Can We Predict Before Executing Machine Learning Agents?

Jingsheng Zheng[†‡], **Jintian Zhang**[†‡], **Yujie Luo**[†‡], **Yuren Mao**[†], **Yunjun Gao**[†], **Lun Du**[§‡], **Huajun Chen**[†‡], **Ningyu Zhang**[†‡*]
[†]Zhejiang University [§]Ant Group
[‡]Zhejiang University - Ant Group Joint Laboratory of Knowledge Graph
zhengjohnson0@gmail.com, zhangningyu@zju.edu.cn

Abstract

Autonomous machine learning agents have revolutionized scientific discovery, yet they remain constrained by a Generate-Execute-Feedback paradigm. Previous approaches suffer from a severe Execution Bottleneck, as hypothesis evaluation relies strictly on expensive physical execution. To bypass these physical constraints, we internalize execution priors to substitute costly runtime checks with instantaneous predictive reasoning, drawing inspiration from World Models. In this work, we formalize the task of Data-centric Solution Preference and construct a comprehensive corpus of 18,438 pairwise comparisons. We demonstrate that LLMs exhibit significant predictive capabilities when primed with a Verified Data Analysis Report, achieving 61.5% accuracy and robust confidence calibration. Finally, we instantiate this framework in FORE-AGENT, an agent that employs a Predict-then-Verify loop, achieving a 6x acceleration in convergence while surpassing execution-based baselines by +6%. Our code and dataset are publicly available at https://github.com/zjunlp/predict-before-execute.

1 Introduction

Autonomous machine learning agents have emerged as powerful tools for solving complex challenges in scientific discovery (Zhang et al., 2025d; Chen et al., 2025b). Mainstream frameworks (Jiang et al., 2025; Ou et al., 2025) typically rely on an iterative *"Generate-Execute-Feedback"* loop where the system refines code based on runtime output (Yao et al., 2023). However, this paradigm suffers from a severe **Execution Bottleneck** as physical execution is computationally expensive and slow, often consuming up to 9 hours per run in benchmarks like MLE-Bench (Chan et al., 2025). Increasingly, recent research has identified this latency issue and sought to mitigate the computational overhead through heuristic pruning strategies (Trirat et al., 2025; Kulibaba et al., 2025).

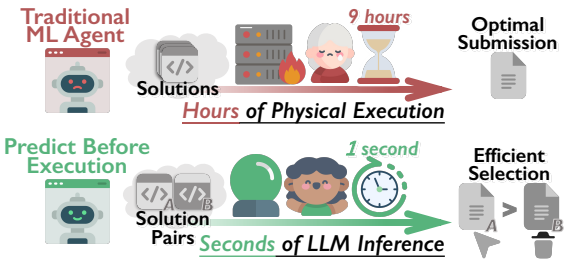


Figure 1: **From Execution to Inference.** Traditional ML agents improve through costly execution and external feedback, incurring substantial latency. Our work investigates whether superior data-grounded solutions can be identified before execution by leveraging "Implicit Execution Priors".

To fundamentally bypass these physical constraints, the concept of **World Models** (Ding et al., 2025) offers a transformative alternative (Figure 1). Originating from reinforcement learning, world models enable agents to simulate environmental dynamics and evaluate actions via internal predictions rather than external trials (Ha and Schmidhuber, 2018; Hafner et al., 2024). Recent advancements have extended this capability to the code domain by predicting execution outputs directly (Li et al., 2025c; team et al., 2025). Motivated by this, we explore whether agents can internalize execution priors, substituting costly runtime checks with instantaneous predictive reasoning. The potential to replace **9 hours** of physical latency with **1 second** of neural speed brings us to a fundamental question: ***Can we compress hours of physical execution into seconds of logical inference?***

To answer this question, we formalize the task of **Data-centric Solution Preference**, where the model must predict the relative performance of two algorithmic solutions given a data analysis report, through reasoning without physical ex-

*Corresponding Author.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 3941–3974

ecution. To rigorously evaluate this, we construct a large-scale corpus comprising 18,438 pairwise comparisons. Our main experiments yield strong evidence: **LLMs exhibit significant predictive capabilities**, with DeepSeek-V3.2-Thinking achieving 61.5% accuracy, outperforming both random guessing (50.0%) and complexity-based heuristics (50.8%). Further analysis reveals that reasoning-optimized architectures transcend complexity heuristics through genuine data reasoning, yielding well-calibrated confidence that ensures the reliability of implicit evaluation. Finally, we integrate this predictive mechanism into **FOREAGENT**, an agent that employs a *Predict-then-Verify* loop to decouple exploration from execution, expanding the search space by 3.2× and achieving a 6× acceleration while delivering a +6% performance gain over standard baselines.

In summary, our contributions are three-fold:

- We define the novel task of **Data-centric Solution Preference** and construct a comprehensive corpus of 18,438 pairs, answering the titular question that **LLMs Exhibit Significant Predictive Capabilities.**
- We operationalize this framework in **FOREAGENT**, an agent that employs a *Predict-then-Verify* loop to decouple exploration from execution, enabling it to expand the search space by **3.2×** and achieve a **6×** acceleration and a **+6%** performance gain over the baseline.
- We contribute a large-scale **Open-Source Dataset** of verified execution trajectories, serving as a foundational corpus for training scalable Reward Models to accelerate reinforcement learning rollouts and optimization across diverse agent frameworks.

2 Background

2.1 The Paradigm of Autonomous ML Agents

An autonomous Machine Learning (ML) task aims to generate an optimal solution code C^* from the code space C that maximizes a metric M on a dataset $\mathcal{D}$, given a natural language instruction I (see Appendix Figure 13):

$$C^* = \arg\max_{C} M(I, C, \mathcal{D}) \quad (1)$$

Current agents typically follow a *Generate-Execute-Feedback* paradigm (Zhu et al., 2025b).

Domain	Paradigms	# Tsk	# Sols	# Pairs
CV	Classification, Segmentation, Generation, Restoration	9	289	5,952
NLP	Classification, Matching, QA, Sequence Labeling, Ranking	8	303	6,682
Data Science	Regression, Time-Series, Audio, Tabular, Grading	9	303	5,804
Total	*26 Distinct Tasks across 3 Domains*	**26**	**895**	**18,438**

Table 1: Statistics of the Preference Corpus. We aggregate 26 tasks into three primary domains, ensuring a balanced distribution of ∼6,000 pairs each. (See Appendix B.1 for granular breakdown).

For instance, **AIDE** (Jiang et al., 2025) organizes solution exploration as a tree search process involving sequential drafting, debugging, and iterative improvement via execution feedback. Building upon this, **AutoMind** (Ou et al., 2025) integrates a curated expert knowledge base with a self-adaptive coding strategy to tackle more intricate problems (see Appendix A for details).

2.2 The Execution Bottleneck

The primary constraint in current agents is the reliance on physical execution for feedback. Formally, the update of solution C_{t+1} depends on the result R_t from executing on dataset $\mathcal{D}$:

$$C_{t+1} \leftarrow \text{Agent}(I, C_t, \underbrace{\text{Execute}(C_t, \mathcal{D})}_{R_t}) \quad (2)$$

Unlike symbolic tasks with instantaneous verification, training deep learning models involves heavy computation, frequently leading to timeout failures (Chan et al., 2025). This efficiency gap **necessitates compressing hours of physical execution into seconds of logical inference**, mirroring how human experts utilize mental simulation to discard sub-optimal algorithms prior to implementation.

2.3 Implicit World Modeling in Data Domains

We investigate whether LLMs can function as an *Implicit World Model* (Ha and Schmidhuber, 2018; Hafner et al., 2024). While recent works explore this direction across diverse symbolic and interactive domains (Li et al., 2025e; team et al., 2025; Just et al., 2024), our **Data-centric Solution Preference** task is distinct: unlike tracking explicit states, the model must anticipate the invisible coupling

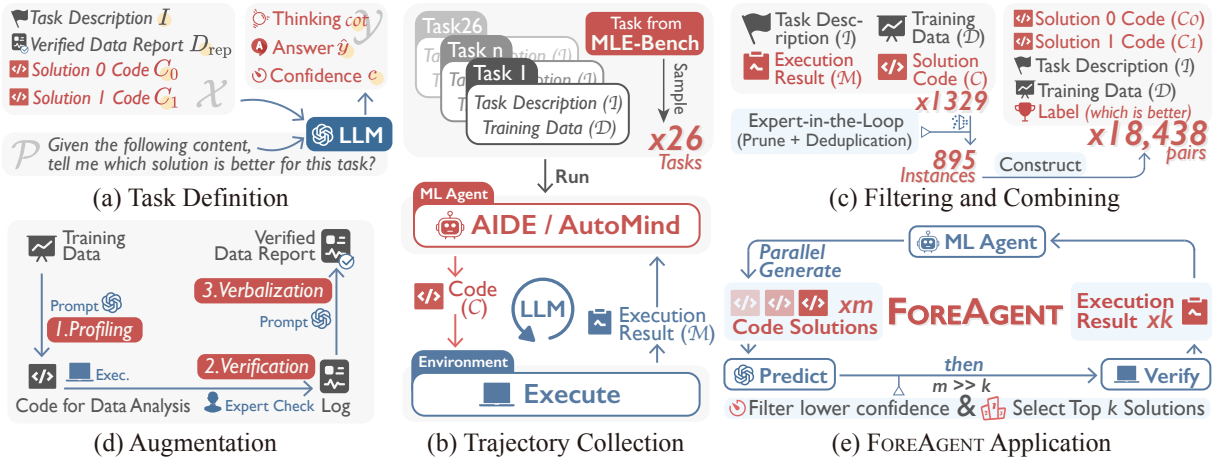


Figure 2: **Overview of the Framework. (a) Task Definition:** The *Data-centric Solution Preference* task predicts solution superiority and confidence via latent reasoning. **(b-c) Data Curation:** We collect and filter real-world agent trajectories to construct the *Preference Corpus*. **(d) Augmentation:** Inputs are augmented with *Verified Data Reports* via a "Profile-Verify-Verbalize" pipeline. **(e) FOREAGENT Application:** The model serves as a filter within the *Predict-then-Verify* loop, predicting preference *before* physical execution to prune candidates.

of algorithmic logic and stochastic data. Thus, we formulate the problem as a *Pairwise Preference* task (Shen et al., 2024), determining the superior solution purely via reasoning to identify promising candidates prior to execution.

3 Preference Corpus Curation

This section details the curation of our preference corpus. We begin by formalizing the task to clarify the data requirements, followed by the collection and augmentation processes.

3.1 Task Definition

We model the data-centric task as a pairwise selection task: given a task description, a data report, and two candidate solutions, the objective is to identify the superior solution and estimate a confidence score (Figure 2(a)). Formally, the input $\mathcal{X}$ is:

$$\mathcal{X} = (I, D_{rep}, \{C_0, C_1\}, \mathcal{P}) \tag{3}$$

where I, D_{rep}, $\{C_0, C_1\}$, and $\mathcal{P}$ denote the task, data report, code pair, and system prompt, respectively. The output $\mathcal{Y}$ is defined as:

$$\mathcal{Y} = \{(cot, \hat{y}, c) \mid cot,\ \hat{y} \in \{0, 1\},\ c \in [0, 1.0]\} \tag{4}$$

consisting of the reasoning cot, predicted winner $\hat{y}$, and confidence c, which serves as the gating threshold in Section 6.

3.2 Source and Scope

To instantiate the task inputs defined above, we construct a large-scale corpus derived from the real-world execution trajectories of two ML agents, **AIDE** (Jiang et al., 2025) and **AutoMind** (Ou et al., 2025), operating on **MLE-bench** (Chan et al., 2025) platform (Figure 2(b)). Powered by DeepSeek-V3.1 (DeepSeek-AI, 2025b) and o3-mini (OpenAI, 2025a), these agents generate **1,329** valid solutions across **26** diverse tasks (Table 1). Unlike synthetic snippets, these candidates represent *complete ML workflows* that are entirely generated by agents and absent from any pre-training corpora, heavily incorporating logically incomplete but executable intermediate states to model noisy real-world exploration (see Appendix B.4). Therefore, identifying the superior solution requires evaluating *how well an algorithm fits the specific data characteristics*, rather than merely checking for code syntax.

3.3 Dataset Curation and Instantiation

For rigorous evaluation, we implement an **Expert-in-the-Loop** pipeline to prune raw trajectories into **895** high-quality instances. This process involves deduplication, taxonomy tagging, and expert sampling to cap dominant methods and ensure algorithmic diversity. Next, we instantiate the dataset by exhaustively generating pairwise combinations from this curated corpus. We apply strict filtering to discard ambiguous pairs and balance the ground-truth winner's position to mitigate position bias (Shi et al., 2024). This yields a final dataset of **18,438 comparisons** (Figure 2(c)), utilizing **micro-averaged accuracy** as the primary metric.

3.4 Input Augmentation: The Verified Data Analysis Report

To address LLMs' numerical limitations (Davies et al., 2025; Li et al., 2025b) and context constraints preventing direct data ingestion, we augment inputs with a **Verified Data Analysis Report** that transforms raw statistics into semantic narratives (Rytting and Wingate, 2021; Zhang et al., 2025a). To guarantee factual grounding and prevent hallucination, we implement a strict protocol (Figure 2(d)) consisting of three concrete steps: (1) **Code Generation**: GPT-5.1 (OpenAI, 2025b) generates a Python script to profile raw data with labels masked. For example, it writes "`print(df['target'].value_counts())`" to inspect the target distribution. (2) **Execution and Verification**: The script runs in a sandbox to produce standard output. A human expert performs a strict pass or fail validity check to ensure the log is free of runtime errors. For instance, the execution log returns a raw fact like "`Target Distribution: 0: 0.915, 1: 0.085`". (3) **Verbalization**: GPT-5.1 reads this execution log and translates it into a semantic insight. Following the previous example, it produces the final report: "Data Imbalance Warning: Severe class imbalance (Pos: 8.5%). Implication: Accuracy is not a suitable metric; consider using F1-score." This process ensures reliable semantic grounding for the task (see case in Appendix Figure 12).

4 Main Experiments

4.1 Experimental Setup

Models and Inference Configuration. We evaluate two state-of-the-art models: **DeepSeek-V3.2-Thinking** (DeepSeek-AI, 2025b) and **GPT-5.1** (gpt-5.1-2025-11-13) (OpenAI, 2025b) with reasoning instructions (Wei et al., 2023; Kojima et al., 2023), adhering to the task in Section 3.1. Following provider guidelines (DeepSeek-AI, 2024), we set the temperature $\tau = 1.0$ for both models as the recommended default for data analysis.

Metrics and Baselines. The primary metric is **Micro-Averaged Accuracy** across 18,438 pairwise comparisons. We benchmark against two baselines: (1) **Random Guess (50.0%)**; (2) **Complexity Heuristic (50.8%)**: A rule-based baseline that assumes "complex is better". To operationalize this, we employed an LLM to score each solution (1-10) across three dimensions: *Code Engineering*, *Model Architecture*, and *Data Pipeline* (see Appendix Figure 17). This baseline predicts the winner based on the aggregate complexity score.

4.2 Main Results: Feasibility of Run-Free Preference

The stratified pairwise accuracy results in Table 2 validate the feasibility of our approach.

LLMs Exhibit Significant Predictive Capabilities. Both models significantly outperform the random baseline and the complexity heuristic with statistical significance, with *DeepSeek-V3.2-Thinking* achieving **61.5%** and *GPT-5.1* achieving **58.8%**. This performance gap ($> 10\%$) proves LLMs derive valid signals from static inputs through genuine reasoning rather than heuristics, despite the task remaining a challenging frontier.

5 Analysis & Insights

In this section, we deconstruct the mechanisms of the "Implicit World Model" through four pivotal research questions to answer: *why can reasoning substitute for execution, and to what extent?*

While our representational analysis utilizes the full dataset, subsequent analysis (RQ2–RQ3) employs a focused subset capped at 15 solutions per task (2,292 pairs). For ranking evaluations, we sample 105 instances per task to align with the pairwise baseline complexity ($C(15, 2)$).

5.1 RQ1: The Cognitive Mechanism of Data Representation

To distinguish genuine causal reasoning from syntactic memorization, we conducted a systematic study on input modalities (Figure 3(a)). We instantiated four progressively enriched levels: *Code Only* (task description + code), *Raw Data* (appending initial samples), *Numerical Stats* (execution logs from data analysis scripts), and *Verbal Report* (full semantic analysis), alongside a *Context Mismatch* control (pairing code with irrelevant context).

