InUS Transactions 2(2), 2026, 59-80
https://doi.org/10.23380/inustrans.2026.2.2.1

InUS
https://acoms.accesson.kr/inustrans

Survey Paper

Idea Generation in AI for Science: A Survey from Hypothesis Generation to Closed-Loop Discovery

Jin-Xia Huang[1,2*†], **Woojin Lee**[1†], **Yoonkyu Woo**[1,2]

[1] Electronics and Telecommunications Research Institute(ETRI), Daejeon, Korea ({hgh, writerwoody, yoon303}@etri.re.kr)

[2] ETRI School, University of Science and Technology(UST), Daejeon, Korea

[*] Corresponding author.

[†] These authors contributed equally to this work.

Abstract: Artificial intelligence increasingly proposes hypotheses, candidate structures, and experimental interventions within scientific discovery workflows. This survey examines *idea generation* in AI for Science as the computational function that produces new candidates (textual hypotheses, symbolic equations, molecular graphs, experimental designs, and simulation-grounded proposals) under domain constraints, and as a systems-level capability whose value depends on how those candidates are validated, operationalized, and integrated into closed-loop discovery pipelines. We present a two-axis taxonomy organized by (i) generation modality and (ii) embodiment level, and use it to analyze seven core paradigms: knowledge-graph and literature-based discovery, symbolic regression and causal discovery, Bayesian experimental design, simulation-grounded scientific machine learning, representation learning and foundation models, diffusion-based and diversity-aware generation, and LLM-based hypothesis generation. We synthesize three recurring integration patterns, namely generator–verifier pipelines, latent-space optimization, and orchestrated closed-loop execution, and consolidate a six-dimensional evaluation framework covering efficiency, reliability, interpretability and idea quality, causal validity, discovery novelty, and mechanistic depth. The survey highlights the engineering interfaces that determine whether generative capability translates into dependable, auditable discovery pipelines.

Keywords: Idea generation; AI for Science; scientific discovery; closed-loop experimentation; scientific foundation models; evaluation framework.

1. Introduction

Artificial intelligence (AI) is increasingly integrated into the scientific discovery workflow, both as a tool for accelerating analysis and as a component that can *propose* candidate hypotheses, experimental interventions, and structured representations [1–3]. In many domains, the practical role of AI has expanded from surrogate modeling and prediction to systems that generate scientific objects—e.g., equations, molecules, experimental sequences, or literature-grounded conjectures—that then require validation through simulation, experiment, or human review [4–6]. This shift motivates a unified survey perspective centered on *idea generation* as an operational function in AI for Science.

Scientific discovery can be viewed as an iterative loop: hypothesis formulation, experimental design, data acquisition, analysis, and theory refinement [7]. Early machine learning contributed primarily to predictive modeling within this loop, improving property prediction, classification, and surrogate simulation without changing how hypotheses were formulated [1]. Contemporary systems increasingly intervene earlier in the loop by proposing candidate explanations, design objects, and experiments [6, 8]. To analyze this landscape consistently across model classes and domains, we focus on the mechanisms that generate new candidates under explicit scientific constraints.

1.1. Motivation and Problem Framing

We define *idea generation in AI for Science* as:

> Computational mechanisms that propose new hypotheses, experimental interventions, structured representations, or candidate designs that extend scientific knowledge under domain-specific constraints.

Fig. 1 illustrates how contemporary idea generation systems intervene earlier in the scientific discovery loop than traditional ML approaches. This definition emphasizes *proposal under constraint*. It is useful to distinguish idea generation from closely related functions that often co-occur in discovery pipelines:

- **Prediction:** estimating outputs for fixed inputs under a learned mapping.

Manuscript received 9 March 2026; revised 8 May 2026; accepted 26 May 2026; and published 30 June 2026. This work was supported by Institute of Information and Communications Technology Planning and Evaluation(IITP) grant funded by the Korea government(MSIT) (RS-2019-II190004, Development of semi-supervised learning language intelligence technology and Korean tutoring service for foreigners). The material in this paper was not presented at any conference. This paper was recommended for publication in revised form by Associate Editor Daxiong Ji.

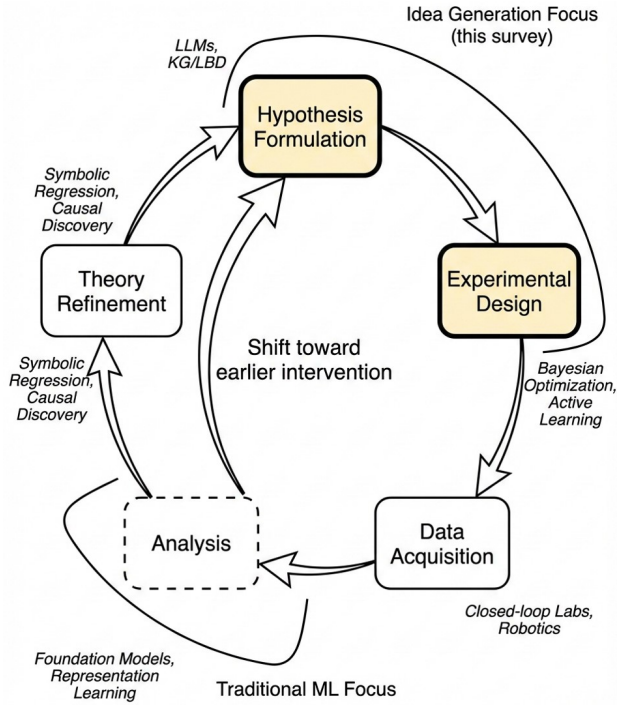

Fig. 1. The scientific discovery loop and AI intervention points. Traditional ML systems primarily target the analysis stage (dashed box); contemporary idea generation systems intervene earlier, at hypothesis formulation and experimental design (solid boxes).

- **Optimization:** selecting candidates from a predefined search space using an objective.
- **Control:** regulating system dynamics toward specified objectives through feedback.

Prediction operates within a fixed hypothesis space, optimization explores a bounded candidate set, and control stabilizes trajectories within known dynamics [1, 9]. Idea generation, in contrast, introduces *new structured candidates*, such as a previously unknown equation, a molecular scaffold, or an experimental protocol, whose scientific validity must then be evaluated, refined, or falsified. Critically, *unconditional* generative sampling (e.g., drawing a random molecule from a diffusion model without a target property or hypothesis) is not itself idea generation under our definition; such sampling qualifies only when conditioned on a prior hypothesis, an identified knowledge gap, an explicit physical or structural constraint, or a validation target.

Concrete instantiations span diverse paradigms—from symbolic regression and diffusion-based molecular design through literature-based discovery, physics-informed learning, and LLM-driven hypothesis generation [2, 10–12]—and are analyzed in Sections 3–5.

1.2. Scope and Positioning

Several surveys review AI for Science from the perspectives of domain applications [1], LLM capabilities [2, 10], or laboratory automation [12, 13]. These works typically organize the field by application area or model family. What remains underdeveloped is a cross-paradigm synthesis that uses *idea generation* as the organizing thread—connecting language-based reasoning, structured inference, experimental design, generative modeling, and closed-loop automation under a single framework.

This survey addresses that gap. We organize heterogeneous approaches by two questions: *how* does the system generate candidates, and *how* are those candidates evaluated? The scope covers systems that (i) generate candidate hypotheses, designs, or structured representations and (ii) evaluate them against explicit scientific constraints such as physical laws, causal consistency, or experimental feasibility. The paradigms covered span knowledge-graph and literature-based discovery, symbolic regression and causal discovery, Bayesian experimental design, simulation-grounded SciML, representation learning and foundation models, diffusion-based and diversity-aware generation, LLM-based hypothesis generation, and closed-loop laboratory integration (the full taxonomy is developed in Section 3).

We do *not* aim to exhaustively survey every application subfield (e.g., chemistry-only or materials-only), nor do we center routine predictive analytics that do not alter the candidate hypothesis space. Prediction and optimization remain essential modules in discovery pipelines [1], but our emphasis is on mechanisms that expand, restructure, or navigate scientific possibility spaces through structured proposal and constraint-aware evaluation.

Contributions

- We present a two-axis taxonomy organized by (i) generation modality and (ii) embodiment level, enabling cross-paradigm comparison across language-based, symbolic, generative, and simulation-grounded approaches.
- We synthesize recurring integration patterns—generator–verifier pipelines, latent-space optimization, and orchestrated closed-loop execution—and analyze constraint enforcement and failure modes across paradigms.
- We analyze closed-loop and self-driving laboratory architectures from a systems-engineering perspective, distinguishing operational from epistemic autonomy [12].

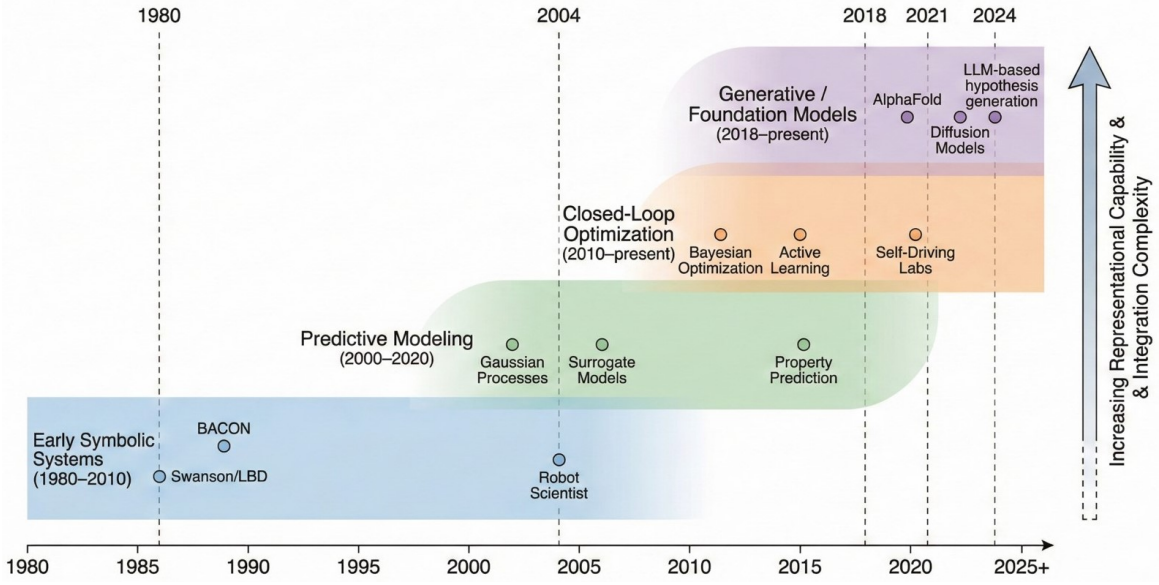

Fig. 2. Evolution of AI for scientific discovery. Overlapping bands indicate progressive integration rather than discrete paradigm shifts. Key milestone systems are annotated along the timeline.

- We consolidate a six-dimensional evaluation framework—efficiency, reliability, interpretability and idea quality, causal validity, discovery novelty, and mechanistic depth—and connect it to emerging benchmarking efforts [12, 13].

2. Historical Evolution of AI for Scientific Discovery

AI adoption in scientific discovery has progressed through advances in algorithms, data infrastructure, and experimental automation [1, 3]. Although boundaries are fluid, four interacting phases can be distinguished (Table 1 and Fig. 2):

1. **Early symbolic discovery systems**—rule-based programs that generated scientific hypotheses via structured search;
2. **Predictive modeling**—ML models serving as fast surrogates for expensive simulations;
3. **Closed-loop optimization**—predictive surrogates embedded within sequential decision systems such as Bayesian optimization;
4. **Generative and foundation models**—large-scale architectures that directly synthesize structured scientific objects (molecules, equations, hypotheses).

These phases overlap and build on each other rather than replacing one another abruptly.

2.1. Early Symbolic Discovery Systems (1980–2010)

Computational idea generation in science predates modern deep learning. Early systems treated scientific discovery as structured search over symbolic hypothesis spaces. Programs such as BACON and related rule-based frameworks formalized heuristic induction of physical laws and empirical regularities [7]. Literature-based discovery further demonstrated that new hypotheses could be generated by linking disjoint knowledge fragments in text corpora, anticipating modern graph-based and retrieval-based methods [14, 15]. In the chemical domain, early database-driven tools automated pharmacophore hypothesis generation [28].

