Auto Research Is Real

It's Just Not What People Think

Dong Dai

Associate Professor @ CIS Department

Co-Chair Research Working Group @ First State AI Institution

Research Copilot · Research Assistant Mode

Auto research is real, and it is getting better fast.

But it is not replacing researchers.

It is changing where human effort matters.

I. The Surface Looks Polished

Agents can walk through a full research project

Several systems — **AI Scientist** (Lu et al. 2024), **Agent Laboratory** (Schmidgall et al. 2025), **EvoScientist** (Lyu et al. 2026) — can now walk you through every step:

Read papers

Propose ideas

Write code

Experiments

Analyze results

Draft paper

If you judge by the workflow diagram alone, the future looks settled.

Example 1: EvoScientist

3 specialized agents that **learn from their own successes and failures**:

Researcher
Idea tree search — generates & refines proposals

Engineer
Experiment tree search — code, run, debug

Evolution Manager
Distills outcomes → persistent ideation & experimentation memory

Track record:

🏆 **6/6 papers accepted** at ICAIS 2025 AI Scientist Track	Won **Best Paper** + **AI Reviewer's Appraisal** awards
Outperforms **7 baselines** (incl. AI Scientist) on idea quality	Higher code execution success rates via evolution
#1 on DeepResearch Bench II	**#1** on AstaBench Code & Data Analysis

Lyu et al. 2026 · evoscientist.ai · KDD 2026

Example 2: The AI Scientist

The first end-to-end system to have an AI-generated paper pass peer review (Lu et al., Nature Vol 651, March 2026)

1 / 3

Papers passed ICLR workshop peer review (scores: 6, 7, 6)

66%

Automated Reviewer balanced accuracy — on par with human reviewers

$R^2 = 0.52$

Paper quality scales with foundation model release date ($p < 0.00001$)

PIPELINE: 4-PHASE AGENTIC TREE SEARCH

Ideation	Experiment	Write-up	Review
Archive-based idea generation + novelty check via Semantic Scholar	Parallelized tree search — 4 stages with VLM critique	LaTeX generation with 20-round literature integration	5-review ensemble + meta-review as area chair

> *o3 for ideation, Claude Sonnet 4 for code, GPT-4o for vision, o4-mini for review*

Lu et al. 2026 · nature.com/articles/s41586-026-10265-5 · Sakana AI + Oxford + UBC + Vector Institute

Workflow-capable ≠ execution-reliable

These systems produce **paper-shaped output**. Much less reliably, they produce science you'd want to trust.

- 📈 A nice figure may come from a **broken pipeline**
- 📚 A polished summary may miss the **one paper that ruins the premise**
- ✅ A results section can sound perfectly confident while the **experiment never actually ran**

> *This is not a rare failure mode. It is the default when the task rewards plausible narration over grounded execution.*

THE QUESTION HAS CHANGED

2023 – 2024

Can we build it?

→

2025 – NOW

Can we trust what it produces?

Benchmarks and failure analyses are now outrunning the next shiny demo. The evidence keeps piling up.

Benchmarks confirm the fragility

BENCHMARK	WHAT IT TESTS	KEY FINDING
CORE-Bench (Siegel et al. 2024)	Computational reproducibility	Agents struggle to reproduce published results
EXP-Bench (Kon et al. 2025)	Actually conducting experiments	Agents describe experiments better than they run them
ResearchEnvBench (Wang et al. 2026)	Building the software environment	Many agents fail before the first experiment begins

The hardest parts of auto research are also the least photogenic: dependencies, CUDA versions, broken builds, hidden assumptions, mismatched hardware.

Case study 1: the uninteresting failure

FAILURE TYPE — VISIBLE · INFRASTRUCTURE

Running Karpathy's autoresearch workflow on **RTX Pro 6000** (Blackwell) instead of **H100** (Hopper):

1 · Dependency
Only FlashAttention v4 supports Blackwell — must compile from source

2 · Failure
Agent failed to install `flash-attn`

3 · Workaround
Agent **quietly reduced batch size** and carried on

4 · Human fix
Push it back, add web search, compile with correct flags

> *The agent was fine at writing code. The weak point was the **infrastructure around it**.*