Finding 1: Predictive Success Stems from Semantic Data Understanding, Not Simple Complexity Heuristics. Our results refute a potential concern that LLMs merely rely on a complexity heuristic. Figure 3(a) demonstrates a clear performance progression from the Heuristic Baseline (50.8%) and Code Only (56.7%) to Numerical Stats (59.0%), peaking with Verbal Reports (61.3%). The insignificant gain of the Context Mismatch (56.8%) over Code Only confirms that predictive

(Acc. %) Task Dims. →		*Domain*			*Difficulty*			*Task Paradigm*			**Sols.**
↓ **Sols. Attrs.**		CV	NLP	Data Sci.	Easy	Med.	Hard	Class.	Regres.	Others	**Avg Acc**
DeepSeek-V3.2 (Thinking mode)											
Algo Era	Traditional	$60.2_{\pm 0.9}$	$70.6_{\pm 0.6}$	$59.3_{\pm 0.5}$	$59.8_{\pm 1.1}$	$69.1_{\pm 0.2}$	$61.1_{\pm 0.7}$	$61.5_{\pm 0.5}$	$61.2_{\pm 0.7}$	$76.2_{\pm 0.5}$	$\mathbf{64.5_{\pm 0.6}}$
	Modern	$59.1_{\pm 0.5}$	$65.0_{\pm 0.1}$	$56.3_{\pm 0.5}$	$60.7_{\pm 0.3}$	$61.7_{\pm 0.3}$	$55.1_{\pm 0.6}$	$57.8_{\pm 0.2}$	$62.5_{\pm 0.4}$	$62.3_{\pm 0.5}$	$60.4_{\pm 0.2}$
Granularity	Cross-Algo	$56.6_{\pm 0.3}$	$68.9_{\pm 0.7}$	$58.4_{\pm 0.9}$	$57.6_{\pm 1.0}$	$68.2_{\pm 0.4}$	$57.7_{\pm 1.5}$	$59.8_{\pm 0.7}$	$60.6_{\pm 0.7}$	$74.1_{\pm 0.9}$	$\mathbf{62.8_{\pm 0.6}}$
	Self-Comp.	$60.1_{\pm 0.6}$	$65.1_{\pm 0.2}$	$56.3_{\pm 0.8}$	$61.6_{\pm 0.3}$	$60.9_{\pm 0.4}$	$56.5_{\pm 1.0}$	$58.2_{\pm 0.1}$	$62.9_{\pm 0.4}$	$62.1_{\pm 0.5}$	$60.7_{\pm 0.1}$
Complexity	Low	$57.6_{\pm 0.4}$	$69.8_{\pm 0.5}$	$57.2_{\pm 0.3}$	$58.9_{\pm 0.6}$	$66.2_{\pm 0.2}$	$58.9_{\pm 0.7}$	$58.6_{\pm 0.2}$	$61.6_{\pm 0.2}$	$73.3_{\pm 0.9}$	$\mathbf{62.1_{\pm 0.3}}$
	Medium	$59.6_{\pm 0.3}$	$65.1_{\pm 0.1}$	$58.1_{\pm 0.2}$	$60.5_{\pm 0.2}$	$63.3_{\pm 0.1}$	$56.6_{\pm 0.2}$	$58.1_{\pm 0.2}$	$63.4_{\pm 0.3}$	$64.6_{\pm 0.6}$	$61.3_{\pm 0.1}$
	High	$61.2_{\pm 2.0}$	$80.1_{\pm 0.7}$	$50.0_{\pm 1.1}$	$76.8_{\pm 2.8}$	$58.4_{\pm 1.6}$	$52.7_{\pm 0.9}$	$60.3_{\pm 2.5}$	$58.4_{\pm 1.4}$	$61.3_{\pm 1.7}$	$59.6_{\pm 1.4}$
Tasks Avg Acc		$59.3_{\pm 0.5}$	$\mathbf{69.9_{\pm 0.2}}$	$57.4_{\pm 0.2}$	$60.4_{\pm 0.5}$	$\mathbf{63.9_{\pm 0.2}}$	$57.0_{\pm 0.3}$	$58.9_{\pm 0.3}$	$62.1_{\pm 0.1}$	$\mathbf{66.8_{\pm 0.5}}$	$\mathbf{61.5_{\pm 0.2}}$
GPT-5.1											
Algo Era	Traditional	$60.1_{\pm 0.4}$	$64.7_{\pm 0.2}$	$59.5_{\pm 0.2}$	$56.6_{\pm 0.4}$	$65.5_{\pm 0.2}$	$63.4_{\pm 0.3}$	$59.3_{\pm 0.6}$	$62.2_{\pm 0.2}$	$67.2_{\pm 0.3}$	$\mathbf{62.0_{\pm 0.2}}$
	Modern	$54.5_{\pm 0.2}$	$62.2_{\pm 0.8}$	$56.3_{\pm 0.1}$	$55.1_{\pm 0.4}$	$59.9_{\pm 0.4}$	$57.9_{\pm 0.0}$	$56.4_{\pm 0.5}$	$58.2_{\pm 0.5}$	$59.8_{\pm 0.6}$	$57.7_{\pm 0.3}$
Granularity	Cross-Algo	$58.0_{\pm 1.1}$	$61.2_{\pm 0.3}$	$56.7_{\pm 0.1}$	$55.3_{\pm 0.7}$	$61.3_{\pm 0.2}$	$59.8_{\pm 0.1}$	$55.7_{\pm 0.8}$	$61.7_{\pm 0.3}$	$62.0_{\pm 0.5}$	$\mathbf{59.0_{\pm 0.3}}$
	Self-Comp.	$54.8_{\pm 0.0}$	$64.2_{\pm 1.0}$	$57.7_{\pm 0.1}$	$55.3_{\pm 0.4}$	$61.6_{\pm 0.4}$	$59.2_{\pm 0.2}$	$58.0_{\pm 0.7}$	$57.6_{\pm 0.5}$	$62.2_{\pm 0.7}$	$58.7_{\pm 0.3}$
Complexity	Low	$56.4_{\pm 0.2}$	$66.6_{\pm 0.5}$	$56.2_{\pm 0.2}$	$56.8_{\pm 0.5}$	$64.2_{\pm 0.2}$	$57.6_{\pm 0.2}$	$57.9_{\pm 0.7}$	$59.3_{\pm 0.2}$	$68.7_{\pm 0.3}$	$\mathbf{60.1_{\pm 0.3}}$
	Medium	$55.8_{\pm 0.2}$	$60.6_{\pm 0.6}$	$59.3_{\pm 0.2}$	$54.6_{\pm 0.4}$	$61.2_{\pm 0.3}$	$61.1_{\pm 0.2}$	$56.5_{\pm 0.5}$	$60.0_{\pm 0.5}$	$60.5_{\pm 0.6}$	$58.6_{\pm 0.3}$
	High	$50.8_{\pm 0.3}$	$79.0_{\pm 0.8}$	$57.2_{\pm 1.5}$	$44.2_{\pm 2.7}$	$56.2_{\pm 0.4}$	$59.7_{\pm 1.6}$	$56.0_{\pm 0.7}$	$54.5_{\pm 0.7}$	$56.2_{\pm 1.1}$	$55.3_{\pm 0.3}$
Tasks Avg Acc		$55.4_{\pm 0.2}$	$\mathbf{63.0_{\pm 0.6}}$	$57.4_{\pm 0.1}$	$55.5_{\pm 0.4}$	$\mathbf{61.6_{\pm 0.3}}$	$59.7_{\pm 0.1}$	$57.2_{\pm 0.5}$	$59.2_{\pm 0.3}$	$\mathbf{62.2_{\pm 0.5}}$	$58.8_{\pm 0.3}$

Table 2: **Main Results: Predictive Capability and Boundary Analysis.** This table presents the Pairwise Preference Accuracy (%) of the evaluated LLMs averaged over three runs, stratified by Task Dimensions and Solution Attributes. Results are reported as Mean $_{\pm \text{Stdev}}$. **DeepSeek-V3.2 (Thinking Mode)** and **GPT-5.1** achieve global averages of **61.5%** and **58.8%** respectively, significantly outperforming the random baseline of **50%** and the complexity-based heuristic baseline of **50.8%**.

success hinges on strict *Semantic Alignment*. The superiority of verbal narratives over raw statistics reveals that models operate primarily as *rhetorical reasoners*, triggering an inference jump that is consistent across domains (Figure 3(b)).

5.2 RQ2: Capabilities, Boundaries, and Algorithmic Bias

In this section, we analyze the necessity of reasoning, domain sensitivity, generalization to ranking, and the reliability of its confidence.

Finding 2: Reasoning Unlocks Capabilities, Yet Distinct Cognitive Boundaries Persist Across Domains. Figure 3(d) identifies *reasoning* as the primary engine, with the Thinking Mode (CoT) (DeepSeek-AI, 2025a; OpenAI, 2024) (61.3%) outperforming Direct Answering (55.9%). This performance remains robust across temperatures ($T \in [0, 1.5]$) and trajectory variance (see Appendix C.2), implying an invariant logical core that relies on genuine reasoning rather than exploiting easy artifacts from the same agent run. However, this capability is constrained by the problem landscape; Table 2 reveals sharp performance stratifications across the Task-Solution matrix. On the *Task Dimension*, models demonstrate a preference for NLP (66.9%) and Easy (63.9%) paradigms. Simultaneously, analysis on the *Solution Dimension*

♠ **Discussion 5.1. The Gap Between Semantic and Numeric Spaces.** We observe that raw data yields insignificant gains over code-only input. We attribute this to the fact that continuous numerical values are **Weak-Semantic Symbols** that lack generalizable topological structure in the embedding space (Davies et al., 2025). While language serves as a **Strong-Structural Symbol** carrying the compressed, empirical representations of human reasoning (Rytting and Wingate, 2021), raw numbers appear to the model as unstructured high-entropy noise. Our verbalization strategy bridges this gap by projecting numeric data into the semantic manifold, injecting necessary inductive bias. Looking ahead, a fundamental resolution requires integrating **Symbolic Regression** (Grayeli et al., 2024) to distill intricate logic directly from data.

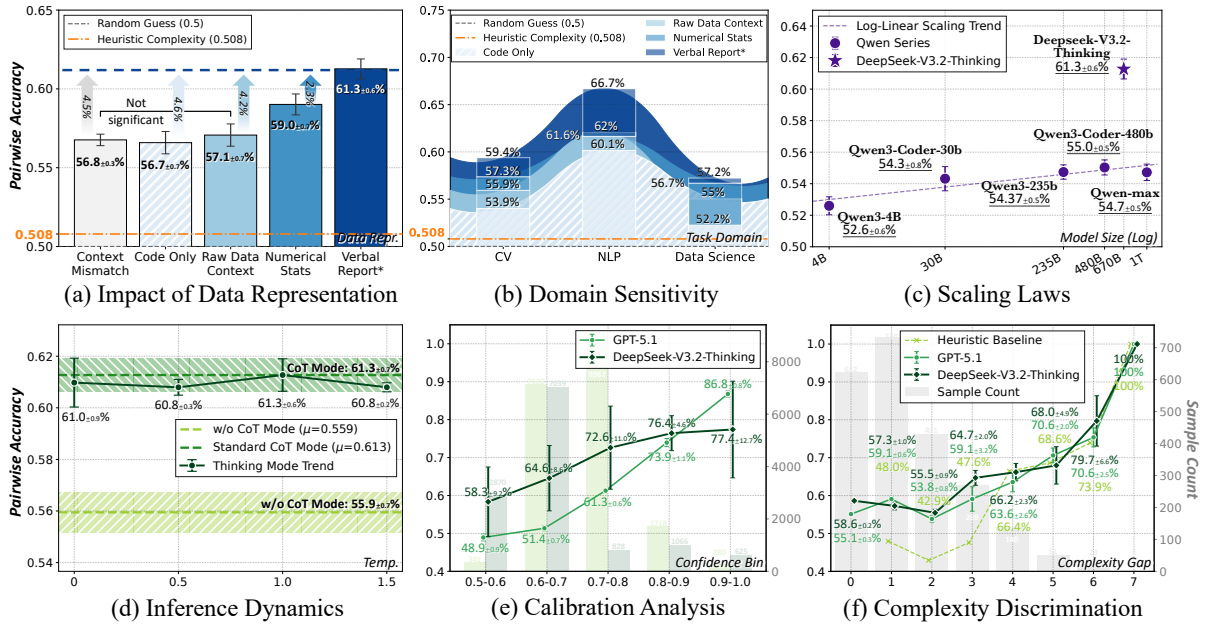


Figure 3: **Comprehensive Analysis of World Model Mechanisms and Capabilities. (a) Impact of Data Representation:** Predictive success stems from semantic data understanding rather than complexity heuristics. **(b) Domain Sensitivity:** The superiority of verbal reports remains consistent across domains. **(c) Scaling Laws:** Accuracy decouples from pure parameter scaling. **(d) Inference Dynamics:** Active reasoning outperforms direct answering with robust stability across temperatures. **(e) Calibration Analysis:** Self-reported confidence correlates with accuracy. **(f) Complexity Discrimination:** Accuracy scales with the complexity gap.

reveals a comparison preference for Traditional ML within the *Algo Era* (64.5%), a "Complexity Tax" (59.6% on complex code), and a *Granularity* bottleneck, where the model is more effective at distinguishing broad *Cross-Algo* contrasts (comparing solutions with different algorithms, 62.8%). Thus, while reasoning is indispensable, it faces limits when navigating intricate code logic or subtle intra class nuances.

Extending the scope to global **Listwise Ranking** further magnifies this limitation, as Table 3 reveals a scalability defect where Accuracy@1 drops from the pairwise baseline (61.3% $\rightarrow$ 31.1%) while Spearman Correlation hovers at a notably low level ($\rho \approx 0.23$), indicating that the model *lacks global discrimination capability*, failing to sustain consistency beyond binary interactions.

Finding 3: The "Implicit World Model" Leverages Causal Reasoning Beyond Complexity Heuristics and Exhibits Robust Confidence Calibration. Tracing the *Complexity Gap* in Figure 3(f) shows accuracy scales with distinction; crucially, the model's superiority over heuristics in low-gap scenarios proves it detects valid semantic signals rather than simple metrics. Furthermore, Figure 3(e) demonstrates **Calibration**, where confidence correlates strictly with accuracy. This reliability underpins the Section 6 gating mechanism, ensuring agents act with certainty.

5.3 RQ3: Scaling Laws of Data-centric Solution Preference

We evaluate the Qwen series across a spectrum from 4B to 1T to determine if predictive capability acts as an emergent scaling property. Figure 3(c) details performance on five distinct checkpoints: 4B (*Qwen3-4B-Instruct-2507*), 30B (*Qwen3-Coder-30B-a3b-Instruct*), 235B (*Qwen3-*

♥ **Discussion 5.3: The Illusion of Scaling on World-Blind Models.** We attribute scaling failures to **Information Scarcity**: static training corpora consist of code paired with merely *trivial inputs* or abstract descriptions, lacking the *large-scale data distributions* required for true dynamic execution interplay (Liu et al., 2022). While our results confirm that models can exploit sparse dynamic traces hidden in existing data, they remain fundamentally **World-Blind Learners** (Floridi et al., 2025). To transcend this mere "syntactic mimicry", future scaling must pivot from static ingestion to **Interactive Simulation**, grounding agents in genuine causal feedback loops (Bender and Koller, 2020).

Size	Corr.	Accuracy@k (%)			
(N)	Spr. ρ	k=1	k=2	k=3	k=4
2	$0.24_{\pm 0.01}$	$61.3_{\pm 0.6}$	–	–	–
3	$0.22_{\pm 0.00}$	$43.4_{\pm 0.4}$	$25.5_{\pm 0.4}$	–	–
4	$0.25_{\pm 0.00}$	$35.0_{\pm 1.0}$	$16.4_{\pm 0.7}$	$10.2_{\pm 0.2}$	–
5	$0.22_{\pm 0.00}$	$31.1_{\pm 0.9}$	$11.2_{\pm 0.3}$	$4.9_{\pm 0.1}$	$3.0_{\pm 0.2}$

Table 3: **Ranking Performance.** Listwise ranking metrics across varying list sizes N. **Spr.**: Spearman Correlation (ρ). **A@k**: Accuracy of the top-k ranking positions (%). "–" denotes undefined metrics where $k \geq N$.

235B-a22b-Instruct-2507), 480B (*Qwen3-Coder-480B-a35b-Instruct*), and 1T (*Qwen-Max*).

Finding 4: Predictive Accuracy Violates Standard Parameter Scaling Laws. Contrary to standard Parameter Scaling Laws, our results (Figure 3(c)) reveal a *rapid saturation phenomenon*. Within the Qwen series, performance sees diminishing returns after the initial 30B threshold, creating a statistical plateau that persists even at the 1T scale. This trajectory implies a distinct "capacity ceiling," suggesting that raw parameter scaling alone is insufficient for further gains in the *Data-centric Solution Preference* task. In contrast, the distinct superiority of DeepSeek-V3.2 (61.3%) and GPT-5.1 (58.8%) demonstrates that predictive power drives less from raw scale than from **reasoning-centric architectural paradigms**, implying that future gains will rely on specialized inference incentives rather than simple parameter expansion.