These systems operated over explicit symbolic representations and established key design principles that persist: (i) hypothesis generation as search over structured spaces, (ii) constraint-guided pruning, and (iii) separation of proposal and empirical validation [7]. By the 2000s, the Robot Scientist paradigm automated full cycles of hypothesis formulation, experiment execution, and model revision [17, 29], while neural-symbolic synthesis planning in chemistry combined learned components with symbolic search [30, 31], and transformer-based models enabled uncertainty-calibrated reaction prediction [32]. Together, these developments trace a clear arc from symbolic search through neural-symbolic integration to embodied experimentation, situating contemporary generative systems as continuations of a longer trajectory.

Table 1. Historical phases of AI for scientific discovery: dominant method, operational role, and limitation.

Phase	Core Method	Role of AI	Limitation
Early symbolic systems	Rule-based search; heuristic induction [7, 14–17]	Explicit hypothesis generation and law discovery	Limited scalability; brittle representations
Predictive modeling	Supervised learning; surrogate models [1, 18, 19]	Fast approximation of expensive simulations; screening and inference	Fixed hypothesis/design space; limited extrapolation and mechanistic interpretability
Closed-loop optimization	Bayesian optimization and acquisition rules [9, 20–22]	Adaptive selection of experiments; sequential decision-making with uncertainty	Search constrained to predefined variables/parameterizations; limited representation invention
Generative / foundation models	Diffusion; energy-based models; large-scale representation learning; LLMs [2, 10, 23–27]	Synthesis of structured candidates (molecules, sequences, equations, hypotheses); transferable representations	Reliability and validation burden; constraint satisfaction and interpretability remain nontrivial

2.2. Predictive Modeling Phase

The predictive modeling phase established ML as a surrogate for scientific computation. Models approximated mappings from structured inputs to measurable outputs, enabling rapid screening and reduced simulation cost across domains [1]. Applications included property prediction, reaction outcome estimation, and emulation of numerical solvers [33, 34]. The dominant architecture was open-loop: data collection → model training → candidate ranking.

Methodologically, supervised learning dominated [1, 35], but two systematic limitations constrained this paradigm: predictive systems operated within fixed hypothesis spaces that degraded under distribution shifts, and high predictive accuracy did not guarantee mechanistic interpretability. These constraints—especially the inability to adaptively acquire data—motivated embedding predictive surrogates within sequential decision-making frameworks, the central innovation of closed-loop optimization [9].

2.3. Closed-Loop Optimization Phase

Closed-loop optimization integrated predictive surrogates with sequential decision strategies under experimental budgets. Bayesian optimization (BO) formalized global optimization with expensive evaluations [9, 20, 21]. A probabilistic surrogate is iteratively updated, and acquisition functions such as predictive entropy search allocate experiments via exploration–exploitation trade-offs [22]. Active learning perspectives further generalized adaptive data acquisition for scientific modeling [34, 36].

This phase operationalized idea generation in the intervention space. Coupled with laboratory automation, BO enabled autonomous materials and chemistry discovery [37, 38], and middleware platforms such as ChemOS standardized orchestration [39, 40]. However, BO generates interventions within a predefined parameterization [9, 21] and cannot reshape the design space itself. This representational gap motivated the development of generative models capable of synthesizing structured scientific objects directly.

2.4. Generative and Foundation Model Phase

Generative models enable direct synthesis of structured scientific objects. Representative examples include end-to-end protein structure prediction [26, 41], transferable sequence representations [27, 42], and diffusion- and energy-based molecular generation [5, 23–25, 43–45]; earlier graph generative models and reinforcement learning methods established foundational design paradigms [46–50]. Complementary threads include symbolic regression for discovering governing equations from data [4, 51–56] and LLM-based generation of hypotheses and experimental plans [2, 10, 57]. Mechanistic analysis of each generative thread is provided in Section 4.

Overall, modern discovery platforms combine symbolic search principles [7], predictive surrogates [1], adaptive experimental design [9, 22], and large-scale generative models [24–27]. This progressive integration motivates a cross-paradigm taxonomy capturing both *what* is generated and *how* it is operationalized, which we develop in Section 3.

3. Taxonomy of Idea Generation Paradigms

As Section 2 illustrated, AI systems that generate scientific candidates vary widely—from symbolic equation finders to LLM-based hypothesis genera-

tors to robotic laboratories. Despite this diversity, every system performs two steps: it *proposes structured candidates under domain constraints* and then passes those candidates to some form of validation. This section introduces a two-axis taxonomy that organizes these systems and provides the structural scaffold for the rest of the survey (Sections 4–6).

3.1. Taxonomy Overview

3.1.1 Classification Dimensions

We classify idea generation paradigms along two orthogonal dimensions. **Generation modality** determines the representation of the generated object and, consequently, the entire validation pipeline it must traverse. **Embodiment level** specifies how tightly generation is coupled to empirical action.

(1) Generation Modality. Modality refers to *what form* the generated candidate takes: a text hypothesis, an equation, a molecular graph, an executable program, or a latent coordinate. This choice matters because it determines which constraints can be checked, how validation is performed, and how the system scales. We identify seven recurring modalities:

- **Textual–conceptual:** hypotheses, research plans, and conceptual links generated from scientific corpora using LLMs or literature-grounded reasoning [2, 10, 15, 16, 57].
- **Symbolic–structural:** equations and causal structures generated via symbolic regression, equation learning, and causal discovery [4, 51, 52, 58–60].
- **Search-based generators:** proposal via structured search (e.g., MCTS, evolutionary search) over discrete hypothesis spaces, including equation discovery and design under constraints [53, 54].
- **Program synthesis-based:** hypothesis generation as synthesis of executable programs or compositional procedures, often using neuro-symbolic priors and library induction [61–63].
- **Graph / molecular:** molecular graphs or structured candidates generated by diffusion, energy-based, or constraint-guided models, with complementary RL and graph-generative baselines [23–25, 45–48, 50].
- **Experimental design / simulation-grounded:** interventions proposed by Bayesian optimization or active design [9, 21, 36, 58, 64], and counterfactual trajectories or solution operators produced under physical residual constraints via physics-informed learning and neural operators [18, 65, 66].
- **Latent–representational:** embedding coordinates learned by pretrained foundation models, enabling downstream screening, conditioned generation, or latent-space optimization. Representations do not directly generate candidates but define the coordinate system within which other modalities propose and evaluate [27, 42, 67, 68].

(2) Embodiment Level. Embodiment captures *how closely* the generation process is connected to real-world experiments. This dimension is independent of modality: the same algorithm (e.g., Bayesian optimization) can run purely on a computer or drive an autonomous robotic lab, with very different latency, cost, and safety implications. We distinguish four levels:

- **Computational-only:** in-silico proposal and validation (e.g., equation discovery [4]).
- **Human-mediated:** candidates proposed by AI and executed/verified by humans (common for LLM-generated hypotheses [2, 10]).
- **Partially autonomous:** automated proposal and execution for some steps, with human gates for safety or interpretation [12].
- **Fully closed-loop:** autonomous proposal–execute–update cycles integrating robotics and orchestration [12, 37, 39, 40].

These dimensions separate *what* is generated from *how* it is operationalized (Fig. 3). For example, BO may remain computational or drive robotic experimentation [9, 37], and LLM generation may remain advisory or be embedded in tool-using agent workflows [2, 10]. To emphasize cross-paradigm comparability, we visualize this taxonomy as an x–y coordinate map in which each paradigm family is represented as a labeled point.

3.1.2 Why Modality × Embodiment

Taxonomies based on model architecture (e.g., "transformer vs. genetic programming") or application domain (e.g., "biology vs. materials") are useful for implementation but obscure functional similarities across fields. A modality–embodiment view instead highlights four engineering interfaces that every idea generation system must address: (i) how the hypothesis space is defined, (ii) how constraints are enforced, (iii) how candidates are validated, and (iv) how tightly the system is coupled to experiments or simulations. These interfaces are the main sources of cross-paradigm similarity and difference [53].

3.2. Cross-Paradigm Comparison

We compare paradigms along three axes: how they define the hypothesis space, how they validate can-

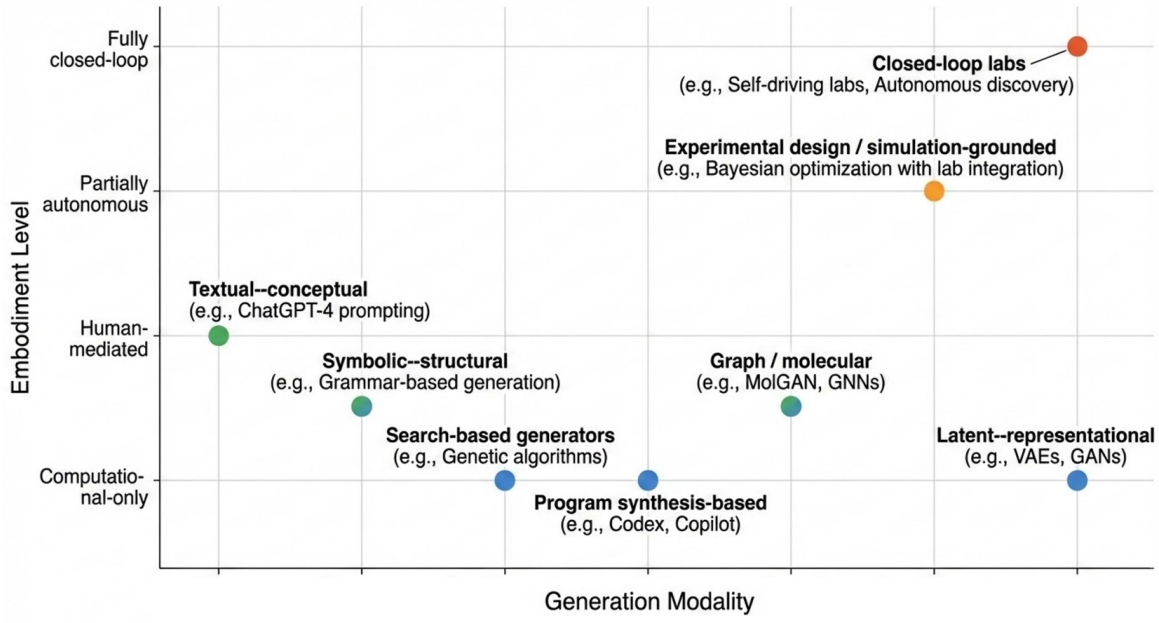

Fig. 3. Two-axis taxonomy of AI-for-Science idea generation. The horizontal axis represents generation modality; the vertical axis represents embodiment level. Each paradigm family is plotted as a labeled point in modality–embodiment space. Point color encodes embodiment level: computational-only (blue), human-mediated (green), partially autonomous (orange), and fully closed-loop (red-orange).

didates, and at what embodiment level they operate (Table 2). A key insight is that similarity along one axis does not imply similarity along others. For example, symbolic regression and search-based generators both explore discrete expression spaces, yet they use different search mechanisms and scale differently. Conversely, systems with very different representations (e.g., LLM text vs. molecular graphs) may share the same embodiment level and similar integration interfaces.

3.2.1 *Hypothesis Space Definition*

Paradigms differ in how hypothesis spaces are parameterized. LLM-based systems induce broad spaces implicitly from corpora [2,10], with retrieval and KG grounding restricting outputs [15,57]. Symbolic regression defines combinatorial spaces shaped by sparsity and operator constraints [4, 52]; causal discovery searches graph structures under interventional assumptions [58–60]. Search-based generators use explicit expansion guided by scoring functions [53, 54]; program synthesis constructs executable programs over typed primitives [61, 62]. Graph-based molecular generators define spaces over graphs or latent manifolds [23, 24, 45]; benchmarks standardize admissibility [50,70]. Experimental design operates over explicit intervention variables [9, 64], and simulation-grounded methods define PDE-residual-constrained function classes [18, 65]. Latent–representational approaches supply coordinate systems via pretrained embedding manifolds [27, 42]. A practical implication is that "generativity" is not uniform: implicit spaces maximize coverage but provide weak guarantees, while explicit structured spaces offer stronger constraints but narrower inductive bias.

