private fine-tuning, and batch-mode orchestration enabling unsupervised overnight analyses. Resource-aware agents should 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) has zero coverage across all six frameworks.

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 $0.6 \times \text{bioactivity} + 0.4 \times \text{drug-likeness}$ " [31]. 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. A peptide that degrades rapidly in serum has no value. Stability modifications (D-amino acids, non-natural residues, cyclization) improve half-life but can reduce affinity, increase aggregation, or complicate synthesis. 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.

The Pareto Frontier Agents Ignore

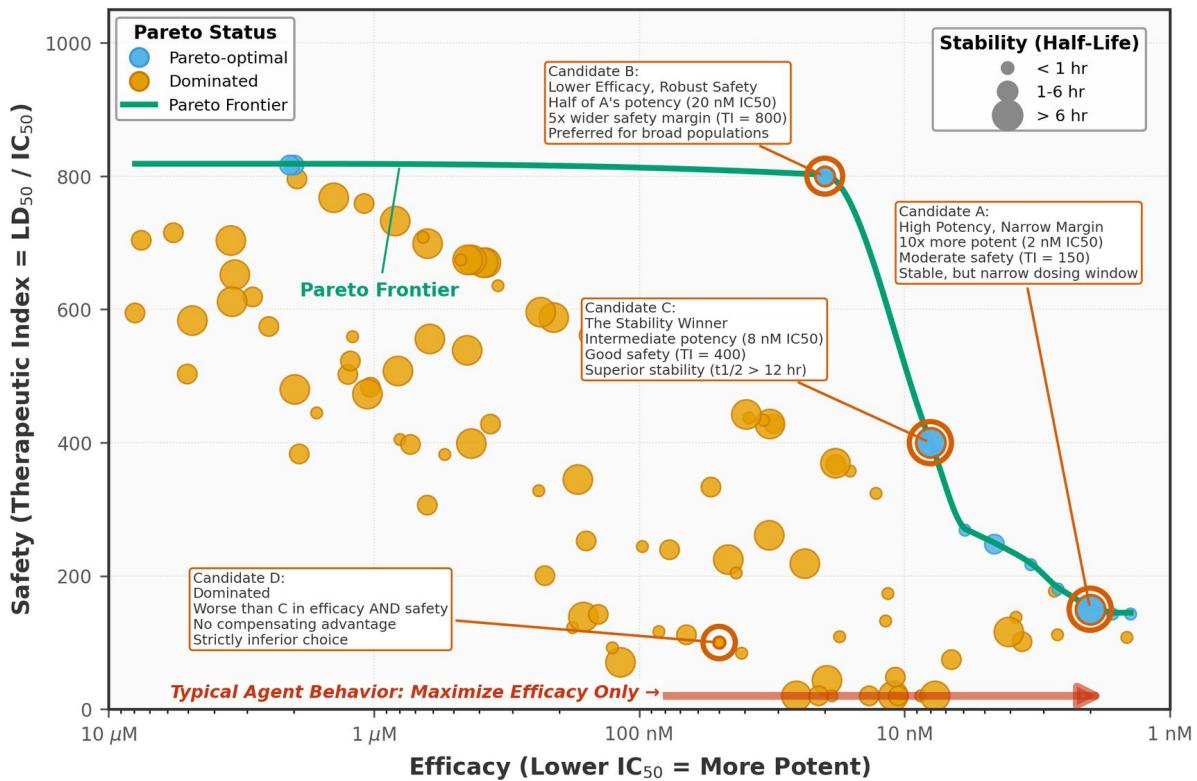


Figure 6: The Pareto Frontier Agents Ignore. Two-dimensional scatter plot of candidate compounds across efficacy (IC₅₀) and safety (LD₅₀ ratio), with stability (half-life) encoded by point size. The Pareto frontier curve identifies candidates where improving one objective requires degrading another. 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.

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 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 [32] 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 [33] 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 (expected improvement, upper confidence bound) balance exploitation and exploration. Over rounds, uncertainty shrinks in explored regions and stays high elsewhere. Agents should visualize uncertainty to guide exploration versus exploitation.

Risk profiles vary by stage. Early discovery tolerates high-risk, high-reward candidates. Late-stage demands well-characterized properties and high confidence. Agents should 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 Derived Requirements

Closing this gap requires multi-objective Bayesian optimization with constraint handling, Pareto frontier visualization with interactive filtering, uncertainty quantification providing epistemic un-

certainty in bioactivity prediction and dose-response confidence bands, and risk-aware candidate selection that adapts recommendations to development stage. Agents must represent Pareto frontiers, quantify uncertainty, support risk-aware decisions, and structure decision spaces, not output single optima.