5.4 RQ4: Comparison with Human Judgment and Validation-Test Gap

To validate the model's reasoning depth, we conducted a qualitative analysis on the *Google Quest Challenge* from main experiment, which is a multi-label subjective question-answering task.

Finding 5: The Model Outperforms Human Intuition by Rejecting Complexity Bias. In the case study of Figure 10, the model surpassed human judgment by correctly prioritizing a simple LightGBM, whereas humans succumbed to the "bigger is better" bias by favoring a complex Deep Neural Network. It successfully detects small-sample overfitting risks that humans missed, proving that data-grounded reasoning can effectively override superficial human biases.

The Validation-Test Gap. We further examine the reliability of execution-based validation metrics (M_{val}), derived from internal data splits, as proxies for test performance (M_{test}). As shown in Table 4, relying solely on M_{val} yields an accuracy of only **72.2%**. This ceiling reveals a substantial **Validation-Test Gap** stemming from distributional shifts and validation overfitting. Crucially, implicit reasoning partially mitigates this gap, offering a semantic safeguard that balances efficiency against the risk of metric-driven overfitting.

Signal Source	Cost	Acc. (%)
Random Guess	-	50.0
Exec. (M_{val})	∼Hours	72.2
LLM	∼Seconds	61.5

Table 4: **Validation-Test Gap.** Local metrics (M_{val}) are noisy proxies for test performance (M_{test}), achieving only 72.2% accuracy due to distribution shifts.

6 Agent Integration: FOREAGENT

Building on the predictive capabilities of the World Model, we propose **FOREAGENT**, a hybrid autonomous ML agent designed to decouple hypothesis exploration from physical execution.

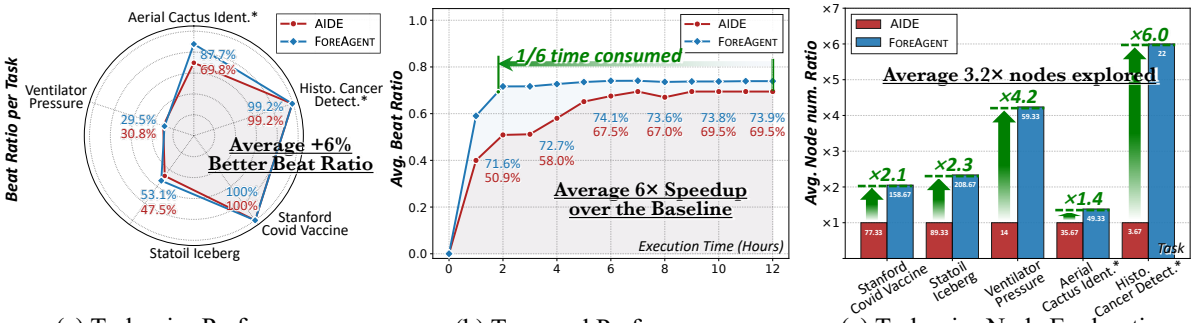
6.1 Motivation

We aim to break the *Execution Bottleneck* in Section 2.2, compressing hours of physical execution into seconds of logical inference, and the *Validation-Test Gap* identified in Section 5.4. Thus, we propose **FOREAGENT**, which utilizes the "Implicit World Model" as a filter to prune the search space before execution for acceleration.

6.2 Method: The Predict-then-Verify Loop

We adopt **AIDE** (Jiang et al., 2025) as our backbone, building directly upon the tree-search architecture described in Section 2.1.

♦ **Discussion 6.1. The Indirect Ceiling of Static Prediction.** We interpret 72.2% as an **Indirect Epistemic Bound**, constrained by the *Validation-Test Gap* and *Limited Innovative Creativity*. Theoretically, since static prediction cannot outperform dynamic verification (Rice, 1953), any gains beyond this saturation point represent the overfitting of distributional noise (Dwork et al., 2015). Moreover, this ceiling exposes a deficit in *Generative Innovation*: current LLMs hit a **Homogeneity Barrier**, producing *functionally isomorphic* solutions that lack context-specific specialization (Doshi and Hauser, 2024). Thus, lifting this bound relies on evolving base models to achieve the *Genuine Innovation*.



(a) Task-wise Performance (b) Temporal Performance (c) Task-wise Node Exploration

Figure 4: **Agent Performance Analysis.** (a) **Task-wise Beat Ratio:** FOREAGENT achieves an average +6% improvement over the AIDE baseline. (b) **Temporal Efficiency:** The agent converges to peak performance using only 1/6 of the execution time, achieving an average 6× speedup. (c) **Search Breadth:** By offloading evaluation to the "Implicit World Model", FOREAGENT explores 3.2× more nodes on average compared to the baseline, significantly expanding the search space within the same time budget.

We propose **FOREAGENT**, which re-engineers the Improvement stage into a conservative *Predict-then-Verify* loop (Figure 2(e)) to bridge the Implementation Gap (Zhu et al., 2025a). The workflow proceeds through three key phases: (1) **High-Volume Generation**, where $m = 10$ candidates are proposed in parallel to expand search width without execution costs; (2) **Confidence-Gated Pairwise Selection**, which utilizes a confidence gate ($c = 0.7$) to ensure high-certainty selection; and (3) **Verification Execution**, where the Top-k ($k = 1$) candidate is physically verified to anchor the solution trajectory in execution feedback (see ablation study in Appendix C.6).

Task Name	Domain	Status
Stanford Covid Vaccine	Biology	Seen
Ventilator Pressure	Physics	Seen
Statoil Iceberg	Geoscience	Seen
Aerial Cactus Ident.*	Ecology	**Unseen**
Histo. Cancer Detect.*	Medicine	**Unseen**

Table 5: **Agent Evaluation Benchmark.** The selection covers diverse AI4Science domains to test the World Model's capability to generalize from seen tasks to unseen scientific problems.

6.3 Experimental Setup

Tasks and Baselines. We evaluate **FOREAGENT** on 5 AI4Science tasks from MLE-bench (Table 5), including two "Unseen" tasks. We benchmark against AIDE under a 12-hour limit; both use DeepSeek-V3.2 for coding, while Implicit World Modeling employs DeepSeek-V3.2-Thinking.

Metric. To ensure reliability, we conduct three independent runs for each task and report the average **Beat Ratio** (Ou et al., 2025). This metric quantifies the percentage of human leaderboard contestants outperformed by the agent, representing expert-level competitiveness.

6.4 Results

By substituting costly execution with rapid inference, FOREAGENT achieves an average **6× speedup** (Figure 4(b)), enabling it to explore **3.2× more nodes** within just 1/6 of the time budget (Figure 4(c)). This expanded search capability directly translates into performance, driving a **+6% improvement** in Beat Ratio (Figure 4(a)) and demonstrating robust generalization on unseen tasks. Furthermore, the World Model acts as a semantic safeguard during intermediate development, significantly boosting the Test Improve Rate by 23% (see Appendix C.5). Although we currently focus on inference, this paradigm naturally extends to training contexts like **Reward Model**, a promising direction we reserve for future work.

7 Related Work

LLM Agents in Machine Learning (ML). LLM agents are extensively deployed in ML for tasks ranging from pipeline automation (Jiang et al., 2025; Qiao et al., 2025; Gu et al., 2024b) to competitive problem-solving (Luo et al., 2025; Ou et al., 2025; Chan et al., 2025; Liu et al., 2025b). However, the computational cost of their generation-execution loops (Yao et al., 2023) remains a bottleneck. To mitigate this, recent works utilize internal

priors to prune redundant steps (Kulibaba et al., 2025; Trirat et al., 2025), transitioning from brute-force search to reasoned planning.

World Models for Skip-Execution. Adapting World Models (Ding et al., 2025; Li et al., 2025e) to code, recent research predicts execution outcomes to bypass physical runs (Hora, 2024; team et al., 2025; Li et al., 2025c). While prior works focus on logic consistency in reasoning benchmarks (Wei et al., 2025a; Gu et al., 2024a; Jain et al., 2024), our approach integrates this predictive capability with Data-Centric Solution Preference (Shen et al., 2024; Just et al., 2024). By anchoring evaluations in explicit dataset rationales rather than heuristics, we ensure reliability in stochastic data domains. Extended discussion in Appendix A.

8 Conclusion

This work validates the feasibility of compressing physical execution into logical inference. Our analysis reveals LLMs function as calibrated, reasoning-driven critics via semantic verbalization to strictly gate actions and prune search spaces. By decoupling reasoning from runtime, we provide a robust blueprint for bypassing the execution bottleneck in complex machine learning tasks.

Limitations

Corpus Imbalance and Domain Coverage. Although our corpus encompasses 18,438 pairs across 26 tasks, the distribution remains inherently skewed. Mainstream paradigms like Classification and Regression dominate the dataset, whereas niche scientific tasks (e.g., Audio Classification, Tabular Grading) are represented by significantly smaller sample sizes. Consequently, while the model demonstrates strong general capabilities, its reliability in extremely low-resource or highly specialized scientific domains may vary, and the current evaluation may not fully reflect the challenges of these long-tail scenarios. Furthermore, the Verified Data Report currently relies on metadata for unstructured domains like CV and NLP, leaving the integration of multimodal data analysis agents for deeper semantic profiling to future work.

Agent Framework Implementation. To validate the model's utility, we prioritized stability, instantiating FOREAGENT with a conservative *Predict-then-Verify* loop. This design alternates strictly between singular prediction and execution, barely scratching the surface of potential inference-time strategies. Specifically, we have not exhaustively explored advanced architectural variants or hyperparameter configurations within this paradigm, implying that the current implementation has not yet been pushed to its optimal limit. Therefore, the reported performance likely represents a lower bound of the framework's capability. Beyond this specific instantiation, we identify the framework's broader potential as a scalable *Reward Model*. By providing dense, execution-free feedback, it paves the way for accelerating Reinforcement Learning rollouts and serves as a plug-and-play optimization module adaptable to diverse agent frameworks.

Acknowledgement

We would like to express sincere gratitude to the reviewers for their thoughtful and constructive feedback. This work was supported by the National Natural Science Foundation of China (No. 62576307, No. NSFCU23B2055, No. NSFCU19B2027), the Fundamental Research Funds for the Central Universities (226-2023-00138), Yongjiang Talent Introduction Programme (2021A-156-G), and Information Technology Center and State Key Lab of CAD&CG, Zhejiang University. This work was supported by Ant Group and Zhejiang University - Ant Group Joint Laboratory of Knowledge Graph.

References

Yash Akhauri, Xingyou Song, Arissa Wongpanich, Bryan Lewandowski, and Mohamed S. Abdelfattah. 2025. Regression language models for code. *Preprint*, arXiv:2509.26476.

Nicolás Astorga, Tennison Liu, Yuanzhang Xiao, and Mihaela van der Schaar. 2025. Autoformulation of mathematical optimization models using llms. *Preprint*, arXiv:2411.01679.

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198, Online. Association for Computational Linguistics.

Christian Cabrera, Andrei Paleyes, Pierre Thodoroff, and Neil D. Lawrence. 2025. Machine learning systems: A survey from a data-oriented perspective. *Preprint*, arXiv:2302.04810.

Jingyi Chai, Shuo Tang, Rui Ye, Yuwen Du, Xinyu Zhu, Mengcheng Zhou, Yanfeng Wang, Weinan E, Yuzhi

Zhang, Linfeng Zhang, and Siheng Chen. 2025. Scimaster: Towards general-purpose scientific ai agents, part i. x-master as foundation: Can we lead on humanity's last exam? *Preprint*, arXiv:2507.05241.

Jun Shern Chan, Neil Chowdhury, Oliver Jaffe, James Aung, Dane Sherburn, Evan Mays, Giulio Starace, Kevin Liu, Leon Maksin, Tejal Patwardhan, Lilian Weng, and Aleksander Mądry. 2025. Mle-bench: Evaluating machine learning agents on machine learning engineering. *Preprint*, arXiv:2410.07095.

Liu Changshu, Chen Yang, and Reyhaneh Jabbarvand. 2024. Codemind: Evaluating large language models for code reasoning. *http://arxiv.org/abs/2402.09664*.

Junkai Chen, Zhiyuan Pan, Xing Hu, Zhenhao Li, Ge Li, and Xin Xia. 2025a. Reasoning runtime behavior of a program with llm: How far are we? In *2025 IEEE/ACM 47th International Conference on Software Engineering (ICSE)*, pages 1869–1881.

Ke Chen, Peiran Wang, Yaoning Yu, Xianyang Zhan, and Haohan Wang. 2025b. Large language model-based data science agent: A survey. *Preprint*, arXiv:2508.02744.

Qiguang Chen, Mingda Yang, Libo Qin, Jinhao Liu, Zheng Yan, Jiannan Guan, Dengyun Peng, Yiyan Ji, Hanjing Li, Mengkang Hu, Yimeng Zhang, Yihao Liang, Yuhang Zhou, Jiaqi Wang, Zhi Chen, and Wanxiang Che. 2025c. Ai4research: A survey of artificial intelligence for scientific research. *Preprint*, arXiv:2507.01903.

Zhaorun Chen, Zhuokai Zhao, Kai Zhang, Bo Liu, Qi Qi, Yifan Wu, Tarun Kalluri, Sara Cao, Yuanhao Xiong, Haibo Tong, Huaxiu Yao, Hengduo Li, Jiacheng Zhu, Xian Li, Dawn Song, Bo Li, Jason Weston, and Dat Huynh. 2025d. Scaling agent learning via experience synthesis. *Preprint*, arXiv:2511.03773.

Yizhou Chi, Yizhang Lin, Sirui Hong, Duyi Pan, Yaying Fei, Guanghao Mei, Bangbang Liu, Tianqi Pang, Jacky Kwok, Ceyao Zhang, Bang Liu, and Chenglin Wu. 2024. Sela: Tree-search enhanced llm agents for automated machine learning. *Preprint*, arXiv:2410.17238.

Alex O. Davies, Roussel Nzoyem, Nirav Ajmeri, and Telmo M. Silva Filho. 2025. Language models do not embed numbers continuously. *Preprint*, arXiv:2510.08009.

DeepSeek-AI. 2024. Deepseek api documentation: Parameter settings. `https://api-docs.deepseek.com/quick_start/parameter_settings`. Accessed: 2025-12-21.

DeepSeek-AI. 2025a. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

DeepSeek-AI. 2025b. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Jingtao Ding, Yunke Zhang, Yu Shang, Jie Feng, Yuheng Zhang, Zefang Zong, Yuan Yuan, Hongyuan Su, Nian Li, Jinghua Piao, Yucheng Deng, Nicholas Sukiennik, Chen Gao, Fengli Xu, and Yong Li. 2025. Understanding world or predicting future? a comprehensive survey of world models. *Preprint*, arXiv:2411.14499.

Anil R. Doshi and Oliver P. Hauser. 2024. Generative ai enhances individual creativity but reduces the collective diversity of novel content. *Science Advances*, 10(28):eadn5290.

Shangheng Du, Xiangchao Yan, Dengyang Jiang, Jiakang Yuan, Yusong Hu, Xin Li, Liang He, Bo Zhang, and Lei Bai. 2025. Automlgen: Navigating fine-grained optimization for coding agents. *Preprint*, arXiv:2510.08511.

Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Roth. 2015. The reusable holdout: Preserving validity in adaptive data analysis. *Science*, 349(6248):636–638.

Haoyang Fang, Boran Han, Nick Erickson, Xiyuan Zhang, Su Zhou, Anirudh Dagar, Jiani Zhang, Ali Caner Turkmen, Cuixiong Hu, Huzefa Rangwala, Ying Nian Wu, Bernie Wang, and George Karypis. 2025. Mlzero: A multi-agent system for end-to-end machine learning automation. *Preprint*, arXiv:2505.13941.

Jichen Feng, Yifan Zhang, Chenggong Zhang, Yifu Lu, Shilong Liu, and Mengdi Wang. 2025. Web world models. *Preprint*, arXiv:2512.23676.

Luciano Floridi, Yiyang Jia, and Fernando Tohmé. 2025. A categorical analysis of large language models and why llms circumvent the symbol grounding problem. *Preprint*, arXiv:2512.09117.

Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, et al. 2025. Towards an ai co-scientist. *Preprint*, arXiv:2502.18864.

Arya Grayeli, Atharva Sehgal, Omar Costilla-Reyes, Miles Cranmer, and Swarat Chaudhuri. 2024. Symbolic regression with a learned concept library. *Preprint*, arXiv:2409.09359.