3.2.2 *Validation Mechanisms*

A central engineering question is: does the system check constraints *during* generation (internalized validation) or *after* generation (externalized validation)? Textual hypotheses require external checks [57]. Symbolic regression partially internalizes via fit and structure penalties [4, 52]; causal discovery uses interventional consistency as internal scoring [58–60]. Search-based generators reject infeasible expansions [53]; program synthesis raises internalization through execution-based testing [61,62]. Molecular generation validates via validity filters and proxy simulation [23, 45]. Latent–representational models are validated indirectly through downstream task performance and probing of learned features [27,42]. At the most deeply internalized end, experimental design validates through in-loop measurements [9,37], and simulation-grounded approaches constrain candidates via physics residuals [18,65,66]. Fig. 4 places paradigms from left to right by this progression: textual and latent paradigms (more

Table 2. Summary taxonomy of idea generation paradigms by hypothesis space, validation mode, and embodiment level.

Paradigm	Hypothesis Space	Validation Mode	Embodiment Level
Textual–conceptual [2, 10, 15, 16, 57]	Implicit space induced by corpora; partially constrained by retrieval/graphs	External verification (retrieval checks, literature consistency, experiment)	Human-mediated to partially autonomous
Symbolic–structural [4, 51, 52, 58–60]	Expression/graph grammars with operator libraries and priors	Data fit; sparsity/structure scores; interventional consistency	Computational-only (often), sometimes human-mediated
Search-based generators [53, 54, 69]	Discrete hypothesis spaces explored by MCTS/evolution; scoring functions define search pressure	Score-based pruning; held-out fit; constraint checks during expansion	Computational-only to human-mediated
Program synthesis-based [61–63]	Program spaces over typed primitives; library induction and compositional priors	Execution-based tests; likelihood/MDL-style objectives; unit constraints	Computational-only (often), sometimes human-mediated
Graph / molecular generation [23–25, 45–48, 50]	Graph/latent spaces with chemical feasibility constraints; optional energy/constraint guidance	Validity filters; proxy simulation; downstream assay	Computational-only to human-mediated
Experimental design / simulation-grounded [9, 18, 21, 36, 58, 64–66]	Explicit design spaces; or PDE/operator-constrained function classes	Measurement-in-loop; uncertainty reduction; residual/consistency checks	Computational-only to fully closed-loop
Latent–representational [27, 42, 67, 68]	Embedding manifolds learned by pretrained foundation models; defines coordinate system for other modalities	Downstream task performance; probing; latent-space consistency checks	Computational-only to human-mediated

externalized), symbolic/search/program/molecular paradigms (mixed), and simulation-grounded plus closed-loop paradigms (more internalized). This spectrum systematically affects throughput, bias, and failure rates.

3.2.3 *Scalability and Integration Interfaces*

Scalability is constrained by fundamentally different bottlenecks across paradigms, meaning that the choice of modality and embodiment level affects not only performance but practical deployability. Foundation models scale in compute and data [2, 10]; diffusion introduces sampling cost [24, 25]. Symbolic and program synthesis face combinatorial growth, motivating priors, learned heuristics, and staged search [54, 61, 62]. Search-based generators share the same combinatorial bottleneck but offer explicit control via branching policies, pruning, and anytime behavior [53]. BO scales for low-to-moderate dimensional spaces but degrades in very high dimensions [9, 21]. Latent–representational models offer high amortized scalability through reusable embeddings, though probing and alignment costs grow with task diversity [27, 71]. Closed-loop systems scale with experimental throughput but require orchestration, metadata standards, and interoperability [39, 40, 72].

Despite these differences, integration across paradigms converges on four recurring interfaces: (i) a *candidate proposer* that outputs structured objects, (ii) *constraint validators* that apply hard or soft checks, (iii) *evaluators* that score candidates via simulation or experiment, and (iv) *memory stores* that retain data and provenance. These four interfaces appear in every hybrid system we surveyed—whether an LLM orchestrates external tools [2, 10], BO drives a robotic lab [12, 37], or a search-based generator uses a neural scorer for pruning [53, 54]. Recognizing them provides a practical blueprint for designing new hybrid systems.

Synthesis

The modality–embodiment taxonomy supports cross-paradigm comparison by separating representational modality from operational coupling. Adding *search-based generators* and *program synthesis-based* paradigms makes explicit a widely used proposal mechanism: structured search over discrete hypothesis spaces with algorithmic control of exploration,

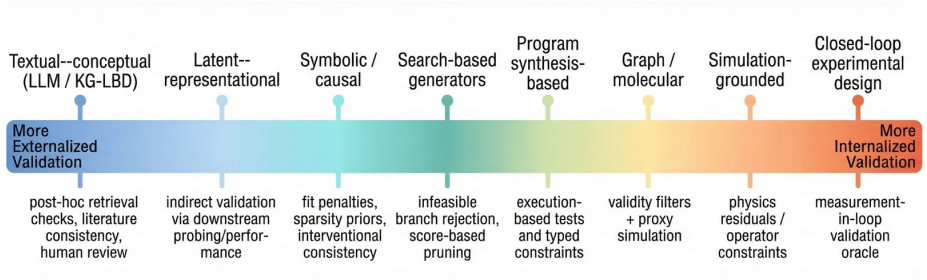

Fig. 4. Internalization–externalization spectrum of validation mechanisms. Paradigms are positioned according to the degree to which validation is embedded within generation (internalized) versus applied post-hoc (externalized), from textual and latent paradigms on the left to simulation-grounded and closed-loop paradigms on the right.

constraints, and anytime refinement [53,61,62]. This taxonomy serves as the organizing scaffold for the remainder of the survey: Section 4 follows the modality axis to examine seven paradigms in approximate historical order; Section 5 follows the embodiment axis to survey closed-loop laboratory architectures; and Section 6 addresses evaluation dimensions and benchmarks for cross-paradigm comparison.

4. Core Idea Generation Paradigms

This section surveys seven core paradigms along the modality axis of the taxonomy introduced in Section 3. Each paradigm uses a different representation (text, symbols, graphs, latent vectors) and a different mechanism to propose and constrain candidates. We present them in approximate historical order, from knowledge graph / literature-based discovery (ca. 1986) to LLM-based generation (ca. 2023), and use a consistent three-part format for each: core mechanism, role in scientific discovery, and limitations.

4.1. Knowledge Graph and Literature-Based Discovery

Knowledge graph (KG) and literature-based discovery (LBD) systems generate hypotheses from *explicit relational structure* extracted from publications and curated resources. A KG encodes entities as nodes and typed relations as edges; hypothesis generation is posed as link prediction, constrained path discovery, or subgraph pattern completion under a schema [73, 74]. Foundational LBD work demonstrates that connecting disjoint literatures via intermediate concepts yields testable hypotheses, establishing graph traversal and bridging-node search as core operators [14].

In scientific discovery workflows, KG/LBD provides auditable candidates with typed provenance links [14, 16, 73, 75], and schema constraints enable cross-dataset reasoning through data fusion [74,75]. Deployments follow a pipeline (ingestion, extraction, graph construction, constrained generation, and prioritization), with biomedicine as the canonical domain [14,57,74].

However, coverage depends on extraction quality and ontology scope [75], entity linking errors create spurious edges that propagate through path-based inference [73], and curation bias skews predictions toward well-studied areas [14, 75]. Multi-hop path search can chain weak edges into plausible but unsupported narratives [14].

Whereas KG/LBD generates hypotheses from discrete relational structures extracted from text and ontologies, a complementary family of methods seeks to discover governing equations and causal structures directly from observational data.

4.2. Symbolic Regression, Equation Learning, and Causal Discovery

Symbolic regression searches a discrete space of symbolic expressions to discover interpretable analytical forms from data [4, 51, 52]. Causal discovery infers directed causal graphs from observational or interventional data, with differentiable relaxations improving scalability [59, 76]. Both methods are distinctive in that their outputs—equations, causal graphs—are themselves verifiable scientific claims.

In scientific discovery, these methods directly contribute to rediscovering physical laws, identifying governing equations of dynamical systems, and elucidating causal mechanisms. Physical constraints such as dimensional consistency, symmetry, and sparsity can be incorporated into the generation process, and active causal discovery couples experimental design with causal reasoning [58]. Hybrid approaches in which LLMs guide symbolic regression have also emerged [53,54].

However, the search space grows combinatorially

with expression complexity, and identifiability is limited in high-dimensional or noisy settings. Causal discovery from observations alone cannot identify beyond the Markov equivalence class, often requiring interventional data.

While symbolic regression and causal discovery produce interpretable symbolic structures, many scientific design problems require efficient sequential search over noisy black-box objectives—the domain of Bayesian optimization.

4.3. Bayesian Optimization and Experimental Design

Bayesian optimization (BO) selects experiments sequentially using a probabilistic surrogate and an acquisition rule that trades off exploitation against uncertainty reduction [9,20,21]. Constrained BO extends this mechanism to feasibility and safety limits [64]. Conceptually, BO is *generative in intervention space but not in representational space*: it proposes candidate actions within a fixed parameterization but does not invent new variables or model classes [9,21].

In AI-for-Science workflows, BO operationalizes *experimental design* by proposing the next intervention via uncertainty-aware acquisition [22]. When coupled to automation layers, it serves as the primary controller for self-driving laboratories (Section 5) [12,37]. Modular separation of surrogate, constraints, and acquisition enables plug-and-play deployment across objectives and instruments [9, 20, 21].

However, BO degrades in high dimensions, surrogate assumptions can fail on complex platforms, and acquisition pathologies arise under miscalibrated uncertainty [9,21,22]. In embodied settings, instrument drift or batch effects further corrupt surrogate updates [12,39].

While BO searches efficiently within numerical design spaces, many scientific domains require candidate dynamics governed by continuous physical laws—a need addressed by simulation-based and SciML approaches.

4.4. Simulation-Based and Scientific Machine Learning Discovery

SciML paradigms couple learning with governing equations and numerical solvers to generate *structured simulation outputs*—fields over space-time, operator responses, or counterfactual rollouts. Physics-informed learning embeds differential constraints into training objectives [18, 19, 77], while operator-learning approaches approximate mappings between function spaces for fast inference [65,66].

In discovery workflows, SciML serves as an inner-loop engine: embedding PDE residuals or conservation laws narrows outputs to physically admissible solutions [18, 19, 78], and learned operators amortize simulation cost for repeated queries inside design loops [65, 66]. Differentiable pipelines enable gradient-based parameter estimation and integration of mechanistic with learned components [18,19,79]. In pipelines, SciML modules integrate as accelerators, inverse-model components, or validators.

However, balancing data and physics losses can create ill-conditioned objectives, sparse observations admit non-unique mechanisms, and operators extrapolate brittly under coverage gaps [19,35,65,66]. Small surrogate errors compound in long-horizon rollouts, producing unstable trajectories [66].

Representation learning and foundation models complement simulation-based methods by providing reusable latent coordinates for downstream search and generation.

4.5. Representation Learning and Foundation Models

Representation learning and foundation models provide *shared hypothesis spaces* by embedding scientific objects (sequences, graphs, structures, materials records) into reusable latent representations [27,42,71,80]. Pretrained encoders define search coordinates for downstream optimization, enable retrieval of analogs, and serve as conditioning priors for generative modules. In materials science, these models are intertwined with standardized data infrastructure that enables multi-source training and cross-lab reuse [67,68].

In scientific discovery, these models amortize feature extraction and allow rapid transfer under limited labels. Deployments take three forms: embedding-for-prediction [27, 42, 71], embedding-for-search, and embedding-conditioned generation [23–25].

However, foundation models primarily encode *statistical regularities* rather than causal structure. Self-supervised objectives do not guarantee alignment with scientific targets [27, 42], embeddings may encode protocol biases [67, 68], and downstream optimizers can push candidates off the pretraining manifold [27,42].

Combinatorial scientific design problems further demand generating diverse candidates under multi-modal objectives.

4.6. Diffusion-Based and Diversity-Aware Generative Models

Diffusion models learn an implicit data distribution and generate valid structures via iterative denoising, with guidance mechanisms steering toward desirable regions [5, 23–25]. GFlowNets train

a stochastic policy to sample structured objects with probability proportional to a reward function, producing *diverse* high-reward candidates rather than collapsing to a single optimum [81]. Both approaches address the need for diversity in multimodal scientific design spaces.

In scientific discovery, constrained diffusion has been applied to de novo protein backbone generation [5] and multi-objective optimization [24]. GFlowNets are particularly valuable when multiple mechanisms or molecular scaffolds may satisfy an objective under uncertainty [70], and their explicit *construction graph* over partial objects makes constraint enforcement modular [81].