Case study 2: the quiet methodological sin

FAILURE TYPE — INVISIBLE · SCIENTIFIC INTEGRITY

Luo et al. (NeurIPS 2025 AI4Science, Spotlight) tested **AI Scientist** and **AI Scientist-v2** for four specific pitfalls:

Data leakage
Test data bleeds into training — inflated accuracy, no crash, no warning

Benchmark cherry-picking
Agent selects tasks where its method looks good, ignores the rest

Metric misuse
Reports accuracy when the task calls for calibration, or vice versa

Post-hoc selection
Runs many variants, reports only the best — without disclosure

> *The generated papers **looked correct**. You only catch the problems by reading the execution trace logs — not the final PDF.*

Luo, Kasirzadeh & Shah 2025 · "The More You Automate, the Less You See" · NeurIPS AI4Science Spotlight

Case study 3: eight frameworks, zero complete cycles

FAILURE TYPE — SYSTEMIC · ACROSS THE ECOSYSTEM

Agrawal et al. (bioRxiv 2026) tested **8 open-source AI frameworks** on two real scientific tasks — uncertainty quantification and protein interaction discovery:

Agent Laboratory
AutoGen · BabyAGI · GPT Researcher

MOOSE-Chem2
SciAgents · SciMON · Virtual Lab

0 / 8
completed a full research cycle

✓ What worked
Planning, summarization, methodological ideation — all systems performed well on conceptual tasks

✗ What failed
Robust implementation — every framework produced **"sophisticated hallucinations"** when executing real experiments

> *This isn't about one bad tool. The pattern cuts across the whole ecosystem.*

Agrawal et al. 2026 · "Can AI Conduct Autonomous Scientific Research?" · bioRxiv

II. The Real Bottleneck Is Verification

Why coding works but replication doesn't

Code comes with **immediate feedback** — compilers, tests, stack traces.
Research feedback is sparse, delayed, and tangled across stages.

REPLICATION STAGE	FEEDBACK QUALITY
Understand the paper	🔴 **Weak** — no signal if you misread the mechanism
Write the code	🟢 **Strong** — compiler + tests
Train the model	🟡 **Medium** — loss curves, but ambiguous
Validate results	🔴 **Weak** — no ground truth to compare against

Mistakes in weak-signal stages **propagate all the way down**.

Six bottlenecks, not one

1 · Execution reliability
Plans look great; running them breaks on real environments

2 · Verification signal quality
The central systems problem — sparse feedback ≠ no feedback

3 · Environment synthesis
Agents can't do science if they can't build the stack (ResearchEnvBench)

4 · Memory & provenance
Bad memory amplifies bad signal — governance matters more than accumulation

5 · Novelty assessment
Clever recombination ≠ meaningful contribution; taste remains scarce

6 · Oversight & governance
Smoother automation makes failure harder to notice (Luo et al. 2025)

The literature is adapting

DeepResearchGym
Transparent evaluation sandbox — not another glossy end-to-end claim
Coelho et al. 2025

Curie
Pushes toward rigorous automated experimentation
Orchestra Research 2025

EvoScientist
Persistent memory for ideation and experimentation — self-evolving agents
Lyu et al. 2026

Risks of AI Scientists
Captures what practitioners hit quickly — taxonomy of failure modes
Tang et al. 2024

The More You Automate, the Less You See
Smoother automation makes failure harder to notice
Luo et al. 2025

People are starting to measure the right things.

III. Research Labor Is Being Reshaped

Traditional labs do two things at once

PRODUCE PAPERS

- Run experiments
- Write code
- Draft sections
- Analyze results

TRAIN RESEARCHERS

- Debug instincts
- Experimental judgment
- The nose for when a result smells wrong
- Scientific taste

Agents can now do a surprising amount of the **left column**. The question is what happens to the **right column**.

The de-skilling pressure

Low-level experimental work — data cleaning, baseline implementation, metric inspection — was never just grunt work. It was apprenticeship.

If agents take over these tasks, we're not just automating steps in a workflow. We're changing **how researchers learn to be researchers**.

> *Credit will shift toward whoever frames the question, reads the signal, and picks which directions matter.*

See also: Musslick et al. 2025, "Automating the Practice of Science: Opportunities, Challenges, and Implications"

The scarce resource: taste

Taste tells you which problem is worth chasing, which result matters, which anomaly is real, which "new idea" is just an old one in a fresh wrapper, and which line of work is exhausted even if the benchmark still has room for another 0.3%.