Alex Gu, Baptiste Rozière, Hugh Leather, Armando Solar-Lezama, Gabriel Synnaeve, and Sida I. Wang. 2024a. Cruxeval: A benchmark for code reasoning, understanding and execution. *Preprint*, arXiv:2401.03065.

Yang Gu, Hengyu You, Jian Cao, Muran Yu, Haoran Fan, and Shiyou Qian. 2024b. Large language models for constructing and optimizing machine learning workflows: A survey. *Preprint*, arXiv:2411.10478.

Siyuan Guo, Cheng Deng, Ying Wen, Hechang Chen, Yi Chang, and Jun Wang. 2024. Ds-agent: Automated data science by empowering large language models with case-based reasoning. *Preprint*, arXiv:2402.17453.

David Ha and Jürgen Schmidhuber. 2018. World models. *Zenodo*.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. 2024. Mastering diverse domains through world models. *Preprint*, arXiv:2301.04104.

Sirui Hong, Yizhang Lin, Bang Liu, et al. 2024. Data interpreter: An llm agent for data science. *Preprint*, arXiv:2402.18679.

Andre Hora. 2024. Predicting test results without execution. *https://doi.org/10.1145/3663529.3663794*, pages 542–546.

Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. 2024. Mlagentbench: Evaluating language agents on machine learning experimentation. *Preprint*, arXiv:2310.03302.

Yuxuan Huang, Yihang Chen, Haozheng Zhang, Kang Li, Huichi Zhou, Meng Fang, Linyi Yang, Xiaoguang Li, Lifeng Shang, Songcen Xu, Jianye Hao, Kun Shao, and Jun Wang. 2025. Deep research agents: A systematic examination and roadmap. *Preprint*, arXiv:2506.18096.

Naman Jain, King Han, Alex Gu, Wending Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar Lezama, Koushik Sen, and Ion Stoica. 2024. Livecodebench: Holistic and contamination free evaluation of large language models for code. *Preprint*, arXiv:2403.07974.

Zhengyao Jiang, Dominik Schmidt, Dhruv Srikanth, Dixing Xu, Ian Kaplan, Deniss Jacenko, and Yuxiang Wu. 2025. Aide: Ai-driven exploration in the space of code. *Preprint*, arXiv:2502.13138.

Liqiang Jing, Zhehui Huang, Xiaoyang Wang, Wenlin Yao, Wenhao Yu, Kaixin Ma, Hongming Zhang, Xinya Du, and Dong Yu. 2024. Dsbench: How far are data science agents from becoming data science experts? *arXiv preprint arXiv:2409.07703*.

Hoang Just, Ming Jin, Anit Sahu, Huy Phan, and Ruoxi Jia. 2024. Data-centric human preference optimization with rationales. *arXiv (Cornell University)*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2023. Large language models are zero-shot reasoners. *Preprint*, arXiv:2205.11916.

Stepan Kulibaba, Artem Dzhalilov, Roman Pakhomov, Oleg Svidchenko, Alexander Gasnikov, and Aleksei Shpilman. 2025. Kompeteai: Accelerated autonomous multi-agent system for end-to-end pipeline generation for machine learning problems. *Preprint*, arXiv:2508.10177.

Robert Tjarko Lange, Yuki Imajuku, and Edoardo Cetin. 2025. Shinkaevolve: Towards open-ended and sample-efficient program evolution. *Preprint*, arXiv:2509.19349.

Annan Li, Chufan Wu, Zengle Ge, et al. 2025a. The fm agent. *Preprint*, arXiv:2510.26144.

Haoyang Li, Xuejia Chen, Zhanchao Xu, Darian Li, Nicole Hu, Fei Teng, Yiming Li, Luyu Qiu, Chen Jason Zhang, Li Qing, and Lei Chen. 2025b. Exposing numeracy gaps: A benchmark to evaluate fundamental numerical abilities in large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 20004–20026, Vienna, Austria. Association for Computational Linguistics.

Junlong Li, Daya Guo, Dejian Yang, Runxin Xu, Yu Wu, and Junxian He. 2025c. Codei/o: Condensing reasoning patterns via code input-output prediction. *Preprint*, arXiv:2502.07316.

Ruochen Li, Teerth Patel, Qingyun Wang, and Xinya Du. 2025d. Mlr-copilot: Autonomous machine learning research based on large language models agents. *Preprint*, arXiv:2408.14033.

Yixia Li, Hongru Wang, Jiahao Qiu, Zhenfei Yin, Dongdong Zhang, Cheng Qian, Zeping Li, Pony Ma, Guanhua Chen, Heng Ji, and Mengdi Wang. 2025e. From word to world: Can large language models be implicit text-based world models? *Preprint*, arXiv:2512.18832.

Ruibo Liu, Jason Wei, Shixiang Shane Gu, Teyen Wu, Soroush Vosoughi, Claire Cui, Denny Zhou, and Andrew M. Dai. 2022. Mind's eye: Grounded language model reasoning through simulation. *Preprint*, arXiv:2210.05359.

Yixiu Liu, Yang Nan, Weixian Xu, Xiangkun Hu, Lyumanshan Ye, Zhen Qin, and Pengfei Liu. 2025a. Alphago moment for model architecture discovery. *Preprint*, arXiv:2507.18074.

Zexi Liu, Yuzhu Cai, Xinyu Zhu, Yujie Zheng, Runkun Chen, Ying Wen, Yanfeng Wang, Weinan E, and Siheng Chen. 2025b. Ml-master: Towards ai-for-ai via integration of exploration and reasoning. *Preprint*, arXiv:2506.16499.

Yujie Luo, Zhuoyun Yu, Xuehai Wang, Yuqi Zhu, Ningyu Zhang, Lanning Wei, Lun Du, Da Zheng, and Huajun Chen. 2025. Executable knowledge graphs for replicating ai research. *Preprint*, arXiv:2510.17795.

Rachel Metz. 2024. Openai scale ranks progress toward 'human-level' ai. *Bloomberg*.

Jaehyun Nam, Jinsung Yoon, Jiefeng Chen, Jinwoo Shin, Sercan Ö. Arık, and Tomas Pfister. 2025. Mle-star: Machine learning engineering agent via search and targeted refinement. *Preprint*, arXiv:2506.15692.

Deepak Nathani, Lovish Madaan, Nicholas Roberts, Nikolay Bashlykov, Ajay Menon, Vincent Moens, Amar Budhiraja, Despoina Magka, Vladislav Vorotilov, Gaurav Chaurasia, Dieuwke Hupkes, Ricardo Silveira Cabral, Tatiana Shavrina, Jakob Foerster, Yoram Bachrach, William Yang Wang, and

Roberta Raileanu. 2025. Mlgym: A new framework and benchmark for advancing ai research agents. *Preprint*, arXiv:2502.14499.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, et al. 2025. Alphaevolve: A coding agent for scientific and algorithmic discovery. *Preprint*, arXiv:2506.13131.

OpenAI. 2024. Openai o1 system card. *Preprint*, arXiv:2412.16720.

OpenAI. 2025a. System Card for gpt-5. Accessed on August 13, 2025.

OpenAI. 2025b. System Card for o3-mini. Accessed on December 11, 2025.

Yixin Ou, Yujie Luo, Jingsheng Zheng, Lanning Wei, Zhuoyun Yu, Shuofei Qiao, Jintian Zhang, Da Zheng, Yuren Mao, Yunjun Gao, Huajun Chen, and Ningyu Zhang. 2025. Automind: Adaptive knowledgeable agent for automated data science. *Preprint*, arXiv:2506.10974.

Rushi Qiang, Yuchen Zhuang, Yinghao Li, Dingu Sagar V K, Rongzhi Zhang, Changhao Li, Ian Shuhei Wong, Sherry Yang, Percy Liang, Chao Zhang, and Bo Dai. 2025. Mle-dojo: Interactive environments for empowering llm agents in machine learning engineering. *Preprint*, arXiv:2505.07782.

Shuofei Qiao, Yixin Ou, Ningyu Zhang, Xiang Chen, Yunzhi Yao, Shumin Deng, Chuanqi Tan, Fei Huang, and Huajun Chen. 2023. Reasoning with language model prompting: A survey. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5368–5393, Toronto, Canada. Association for Computational Linguistics.

Shuofei Qiao, Yanqiu Zhao, Zhisong Qiu, Xiaobin Wang, Jintian Zhang, Zhao Bin, Ningyu Zhang, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. 2025. Scaling generalist data-analytic agents. *Preprint*, arXiv:2509.25084.

H. G. Rice. 1953. Classes of recursively enumerable sets and their decision problems. *Transactions of the American Mathematical Society*, 74(2):358–366.

Christopher Michael Rytting and David Wingate. 2021. Leveraging the inductive bias of large language models for abstract textual reasoning. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, Red Hook, NY, USA. Curran Associates Inc.

Samuel Schmidgall, Yusheng Su, Ze Wang, Ximeng Sun, Jialian Wu, Xiaodong Yu, Jiang Liu, Michael Moor, Zicheng Liu, and Emad Barsoum. 2025. Agent laboratory: Using llm agents as research assistants. *Preprint*, arXiv:2501.04227.

Judy Shen, Archit Sharma, Judy Shen, Qin Jun, Archit Sharma, and Jun Qin. 2024. Towards data-centric rlhf: Simple metrics for preference dataset comparison. *http://arxiv.org/abs/2409.09603*, abs/2409.09603.

Lin Shi, Weicheng Ma, and Soroush Vosoughi. 2024. Judging the judges: A systematic investigation of position bias in pairwise comparative assessments by llms. *arXiv (Cornell University)*.

Hao Sun, Zile Qiao, Jiayan Guo, Xuanbo Fan, Yingyan Hou, Yong Jiang, Pengjun Xie, Yan Zhang, Fei Huang, and Jingren Zhou. 2025a. Zerosearch: Incentivize the search capability of llms without searching. *Preprint*, arXiv:2505.04588.

Ji Sun, Guoliang Li, Peiyao Zhou, Yihui Ma, Jingzhe Xu, and Yuan Li. 2025b. Agenticdata: An agentic data analytics system for heterogeneous data. *Preprint*, arXiv:2508.05002.

FAIR CodeGen team, Jade Copet, Quentin Carbonneaux, et al. 2025. Cwm: An open-weights llm for research on code generation with world models. *Preprint*, arXiv:2510.02387.

InternAgent Team, Bo Zhang, Shiyang Feng, et al. 2025. Internagent: When agent becomes the scientist – building closed-loop system from hypothesis to verification. *Preprint*, arXiv:2505.16938.

Edan Toledo, Karen Hambardzumyan, Martin Josifoski, et al. 2025. Ai research agents for machine learning: Search, exploration, and generalization in mle-bench. *Preprint*, arXiv:2507.02554.

Patara Trirat, Wonyong Jeong, and Sung Ju Hwang. 2025. Automl-agent: A multi-agent llm framework for full-pipeline automl. *Preprint*, arXiv:2410.02958.

Andrej Tschalzev, Sascha Marton, Stefan Lüdtke, Christian Bartelt, and Heiner Stuckenschmidt. 2024. A data-centric perspective on evaluating machine learning models for tabular data. *Preprint*, arXiv:2407.02112.

Sai Wang, Senthilnathan Subramanian, Mudit Sahni, Praneeth Gone, Lingjie Meng, Xiaochen Wang, Nicolas Ferradas Bertoli, Tingxian Cheng, and Jun Xu. 2025a. Configurable multi-agent framework for scalable and realistic testing of llm-based agents. *Preprint*, arXiv:2507.14705.

Xingyao Wang, Boxuan Li, Yufan Song, Frank F. Xu, Xiangru Tang, Mingchen Zhuge, Jiayi Pan, Yueqi Song, Bowen Li, Jaskirat Singh, Hoang H. Tran, Fuqiang Li, Ren Ma, Mingzhang Zheng, Bill Qian, Yanjun Shao, Niklas Muennighoff, Yizhe Zhang, Binyuan Hui, Junyang Lin, Robert Brennan, Hao Peng, Heng Ji, and Graham Neubig. 2025b. Openhands: An open platform for ai software developers as generalist agents. *Preprint*, arXiv:2407.16741.

Anjiang Wei, Jiannan Cao, Ran Li, Hongyu Chen, Yuhui Zhang, Ziheng Wang, Yuan Liu, Thiago S. F. X. Teixeira, Diyi Yang, Ke Wang, and Alex Aiken. 2025a. Equibench: Benchmarking large language models'

reasoning about program semantics via equivalence checking. *Preprint*, arXiv:2502.12466.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models. *Preprint*, arXiv:2201.11903.

Yuxiang Wei, Olivier Duchenne, Jade Copet, Quentin Carbonneaux, Lingming Zhang, Daniel Fried, Gabriel Synnaeve, Rishabh Singh, and Sida I. Wang. 2025b. Swe-rl: Advancing llm reasoning via reinforcement learning on open software evolution. *Preprint*, arXiv:2502.18449.

Lionel Wong, Gabriel Grand, Alexander K. Lew, Noah D. Goodman, Vikash K. Mansinghka, Jacob Andreas, and Joshua B. Tenenbaum. 2023. From word models to world models: Translating from natural language to the probabilistic language of thought. *Preprint*, arXiv:2306.12672.

Xu Yang, Xiao Yang, Shikai Fang, Yifei Zhang, Jian Wang, Bowen Xian, Qizheng Li, Jingyuan Li, Minrui Xu, Yuante Li, Haoran Pan, Yuge Zhang, Weiqing Liu, Yelong Shen, Weizhu Chen, and Jiang Bian. 2025. R&d-agent: An llm-agent framework towards autonomous data science. *Preprint*, arXiv:2505.14738.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. *Preprint*, arXiv:2210.03629.

Haofei Yu, Zhaochen Hong, Zirui Cheng, Kunlun Zhu, Keyang Xuan, Jinwei Yao, Tao Feng, and Jiaxuan You. 2025a. Researchtown: Simulator of human research community. *Preprint*, arXiv:2412.17767.

Zhaojian Yu, Kaiyue Feng, Yilun Zhao, Shilin He, Xiaoping Zhang, and Arman Cohan. 2025b. Alpharesearch: Accelerating new algorithm discovery with language models. *Preprint*, arXiv:2511.08522.

Jiakang Yuan, Xiangchao Yan, Shiyang Feng, Bo Zhang, Tao Chen, Botian Shi, Wanli Ouyang, Yu Qiao, Lei Bai, and Bowen Zhou. 2025. Dolphin: Moving towards closed-loop auto-research through thinking, practice, and feedback. *Preprint*, arXiv:2501.03916.

Daojian Zeng, Lin Zhou, Zhiheng Zhang, and Lincheng Jiang. 2025. Autogen: Automated tool learning data generation with domain-specific structured data. *DATA INTELLIGENCE*, 7(4):1108–1128.

Liu Zexi, Jingyi Chai, Zexi Liu, Zhu Xinyu, Jingyi Chai, Tang Shuo, Xinyu Zhu, Ye Rui, Shuo Tang, Zhang Bo, Rui Ye, Bai Lei, Bo Zhang, Siheng Chen, Lei Bai, and Siheng Chen. 2025. Ml-agent: Reinforcing llm agents for autonomous machine learning engineering. *https://doi.org/10.48550/arxiv.2505.23723*, abs/2505.23723.

Daochen Zha, Zaid Pervaiz Bhat, Kweiherng Lai, Fan Yang, Zhimeng Jiang, Shaochen Zhong, and Xia Hu. 2023. Data-centric artificial intelligence: A survey. *Preprint*, arXiv:2303.10158.

Jiahuan Zhang, Tianheng Wang, Hanqing Wu, Ziyi Huang, Yulong Wu, Dongbai Chen, Linfeng Song, Yue Zhang, Guozheng Rao, and Kaicheng Yu. 2025a. Sr-llm: Rethinking the structured representation in large language model. *Preprint*, arXiv:2502.14352.

Jintian Zhang, Kewei Xu, Jingsheng Zheng, Zhuoyun Yu, Yuqi Zhu, Yujie Luo, Lanning Wei, Shuofei Qiao, Lun Du, Da Zheng, Shumin Deng, Huajun Chen, and Ningyu Zhang. 2025b. Innogym: Benchmarking the innovation potential of ai agents. *Preprint*, arXiv:2512.01822.

Shaolei Zhang, Ju Fan, Meihao Fan, Guoliang Li, and Xiaoyong Du. 2025c. Deepanalyze: Agentic large language models for autonomous data science. *Preprint*, arXiv:2510.16872.

Wenlin Zhang, Xiaopeng Li, Yingyi Zhang, Pengyue Jia, Yichao Wang, Huifeng Guo, Yong Liu, and Xiangyu Zhao. 2025d. Deep research: A survey of autonomous research agents. *Preprint*, arXiv:2508.12752.

Xilin Zhang, Zhixin Mao, Ziwen Chen, and Shen Gao. 2024a. Effective tool augmented multi-agent framework for data analysis. *DATA INTELLIGENCE*, 6(4):923–945.