However, diffusion models incur high sampling cost, GFlowNet scalability depends on state-space factorization, and reward misalignment can bias sampling toward proxy artifacts [23, 81].

Large language models provide the most recent general-purpose interface, unifying hypothesis generation, tool use, and multi-agent coordination through natural language.

4.7. LLM-Based Hypothesis Generation

LLM-based systems generate candidate hypotheses, research plans, and conceptual links from scientific corpora, increasingly as components in tool-augmented or multi-agent pipelines [2, 10, 82]. However, LLMs map context to plausible *formulations* rather than providing mechanistic guarantees.

Practical deployments emphasize *grounding*, *verification*, and *workflow integration* [57]. Retrieval augmentation constrains outputs [57], and multi-agent coordination with proposer–critic–planner roles improves completeness [6, 8, 83–87].

However, benchmarks reveal gaps in causal/mechanistic reasoning relative to domain-specialized solvers [88–90], and physical feasibility is not enforced by default. Key failure modes include hallucination [57], mechanistic confabulation [88, 90], and error amplification in multi-step pipelines [6, 84]. The most reliable pattern is proposal generation followed by verification with specialized evaluators [57, 89].

Synthesis

Viewed through Section 3's modality–embodiment taxonomy, these paradigms separate along the *modality axis* (how candidates are parameterized) and their *constraint enforcement strategy*. KG/LBD emphasizes auditability but is bounded by extraction quality [14]. Symbolic regression and causal discovery directly produce interpretable equations and causal graphs but face combinatorial search-space growth [4, 52, 59]. BO is most effective when controllable variables are well-defined and evaluation is expensive [9, 22]. SciML provides constraint-consistent candidate dynamics but is limited by identifiability and regime shift [18, 19, 65]. Foundation representations define compact coordinates for downstream search [27, 42]; diffusion models and GFlowNets generate diverse candidates via guided denoising and reward-proportional sampling [23, 81]. LLM pipelines maximize interface flexibility but externalize most feasibility checks [2, 10, 57]. Table 3 consolidates these mechanism-level differences across the seven paradigms. The complementary embodiment axis is addressed in Section 5.

5. Closed-Loop and Self-Driving Laboratory Systems

Section 4 examined paradigms along the modality axis; this section turns to the embodiment axis. Closed-loop and self-driving laboratories sit at the highest embodiment level, where idea generation is inseparable from physical execution. These systems combine three capabilities into a single cyber-physical workflow: (i) algorithmic experiment selection, (ii) automated execution via robotics, and (iii) continuous model updating from new measurements [12, 37–39]. In practice, "idea generation" here means choosing the next physical experiment under cost, time, and feasibility constraints—most commonly via Bayesian optimization [9, 21, 22, 64]. As a result, system performance depends as much on engineering (middleware, robotics, data pipelines) as on the learning algorithm itself.

5.1. Architectural Components

A typical self-driving laboratory decomposes into four layers: decision engines, orchestration middleware, robotic execution/sensing, and interoperability standards, including shared laboratory-action ontologies and data/protocol schemas such as SiLA 2- or Allotrope-style specifications [39, 40, 72]. The control loop is:

$$\textit{Decision Engine} \rightarrow \textit{Middleware} \rightarrow \textit{Robotics} \rightarrow \textit{Data Ingestion} \rightarrow \textit{Model Update} \rightarrow (\text{repeat}).$$

Each layer constrains stability, throughput, and extensibility [39, 40] (Fig. 5).

Decision engines. BO remains the dominant controller, using a surrogate to predict outcomes and an acquisition function to propose experiments under uncertainty [9, 21]. Information-theoretic acquisition optimizes expected information gain [22], and constrained BO encodes feasibility limits [64]. Alternative controllers include multi-objective optimizers and reinforcement-learning policies [91]. Two

Table 3. Cross-paradigm comparison of seven proposal mechanisms.

Paradigm	Strength	Limitation	Scalability	Interpretability
KG/LBD (1986–)	Typed, auditable candidates via paths/links [14–16]	Bounded by schema coverage and extraction noise [73,75]	Medium (graph size manageable; curation costly) [75]	High (explicit relations + provenance) [14,74]
Symbolic regression / causal discovery (2009–)	Interpretable equations and causal graphs; physics constraints embeddable [4,51]	Combinatorial search-space explosion; observational identifiability limits [52,59]	Low–Medium (expression complexity) [4,51]	High (symbolic outputs directly auditable) [4,51]
Bayesian optimization (2015–)	Sample-efficient sequential experiment selection [20,22]	Needs well-specified design space; struggles in high dimension [9,21]	Medium (degrades with dimension) [9, 21]	Medium (explicit policy, but model assumptions) [9,22]
Simulation/SciML (2019–)	Physics-consistent rollouts; fast surrogate queries [18,66]	Stiff training; regime shift; identifiability limits [19,35]	Medium (depends on solver/surrogate regime) [65,79]	Medium (structured outputs; latent internals opaque) [18,19]
Representation/ foundation models (2021–)	Reusable embeddings for screening and conditioning [27,42,80]	Proxy objective mismatch; corpus bias; off-manifold optimization [27,42,67,68]	High (amortized reuse across tasks) [27,71]	Low–Medium (latent features need probing) [27,42]
Diffusion & GFlowNets (2021–)	Diverse candidates under multimodal objectives; reward-proportional sampling [5,25,81]	High sampling cost (diffusion); reward misalignment; state-space tractability; proxy artifacts [23,81]	Medium (depends on state-space factorization) [24,81]	Low–Medium (latent generative process) [23,81]
LLM-based generation (2023–)	Flexible candidate formats; integrates tools and roles [6,83–87]	Feasibility and evidence faithfulness require external gates [57]	High (compute- and retrieval-dependent) [2,10]	Low–Medium (prompt- and tool-dependent traces) [6,57]

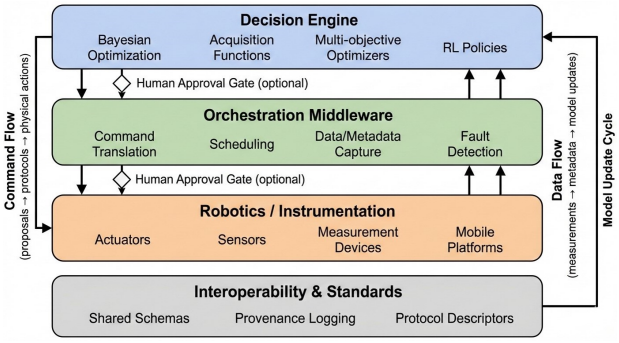

Fig. 5. Four-layer architecture of a self-driving laboratory. Downward arrows represent command flow (proposals to executable protocols to physical actions); upward arrows represent data flow (measurements to model updates). In addition to human oversight gates at layer boundaries, a middleware-level *Safety Filter Gate* screens proposed protocols for hazardous materials, biosecurity-sensitive procedures, and unsafe operating conditions before any actuation command is dispatched to robotics (see §5.2).

interfaces largely determine behavior: (i) design-space parameterization—mis-specification induces structural blind spots [9, 21]—and (ii) asynchrony handling—controllers must support batched updates and delayed feedback without destabilizing learning [39, 40].

Middleware. Orchestration middleware (e.g., ChemOS [39, 40]) translates abstract proposals into device-specific commands, managing scheduling, data/metadata capture, and fault isolation [72]. Middleware stability often dominates long-term maintainability; standardized schemas directly reduce integration friction. Middleware is also the natural insertion point for a *Safety Filter Gate*: every actuation command must pass deterministic checks against hazard ontologies (e.g., CWA/CWC chemicals, BSL-restricted reagents, explosive-mixture thresholds) before reaching the robotics layer, and these gates should be versioned independently of the decision-engine policy to preserve auditability.

Robotics and interoperability. Robotic subsystems—autonomous materials platforms [37], robotic chemists [38], and exploratory mobile systems [92]—define the effective search space via action realization constraints (discretization, reachable sets, actuation limits) and feedback reliability via measurement properties (noise, drift) [39, 40]. Recent work links vision-language-action control, multi-robot coordination, and polymer electronics platforms to autonomous workflows [93, 94]. Because systems become brittle when formats and semantics

are implicit, interoperability through shared experiment descriptors, normalized schemas, laboratory-action ontologies, and standard interfaces such as SiLA 2- or Allotrope-style specifications is a prerequisite for scaling across labs [72].

5.2. Autonomy, Objective Design, and Governance

The architecture above determines what a closed-loop system *can* do; governance determines what it *should* do and who decides. In practice, autonomy is not all-or-nothing: it depends on which steps are automated and where human approval is still required [12]. Most systems keep humans in the loop for revising goals, setting safety boundaries, and handling anomalies [95, 96]. In high-consequence closed-loop laboratories, however, governance must also be implemented as *executable safety constraints* rather than relying on human oversight alone: middleware-level *Safety Filter Gates* (§5.1) screen proposed protocols against hazard ontologies, dual-use procedures, and facility-specific limits before any actuation, and they complement—rather than replace—the policy-level safeguards detailed below. Three design dimensions constrain autonomy:

- **Objective specification:** systems typically optimize scalarized proxies (yield, score, cost) rather than open-ended scientific goals [9, 12, 21].
- **Constraint encoding:** feasibility boundaries must be represented algorithmically or enforced procedurally [64].
- **Human oversight:** supervision layers mitigate anomaly propagation and reinterpret intermediate results [95, 96].

Objective mis-specification and reward hacking. A critical failure mode is *reward hacking*: if the optimization target is a proxy metric (e.g., predicted yield) that does not fully capture scientific value, sample-efficient optimizers like BO can quickly converge to solutions that score highly on the proxy but are scientifically meaningless [9, 12, 21]. For example, an optimizer might exploit measurement artifacts or narrow chemical conditions that inflate the score without genuine discovery. Human-in-the-loop governance and periodic objective auditing are therefore essential safeguards [95, 96].

Biosecurity and dual-use risk. Closed-loop autonomous laboratories that accelerate synthetic chemistry or biology capabilities also amplify *dual-use* risks: a synthesis route that appears benign to a human reviewer can be expanded by middleware into combinations of restricted reagents, accumulating precursors, or actions exceeding control thresholds. Decision-engine-level oversight alone is therefore insufficient. Governance should be layered into (i) a middleware-level *Safety Filter Gate* performing deterministic checks against hazard ontologies (e.g., CWA/CWC chemicals, BSL-restricted reagents, controlled precursors) and quantitative thresholds, blocking policy-violating actions before any actuation; (ii) immutable audit logs recording provenance and justification for every permitted or denied action [72, 97]; and (iii) external-review triggers that escalate ambiguous cases to expert review via human-in-the-loop gates [95, 98, 99]. Governance becomes operationally enforceable only when these mechanisms are implemented as a *technical control plane*, separate from the decision-engine policy.

5.3. Scalability Constraints

Governance choices set the boundaries of autonomous operation; scalability determines how far the system can push within those boundaries. Scaling is limited primarily by physical throughput and integration overhead:

- **Turnaround time:** throughput is bounded by the slowest experimental stage; scaling requires parallelization and non-blocking scheduling [39, 40].
- **Consumables and maintenance:** reagent depletion, calibration cycles, and hardware downtime cap continuous operation [37, 38, 92].
- **Data infrastructure limits:** higher throughput stresses storage, versioning, and provenance systems; schema drift can silently corrupt model updates [72].
- **Design-space expansion:** enlarging controllable variables increases search complexity and can outpace sample-efficient methods unless structural priors are introduced [9, 21].

Networked laboratories amplify coordination challenges: shared ontologies, synchronized protocol definitions, and consistent metadata become mandatory for data pooling [72].

Synthesis

Closed-loop laboratories represent the highest embodiment level in our taxonomy: idea generation is fully coupled to physical execution, and the experiment itself is the ultimate validator. In practice, system performance depends as much on middleware robustness and data standards as on the learning algorithm [9, 21, 22, 64, 72]. Automation increases throughput, but poorly chosen objectives can lead the system to exploit proxy metrics rather than make genuine discoveries. Sustained reliability therefore

requires well-calibrated objectives, constraint-aware policies, interoperability standards, and structured human oversight [12, 72, 95, 96]. Section 6 develops evaluation frameworks that apply across all embodiment levels.