4 Design Requirements for Next-Generation Frameworks

The gaps identified in the preceding analysis reflect architectural assumptions that must be revisited. This section synthesizes five design requirements derived from the gap analysis and illustrates them with concrete use cases.

4.1 Requirements Derived from Gap Analysis

4.1.1 Requirement 1: Multi-Paradigm Orchestration

Agents must support ML training, RL, simulation, and optimization as first-class primitives. Practitioners should specify workflows declaratively and receive an executable workflow graph that supports parallelization, checkpointing, and branching when validation fails. Human-in-the-loop decision points must be explicit when models trade off accuracy, calibration, or interpretability. Workflow orchestration systems (Airflow, Kubeflow, Nextflow) already provide these abstractions; what is missing is tight integration with agent reasoning and iterative improvement.

4.1.2 Requirement 2: Modality-Aware Representations

Agents must provide first-class support for peptides, proteins, and other modalities, not just small molecules. This requires PLM fine-tuning pipelines, structural biology integration (AlphaFold, flexible docking, molecular dynamics), and peptide-specific property prediction (aggregation, protease resistance, immunogenicity). Modality awareness must flow through data loaders, feature extractors, generative models, and evaluation metrics. Tool selection should be contextual: peptides default to ESM-2 embeddings, protein structures to structural alignment, not SMILES-based similarity.

4.1.3 Requirement 3: In Vivo-In Silico Data Fusion

Agents must support temporal modeling for longitudinal efficacy and safety data, multi-modal fusion (behavior, imaging, transcriptomics), and causal inference for mechanistic hypotheses. This requires bioinformatics pipelines (RNA-seq alignment, differential expression, pathway enrichment), computer vision, and statistical models for dose-response and mixed effects. Predictive efficacy modeling should align heterogeneous timepoints, handle missing data, and validate on temporally ordered splits. Mechanistic inference should integrate KEGG, GO, and Reactome to translate results into hypotheses and validation experiments.

4.1.4 Requirement 4: Data-Efficient Learning

Agents must optimize for few-shot adaptation, active learning, and knowledge transfer across related tasks. This requires meta-learning, Bayesian uncertainty quantification, and transfer pipelines combining public pre-training (UniProt, PDB, ChEMBL) with private fine-tuning. Agents should recommend strategies based on dataset size: deep nets for 500 examples, simpler models for 50, few-shot for 10. Active learning loops should select candidates via acquisition functions, update models with experimental feedback, and signal when marginal information gain is low.

4.1.5 Requirement 5: Multi-Objective, Risk-Aware Optimization

Agents must support Pareto optimization with constraints, uncertainty quantification, sensitivity analysis, and interactive trade-off visualization. Results should be Pareto frontiers with confidence intervals, constraint status, and robustness scores. Practitioners should filter by feasibility, adjust objective weights, and highlight low-uncertainty candidates. Risk awareness should adapt recommendations to stage: early exploration versus late-stage de-risking.

4.2 Illustrative Use Cases

These requirements imply a shift from chat-first tooling to workflow-first systems. Agents should maintain state across iterations, log decisions, and make assumptions explicit so practitioners can audit results and reproduce outcomes. This is essential for 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

Table 4: 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

Realizing these capabilities requires infrastructure beyond current agents. API standards must 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 essential for reproducibility and auditability.

Organizational alignment matters. Systems must fit small biotech budgets with modest compute, minimal storage, and lightweight deployment. Documentation should target non-ML-expert biologists. 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.

Our binary capability assessment (supported or not supported) may oversimplify cases of nuanced partial support. Some frameworks provide limited functionality for certain task classes that falls short of full workflow integration but represents meaningful progress. Where we identified partial support, we noted this in the analysis, but a more granular scoring framework could reveal additional nuance.

The field is rapidly evolving. Since beginning this analysis, systems like Agentomics [28] and ML-Agent [29] have demonstrated single-paradigm ML automation, and tools like PepTune [13] and PepMLM [14] 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 agentic AI adoption in biopharma. The adoption they advocate requires the engineering solutions we propose; 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. Workflow orchestration systems (Airflow, Kubeflow, Nextflow) provide task graphs, dependency resolution, and resource allocation. Integrating agent reasoning with these systems, 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.

These systems should 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 AI 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 are structural, not incremental: addressing them requires architectural changes, not feature additions.

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] Jiazen 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.adc2574. 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] 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.

- [13] 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.
- [14] 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.
- [15] 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.
- [16] 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.
- [17] 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.
- [18] 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.
- [19] 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.
- [20] 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.
- [21] 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.

- [22] 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.
- [23] 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.
- [24] 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.
- [25] 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.
- [26] 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.
- [27] 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.
- [28] 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.
- [29] 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.
- [30] 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.
- [31] G. Richard Bickerton, Gaia V. Paolini, Jérémie 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.

- [32] 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.
- [33] 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.