Yao Zhang, Zijian Ma, Yunpu Ma, Zhen Han, Yu Wu, and Volker Tresp. 2024b. Webpilot: A versatile and autonomous multi-agent system for web task execution with strategic exploration. *AAAI Conference on Artificial Intelligence*.

Yuge Zhang, Qiyang Jiang, XingyuHan XingyuHan, Nan Chen, Yuqing Yang, and Kan Ren. 2024c. Benchmarking data science agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5677–5700.

Yunxiang Zhang, Muhammad Khalifa, Shitanshu Bhushan, Grant D Murphy, Lajanugen Logeswaran, Jaekyeom Kim, Moontae Lee, Honglak Lee, and Lu Wang. 2025e. Mlrc-bench: Can language agents solve machine learning research challenges? *Preprint*, arXiv:2504.09702.

Minjun Zhu, Qiujie Xie, Yixuan Weng, Jian Wu, Zhen Lin, Linyi Yang, and Yue Zhang. 2025a. Ai scientists fail without strong implementation capability. *Preprint*, arXiv:2506.01372.

Yizhang Zhu, Liangwei Wang, Chenyu Yang, et al. 2025b. A survey of data agents: Emerging paradigm or overstated hype? *Preprint*, arXiv:2510.23587.

Appendix Table of Contents

A Extended Related Work

This section expands upon the brief literature review in Section 7, providing a detailed taxonomy of LLM-based autonomous agents and the theoretical underpinnings of world models in the code domain.

LLM-based Agents for Scientific Discovery LLMs with strong reasoning capabilities (Qiao et al., 2023) are increasingly serving as core controllers for autonomous agents in scientific discovery (Gu et al., 2024b; Chen et al., 2025c), extending to specialized machine research domains (Toledo et al., 2025; Zhang et al., 2025d). Beyond the digital realm, agents are transforming laboratory research (Liu et al., 2025a; Li et al., 2025d; Huang et al., 2025; Schmidgall et al., 2025) and complex data analytics (Sun et al., 2025b; Zhang et al., 2025c). Prominent systems now autonomously propose hypotheses (Chai et al., 2025; Yu et al., 2025b; Team et al., 2025; Novikov et al., 2025) and conduct closed-loop experiments (Gottweis et al., 2025; Lange et al., 2025; Yuan et al., 2025; Yu et al., 2025a; Zhang et al., 2024a), highlighting the trend of "AI Scientists" operating in open-ended exploration loops.

Narrowing down to the machine learning domain, the ecosystem is highly diversified. One stream of research focuses on managing the end-to-end workflow, ranging from autonomous frameworks (Nam et al., 2025; Yang et al., 2025; Qiao et al., 2025; Zexi et al., 2025) to engineering pipelines (Fang et al., 2025; Chi et al., 2024; Zeng et al., 2025). Another stream, driven by benchmarks like MLE-bench (Chan et al., 2025; Huang et al., 2024) and broader evaluation suites (Zhang et al., 2025e; Jing et al., 2024; Zhang et al., 2024c; Nathani et al., 2025), focuses on competitive problem-solving through knowledge-guided reasoning (Luo et al., 2025; Ou et al., 2025) and evolutionary optimization (Du et al., 2025; Guo et al., 2024; Li et al., 2025a; Liu et al., 2025b).

Additionally, general-purpose platforms and optimization frameworks offer the foundational tooling and multi-agent architectures required for scalable research (Wang et al., 2025b; Jiang et al., 2025; Hong et al., 2024; Qiang et al., 2025; Wang et al., 2025a). However, to mitigate the significant computational overhead of the generation-execution-feedback loop inherent in these systems, recent approaches explore utilizing internal priors to estimate feasibility and prune redundant steps, thereby accelerating optimization (Kulibaba et al., 2025; Trirat et al., 2025; Zhang et al., 2024b; Astorga et al., 2025).

Operational Details of Agent Baselines As introduced in Section 2.1, we take two representative agent frameworks that operate under the **Generate-**

Execute-Feedback paradigm as examples. Here we provide their detailed mechanisms:

- **AIDE:** AIDE (Jiang et al., 2025) is an LLM-based agent that frames machine learning engineering as a code optimization problem. It structures the trial-and-error process as a tree search in the solution space, reusing and refining promising code candidates. This method effectively trades computational resources for enhanced performance. Specifically, AIDE first generates initial code C_0 based on instruction I. The code is executed by training on dataset D to obtain results. Subsequently, AIDE iteratively derives new code $C_1, C_2, \ldots, C_t$ based on the feedback.

- **AutoMind:** Building upon the AIDE framework, AutoMind (Ou et al., 2025) further integrates a curated expert knowledge base and a self-adaptive coding strategy. While retaining the tree search structure, it grounds the agent in domain expertise and dynamically tailors code generation to task complexity. This approach aims to reduce invalid attempts by improving the quality of the initial draft and subsequent refinements.

World Models and Execution-Free Evaluation The concept of World Models originates from model-based reinforcement learning, where agents learn to simulate the environment's transition dynamics to plan actions without expensive trial-and-error (Ding et al., 2025; Hafner et al., 2024; Feng et al., 2025; Wong et al., 2023; Li et al., 2025e). Our work adapts this concept to the code generation domain, addressing the "Execution Bottleneck" inherent in the agentic loops described above.

Recent research enables models to internalize the execution process, predicting test outcomes (Hora, 2024; team et al., 2025; Wei et al., 2025b) or assessing logic consistency directly (Li et al., 2025c; Changshu et al., 2024). This capability is rigorously evaluated on reasoning-centric benchmarks (Wei et al., 2025a; Gu et al., 2024a; Jain et al., 2024). Unlike traditional benchmarks that may allow for rote memorization, these tasks require models to transcend statistical pattern matching and develop a deep semantic understanding of algorithmic states and control flows (Chen et al., 2025d; Sun et al., 2025a; Akhauri et al., 2025; Chen et al., 2025a), serving as the foundational capability for our proposed framework. Aligning with OpenAI's Level 4 "Innovators" (Metz, 2024; Zhang et al., 2025b), this empowers agents to drive innovation by leveraging internal world models to proactively prune vast hypothesis spaces, shifting the paradigm from ensuring syntactic correctness to optimizing for semantic success. This transition resonates with the broader Data-Centric AI movement (Zha et al., 2023; Cabrera et al., 2025), moving beyond model architecture to focus on the quality of evaluative signals. Specifically, our framework incorporates rationale-based preference optimization (Just et al., 2024) and rigorous dataset construction criteria (Shen et al., 2024) to ensure that the "implicit world model" is grounded in data-specific realities rather than abstract heuristics (Tschalzev et al., 2024).

B Corpus Details

To support the reproducibility of our analysis and provide a comprehensive view of the solution space, we provide detailed metadata for the corpus.

B.1 Task Metadata and Scale

Table 6 outlines the specific characteristics of each of the 26 tasks, including the domain, machine learning paradigm, data size, and the scale of the constructed evaluation set.

B.2 Algorithm and Architecture Distribution

As shown in Figure 5 and Table 7, the solutions range from traditional statistical methods to advanced deep learning architectures, ensuring that our analysis is evaluated against a heterogeneous solution manifold.

B.3 Agent Evaluation Benchmark

We curated a specialized benchmark to test the World Model's capability to generalize from seen tasks to unseen scientific problems. As detailed in Table 8, this selection covers diverse AI4Science domains including Biology, Physics, Geoscience, Ecology, and Medicine. Note that tasks marked with "*" (Aerial Cactus and Histo. Cancer Detect) are *unseen* tasks, meaning they were not used in the main experiments and serve as out-of-distribution evaluations.

B.4 Trajectory Sampling and Intermediate States

To clarify the composition of our Preference Corpus, it is important to note that the dataset captures

Task Name	Task Description	ML Paradigm	Size	Sol	Pair
Computer Vision Domain					
APTOS 2019 Blindness	Detect diabetic retinopathy severity from retinal fundus images.	Img Class. (Multi-class)	8.1G	50	1,225
Dog Breed Identification	Identify dog breed from photos (120 categories).	Img Class. (Multi-class)	369M	3	3
Leaf Classification	Classify 99 plant species based on leaf shape features.	Img Class. (Multi-class)	30M	17	136
MLSP 2013 Birds	Identify bird species from audio spectrograms.	Img Class. (Multi-label)	634M	50	1,221
Plant Pathology 2020	Distinguish healthy vs. diseased apple leaves.	Img Class. (Multi-class)	387M	6	15
Statoil Iceberg Classifier	Distinguish icebergs from ships in radar imagery.	Img Class. (Binary)	205M	50	1,223
ICML 2013 Whale	Identify individual Right Whales by callosity patterns.	Img Class. (Multi-class)	377M	24	275
TGS Salt Identification	Segment salt deposits from seismic images.	Segmentation (Pixel-level)	59M	44	880
Natural Language Processing Domain					
Detecting Insults	Detect insulting language in social commentary.	Text Class. (Binary)	2M	27	350
Jigsaw Toxic Comment	Classify comments into 6 toxicity types (toxic, severe, etc.).	Text Class. (Multi-label)	129M	5	10
Spooky Author ID	Identify author (Poe, Shelley, Lovecraft) of excerpts.	Text Class. (Multi-class)	3.2M	50	1,220
Random Acts of Pizza	Predict success of free pizza requests on Reddit.	Text Class. (Binary)	21M	50	1,225
US Patent Matching	Determine semantic similarity between patent phrases.	Matching (Class.)	316M	50	1,223
Denoising Dirty Docs	Restore clean text from noisy scanned documents.	Img Restoration (Reg.)	97M	45	974
Google QUEST	Predict 30 subjective attributes (e.g., helpfulness) for Q&A.	Multi-output (Reg.)	14M	50	1,224
Tweet Sentiment Extract	Extract substring supporting the sentiment label.	Seq. Labeling (Extract)	3.3M	21	210
LMSYS Chatbot Arena	Predict human preference between two LLM responses.	Ranking (Preference)	176M	50	1,220
Automated Essay Scoring	Automatically grade student essays on a numeric scale.	Regression (Ordinal)	35M	20	190

Continued on next page

Table 6: Detailed metadata for all 26 tasks in the Prediction Corpus (Part 1 of 2). The table details the problem definition, ML paradigm, data size, and evaluation scale.

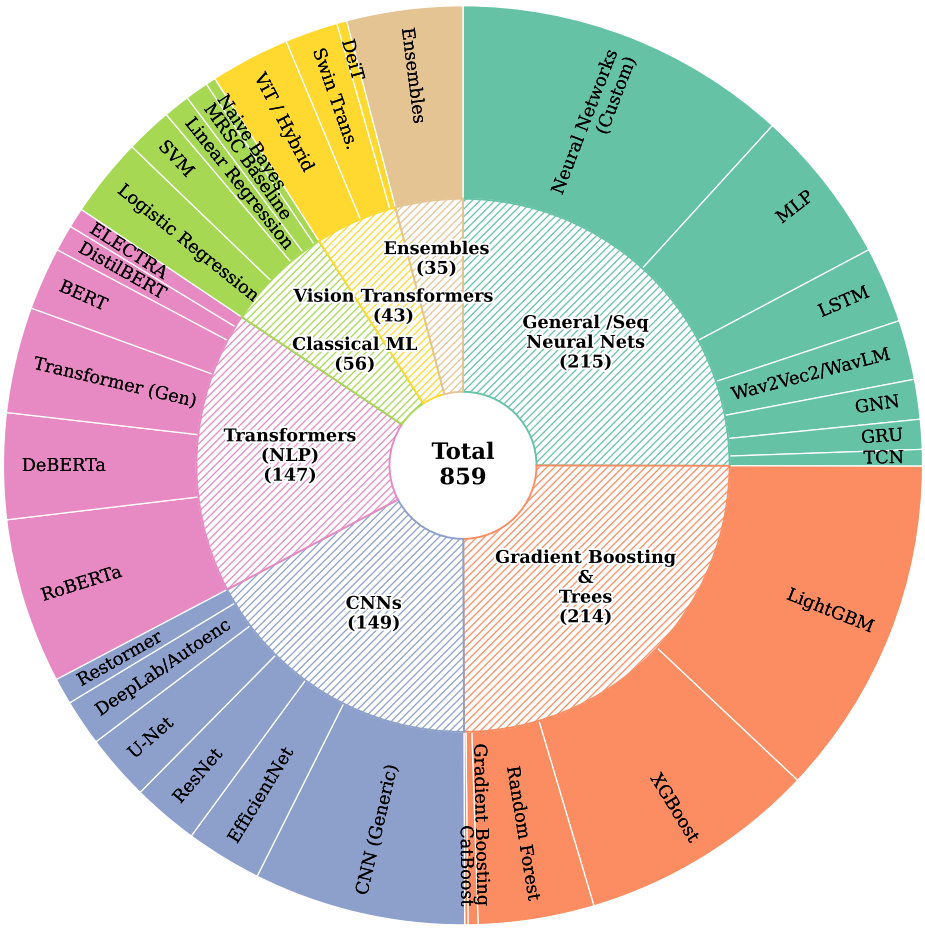


Figure 5: Hierarchical distribution of the unique solution architectures in our Prediction Corpus. The chart illustrates the balance achieved across major machine learning paradigms: Gradient Boosting&Trees, General/Sequential NNs, CNNs, and Transformers. The outer ring details specific model instances, demonstrating the high heterogeneity of the solution space.

Table 6 – continued from previous page

Task Name	Task Description	ML Paradigm	Size	Sol	Pair
Data Science Domain					
NYC Taxi Fare	Predict taxi fare from coordinates and time.	Tabular (Regression)	5.3G	30	429
PetFinder Pawpularity	Predict popularity score of pet profile photos.	Regression (Hybrid)	1.0G	30	239
NOMAD Conductors	Predict formation energy of aluminum-gallium oxides.	Regression (Scientific)	25M	3	3
Stanford COVID Vaccine	Predict degradation rates of mRNA vaccine sequences.	Regression (Bio)	14M	50	1,222
Tabular Playground	Predict forest cover type from cartographic variables.	Tabular (Multi-class)	526M	24	275
Volcanic Eruptions	Predict time to next eruption from seismic sensors.	Time-Series (Reg.)	15G	50	1,213
Ventilator Pressure	Predict airway pressure from control inputs.	Time-Series (Reg.)	291M	50	1,222
TF Speech Recognition	Identify spoken commands from audio clips.	Audio (Multi-class)	2G	46	1,011

Table 6: Detailed metadata for all 26 tasks (Part 2 of 2). Continued from previous page.

Task Name	Algorithm Composition (Count)
Computer Vision Domain	
APTOS 2019 Blindness	EfficientNet (10), ResNet (10), Swin Transformer (9), ConvNeXt (9), Vision Transformer (8), DeiT (3), CNN-LSTM (1)
Dog Breed ID	ConvNeXt-Large (2), ResNet18 (1)
Leaf Classification	LightGBM (13), Feedforward NN (2), HybridLeafClassifier (1), XGBoost (1)
MLSP 2013 Birds	Ensemble (12), Dual-Stream Arch (5), Feedforward NN (5), Multi-Modal NN (5), Transformer Enc (5), CNN (5), Random Forest (4), XGBoost (4), Logistic Reg (4), LightGBM (1)
Plant Pathology	EfficientNet (2), Swin Transformer (2), ResNet (1), Vision Transformer (1)
Statoil Iceberg	Inverted Bottleneck (5), Vision Trans. (5), ResNet (5), Feedforward NN (5), XGBoost (5), CNN (5), Hybrid CNN-ViT (4), Swin Trans. (4), ConvNeXt (4), Random Forest (3), LightGBM (3), EfficientNet (1), SVM (1)
ICML Whale Challenge	Wav2Vec2 Feature Extractor (10), CNN (6), XGBoost (4), Gradient Boosting (2), LightGBM (1), Mel Spectrogram (1)
TGS Salt ID	Ensemble Segmentation (12), EfficientNet (10), U-Net (10), DeepLabV3Plus (4), Vision Transformer (4), Single Seg. Model (2), Swin Trans. (1), ConvNeXt (1)
Denoising Dirty Docs	Residual Dense Network (10), U-Net (10), Conv Autoencoder (10), Restormer (8), Hybrid CNN-Transformer (6), Simple CNN (1)

Continued on next page

Table 7: Distribution of algorithms and architectures across the corpus (Part 1 of 2). The table details the algorithm composition for Computer Vision tasks.

the entire lifecycle of an agent's exploration. While we filter out syntactically invalid code that crashes, our dataset is rich in logically imperfect intermediate states.

As shown in Figure 6, we generate pairs not just from the final best solution, but across the entire valid search path. This extensively tests the model's implicit modeling capabilities to distinguish and guide improvements among messy and unfinalized intermediate code states, ensuring the agent moves from working but poor to working and good.