6. Evaluation Frameworks

How should we evaluate AI systems that generate scientific ideas? Unlike static prediction benchmarks, these systems propose interventions, allocate resources, and update their models under noise and latency. Evaluation must therefore be multi-dimensional and practical—metrics should be computable from system logs, model states, and validated outcomes. We organize evaluation into six dimensions that apply across all paradigms and embodiment levels: (i) efficiency, (ii) reliability, (iii) interpretability and idea quality, (iv) causal validity, (v) discovery novelty vs. rediscovery rate, and (vi) mechanistic depth.

6.1. Efficiency

Efficiency measures progress per unit resource (time, experiments, compute).

Sequential efficiency. In BO and related controllers, cumulative regret summarizes performance over T trials relative to the best achievable value [9, 21]. In multi-objective settings, hypervolume improvement or Pareto-front expansion provides a more appropriate metric [100]. Under constraints, efficiency must be reported jointly with feasibility rate; improvements achieved via frequent invalid proposals are operationally misleading [64].

Information efficiency. When the goal is model identification, information-theoretic acquisition (e.g., predictive entropy search) quantifies expected uncertainty reduction per experiment [22, 101]. Information gain should be paired with uncertainty calibration metrics; miscalibrated posteriors invalidate efficiency claims [9, 22].

System-level efficiency. In embodied platforms, efficiency includes cycle time, queue utilization, and time-to-threshold discovery [13]. Orchestration logs enable decomposition into decision latency, execution time, and update overhead [39, 40]. For large generative models, candidate yield per unit compute and end-to-end shortlist latency should be reported [2, 10].

6.2. Reliability

Reliability concerns stability under constraints, noise, and distribution shift.

Constraint adherence. Metrics include feasibility rate, unsafe proposal rate, and executed violation rate [64, 102]. For robotic labs, protocol validity and instrument envelope adherence must be logged separately for proposed vs. executed actions [39, 40].

Reproducibility and provenance. Reproducibility metrics emphasize versioned models, complete metadata, and replayable pipelines [97]. Interoperability standards can be evaluated via schema compliance and provenance completeness [72]. For retrieval-augmented systems, evidence traceability and deterministic re-run consistency provide additional reliability indicators [57].

Update stability. Closed-loop systems require robustness to drift and noise. Calibration error, change-point detection frequency, and rollback events provide measurable indicators [39, 40, 97].

6.3. Interpretability and Idea Quality

Interpretability evaluates whether outputs are structured, auditable, and decision-relevant.

Structural compactness. In symbolic regression and equation learning, interpretability can be quantified via expression length, sparsity, and term count [4, 52]. Reporting complexity–fit trade-offs avoids conflating accuracy with explanatory simplicity.

Evidence-grounded hypotheses. In KG/LBD systems, relational paths and provenance edges provide explicit justification [14, 15]. Retrieval-augmented LLMs can be evaluated via evidence alignment rate and attribution accuracy [57]. These metrics prioritize traceable justification over subjective creativity.

Feasible novelty. Novelty metrics should be paired with feasibility filters. Graph-distance or semantic divergence from prior literature provides one operationalization [14, 15]. In design domains, novelty must be reported alongside validity or executability rates to avoid rewarding out-of-manifold artifacts.

Together, these metrics define an *audit view* that human-mediated embodiments (§3.1) can surface as a dashboard—typically combining provenance trails, uncertainty calibration, constraint-violation logs, causal-test status, and the system's proposed next action—so that scientists can inspect a candidate before committing experimental resources.

6.4. Causal Validity and Mechanistic Depth

Causal validity asks a simple but demanding question: does the generated hypothesis hold up when we intervene, or does it merely fit the observed data [60, 103–105]? Testing requires intervention experiments, counterfactual consistency checks, or invariant prediction across different contexts [14, 15].

Mechanistic depth poses a stricter requirement: does the output reveal *why* something works, such as through conserved quantities, invariants, or physically interpretable components, rather than only *that*

Table 4. Evaluation matrix: six dimensions, representative metrics, applicable paradigms, and limitations.

Dimension	Metric Example	Applicable Paradigms	Limitation
Efficiency	Regret; time-to-threshold [9, 13]	BO; closed-loop labs [9, 21, 39, 40]	Requires well-defined objective.
Reliability	Feasibility / violation rate [64, 102]	Constrained BO; embodied systems [37, 38, 64, 92]	Depends on accurate constraint models.
Interpretability	Symbolic complexity [4, 52]	Equation discovery [4, 52, 54]	Parsimony may trade off accuracy.
Causal validity	Interventional confirmation rate [60, 103, 105]	Causal discovery; closed-loop labs [37, 60, 103]	Requires costly experiments.
Novelty vs. rediscovery	Temporal holdout rediscovery ratio; graph-distance or semantic divergence [14–16]	KG/LBD; literature-based systems [14]	Reference-set dependent.
Mechanistic depth	Invariance / conserved-quantity alignment [105, 106]	Symbolic; causal; physics-informed models [4, 52, 105]	Hard to quantify across domains.

it works [4, 52, 77, 106]? For foundation models, one practical test is probing whether latent features correspond to known physical or biochemical variables [27, 42]. Systems that achieve high predictive scores without exposing transferable structure should be distinguished from those that yield compact, mechanistically interpretable models. Table 4 consolidates all six evaluation dimensions.

Synthesis

A rigorous evaluation framework must extend beyond efficiency, reliability, and interpretability to include causal validity, novelty vs. rediscovery rate, and mechanistic depth. Each dimension applies across the full modality–embodiment space, though the required evidence differs: efficiency must account for both algorithmic sample complexity and physical cycle time, while causal validity depends on whether validation is internalized or externalized [9, 22, 72, 97, 105]. The metrics defined above are operational and composable—for instance, throughput claims can be penalized by the unsafe-proposal rate as $\eta_{\text{eff}}\,(1 - r_{\text{unsafe}})$ [64, 102]—but a canonical cross-paradigm scalar score is left to dedicated benchmarking efforts [13]; Fig. 6 therefore reports illustrative qualitative profiles for four paradigm families rather than a normative ranking. Section 7 examines how structured integration patterns operationalize these dimensions within end-to-end discovery systems.

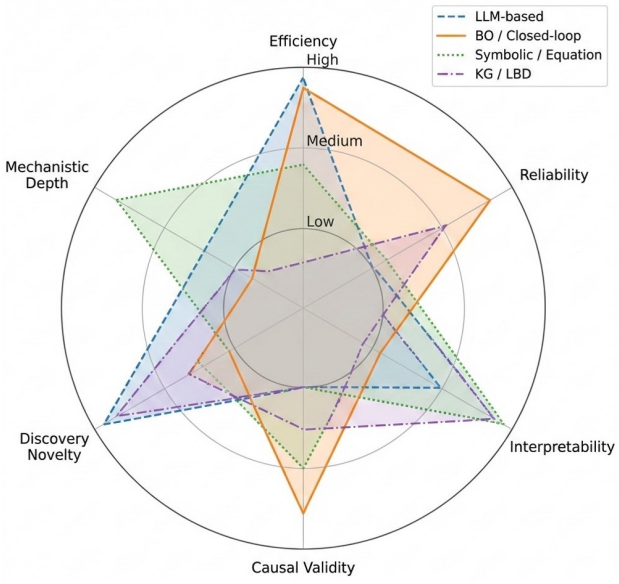

Fig. 6. Qualitative evaluation profiles for four paradigm families across six assessment dimensions. Each polygon represents a paradigm's relative strength on each dimension.

7. Toward Structured and Integrated Discovery Systems

Sections 4–6 examined individual paradigms and evaluation dimensions largely in isolation, but practical discovery systems must combine multiple paradigms. For example, an LLM may generate hypotheses that a physics simulator validates, or a pre-trained encoder may feed a Bayesian optimizer that drives a robotic lab. Building such integrated systems requires solving two engineering problems: (i) *representation alignment*—ensuring that different modules can exchange candidates in compatible formats, and (ii) *control-flow orchestration*—keeping long-running evaluations (simulations, assays, lab runs) synchronized with the decision-making policy. This section describes three recurring integration patterns observed across AI-for-Science platforms (Fig. 7), focusing on how they work, what they enable, and what engineering challenges remain.

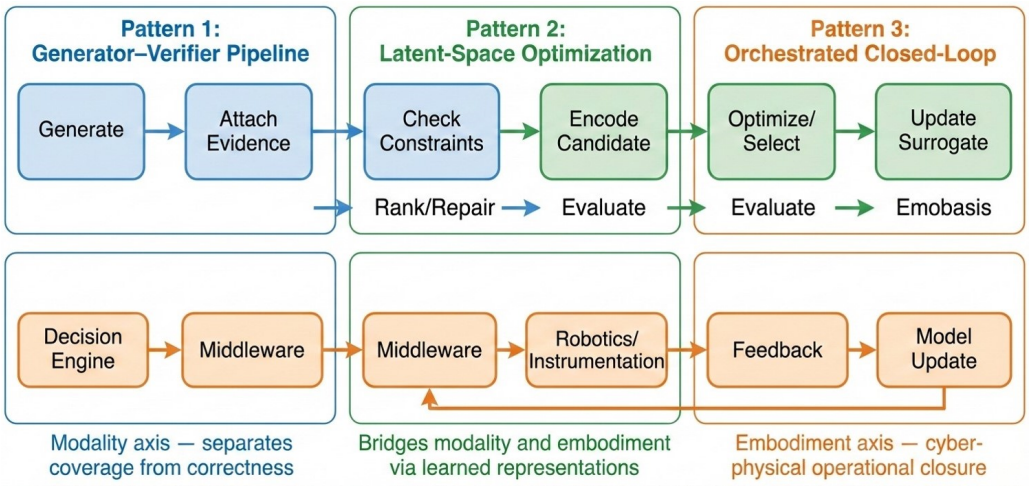

Fig. 7. Three recurring integration patterns in AI-for-Science discovery systems. Pattern 1 operates primarily along the modality axis, Pattern 2 bridges modality and embodiment through learned representations, and Pattern 3 operates along the embodiment axis.

7.1. Generator–Verifier Pipelines

A high-recall generator—typically an LLM producing hypotheses, experimental plans, or structured summaries [2, 10]—proposes candidates, which a verifier stack then filters, scores, and repairs before downstream use. Verifiers include retrieval-grounding modules for evidence attachment [57], knowledge-graph consistency checks for typed relations and provenance [14], and physics/constraint modules (e.g., PDE residual checks) when candidates can be expressed in simulator-compatible form [18, 19]. The pipeline is implemented as a staged gate:

$$\texttt{Generate} \rightarrow \texttt{Attach Evidence} \rightarrow \texttt{Check Constraints} \rightarrow \texttt{Rank/Repair}.$$

This pattern separates coverage from correctness: the generator explores broadly while reliability duties move into independently testable, versionable modules. Verifiers can be swapped per domain—literature-heavy tasks emphasize retrieval and KG checks [57], physics-heavy tasks emphasize residual checks [18, 19]—and the staged design supports incremental adoption [57]. The dominant engineering challenge is *schema alignment* between free-form text and structured verifiers: mapping LLM hypotheses to KG entities requires robust entity linking, and converting textual mechanisms into parameterized models requires intermediate representations [18, 19, 75]. Multi-stage gating also introduces *latency*, requiring caching and partial evaluation for throughput [57].

Repair requires more than reranking: when a verifier rejects a candidate, the failure signal must be structured enough for the generator to produce a *patched* hypothesis rather than resampling blindly. Three repair mechanisms recur—*iterative symbolic refinement*, where residual or constraint violations drive sub-tree swaps and parsimony-aware pruning in symbolic regression [4, 51, 52]; *critic-driven prompting*, where an LLM critic surfaces evidence gaps or constraint violations as structured feedback for the next generation step (e.g., Self-Refine and reviewer loops in AI co-scientist systems) [6, 57, 84]; and *counterexample-guided revision*, where a falsifying counterexample (a violated PDE residual, an inconsistent KG path, or a failed experiment) is fed back as an explicit constraint on the next proposal [18, 19]. Without structured failure signals, the loop degenerates to rejection sampling.