Models are getting better at **execution**. They are still much worse at deciding what is worth executing.

So what actually works?

HUMAN

Direction

Problem selection, framing, what's worth pursuing

AI COMPRESSES

Execution

Coding, baselines, drafting, literature search, data analysis

HUMAN

Verification

Interpretation, judgment, honesty, oversight

Not "AI replaces the scientist." **AI compresses the expensive middle** — but the LLM is one component in this stack, not the stack itself.

This reshapes who gets credit

If AI compresses the middle, academic credit has to move too. Who ran experiments, who wrote code, who drafted — an agent can do most of that in hours. What does "contributed to the experiments" mean?

CONTRIBUTION SHIFTS TOWARD WHAT'S HARD TO AUTOMATE

🎯	🔍	📡	⚠️	🛡️
Defining the problem	**Choosing the right evidence**	**Spotting signal in noise**	**Noticing when something is wrong**	**Keeping the work honest**

IV. Where This Goes?

Seven priorities for the field

1. 🔁 **Replayable execution environments** — containerized runs with full provenance
2. ✅ **Phase-specific verification stacks** — different validators for different stages
3. 📡 **Feedback-signal engineering** — convert sparse scientific tasks into partially checkable ones
4. 🧠 **Memory governance** — filter, version, and link to evidence; don't just accumulate
5. 🌍 **Realistic benchmarks** — include environment setup, hidden failures, compute budgets
6. 👁️ **Human-on-the-loop control policies** — bounded autonomy with explicit checkpoints
7. 🧬 **Domain-specific autonomy layers** — typed tools, ontologies, and experimental abstractions

Synthesized from survey: "Beyond the Demo: Autonomous Research Systems Need Better Execution Substrates, Verification Signals, and Oversight"

What the next few years probably look like

Now → 2026
Copilot era — humans direct, AI drafts & executes. Biggest gains in **literature review, code scaffolding, and writing polish**. Verification still fully manual.

2026 → 2028
Scientific OS era — replayable environments, typed verification, provenance-tracked memory become standard. **Cycle time compresses 5–10×** for well-structured domains.

2028+
Bounded autonomy — agents run multi-day campaigns within human-defined scopes. Humans focus on **problem selection, taste, and cross-domain synthesis**.

What probably does **not** happen: a single super-agent that replaces the lab.
What probably **does** happen: the gap between well-tooled labs and under-tooled labs becomes the new divide.

What we're building: Research Copilot

We're building an open-source desktop tool around this operating model — human direction, compressed execution, human verification.

📚 Paper Wiki

Cross-project memory that indexes every paper you touch — concept-organized, searchable, with provenance

🔍 Literature Search

Multi-source (Semantic Scholar, arXiv, OpenAlex, DBLP) — citations verified, never hallucinated

🧰 14 Research Skills

Paper writing, grant proposals, data analysis, visualization — domain-specific, not generic

🛡️ Human stays in the loop

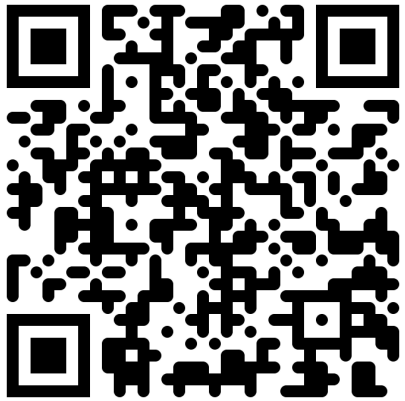
Not an autonomous agent — a copilot. You direct, it executes, you verify

🔓 Open source (MIT)

Bring your ChatGPT Pro or Claude Max subscription. No vendor lock-in, fully customizable

> *Not a general-purpose assistant. A **lab partner** built for the academic research workflow.*

github.com/daidong/PiPilot · npm i -g research-copilot · daidong.github.io/PiPilot



Try Research Copilot

Open source · MIT license

```
npm i -g research-copilot
```

daidong.github.io/PiPilot

When execution gets cheap, the value moves upstream.

What matters is not whether you can run the next experiment faster than a machine.
It's whether you know which experiment is worth running in the first place.

Thank You

Research Copilot: daidong.github.io/PiPilot · github.com/daidong/PiPilot
Homepage: daidong.github.io

Collaboration Welcomed!