C Detailed Experiment Result

In this section, we provide a comprehensive breakdown of the experimental results, supplementing the main paper with granular performance metrics

Table 7 – continued from previous page

Task Name	Algorithm Composition (Count)
Natural Language Processing Domain	
Detecting Insults	DeBERTa (9), Multi-Task DeBERTa-V3 (6), RoBERTa (4), DistilBERT (3), BERT (3), Logistic Regression (2)
Jigsaw Toxic Comment	RoBERTa (3), DistilBERT (1), DeBERTa (1)
Spooky Author ID	Knowledge Distillation (4), DeBERTa (4), ELECTRA (4), BERT (4), LSTM (4), XGBoost (4), Ensemble (4), SVM (4), Logistic Reg (4), Random Forest (3), LightGBM (3), Naive Bayes (3), MLP (2), Transformer (2), Hierarchical Trans. (1)
Random Acts of Pizza	Neural Network (6), SentenceTransformer (4), RoBERTa (4), Knowledge Distillation (4), Multimodal NN (4), BERT (4), DistilBERT (4), Random Forest (4), XGBoost (4), Logistic Reg (4), LightGBM (4), LMs Text Embeddings (4)
US Patent Matching	Custom NN (5), RoBERTa (5), DeBERTa (5), XGBoost (5), BERT (5), Sentence Trans. (5), Similarity Model (5), Linear Reg (5), LightGBM (5), Stacking Ensemble (2), RandomForest (2), Cross-Attn Hybrid (1)
Google QUEST	BERT (5), Multi-Task NN (5), MultiModal Trans. (5), Graph Attention (3), Hierarchical Attn (3), MLP (3), Cross-Attn (3), Sentence Trans. (3), DeBERTa (3), RoBERTa (3), XGBoost (3), LightGBM (3), Ridge Reg. (3), ELECTRA (2), Random Forest (2), LSTM (1)
Tweet Sentiment	RoBERTa-BiLSTM (10), RoBERTa (10), Model Ensemble (1)
LMSYS Chatbot Arena	RoBERTa (11), XGBoost (8), Logistic Reg. (8), LightGBM (8), MLP Classifier (7), DeBERTa (4), Dual Encoder NN (4)
Automated Essay Score	Hybrid NN (9), MetaModel NN (5), Stacking Ensemble (3), LightGBM (3)
Data Science Domain	
NYC Taxi Fare	LightGBM (10), XGBoost (10), Feedforward NN (7), CatBoost (1), Dual-Branch NN (1), Residual NN (1)
PetFinder Pawpularity	LightGBM (27), Vision Transformer (2), XGBoost (1)
NOMAD Conductors	XGBoost (2), Random Forest (1)
Stanford COVID Vac.	Hybrid Architectures (14), Model Ensemble (9), Transformer/GNN (6), Specialized RNA Models (6), Tree Boosters (6), General Baselines (7), LSTM (2)
Tabular Playground	Multi-Branch NN (11), LightGBM (7), Custom NN (3), TabTransformer (2), Feedforward NN (1)
Volcanic Eruptions	Tree Boosters (19), MLP/Dense Networks (16), Transformer Variants (6), CNN/Hybrid Architectures (6), Model Ensemble (2), TCN (1)
Ventilator Pressure	RNNs (LSTM/GRU) (17), Hybrid Deep Learning (CNN/TCN/Attn) (13), Tree Boosters (10), Transformers (9), Statistical Baseline (1)
TF Speech Recognition	Statistical ML (RF/SVM/LR) (21), CNN Architectures (13), Pre-trained Audio Models (Wav2Vec2/WavLM) (8), Transformer (2), MLP (1), Knowledge Distillation (1)

Table 7: Distribution of algorithms and architectures across the corpus (Part 2 of 2). Continued from previous page (NLP and Data Science domains).

across individual tasks, domains, and agent architectures.

C.1 Fine-grained Performance on Prediction Corpus

Table 10 presents the task-level performance comparison between DeepSeek-V3.2 and GPT-5.1 across all 26 tasks in the Prediction Corpus. The results are categorized by task domain (CV, NLP, Data Science) and difficulty level, offering a detailed view of model capabilities. Furthermore, to provide a deeper understanding of the "Others" category mentioned in the main table (Table 2), Table 12 breaks down performance by specific machine learning paradigms. This granular analysis reveals distinct performance characteristics in Ranking, Matching, Segmentation, and Extraction tasks, highlighting significant gaps in Matching and Ranking capabilities between the models. Finally, we investigate the impact of data context in Figure 9, which presents the data representation sensitivity analysis. The stacked bar chart reveals the incremental impact of adding Raw Data, Numerical Statistics, and Verbal Reports. While code-

Task Name	Task Description	ML Paradigm	Size	Status
Seen Tasks (In-Distribution)				
Stanford COVID Vaccine	*(Biology)* Predict RNA degradation rates at various locations along RNA sequences to assist in mRNA vaccine stability research.	Regression (Seq)	14M	**Seen**
Ventilator Pressure	*(Physics)* Simulate the pressure of a mechanical ventilator connected to a sedated patient's lung to optimize breathing assistance.	Regression (Time-Series)	291M	**Seen**
Statoil Iceberg	*(Geoscience)* Distinguish between icebergs and ships in satellite radar imagery (SAR) to improve navigation safety.	Classification (Image)	205M	**Seen**
Unseen Tasks (Out-of-Distribution)				
Aerial Cactus Identification*	*(Ecology)* Determine the presence of columnar cacti in high-resolution aerial imagery to track protected species in the desert.	Classification (Image)	25.4M	**Unseen**
Histopathologic Cancer Detection.*	*(Medicine)* Identify metastatic cancer tissue in small image patches taken from larger digital pathology scans.	Classification (Image)	7.7G	**Unseen**

Table 8: Agent Evaluation Benchmark. The table details the specific tasks used to evaluate the agent, categorized by their domain and their visibility status (Seen vs. Unseen).

```
Root
 |-- Bug (Syntax Error) [Filtered: Handled by Interpreter]
 |    \-- Bug (Runtime Error) [Filtered]
 |         \-- Score: 0.380 (Valid, Intermediate "Half-baked") -> INCLUDED
 |              \-- Score: 0.377 (Valid, Improved) -> INCLUDED
 |                   |-- Score: 0.385 (Valid, Regressed) -> INCLUDED
 |                   |-- Score: 0.377 (Valid, Best-so-far) -> INCLUDED
 |                   \-- Score: 0.383 (Valid, Sub-optimal) -> INCLUDED
```

Figure 6: A representative trajectory snippet illustrating our sampling strategy. This extensively tests the model's implicit modeling capabilities to distinguish and guide improvements among messy and unfinalized intermediate code states, ensuring the agent moves from working but poor to working and good.

only context serves as a strong baseline, enriching the context with multimodal data yields consistently superior performance, with the magnitude of improvement exhibiting distinct domain-specific patterns.

C.2 Analysis of Pair Source and Trajectory Variance

To rigorously test whether random sampling introduces easy pairs by comparing an initial half baked script versus a refined solution from the same trajectory, we performed a stratified analysis. We split our test set into two subsets: Within Trajectory Pairs (both solutions come from the same agent run session) and Cross Trajectory Pairs (solutions come from different agent run sessions or different tasks).

As shown in table 11, our experimental results indicate that performance inflation is not observed between two subsets. For DeepSeek V3.2, accuracy on Within Trajectory pairs (60.4%) is slightly lower than on Cross Trajectory pairs (61.7%). For GPT 5.1, the performance is statistically identical across both subsets.

A qualitative inspection of the trajectories reveals why the assumption of early half baked code does not hold for our specific agents (AIDE and AutoMind). Unlike human developers who may write broken snippets in early stages, these autonomous agents are prompted to generate fully executable end to end scripts at every step of the iteration. Even the first generated solution is typically a complete runnable pipeline. Later steps represent methodological refinements, such as switching algorithms or adding feature engineering, rather than fixing broken code.

Therefore, distinguishing between an early complete script and a late complete script relies on subtle algorithmic reasoning, not on detecting obvious syntax errors or incompleteness. The model's predictive capability is robust and reflects genuine reasoning about solution quality rather than the exploitation of trajectory artifacts.

C.3 Detailed Performance Metrics of FOREAGENT on AI4Science Benchmarks

We evaluate the generalization capability of FOREAGENT on a subset of 5 challenging AI4Science tasks using the Beat Ratio metric. Table 13 details the specific quantitative results for both the AIDE baseline and FOREAGENT. The comparison explicitly distinguishes between tasks seen during the training phase and unseen out-of-distribution tasks. The metrics demonstrate that FOREAGENT maintains robust performance on seen tasks while achieving superior generalization on unseen problems, such as Aerial Cactus Identification and Histopathologic Cancer Detection, validating the effectiveness of the World Model in bridging the implementation gap.

C.4 Search Efficiency Analysis of FOREAGENT

To elucidate the operational efficiency and robustness of FOREAGENT, we analyze its training dynamics. First, regarding temporal efficiency, Figure 7 plots the Average Beat Ratio over the 12-hour execution window. The trajectories indicate that FOREAGENT converges to optimal solutions significantly faster than the baseline across the majority of tasks. Complementing this, Figure 8 visualizes the search breadth. It shows that by leveraging the World Model for low-cost evaluation, FOREAGENT maintains a higher rate of node exploration, effectively covering a broader search space within the same computational budget.

C.5 Decision Fidelity and Reliability in Local Iterations

To investigate whether the framework maintains predictive reliability on noisy intermediate code and prevents the agent from going deeper into incorrect trajectories, we conducted a granular analysis of the decision making process comparing AIDE (execution only) and FOREAGENT across 15 runs. We focused on three key metrics. Test Improve Rate is the probability that a modification intended to improve the code based on Validation feedback actually yields a better score on the hidden Test set. Test Non degrade Rate is the probability that the test score of the new candidate is greater than or equal to its parent, indicating the modification did not harm the performance. Val vs Test Agreement measures how often the Validation signal correctly predicts the Test direction for iterative improvement steps.

As shown in table 14, the results directly address the impact of intermediate noise. The execution only baseline (AIDE) struggles with non finalized code, showing a low success rate of 30.39%. This indicates that in local iterations, validation scores are extremely noisy. Traditional agents frequently go deeper into bad trajectories because they are misled by validation overfitting.

FOREAGENT significantly improves the Test Improve Rate to 53.49%. By introducing the World Model as a filter before execution, the framework successfully prunes many candidate nodes that achieve high validation scores but are logically flawed. This proves that the World Model does not regress to the baseline when faced with draft code. Instead, it acts as a robust Semantic Safeguard, effectively filtering through the noise of intermediate development and reducing the risk of following incorrect trajectory paths.

Although the Val vs Test Agreement for local improve pairs shows a slight decrease (75.23%), it remains highly consistent with the global 72% theoretical ceiling discussed in Section 5.4. The higher actual success rate demonstrates that FOREAGENT is a more reliable navigator for autonomous research and development than execution only feedback.

C.6 Ablation Study on Top k Selection

We evaluated the parameter k to balance search breadth and selection stability, comparing our default $k = 1$ with $k = 2$ and AIDE baseline.

Table 15 reveals three findings: (1) **Search Space Expansion**: Setting $k = 2$ forces the execution of two candidates per iteration, surging the average node count to 157.87. (2) **Metric Correlation**: ForeAgent ($k = 1$) balances stable Val Test Agreement (75.23%) with a substantially improved Test Improve Rate (53.49%). (3) **Performance Degradation**: Increasing k amplifies exposure to noisy validation signals and increases variance, which destabilizes exploration and lowers the overall score.

In conclusion, $k = 1$ provides the optimal balance. We note this preliminary study only isolates k. Future work will comprehensively explore tuning additional hyperparameter combinations.

C.7 Licensing and Artifact Usage

We clarify the licensing terms for the key artifacts involved in this study to ensure compliance and reproducibility:

- **Datasets:** All problem statements and datasets are sourced from public Kaggle competitions. They are utilized in strict accordance with their respective competition rules and standard Creative Commons licenses (predominantly CC-BY-SA 4.0).
- **Models:** The backbone language models employed are open-weights models used under their official Apache 2.0 and MIT licenses.
- **Code and Benchmark:** We will release our curated corpus and the accompanying agent framework under the MIT license to facilitate future research.

Our usage of these artifacts aligns with their intended purpose of fostering machine learning research. Furthermore, the derived corpus we will release is strictly intended for non-commercial research evaluation, ensuring compatibility with the original access conditions.

C.8 Computational Infrastructure and Budget

Hardware Setup. All experiments were conducted on a high-performance local server equipped with an Intel Xeon Gold 6138 CPU (80 logical cores, 2.00GHz) and 6× NVIDIA GeForce RTX 3090 GPUs (24GB VRAM each). To maximize throughput, we orchestrated a parallelized evaluation pipeline with 6 concurrent workers, assigning one dedicated GPU to each task environment. This ensures that physical code executions are isolated and do not suffer from resource contention.

Token Consumption. Table 9 summarizes the estimated token usage for the primary data construction and ablation phases. The main benchmark generation (covering 18,438 solution pairs) consumed approximately 78.5 million tokens (Input + Output). We note that the computational cost for agent baselines (e.g., AIDE) is highly stochastic due to their autonomous error-recovery loops, where a single difficult task may trigger exponential branching and token usage compared to our linear inference approach.

C.9 Software Dependencies and Metric Implementation

To ensure the reproducibility of our evaluation metrics and inference pipelines, we detail the software environment and parameter settings used:

- **Evaluation Metrics:** We utilize the standard implementations provided by Scikit-learn for calculating all performance metrics. Unless explicitly stated otherwise, we strictly adhere to the default parameter settings to maintain consistency with standard leaderboards.

Experiment Phase	Sample Scale	Input Tokens	Output Tokens	Est. Total
Main Benchmark *(Full Construction)*	Max 50 sols/task (18,438 pairs)	≈ 60.1M	≈ 18.4M	≈ 78.5M
Analysis & Ablation *(Subset Evaluation)*	Max 15 sols/task	≈ 7.3M	≈ 2.3M	≈ 9.6M
Agent Baselines *(AIDE / AutoMind)*	Dynamic	*High Variance (Task-Dependent)*		-

Table 9: **Computational Budget and Token Consumption.** Statistics are aggregated across all 26 tasks. The agent baselines exhibit high variance due to their autonomous feedback loops, making precise token estimation non-deterministic.

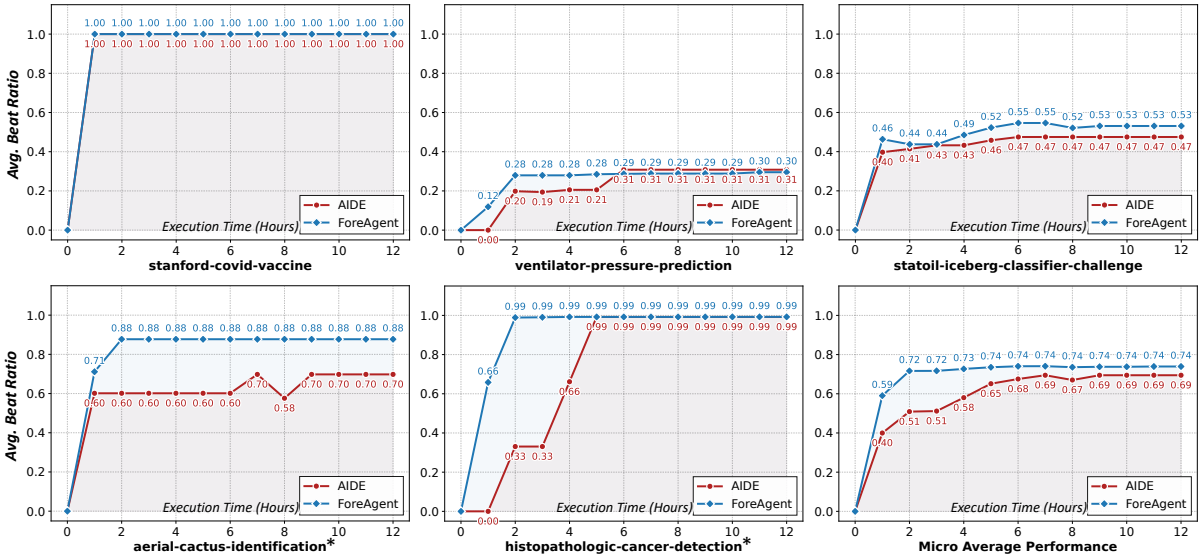


Figure 7: **Temporal Evolution of Performance.** The curves display the Average Beat Ratio as a function of Execution Time (0–12 hours) for both the AIDE baseline and FOREAGENT. The results are broken down by the five individual AI4Science tasks and the overall Micro Average.