7.2. Latent-Space Optimization with Pretrained Encoders

A pretrained encoder—protein language models [27, 42], graph-based representations for molecules and relational structures [71, 80]—defines a continuous search space over which BO allocates evaluations via acquisition rules [9, 21, 22], with constrained variants when feasibility must be modeled explicitly [64]. The minimal loop is:

$$\texttt{Encode Candidate} \rightarrow \texttt{Optimize/Select} \rightarrow \texttt{Evaluate} \rightarrow \texttt{Update Surrogate}.$$

This pattern makes high-dimensional design problems operational: encoders supply priors that reduce sample requirements, and the modular integration allows the optimizer and surrogate to be tuned independently of the encoder. The same representation supports multi-objective Pareto-front exploration [100] and constrained feasibility model-

ing [64]. However, self-supervised embeddings may not preserve distances relevant to experimental objectives, causing BO to over-exploit spurious geometry (*representation-to-objective mismatch*) [27, 42]. Off-manifold proposals can produce invalid candidates unless decoding is constrained (*decoder validity*) [71], and surrogate *uncertainty calibration* degrades under distribution shift [9, 22].

Several mitigations address these failure modes without retraining encoders from scratch. *Property-aware contrastive fine-tuning* and metric learning with experimentally validated triplets pull candidates with similar measured outcomes closer in latent space and push apart those with divergent outcomes, aligning the geometry to the experimental objective [27, 42, 71]; multi-task pretraining with auxiliary property heads further reduces representation-to-objective mismatch. *Supervised calibration* of surrogate uncertainty against held-out experimental measurements addresses calibration drift [9, 22], and *active-learning-based embedding adaptation*—periodically refreshing the encoder with newly labeled candidates from the loop—prevents geometry drift as the campaign explores under-represented regions [36]. These should be treated as possible mitigations rather than universal remedies, since each technique adds inductive bias that may shift failure modes elsewhere.

7.3. Orchestrated Closed-Loop Execution

A decision engine—often a BO controller [9, 21] with constrained [64] or information-gain acquisitions [22]—selects the next experiment; middleware translates selections into executable protocols, schedules instruments, and captures data/metadata. ChemOS and ChemOS 2.0 are representative orchestration layers [39, 40], while robotic lab platforms provide automated actuation and sensing [37, 38, 92]. Autonomy levels can be treated as gate and scheduling-policy configuration [12]:

$$\texttt{Decision Engine} \rightarrow \texttt{Middleware} \rightarrow \texttt{Robotics/Instrumentation} \rightarrow \texttt{Feedback} \rightarrow \texttt{Model Update}.$$

The pattern's value is *operational closure*: consistent protocol execution, standardized metadata, and automated ingestion enable iterative surrogate and policy improvement over long campaigns, while middleware decouples algorithm iteration from hardware integration [37–40]. Engineering challenges include *time-scale mismatch* between fast algorithmic cycles and variable physical latency [39, 40], *interface standardization* across vendor-specific devices [72], and non-computational *scalability constraints* (calibration, consumables, facility throughput) that bound achievable iteration rates regardless of algorithm speed [12, 37, 92].

Synthesis

Across patterns, integration succeeds when (i) representations exchanged between modules are explicit, (ii) verification and constraint checks are implemented as separable gates, and (iii) orchestration layers provide stable APIs for long-running, asynchronous evaluation [39, 40, 57, 72]. Together, the three patterns cover complementary ways systems combine generative breadth (LLMs [2, 10]), structural priors (foundation representations [27, 42]), and resource-efficient decision policies (BO [9, 21, 22, 64]) within deployable discovery pipelines.

8. Open Problems and Research Directions

The integration patterns in Section 7 show that deployable pipelines already combine generative, representational, and experimental modules. However, each pattern also reveals unresolved bottlenecks. As AI assumes a larger role in scientific workflows, generating candidates and increasingly influencing what gets tested, the main risks shift from model accuracy to *system-level concerns*: Can the system ground its claims in evidence? Does it transfer across domains? Can outputs be verified at scale? This section examines these persistent challenges, including the emerging class of *AI co-scientist systems*.

8.1. Epistemic Reliability

Generative modules frequently decouple plausibility from correctness, most visibly in LLM-based systems where fluent reasoning masks factual or mechanistic error [2, 10, 88–90]. Benchmark gains do not ensure scientific robustness: performance varies across domains and formats [88, 90], tool augmentation does not guarantee correct integration into falsifiable hypotheses [57, 89], and uncertainty proxies in generative models are not calibrated to epistemic risk unlike probabilistic surrogates in Bayesian design [9, 21]. Reliability requires *evidence-carrying outputs* and *constraint-carrying representations*: structured provenance fields linking claims to sources [57], domain validators integrated at generation time [18, 19], and calibration protocols evaluated under controlled distribution shifts [88, 90].

These failure modes are not exclusive to LLMs; each generation modality has its own characteristic failure mode. Symbolic regression and equation-learning systems overfit by including spurious operators that improve information criteria yet violate units, dimensions, or symmetries [4, 51, 52]; SciML

systems (physics-informed networks, neural operators) can reduce PDE residuals while breaking conservation laws or degrading under regime shift [18, 19, 66]; and latent / foundation models exploit representation geometry, surfacing high-similarity but mechanism-blind candidates [27, 42, 71]. Detection therefore varies by modality—source-attribution checks for text, unit/symmetry violation tests for symbolic forms, conservation-law residual and OOD calibration for SciML, and probe-based property alignment for latent models—and reliability requires verifiers matched to each modality's failure form.

8.2. AI Co-Scientist Systems

Recent work extends multi-agent coordination toward full *co-scientist* architectures that generate, critique, rank, and prioritize hypotheses across sustained research cycles [6, 83–87]. Following the operational/epistemic autonomy distinction of §5.2, we reserve the term *co-scientist* for systems that combine generative proposal, structured critique, calibrated prioritization, tool use, and sustained state across research cycles—i.e., that exhibit at least partial epistemic autonomy—and describe systems lacking one or more of these capabilities as *generative assistants*. Unlike standalone generators, these systems coordinate proposer–critic–planner roles and integrate tools; however, most implementations remain research prototypes, and analyses of AI-assisted mathematical discovery further highlight epistemic opacity and validation gaps [107]. Key challenges include coordination stability (critique loops can amplify shared misconceptions) [84, 85], hypothesis ranking without calibrated ground truth [2, 88], and underdeveloped coupling between LLM-driven ideation and formal acquisition policies [9, 64]. Progress requires explicit separation between generative proposal and probabilistic prioritization [9, 21], measurable critique effectiveness [6, 57, 84], and integration with executable experimental platforms for closed-loop validation [12, 37, 39].

8.3. Transfer, Verification, and Governance

Scientific transfer demands preservation of causal and physical structure [1, 105, 108]; physics-informed models fail when priors are partial [18, 19, 35], neural operators degrade under regime shifts [65, 66, 79], causal discovery remains hard to scale [58–60], and cross-modal translation is fragile without standardized ontologies [72]. For instance, ESM-style protein language models transfer to AlphaFold-type structure prediction because sequence/structure invariants are shared [27, 42], whereas neural operators trained on Burgers' equation degrade on Navier–Stokes under regime shift [65, 79], and fluid-dynamics symbolic regressions fail on plasma without explicit symmetry priors [4, 52]. Output volume grows faster than validation capacity [2, 6]; verification should be layered into CI-style regression tests [97], calibrated simulation filters [18, 65], and uncertainty-aware allocation of high-fidelity experiments [22, 64], complemented by formal methods where applicable [109–111]. Governance mechanisms are rarely encoded as executable constraints [72, 99]; progress requires enforceable policy gates at module boundaries, interoperable schemas with replayable logs [72, 97], and measurable compliance metrics [98, 99].

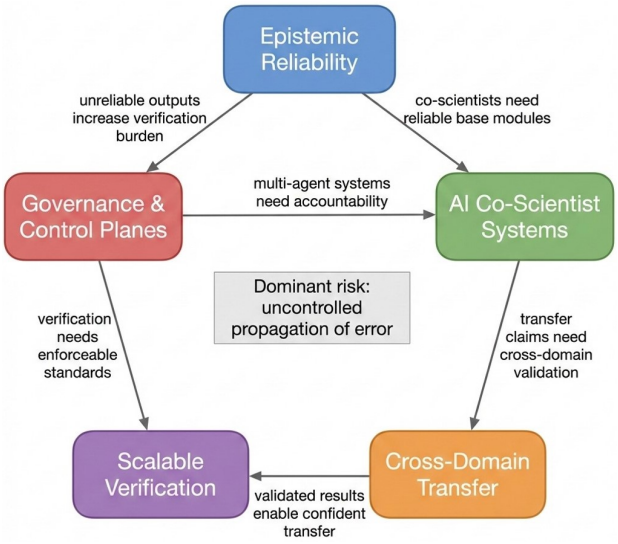

Fig. 8. Interdependencies among open challenges. Arrows indicate that progress on the source challenge is likely necessary for addressing the target. The central annotation reflects the shared risk identified across all frontiers.

Synthesis

Across frontiers—including epistemic reliability, AI co-scientist coordination, cross-domain transfer, scalable verification, and governance—the dominant risk is uncontrolled propagation of error. Addressing these challenges requires evidence-linked generation [57], probabilistic prioritization [9, 64], mechanism-aware modeling [19, 105], layered validation [18, 97], interoperable infrastructure [72], and structured human–AI collaboration [95, 96]. The central research agenda is transforming generative capability into verifiable, controllable, and experimentally grounded scientific co-reasoning (Fig. 8).

9. Conclusion

This survey examined how AI systems support scientific discovery through idea generation—from proposing hypotheses and constructing structured representations to selecting experiments and updating models from results within increasingly integrated discovery workflows. We showed how these functions are implemented across diverse paradigms: language-based systems, symbolic and graph-structured methods, probabilistic experimental design, simulation-grounded learning, and closed-loop laboratory platforms. Each paradigm brings distinct strengths but also characteristic failure modes and engineering requirements.

Our two-axis taxonomy, organized by generation modality (what is produced) and embodiment level (how closely it connects to experiments), provides a framework for comparing systems that would otherwise appear unrelated. In practice, modern discovery platforms increasingly combine complementary modules, making interface design and fault isolation as important as the algorithms themselves. AI co-scientist architectures, in which multi-agent systems generate, critique, and prioritize hypotheses, are an active research direction; however, epistemic reliability and scalable verification remain central open problems. Evaluation must extend beyond efficiency and reliability to include causal validity, discovery novelty, and mechanistic depth. Responsible deployment requires auditable controls, reproducible execution, and clear accountability at every module boundary.

CONFLICT OF INTEREST

Conflicts of Interest: The authors declare no conflicts of interest.

REFERENCES

[1] H. Wang, T. Fu, Y. Du, W. Gao, K. Huang, Z. Liu, P. Chandak, S. Liu, P. Van Katwyk, A. Deac, *et al.*, "Scientific discovery in the age of artificial intelligence," *Nature*, vol. 620, no. 7972, pp. 47–60, 2023.

[2] Y. Zhang, X. Chen, B. Jin, S. Wang, S. Ji, W. Wang, and J. Han, "A comprehensive survey of scientific large language models and their applications in scientific discovery," in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 8783–8817, 2024.

[3] H. Kitano, "Nobel turing challenge: creating the engine for scientific discovery," *NPJ systems biology and applications*, vol. 7, no. 1, p. 29, 2021.

[4] M. Schmidt and H. Lipson, "Distilling free-form natural laws from experimental data," *science*, vol. 324, no. 5923, pp. 81–85, 2009.

[5] J. L. Watson, D. Juergens, N. R. Bennett, B. L. Trippe, J. Yim, H. E. Eisenach, W. Ahern, A. J. Borst, R. J. Ragotte, L. F. Milles, *et al.*, "De novo design of protein structure and function with rfdiffusion," *Nature*, vol. 620, no. 7976, pp. 1089–1100, 2023.

[6] C. Lu, C. Lu, R. T. Lange, J. Foerster, J. Clune, and D. Ha, "The ai scientist: Towards fully automated open-ended scientific discovery," *arXiv preprint arXiv:2408.06292*, 2024.

[7] P. Langley, *Scientific discovery: Computational explorations of the creative processes*. MIT press, 1987.

[8] J. Gottweis, W.-H. Weng, A. Daryin, T. Tu, A. Palepu, P. Sirkovic, A. Myaskovsky, F. Weissenberger, K. Rong, R. Tanno, *et al.*, "Towards an ai co-scientist," *arXiv preprint arXiv:2502.18864*, 2025.

[9] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. De Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2015.