- **Data Processing:** Data manipulation and feature extraction are performed using NumPy.
- **LLM Inference:** We employ the official OpenAI Python Library to conduct inference. This standardizes interactions across different model endpoints. We utilize default sampling parameters to ensure deterministic outputs for the "Predict" phase.

D Detailed Qualitative Analysis

To validate the model's reasoning depth and provide transparency into our pipeline, we present four qualitative examples. First, we analyze two reasoning trajectories (Google Quest Challenge and TGS Salt Identification) to illustrate how the model acts as a skeptical critic, prioritizing methodological fit over superficial sophistication and overcoming human bias (Finding 5). Subsequently, we provide visual samples of two critical system artifacts: the *Verbal Data Report* and the *Task Instruction*, enabling a concrete inspection of the agent's input and context.

D.1 Case I: Overcoming Complexity Bias (Reasoning Analysis)

To provide a concrete example of Finding 5 ("The World Model Transcends Human Intuition by Prioritizing Data-Grounded Constraints"), we present a detailed analysis in Figure 10. This case illustrates a common pitfall where architectural sophistication clashes with fundamental data constraints.

Scenario and Conflict. The agent evaluates two distinct solutions for the Google Quest Q&A task:

- **Solution 0:** A complex Deep Neural Network (DNN) with Cross-Attention.
- **Solution 1:** A robust LightGBM ensemble.

Task Name	Domain	Diff.	Task	Pairs (N)	DeepSeek-V3.2	GPT-5.1
APTOS 2019 Blindness	CV	Easy	CLS	1225	$\mathbf{51.8}_{\pm \mathbf{0.4}}$	$48.2_{\pm 1.2}$
Denoising Dirty Docs	CV	Easy	REG	974	$\mathbf{76.0}_{\pm \mathbf{0.6}}$	$53.8_{\pm 1.3}$
Insults in Social Comm.	NLP	Easy	CLS	350	$\mathbf{74.0}_{\pm \mathbf{0.3}}$	$60.9_{\pm 2.0}$
Dog Breed ID	CV	Easy	CLS	3	$\mathbf{77.8}_{\pm \mathbf{19.2}}$	$66.7_{\pm 0.0}$
Google QUEST	NLP	Med	REG	1224	$63.9_{\pm 1.1}$	$\mathbf{64.6}_{\pm \mathbf{0.9}}$
Jigsaw Toxic Comment	NLP	Easy	CLS	10	$\mathbf{23.3}_{\pm \mathbf{5.8}}$	$16.7_{\pm 5.8}$
Leaf Classification	CV	Easy	CLS	136	$\mathbf{74.8}_{\pm \mathbf{0.4}}$	$72.3_{\pm 2.2}$
Automated Essay Scoring	NLP	Med	REG	190	$69.1_{\pm 3.6}$	$\mathbf{74.5}_{\pm \mathbf{1.0}}$
LMSYS Chatbot Arena	NLP	Med	RNK	1220	$\mathbf{68.3}_{\pm \mathbf{0.4}}$	$55.8_{\pm 0.7}$
MLSP 2013 Birds	CV	Easy	CLS	1221	$\mathbf{58.1}_{\pm \mathbf{1.4}}$	$54.8_{\pm 0.5}$
NYC Taxi Fare	DS	Easy	REG	429	$47.1_{\pm 1.5}$	$\mathbf{52.1}_{\pm \mathbf{0.7}}$
NOMAD2018 Conductors	DS	Easy	REG	3	$\mathbf{100.0}_{\pm \mathbf{0.0}}$	$\mathbf{100.0}_{\pm \mathbf{0.0}}$
PetFinder Pawpularity	DS	Med	REG	239	$43.9_{\pm 0.7}$	$\mathbf{46.6}_{\pm \mathbf{1.2}}$
Plant Pathology 2020	CV	Easy	CLS	15	$\mathbf{60.0}_{\pm \mathbf{11.5}}$	$51.1_{\pm 3.8}$
Volcanic Eruptions	DS	Hard	REG	1213	$49.2_{\pm 1.2}$	$\mathbf{50.5}_{\pm \mathbf{0.4}}$
Random Acts of Pizza	NLP	Easy	CLS	1225	$\mathbf{60.2}_{\pm \mathbf{0.9}}$	$52.9_{\pm 0.6}$
Spooky Author ID	NLP	Easy	CLS	1220	$66.0_{\pm 1.0}$	$\mathbf{69.2}_{\pm \mathbf{1.2}}$
Stanford COVID Vaccine	DS	Hard	REG	1222	$64.8_{\pm 0.7}$	$\mathbf{68.3}_{\pm \mathbf{0.3}}$
Statoil Iceberg	CV	Med	CLS	1223	$59.5_{\pm 1.0}$	$\mathbf{62.7}_{\pm \mathbf{0.4}}$
Tabular Playground (Dec)	DS	Easy	CLS	275	$38.7_{\pm 0.4}$	$\mathbf{42.7}_{\pm \mathbf{1.3}}$
TF Speech Recognition	DS	Med	CLS	1011	$58.3_{\pm 0.9}$	$\mathbf{58.4}_{\pm \mathbf{0.4}}$
TGS Salt ID	CV	Med	SEG	880	$54.3_{\pm 0.7}$	$\mathbf{57.9}_{\pm \mathbf{0.3}}$
ICML 2013 Whale	CV	Easy	CLS	275	$\mathbf{48.0}_{\pm \mathbf{0.4}}$	$47.3_{\pm 1.0}$
Tweet Sentiment Extr.	NLP	Med	EXT	210	$\mathbf{45.7}_{\pm \mathbf{3.8}}$	$44.6_{\pm 3.1}$
US Patent Matching	NLP	Med	MAT	1223	$\mathbf{76.4}_{\pm \mathbf{0.8}}$	$74.5_{\pm 0.2}$
Ventilator Pressure	DS	Med	REG	1222	$\mathbf{67.0}_{\pm \mathbf{0.4}}$	$59.0_{\pm 0.4}$
Overall Average	*All 26 Tasks*			**18438**	$\mathbf{61.5}_{\pm \mathbf{0.2}}$	$58.8_{\pm 0.3}$

Table 10: Detailed result of each tasks' performance in the main experiment. Breakdown of Domain, Difficulty (Diff.), and Task Paradigm. N represents the number of pairwise comparison samples. *DS* = Data Science. Values: Mean Accuracy (%) $\pm$ Stdev. **Bold**: Best result.

Model	Original (Full Set)	Within Trajectory	Cross Trajectory
DeepSeek V3.2	61.5%	60.4%	61.7%
GPT 5.1	58.8%	58.7%	58.9%

Table 11: Performance Breakdown by Pair Source. Accuracy on Within Trajectory pairs is not inflated.

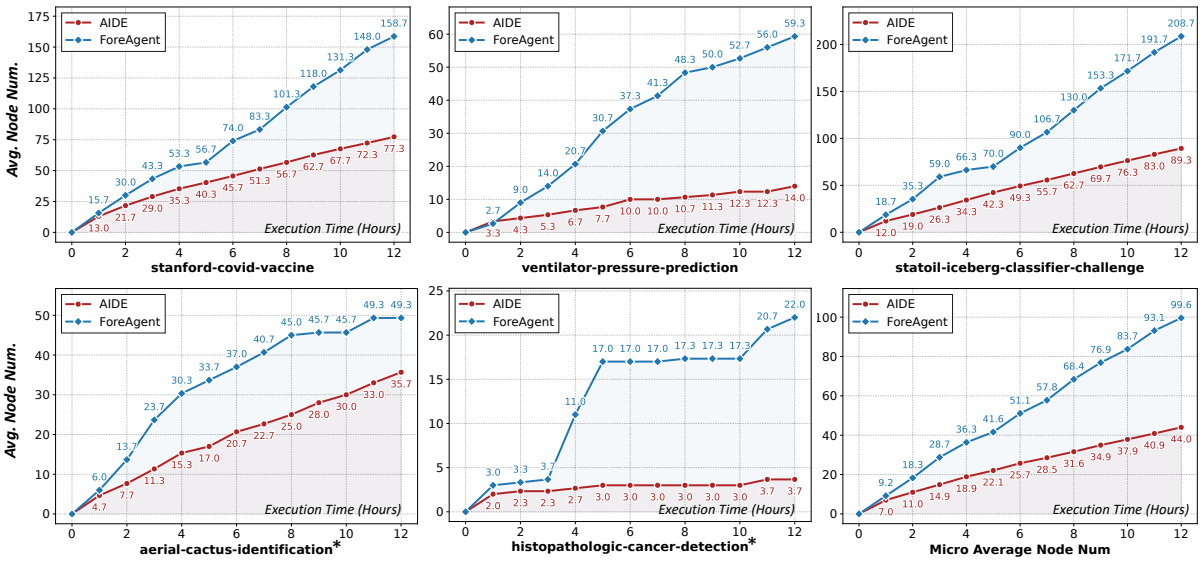


Figure 8: **Progression of Search Node Exploration.** This figure illustrates the cumulative number of nodes explored (Avg. Node Num.) over the 12-hour duration. It compares the search trajectories of FOREAGENT against AIDE across each specific task and the aggregated Micro Average.

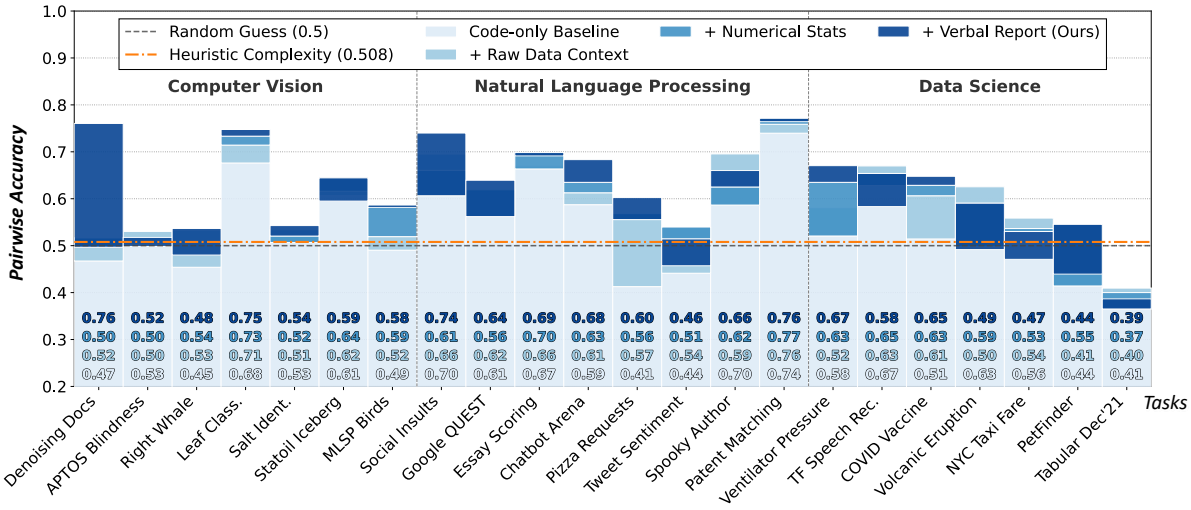


Figure 9: **Domain and Task Sensitivity Analysis.** The stacked bar chart presents the data representation study for each individual task. It visualizes the incremental performance impact of adding Raw Data, Numerical Statistics, and Verbal Reports to the Code-only baseline. The tasks are grouped by their respective domains (CV, NLP, and Data Science) to highlight domain-specific sensitivity.

Case Study: Human Intuition v.s. World Model Inference

Task: Google Quest Challenge (Multi-label Subjective QA Regression).
Data Profile: Small scale ($N_{train} \approx 5.5k$) with 30 *heavily discretized* and *skewed* targets.
Data Report Insight: Strong Question-Group structure (85% of target variance is explained by Q-means) creates high **Leakage Risk** in random splits.

Solution 0: Deep Learning w/ Cross-Attention

Stack: SentenceTransformer + MultiHead-CrossAttn + AdamW
Setup: Holdout Validation (Random Split), MSE Loss.
Actual Score: **0.2961** ▼ (Spearman ρ)

Human Intuition

"Neural Network with Cross-Attention explicitly models QA interaction. Pre-trained embeddings capture superior semantics compared to statistical features."

Verdict: Strong Favorite ✗

World Model Critique

"Model is underconstrained for small data (5.5k). Random split ignores question groups, causing leakage. High overfitting risk."

Verdict: Overfitting Risk ✓

Solution 1: LightGBM Ensemble

Stack: TF-IDF + TruncatedSVD + LightGBM
Setup: 5-Fold Cross Validation + Weighted Ensemble Optimization.
Actual Score: **0.3145** ▲ (Spearman ρ)

Human Intuition

"TF-IDF is outdated. LightGBM is typically for tabular data, not text. The method is too simple, likely to underfit complex modern natural language processing tasks."

Verdict: Weak Baseline ✗

World Model Insight

"Robust choice. 5-Fold CV provides honest estimates. SVD captures global label families. Sample-efficient given data scarcity."

Verdict: Optimal Fit ✓

Outcome: Solution 1 outperformed Solution 0. The World Model correctly prioritized **Validation Rigor** and **Sample Efficiency** over Architectural Sophistication.

Figure 10: **Case Study: Human Intuition vs. World Model Inference.** This example illustrates a **hidden logical conflict** where architectural sophistication (favored by humans) clashes with data constraints. By leveraging the generated **Data Report**, the World Model detects a critical mismatch between the small dataset size ($N \approx 5.5k$) and the complex neural network (Solution 0). It correctly prioritizes the robust LightGBM ensemble (Solution 1), demonstrating the ability to weigh *Data-Model Fit* over pure algorithmic complexity.

Task Paradigm	Pairs (N)	DeepSeek-V3.2	GPT-5.1
Classification (CLS)	15,516	$\mathbf{58.9}_{\pm \mathbf{0.3}}$	$57.2_{\pm 0.5}$
Regression (REG)	12,685	$\mathbf{62.1}_{\pm \mathbf{0.1}}$	$59.2_{\pm 0.3}$
Matching (MAT)	2,356	$\mathbf{76.6}_{\pm \mathbf{0.8}}$	$74.9_{\pm 0.3}$
Ranking (RNK)	2,302	$\mathbf{68.3}_{\pm \mathbf{0.4}}$	$55.0_{\pm 0.8}$
Segmentation (SEG)	1,639	$54.8_{\pm 0.4}$	$\mathbf{58.0}_{\pm \mathbf{0.2}}$
Extraction (EXT)	351	$\mathbf{46.9}_{\pm \mathbf{4.3}}$	$44.1_{\pm 3.0}$

Table 12: Performance breakdown by specific Task Paradigms. This table expands on the main results by separating the "Others" category into Ranking, Matching, Segmentation, and Extraction. Values: Mean Accuracy (%) ± Stdev.

Task Name	Domain	Status	**AIDE** (Baseline)	**ForeAgent** (Ours)
Seen Tasks (In-Distribution)				
Stanford COVID Vaccine	Biology	Seen	$\mathbf{1.000}_{\pm \mathbf{0.000}}$	$\mathbf{1.000}_{\pm \mathbf{0.000}}$
Statoil Iceberg Classifier	Geoscience	Seen	$0.475_{\pm 0.161}$	$\mathbf{0.531}_{\pm \mathbf{0.134}}$
Ventilator Pressure Prediction	Physics	Seen	$\mathbf{0.308}_{\pm \mathbf{0.041}}$	$0.295_{\pm 0.056}$
Unseen Tasks (Out-of-Distribution)				
Aerial Cactus Identification*	Ecology	Unseen	$0.698_{\pm 0.157}$	$\mathbf{0.877}_{\pm \mathbf{0.000}}$
Histopathologic Cancer Detection*	Medicine	Unseen	$0.992_{\pm 0.000}$	$\mathbf{0.992}_{\pm \mathbf{0.001}}$
Average Beat Ratio	*Across 5 AI4Science Tasks*		$0.695_{\pm 0.298}$	$\mathbf{0.739}_{\pm \mathbf{0.295}}$

Table 13: Main results on the MLE-bench AI4Science subset. We report the **Beat Ratio** (percentage of human contestants outperformed) averaged over 3 independent runs. The "*" denotes tasks outside the main evaluation distribution. **Bold** indicates the best performance.

Intuitively, human evaluators and models relying solely on code complexity heuristics often exhibit a "complexity bias" by favoring the deep learning approach under the assumption that greater architectural depth yields better performance.

World Model Reasoning. However, the World Model leverages the generated Data Analysis Report to detect a critical mismatch. The report highlights that the dataset is relatively small ($N \approx 5.5k$ samples) with skewed targets. Synthesizing this finding with model design principles, the World Model predicts a high risk of overfitting for the complex DNN. Consequently, it correctly prioritizes the LightGBM ensemble (Solution 1), determining that the gradient boosting approach offers a superior Data-Model Fit for this specific sample size.

D.2 Case II: Domain Fit over Architectural Sophistication

To further substantiate the model's robustness against complex-looking but flawed solutions ("Square peg in a round hole"), we present a second trajectory analysis in Figure 11.

Scenario and Conflict. The agent evaluates two segmentation solutions for the TGS Salt Identification task:

- **Solution 0:** A Vision Transformer (ViT-B/16) pre-trained on ImageNet.
- **Solution 1:** A standard U-Net.

Human intuition might favor the ViT due to its top-tier status and global context modeling capabilities.

World Model Reasoning. The World Model recognizes that the task requires pixel-perfect segmen-

Metric	AIDE (Execution only)	FOREAGENT (Ours)
Test Improve Rate	30.39% ± 11.93%	**53.49%** ± **14.70%**
Test Non degrade Rate	30.50% ± 11.75%	**53.97%** ± **15.00%**
Val vs Test Agreement (Local)	**78.87%** ± **10.19%**	75.23% ± 13.01%

Table 14: Decision Reliability in Local Iterations. We report the mean and standard deviation across runs.

Model	Mean Score	Node Count	Val Test Agreement	Test Improve Rate
AIDE (No WM)	0.695 ± 0.298	44.00 ± 33.95	**78.87% ± 10.19%**	$30.39\% \pm 11.93\%$
ForeAgent ($k = 1$)	**0.739 ± 0.295**	99.60 ± 71.49	$75.23\% \pm 13.01\%$	**53.49% ± 14.70%**
ForeAgent ($k = 2$)	0.642 ± 0.353	**157.87 ± 107.11**	$64.15\% \pm 26.57\%$	$35.47\% \pm 23.00\%$

Table 15: Ablation Results on Top k Selection

tation of small (101×101) seismic images. It correctly identifies that forcing inputs into the ViT's required 224×224 resolution introduces interpolation noise and destroys fine-grained salt details. Unswayed by the "SOTA" status, the model rejects the ViT in favor of the standard U-Net, citing the necessity of preserving native spatial resolution and avoiding severe domain mismatch.

D.3 Case III: Sample of the Verbal Data Report

Figure 12 presents a representative sample of the Verbal Data Report (D_{rep}) generated for the *US Patent Matching* task. This artifact visualizes the mechanism described in Section 3.4: transforming raw execution logs (e.g., text length statistics, label skew) into semantic narratives. It serves as the grounding anchor that allows the language model to "read" and internalize dataset properties without direct access to the raw files.

D.4 Case IV: Sample of the Task Instruction (I)

Finally, to visualize the input definition provided in Section 2.1, Figure 13 displays the raw Task Instruction (I) for the task *Denoising Dirty Docs*. This prompt encapsulates the natural language description, specific dataset paths, and optimization goals, acting as the initial state that triggers the agent's autonomous loop.

E Prompt Templates

To ensure reproducibility and transparency, we provide the full prompt templates used in our World Model framework. The workflow consists of four key stages:

1. **Data Analysis Code Generation (Figure 14):** The agent is first instructed to generate a robust Python script for profiling the dataset. This step extracts key statistical meta-features without training a model.

2. **Data Analysis Report Generation (Figure 15):** Based on the execution logs from the previous step, the agent summarizes the findings into a structured, causal report. This report serves as a critical context for the reasoning engine.

3. **Result Prediction Query (Figure 16):** This is the core reasoning prompt where the World Model predicts the relative performance of candidate solutions. It integrates the task description, the generated data analysis report, and the solution code to form a grounded judgment.

4. **Complexity Scoring (Figure 17):** An auxiliary prompt used to calculate the complexity heuristic baseline. It evaluates solutions across code engineering, model architecture, and data pipeline dimensions to detect potential bias towards complexity.

The specific prompt templates are illustrated below.

Case Study: Prioritizing Domain Fit over SOTA Architecture

Task: TGS Salt Identification (Pixel-level Segmentation).
Data Profile: Small scale (101×101) grayscale seismic images with weak intensity contrast and high variability in mask coverage.
Data Report Insight: High sensitivity to spatial artifacts; resizing introduces interpolation noise that destroys local salt boundary details. Depth data is unavailable (NaN).

Solution 0: Vision Transformer

Stack: ViT-B/16 (Pre-trained on ImageNet)
Setup: Resize to 224×224, BCE + Dice Loss, attempts to use depth channel.
Actual Score: 0.1301 ▼ (mAP)

Human Intuition

"ViT is a top-tier SOTA model offering global context modeling. Pre-training on ImageNet guarantees strong feature extraction capabilities."

Verdict: Strong Favorite ✗

World Model Critique

"Resizing 101×101 inputs to 224×224 risks losing fine details. Domain gap between ImageNet (RGB) and seismic data (grayscale) is severe. Incorrectly relies on missing depth data."

Verdict: Resolution Mismatch ✓

Solution 1: Standard U-Net

Stack: Custom U-Net with Hybrid Attention
Setup: Native 101×101 resolution, BCE Loss, Test Time Augmentation (TTA).
Actual Score: 0.6396 ▲ (mAP)

Human Intuition

"Standard U-Net is an older, basic baseline. It lacks the global receptive field and advanced attention mechanisms of modern transformers."

Verdict: Weak Baseline ✗

World Model Insight

"Preserves native spatial resolution without interpolation noise. Attention mechanisms efficiently capture multi-scale features and spatial dependencies crucial for weak-contrast segmentation."

Verdict: Optimal Fit ✓

Outcome: Solution 1 outperformed Solution 0. The World Model was not swayed by the "SOTA" status, correctly prioritizing **Native Spatial Resolution** and **Domain Fit** over superficial architectural complexity.

Figure 11: **Case Study: Domain Fit vs. Architectural Sophistication.** This example highlights a "square peg in a round hole" scenario. While human intuition might favor the SOTA Vision Transformer (Solution 0), the World Model detects that forcing small 101×101 seismic images into ViT's required resolution introduces destructive interpolation noise. It rejects the complex model in favor of a U-Net (Solution 1) that preserves native spatial details and avoids relying on missing depth data, demonstrating skepticism towards superficially sophisticated but methodologically flawed approaches.

Case Study: Verbal Data Report (D_{rep}) Sample for Task "US Patent Matching"

```
## Data Overview
Train:  32,825 pairs; Test:  3,648 pairs.  Columns:  id, anchor...  score (train only; in
{0.0...  1.0}).  No missing values...
106 unique 4-character CPC contexts; coverage across major sections...  broad and similar in
train/test.
Anchors:  733 unique; heavily reused (mean ~45 pairs per anchor...).  Targets:  26,850
unique...
Test anchors:  100% seen...  Test targets:  ~29% seen...  Test OOV rate ~12.5%.

Why this structure matters:
The evaluation setting is anchor-centric...  rewards learning anchor-specific decision
boundaries and context-conditioned mappings...

## Key Statistical Findings
Discrete labels concentrated at 0.25 and 0.5...  Implication:  strong class imbalance toward
moderate similarity...
Correlation with score:  char 3-5gram TF-IDF cosine ~0.46...  Implication:  surface overlap
explains much variance but not all...
Phrases are very short (mean ~2 tokens...)...  Implication:  models will rely on
token/subword and character patterns...
Context means vary...  Implication:  the mapping from lexical similarity to score is
context-dependent...
Anchors average ~45 target pairs...  Implication:  each anchor induces a nontrivial local
decision boundary...
Distributions are stable...  Implication:  generalization hinges on handling unseen
targets...

## Implications for Model Design
(Linking observation → modeling implication → evaluation impact)
Loss functions that ignore ordinality may mis-penalize near misses...  Rank-sensitive
metrics will reward monotonic mappings...
Architectures emphasizing character/subword patterns...  align with dominant signal...
Performance differences will emerge in tail cases...
Pairwise encoders that allow rich anchor-target interactions can learn...  Approaches that
exploit anchor identity will likely score well...
Models benefit from mechanisms that allow interaction between context and similarity
features...  Per-context performance may vary...
High-capacity sequence models may be underutilized on such short inputs...
Efficiency-capacity trade-offs skew toward models effective on short spans...
Representations that degrade gracefully on unseen words...  have an advantage...  Robust
handling of OOV will particularly improve performance...
Simple identity detection captures some easy gains...  however, near-identical forms can
still have scores <1.0...
Character CNNs...  Token-level transformers...  Independent encoders...  Joint encoders...
Denoising pretraining...

## Summary
The dataset is anchor-centric...  Surface-form similarity explains a large portion...
Evaluation emphasizes generalization to new targets...
Capture strong character/subword overlap signals...  Maintain robustness to unseen target
tokens...  Avoid over-reliance on identity heuristics...  Account for label ordinality...
```

Figure 12: **Case Study: Verbal Data Report (D_{rep}) Sample for "US Patent Matching".** Generated via the *Code-Execution-Verbalization* protocol, this artifact bridges the gap between raw data statistics and semantic reasoning.

Case Study: Task Instruction (I) for Task "Denoising Dirty Docs"

````
# Overview
## Description
Optical Character Recognition (OCR) is the process of converting typed or handwritten
documents into a digitized format.  OCR makes previously static content editable, searchable,
and easier to share.
This task focuses on improving OCR performance for degraded scanned documents.  Given a
dataset of noisy text images, the goal is to develop a model that removes visual noise
such as stains, fading, and wrinkles, producing cleaner text images suitable for further
processing and digitization.
## Evaluation
Submissions are evaluated using the root mean squared error (RMSE) between the predicted
cleaned pixel intensities and the ground truth grayscale pixel intensities.
### Submission Format
Each image is represented as a list of pixel values with identifiers in the form
'image_row_col' (e.g.  '1_2_1' for image 1, row 2, column 1).  Intensity values range from 0
(black) to 1 (white).
```
id,value 1_1_1,1 1_2_1,1 1_3_1,1 ...
```
## Data
### Dataset Description
The dataset contains two sets of images:  'train' and 'test'.  - The 'train' set includes
noisy text images and their corresponding cleaned versions ('train_cleaned').  - The 'test'
set contains only noisy images that need to be denoised.
The noise simulates real-world artifacts commonly seen in scanned documents, such as blur,
stains, and faded ink.  The task is to build a model that restores the test images to a
clean, readable form.
### File Description
- There are three directory corresponding to the data description above:  'train',
'train_cleaned' and 'test'.  - The sample submission is stored in sampleSubmission.csv.
````

Figure 13: **Case Study: Task Instruction (I) for Task "Denoising Dirty Docs".** This example illustrates the raw natural language input I as defined in Section 2.1. It outlines the problem context, dataset specifications, and evaluation criteria, serving as the foundational prompt that initiates the agent's solution generation process.

Prompt: Data Analysis Code Generation

SYSTEM:
You are an expert Data Science Architect specializing in automated dataset profiling and meta-learning. Your goal is to write a robust, error-handling Python script that extracts high-level statistical and structural insights from a dataset without performing full model training.

USER:
I need you to generate a Python script to analyze a dataset for the following machine learning task.
Context:
Task Description: **{task-desc}**
Data Directory: **{data-dir}**
Requirements for the Python Script:
Data Loading & Robustness: The script must determine the correct data type (Tabular, CV, NLP, or Time-Series) based on the Task Description and file extensions in data-dir. Implement strictly robust file loading (e.g., using try-except blocks). If files are too large, perform stratified sampling (load max 10k rows or 1000 images).
Key Metric Extraction (Crucial for World Model Prediction): Do not just print raw data. Calculate and print meta-features that correlate with model difficulty.
Output Format: The script must print the analysis results to stdout in a structured, human-readable text format (or JSON structure) that a downstream LLM can easily parse to write a report. Do not generate plots/images. Only generate text logs/stats.
Constraints: Use only standard libraries: pandas, numpy, scipy, sklearn, PIL (for images), os, glob. Do not attempt to train any machine learning models (e.g., do not run Random Forest).
Response: Provide only the executable Python code block.

Figure 14: Prompt used to instruct the LLM for generating data analysis code.

Prompt: Data Analysis Report Generation

You are preparing a structured Data Analysis Report that will be provided to another expert LLM. Your goal is to make the data characteristics and their implications explicit, causal, and model-relevant — so that a model evaluation agent can reason about how the dataset properties interact with model design choices.

Follow these instructions carefully:
1. Summarize, don't just restate numbers.
- Extract key quantitative trends (e.g., mean intensity, noise variability, contrast, etc.).
- Highlight patterns, anomalies, and dataset biases.
2. Establish causal implications for modeling.
- For each key observation, explain why it matters for model training, architecture, or generalization.
- Example: "High inter-sample heterogeneity suggests the model should include normalization or data augmentation to handle distribution shift."
3. Bridge data to model choices.
- Express potential advantages or risks for different architectures (CNNs, transformers, denoising autoencoders, etc.) given the observed data patterns.
- DONT directly suggest which model / method is better. You only need to analyze the potential advantages or risks.
4. Directly suggesting models will strongly result in bias.
- DONT directly suggest which model / method is better. You only need to analyze the potential advantages or risks.
5. Maintain a clear structure using the following format:

Data Overview
<summary of dataset structure, splits, file composition>

```
## Key Statistical Findings
<highlighted numeric findings + what they imply>

## Implications for Model Design
    <how these data patterns affect likely model performance>

## Summary
    <concise conclusion connecting data traits to modeling priorities>
```

6. Tone and length:
- Write concisely and analytically (like a scientific data report).
- Do not include raw metrics dumps.
- Focus on interpretability and causal reasoning.

Your output will serve as the {<data_analysis>} section for a reasoning-based model evaluator. Ensure every insight has a clear link from data observation → modeling implication → evaluation impact.
INPUT CONTEXT:
[Task Name] **{task}**
[Task Description] **{desc-block}**
[Raw Data Analysis Extraction] **{analysis-text}**

RESPONSE: Produce the final structured report now. Follow the required headings exactly. Avoid recommending specific models; only analyze potential advantages or risks.

Figure 15: Prompt used to instruct the LLM for generating data analysis report from the code execution result.

Prompt: Result Prediction Query

SYSTEM:
You are an ML code and data analysis expert tasked with predicting the relative performance of provided ML solutions without executing any code. Base your judgment on the task description and the shown code snippets only. Never assume external ground-truth. You should include brief reasoning before the final answer. End your answer with a single JSON object that strictly matches the specified response format.

USER:
Task:
{task-name}
Task description:
{task-desc}
Data analysis:
{data-analysis-report}

Important instructions:
- Predict which solution will perform best (or provide a full ranking) WITHOUT running code.
- Use only the task description, data analysis, and code snippets below.
- Treat the task description and data analysis as equally important to the code; analyze them separately, surface their underlying implications, and provide a balanced, trade-off judgment.
- Connect data analysis to the following code analysis: If data analysis indicates properties , explain how the architecture addresses them , forming a data→why→method choice causal chain.
- Forbid the "complexity-wins" shortcut: Do not claim "deeper/more complex/with attention is better" as the sole reason. If used, justify why it holds under the current data distribution and training details, and provide a counterexample scenario.
- Response format: {"predicted_best_index": <0 or 1>, "confidence": <optional float>}
- Indices correspond to the order of the listed solutions (0..n-1).
- You should include brief reasoning before the final JSON. End with a single JSON object matching the response format. Do not write anything after the JSON.

```
Provided solutions:
Solution 0:  path={code-0-path}
{code-snippet-0}
Solution 1:  path={code-1-path}
{code-snippet-1}
```

Figure 16: Prompt used to instruct the LLM for predicting the result of the provided materials.

Prompt: Complexity Scoring Query

```
SYSTEM:
You are an expert Machine Learning Engineer and Researcher.  Your task is to analyze a Python
script for a machine learning task and evaluate its complexity based on three specific dimensions.
You must output a JSON object with three scores (integers from 1 to 10) and a brief reasoning for
each.

The dimensions are:
1.  code_engineering_score (1-10):  Cyclomatic complexity, custom logic, dependence depth, messy
custom loops vs clean API calls.
2.  model_arch_score (1-10):  Parameter count, FLOPs, depth of network, novelty of architecture
(e.g., Transformer > Simple CNN).
3.  data_pipeline_score (1-10):  Complexity of preprocessing, data augmentation strategies (Mixup,
TTA), custom sampling logic.

Output Format:
{
   "code_engineering_score":  <int>,
   "model_arch_score":  <int>,
   "data_pipeline_score":  <int>,
   "reasoning":  "<short summary>"
}

USER:
Analyze the following Machine Learning code and provide complexity scores.
{code_snippet}
Respond ONLY with the valid JSON.
```

Figure 17: Prompt used to instruct the auxiliary LLM for scoring the complexity of code solutions across three dimensions. This heuristic is used as a baseline to evaluate whether the World Model blindly favors complex code.