[10] T. Zheng, Z. Deng, H. T. Tsang, W. Wang, J. Bai, Z. Wang, and Y. Song, "From automation to autonomy: A survey on large language models in scientific discovery," in *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 17744–17761, 2025.

[11] C. Si, D. Yang, and T. Hashimoto, "Can llms generate novel research ideas? a large-scale human study with 100+ nlp researchers," *arXiv preprint arXiv:2409.04109*, 2024.

[12] M. Abolhasani and E. Kumacheva, "The rise of self-driving labs in chemical and materials sciences," *Nature Synthesis*, vol. 2, no. 6, pp. 483–492, 2023.

[13] A. D. Adesiji, J. Wang, C.-S. Kuo, and K. A. Brown, "Benchmarking self-driving labs," *Digital Discovery*, 2026.

[14] D. R. Swanson, "Fish oil, raynaud's syndrome, and undiscovered public knowledge," *Perspectives in biology and medicine*, vol. 30, no. 1, pp. 7–18, 1986.

[15] D. R. Swanson, "Undiscovered public knowledge," *The Library Quarterly*, vol. 56, no. 2, pp. 103–118, 1986.

[16] N. R. Smalheiser and D. R. Swanson, "Using arrowsmith: a computer-assisted approach to formulating and assessing scientific hypotheses," *Computer methods and programs in biomedicine*, vol. 57, no. 3, pp. 149–153, 1998.

[17] R. D. King, K. E. Whelan, F. M. Jones, P. G. Reiser, C. H. Bryant, S. H. Muggleton, D. B. Kell, and S. G. Oliver, "Functional genomic hypothesis generation and experimentation by a robot scientist," *Nature*, vol. 427, no. 6971, pp. 247–252, 2004.

[18] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational physics*, vol. 378, pp. 686–707, 2019.

[19] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," *Nature Reviews Physics*, vol. 3, no. 6, pp. 422–440, 2021.

[20] P. I. Frazier, "A tutorial on bayesian optimization," *arXiv preprint arXiv:1807.02811*, 2018.

[21] R. Garnett, *Bayesian optimization*. Cambridge University Press, 2023.

[22] J. M. Hernández-Lobato, M. W. Hoffman, and Z. Ghahramani, "Predictive entropy search for efficient global optimization of black-box functions," *Advances in neural information processing systems*, vol. 27, 2014.

[23] J. K. Christopher, S. Baek, and N. Fioretto, "Constrained synthesis with projected diffusion models," *Advances in Neural Information Processing Systems*, vol. 37, pp. 89307–89333, 2024.

[24] X. Zhou, X. Cheng, Y. Yang, Y. Bao, L. Wang, and Q. Gu, "Decompopt: Controllable and decomposed diffusion models for structure-based molecular optimization," *arXiv preprint arXiv:2403.13829*, 2024.

[25] S. Yang, S. Batzner, R. Gao, M. Aykol, A. Gaunt, B. McMorrow, D. Rezende, D. Schuurmans, I. Mordatch, and E. D. Cubuk, "Generative hierarchical materials search," *Advances in Neural Information Processing Systems*, vol. 37, pp. 38799–38819, 2024.

[26] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, *et al.*, "Highly accurate protein structure prediction with alphafold," *nature*, vol. 596, no. 7873, pp. 583–589, 2021.

[27] A. Rives, J. Meier, T. Sercu, S. Goyal, Z. Lin, J. Liu, D. Guo, M. Ott, C. L. Zitnick, J. Ma, *et al.*, "Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences," *Proceedings of the national academy of sciences*, vol. 118, no. 15, p. e2016239118, 2021.

[28] P. W. Sprague, "Automated chemical hypothesis generation and database searching with catalyst®," *Perspectives in drug discovery and design*, vol. 3, no. 1, pp. 1–20, 1995.

[29] R. D. King, J. Rowland, S. G. Oliver, M. Young, W. Aubrey, E. Byrne, M. Liakata, M. Markham, P. Pir, L. N. Soldatova, *et al.*, "The automation of science," *Science*, vol. 324, no. 5923, pp. 85–89, 2009.

[30] M. H. Segler, M. Preuss, and M. P. Waller, "Planning chemical syntheses with deep neural networks and symbolic ai," *Nature*, vol. 555, no. 7698, pp. 604–610, 2018.

[31] C. W. Coley, W. H. Green, and K. F. Jensen, "Machine learning in computer-aided synthesis planning," *Accounts of chemical research*, vol. 51, no. 5, pp. 1281–1289, 2018.

[32] P. Schwaller, T. Laino, T. Gaudin, P. Bolgar, C. A. Hunter, C. Bekas, and A. A. Lee, "Molecular transformer: a model for uncertainty-calibrated chemical reaction prediction," *ACS central science*, vol. 5, no. 9, pp. 1572–1583, 2019.

[33] J. Vamathevan, D. Clark, P. Czodrowski, I. Dunham, E. Ferran, G. Lee, B. Li, A. Madabhushi, P. Shah, M. Spitzer, *et al.*, "Applications of machine learning in drug discovery and development," *Nature reviews Drug discovery*, vol. 18, no. 6, pp. 463–477, 2019.

[34] T. Lookman, P. V. Balachandran, D. Xue, and R. Yuan, "Active learning in materials science with emphasis on adaptive sampling using uncertainties for targeted design," *npj Computational Materials*, vol. 5, no. 1, p. 21, 2019.

[35] J. Willard, X. Jia, S. Xu, M. Steinbach, and V. Kumar, "Integrating scientific knowledge with machine learning for engineering and environmental systems," *ACM Computing Surveys*, vol. 55, no. 4, pp. 1–37, 2022.

[36] B. Settles, "Active learning literature survey," 2009.

[37] B. P. MacLeod, F. G. Parlane, T. D. Morrissey, F. Häse, L. M. Roch, K. E. Dettelbach, R. Moreira, L. P. Yunker, M. B. Rooney, J. R. Deeth, *et al.*, "Self-driving laboratory for accelerated discovery of thin-film materials," *Science Advances*, vol. 6, no. 20, p. eaaz8867, 2020.

[38] B. Burger, P. M. Maffettone, V. V. Gusev, C. M. Aitchison, Y. Bai, X. Wang, X. Li, B. M. Alston, B. Li, R. Clowes, *et al.*, "A mobile robotic chemist," *Nature*, vol. 583, no. 7815, pp. 237–241, 2020.

[39] L. M. Roch, F. Häse, and A. Aspuru-Guzik, "Chemos: An orchestration software to democratize autonomous discovery," 2020.

[40] M. Sim, M. G. Vakili, F. Strieth-Kalthoff, H. Hao, R. J. Hickman, S. Miret, S. Pablo-García, and A. Aspuru-Guzik, "Chemos 2.0: An orchestration architecture for chemical self-driving laboratories," *Matter*, vol. 7, no. 9, pp. 2959–2977, 2024.

[41] R. Evans, M. O'neill, A. Pritzel, N. Antropova, A. Senior, T. Green, A. Žídek, R. Bates, S. Blackwell, J. Yim, *et al.*, "Protein complex prediction with alphafold-multimer," *biorxiv*, pp. 2021–10, 2021.

[42] Z. Lin, H. Akin, R. Rao, B. Hie, Z. Zhu, W. Lu, N. Smetanin, R. Verkuil, O. Kabeli, Y. Shmueli, *et al.*, "Evolutionary-scale prediction of atomic-level protein structure with a language model," *Science*, vol. 379, no. 6637, pp. 1123–1130, 2023.

[43] Y. Du and I. Mordatch, "Implicit generation and modeling with energy based models," *Advances in neural information processing systems*, vol. 32, 2019.

[44] A. Madani, B. Krause, E. R. Greene, S. Subramanian, B. P. Mohr, J. M. Holton, J. L. Olmos Jr, C. Xiong, Z. Z. Sun, R. Socher, *et al.*, "Large language models generate functional protein sequences across diverse families," *Nature biotechnology*, vol. 41, no. 8, pp. 1099–1106, 2023.

[45] G. Liu, J. Xu, T. Luo, and M. Jiang, "Graph diffusion transformers for multi-conditional molecular generation," *Advances in Neural Information Processing Systems*, vol. 37, pp. 8065–8092, 2024.

[46] J. You, R. Ying, X. Ren, W. Hamilton, and J. Leskovec, "Graphrnn: Generating realistic graphs with deep auto-regressive models," in *International conference on machine learning*, pp. 5708–5717, PMLR, 2018.

[47] W. Jin, R. Barzilay, and T. Jaakkola, "Junction tree variational autoencoder for molecular graph generation," in *International conference on machine learning*, pp. 2323–2332, PMLR, 2018.

[48] M. Olivecrona, T. Blaschke, O. Engkvist, and H. Chen, "Molecular de-novo design through deep reinforcement learning," *Journal of cheminformatics*, vol. 9, no. 1, p. 48, 2017.

[49] M. Popova, O. Isayev, and A. Tropsha, "Deep reinforcement learning for de novo drug design," *Science advances*, vol. 4, no. 7, p. eaap7885, 2018.

[50] N. Brown, M. Fiscato, M. H. Segler, and A. C. Vaucher, "Guacamol: benchmarking models for de novo molecular design," *Journal of chemical information and modeling*, vol. 59, no. 3, pp. 1096–1108, 2019.

[51] S.-M. Udrescu and M. Tegmark, "Ai feynman: A physics-inspired method for symbolic regression," *Science advances*, vol. 6, no. 16, p. eaay2631, 2020.

[52] S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proceedings of the national academy of sciences*, vol. 113, no. 15, pp. 3932–3937, 2016.

[53] F. Sun, Y. Liu, J.-X. Wang, and H. Sun, "Symbolic physics learner: Discovering governing equations via monte carlo tree search," *arXiv preprint arXiv:2205.13134*, 2022.

[54] P. Lemos, N. Jeffrey, M. Cranmer, S. Ho, and P. Battaglia, "Rediscovering orbital mechanics with machine learning," *Machine Learning: Science and Technology*, vol. 4, no. 4, p. 045002, 2023.

[55] P. Shojaee, K. Meidani, S. Gupta, A. B. Farimani, and C. K. Reddy, "Llm-sr: Scientific equation discovery via programming with large language models," *arXiv preprint arXiv:2404.18400*, 2024.

[56] M. Cranmer, "Interpretable machine learning for science with pysr and symbolicregression. jl," *arXiv preprint arXiv:2305.01582*, 2023.

[57] J. Baek, S. K. Jauhar, S. Cucerzan, and S. J. Hwang, "Researchagent: Iterative research idea generation over scientific literature with large language models," in *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 6709–6738, 2025.

[58] A. Ghassami, S. Salehkaleybar, N. Kiyavash, and E. Bareinboim, "Budgeted experiment design for causal structure learning," in *International Conference on Machine Learning*, pp. 1724–1733, PMLR, 2018.

[59] P. Brouillard, S. Lachapelle, A. Lacoste, S. Lacoste-Julien, and A. Drouin, "Differentiable causal discovery from interventional data," *Advances in Neural Information Processing Systems*, vol. 33, pp. 21865–21877, 2020.

[60] J. M. Mooij, S. Magliacane, and T. Claassen, "Joint causal inference from multiple contexts," *Journal of machine learning research*, vol. 21, no. 99, pp. 1–108, 2020.

[61] E. Parisotto, A.-r. Mohamed, R. Singh, L. Li, D. Zhou, and P. Kohli, "Neuro-symbolic program synthesis," *arXiv preprint arXiv:1611.01855*, 2016.

[62] K. Ellis, C. Wong, M. Nye, M. Sablé-Meyer, L. Morales, L. Hewitt, L. Cary, A. Solar-Lezama, and J. B. Tenenbaum, "Dreamcoder: Bootstrapping inductive program synthesis with wake-sleep library learning," in *Proceedings of the 42nd acm sigplan international conference on programming language design and implementation*, pp. 835–850, 2021.

[63] T. D. Kulkarni, P. Kohli, J. B. Tenenbaum, and V. Mansinghka, "Picture: A probabilistic programming language for scene perception," in *Proceedings of the ieee conference on computer vision and pattern recognition*, pp. 4390–4399, 2015.

[64] J. P. Folch, C. Tsay, R. Lee, B. Shafei, W. Ormaniec, A. Krause, M. van der Wilk, R. Misener, and M. Mutny, "Transition constrained bayesian optimization via markov decision processes," *Advances in Neural Information Processing Systems*, vol. 37, pp. 88194–88235, 2024.

[65] J. Brandstetter, D. Worrall, and M. Welling, "Message passing neural pde solvers," *arXiv preprint arXiv:2202.03376*, 2022.

[66] D. Kochkov, J. A. Smith, A. Alieva, Q. Wang, M. P. Brenner, and S. Hoyer, "Machine learning–accelerated computational fluid dynamics," *Proceedings of the National Academy of Sciences*, vol. 118, no. 21, p. e2101784118, 2021.

[67] A. Jain, K. A. Persson, and G. Ceder, "Research update: The materials genome initiative: Data sharing and the impact of collaborative ab initio databases," *Apl Materials*, vol. 4, no. 5, 2016.

[68] K. Rajan, "Materials informatics," *Materials Today*, vol. 8, no. 10, pp. 38–45, 2005.

[69] D. C. Lonie and E. Zurek, "Xtalopt: An open-source evolutionary algorithm for crystal structure prediction," *Computer Physics Communications*, vol. 182, no. 2, pp. 372–387, 2011.

[70] D. Polykovskiy, A. Zhebrak, B. Sanchez-Lengeling, S. Golovanov, O. Tatanov, S. Belyaev, R. Kurbanov, A. Artamonov, V. Aladinskiy, M. Veselov, *et al.*, "Molecular sets (moses): a benchmarking platform for molecular generation models," *Frontiers in pharmacology*, vol. 11, p. 565644, 2020.

[71] C. Ying, T. Cai, S. Luo, S. Zheng, G. Ke, D. He, Y. Shen, and T.-Y. Liu, "Do transformers really perform badly for graph representation?," *Advances in neural information processing systems*, vol. 34, pp. 28877–28888, 2021.

[72] R. B. Canty, B. A. Koscher, M. A. McDonald, and K. F. Jensen, "Integrating autonomy into automated research platforms," *Digital Discovery*, vol. 2, no. 5, pp. 1259–1268, 2023.

[73] M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich, "A review of relational machine learning for knowledge graphs," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 11–33, 2015.

[74] D. S. Himmelstein, A. Lizee, C. Hessler, L. Brueggeman, S. L. Chen, D. Hadley, A. Green, P. Khankhanian, and S. E. Baranzini, "Systematic integration of biomedical knowledge prioritizes drugs for repurposing," *elife*, vol. 6, p. e26726, 2017.

[75] S. Ji, S. Pan, E. Cambria, P. Marttinen, and P. S. Yu, "A survey on knowledge graphs: Representation, acquisition, and applications," *IEEE transactions on neural networks and learning systems*, vol. 33, no. 2, pp. 494–514, 2021.

[76] P. Spirtes, C. N. Glymour, and R. Scheines, *Causation, Prediction, and Search*. Cambridge, MA: MIT Press, 2nd ed., 2000.

[77] S. Greydanus, M. Dzamba, and J. Yosinski, "Hamiltonian neural networks," *Advances in neural information processing systems*, vol. 32, 2019.

[78] Z. Liu and M. Tegmark, "Machine learning conservation laws from trajectories," *Phys. Rev. Lett.*, vol. 126, p. 180604, May 2021.

[79] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Fourier neural operator for parametric partial differential equations," *arXiv preprint arXiv:2010.08895*, 2020.

[80] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, *et al.*, "Relational inductive biases, deep learning, and graph networks," *arXiv preprint arXiv:1806.01261*, 2018.

[81] E. Bengio, M. Jain, M. Korablyov, D. Precup, and Y. Bengio, "Flow network based generative models for non-iterative diverse candidate generation," *Advances in neural information processing systems*, vol. 34, pp. 27381–27394, 2021.

[82] M. Tantakoun, C. Muise, and X. Zhu, "Llms as planning formalizers: A survey for leveraging large language models to construct automated planning models," in *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 25167–25188, 2025.

[83] A. Ghafarollahi and M. J. Buehler, "Sciagents: automating scientific discovery through bioinspired multi-agent intelligent graph reasoning," *Advanced Materials*, vol. 37, no. 22, p. 2413523, 2025.

[84] X. Bo, Z. Zhang, Q. Dai, X. Feng, L. Wang, R. Li, X. Chen, and J.-R. Wen, "Reflective multi-agent collaboration based on large language models," *Advances in Neural Information Processing Systems*, vol. 37, pp. 138595–138631, 2024.

[85] H. Su, R. Chen, S. Tang, Z. Yin, X. Zheng, J. Li, B. Qi, Q. Wu, H. Li, W. Ouyang, *et al.*, "Many heads are better than one: Improved scientific idea generation by a llm-based multi-agent system," in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 28201–28240, 2025.

[86] Z. Chen, S. Chen, Y. Ning, Q. Zhang, B. Wang, B. Yu, Y. Li, Z. Liao, C. Wei, Z. Lu, *et al.*, "Scienceagentbench: Toward rigorous assessment of language agents for data-driven scientific discovery," *arXiv preprint arXiv:2410.05080*, 2024.

[87] S. Schmidgall, Y. Su, Z. Wang, X. Sun, J. Wu, X. Yu, J. Liu, M. Moor, Z. Liu, and E. Barsoum, "Agent laboratory: Using llm agents as research assistants," *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 5977–6043, 2025.

[88] X. Wang, Z. Hu, P. Lu, Y. Zhu, J. Zhang, S. Subramaniam, A. R. Loomba, S. Zhang, Y. Sun, and W. Wang, "Scibench: Evaluating college-level scientific problem-solving abilities of large language models," *arXiv preprint arXiv:2307.10635*, 2023.

[89] Y. Ma, Z. Gou, J. Hao, R. Xu, S. Wang, L. Pan, Y. Yang, Y. Cao, and A. Sun, "Sciagent: Tool-augmented language models for scientific reasoning," in *Proceedings of the 2024 conference on empirical methods in natural language processing*, pp. 15701–15736, 2024.

[90] D. Rein, B. L. Hou, A. C. Stickland, J. Petty, R. Y. Pang, J. Dirani, J. Michael, and S. R. Bowman, "Gpqa: A graduate-level google-proof q&a benchmark," in *First conference on language modeling*, 2024.

[91] B. Andrew and S. Richard S, *Reinforcement learning: an introduction*. The MIT Press, 2018.

[92] T. Dai, S. Vijayakrishnan, F. T. Szczypiński, J.-F. Ayme, E. Simaei, T. Fellowes, R. Clowes, L. Kotopanov, C. E. Shields, Z. Zhou, *et al.*, "Autonomous mobile robots for exploratory synthetic chemistry," *Nature*, vol. 635, no. 8040, pp. 890–897, 2024.

[93] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, *et al.*, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," in *Conference on Robot Learning*, pp. 2165–2183, PMLR, 2023.

[94] A. Vriza, H. Chan, and J. Xu, "Self-driving laboratory for polymer electronics," *Chemistry of Materials*, vol. 35, no. 8, pp. 3046–3056, 2023.

[95] H. Hysmith, E. Foadian, S. P. Padhy, S. V. Kalinin, R. G. Moore, O. S. Ovchinnikova, and M. Ahmadi, "The future of self-driving laboratories: from human in the loop interactive ai to gamification," *Digital Discovery*, vol. 3, no. 4, pp. 621–636, 2024.

[96] D. Dellermann, P. Ebel, M. Söllner, and J. M. Leimeister, "Hybrid intelligence," *Business & Information Systems Engineering*, vol. 61, p. 637–643, Mar. 2019.

[97] J. Pineau, P. Vincent-Lamarre, K. Sinha, V. Larivière, A. Beygelzimer, F. d'Alché Buc, E. Fox, and H. Larochelle, "Improving reproducibility in machine learning research (a report from the neurips 2019 reproducibility program)," *Journal of machine learning research*, vol. 22, no. 164, pp. 1–20, 2021.

[98] D. B. Resnik and M. Hosseini, "The ethics of using artificial intelligence in scientific research: new guidance needed for a new tool," *AI and Ethics*, vol. 5, no. 2, pp. 1499–1521, 2025.

[99] X. Tang, Q. Jin, K. Zhu, T. Yuan, Y. Zhang, W. Zhou, M. Qu, Y. Zhao, J. Tang, Z. Zhang, *et al.*, "Risks of ai scientists: prioritizing safeguarding over autonomy," *Nature Communications*, vol. 16, no. 1, p. 8317, 2025.

[100] S. Daulton, M. Balandat, and E. Bakshy, "Differentiable expected hypervolume improvement for parallel multi-objective bayesian optimization," *Advances in neural information processing systems*, vol. 33, pp. 9851–9864, 2020.

[101] K. Chaloner and I. Verdinelli, "Bayesian experimental design: A review," *Statistical science*, pp. 273–304, 1995.

[102] S. X. Leong, C. E. Griesbach, R. Zhang, K. Darvish, Y. Zhao, A. Mandal, Y. Zou, H. Hao, V. Bernales, and A. Aspuru-Guzik, "Steering towards safe self-driving laboratories," *Nature Reviews Chemistry*, vol. 9, no. 10, pp. 707–722, 2025.

[103] J. Peters, D. Janzing, and B. Scholkopf, *Elements of causal inference: foundations and learning algorithms*. MIT press, 2017.

[104] J. Pearl, *Causality*. Cambridge university press, 2009.

[105] J. Peters, P. Bühlmann, and N. Meinshausen, "Causal inference by using invariant prediction: identification and confidence intervals," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 78, no. 5, pp. 947–1012, 2016.

[106] R. Iten, T. Metger, H. Wilming, L. Del Rio, and R. Renner, "Discovering physical concepts with neural networks," *Physical review letters*, vol. 124, no. 1, p. 010508, 2020.

[107] E. Duede and K. Davey, "Apriori knowledge in an era of computational opacity: The role of ai in mathematical discovery," *Philosophy of Science*, pp. 1–11, 2024.

[108] B. Schölkopf, F. Locatello, S. Bauer, N. R. Ke, N. Kalchbrenner, A. Goyal, and Y. Bengio, "Toward causal representation learning," *Proceedings of the IEEE*, vol. 109, no. 5, pp. 612–634, 2021.

[109] S. Polu and I. Sutskever, "Generative language modeling for automated theorem proving," *arXiv preprint arXiv:2009.03393*, 2020.

[110] K. Yang, A. Swope, A. Gu, R. Chalamala, P. Song, S. Yu, S. Godil, R. J. Prenger, and A. Anandkumar, "Leandojo: Theorem proving with retrieval-augmented language models," *Advances in Neural Information Processing Systems*, vol. 36, pp. 21573–21612, 2023.

[111] A. Q. Jiang, S. Welleck, J. P. Zhou, W. Li, J. Liu, M. Jamnik, T. Lacroix, Y. Wu, and G. Lample, "Draft, sketch, and prove: Guiding formal theorem provers with informal proofs," *arXiv preprint arXiv:2210.12283*, 2022.

JIN-XIA HUANG received a B.S. degree in physics from Jilin University, China, in 1991, an M.S. degree in computer science from KAIST, Republic of Korea, in 2001, and a Ph.D. degree in computer science from Jeonbuk National University, Korea, in 2018. From 1994 to 1997, she worked as an Engineer at Yanbian University of Science and Technology (YUST), China, and from 2001 to 2003 as a Researcher at Microsoft Research Asia (MSRA), China. Since 2008, she has been with the Electronics and Telecommunications Research Institute (ETRI), Korea, where she is currently a Principal Researcher in the Language Intelligent Research Section. Since March 2024, she has also served as an Associate Professor in the AI Major at UST ETRI School. Her research has focused on natural language processing, including dialogue systems and intelligent tutoring agents, and her current research interests include intelligent multi-agent orchestration and AI-driven scientific discovery.

WOOJIN LEE received the B.S. and M.S. degrees from Konkuk University, Seoul, South Korea, in 2023 and 2025, respectively. He is currently a Researcher with the Electronics and Telecommunications Research Institute (ETRI), Daejeon, Republic of Korea. His research interests include large language models, interpretability, and autonomous agents.

YOONKYU WOO received the B.S. degree in Industrial Management Engineering from Korea University, Seoul, Republic of Korea, in 2024. He is currently a graduate student with the University of Science and Technology (UST), Republic of Korea. His research interests include natural language generation, multi-agent systems, and reinforcement learning.