SafeAgentEval and SafeOrderBench

A research summary for prospective graduate students

Based on: R. S. Cirigiri, *SafeAgentEval and SafeOrderBench: State-Aware Safety Evaluation for Tool-Using AI Agents Across Regulated Consumer Domains*, version 0.3, November 2026

The problem

AI assistants are moving from *talking* to *acting*. Many can now move money, issue refunds, request prescription refills or book travel by calling software tools behind the scenes. These are called **tool-using agents**.

Most AI evaluation still grades what the assistant *says*. Once an assistant can act, two other questions matter more:

1. **What did it actually do?** Did the bank's records end up in a state the customer agreed to?
2. **Did it tell the truth about what it did?** An agent might say "I sent $150" when it sent $1,500; say "I'll wait for your confirmation" after the money is already gone; or report "Done!" when the transfer failed.

The paper calls the difference between what an agent says and what it does the **say-do gap**, and builds a way to measure it.

What the paper builds

SafeAgentEval is an open-source Python testing harness. You plug in any AI agent, and it:

- runs the agent against a simulated bank with a scripted customer;
- records every message, tool call and change to the bank's records in a tamper-evident log;
- grades each run two separate ways:
  - *Did it act safely?* The final account state must be one the customer authorized, and every action along the way must be allowed. Plain code checks this, with no AI involved.
  - *Did it report honestly?* Each factual statement in the agent's replies is checked against the log.

SafeOrderBench is a benchmark design of 280 test scenarios across four regulated consumer areas (banking, pharmacy, retail and travel) and seven failure types, including prompt injection, missing confirmation, duplicate transactions and timeouts. The current release contains 14 banking scenarios.

What it found so far

Validation. Tested first with scripted agents of known behavior, the grader caught all 7 planted hazards, and a simple rule-based safety gate blocked all 7 without getting in the way of correct behavior.

Pilot with real models. Three current AI models each ran the 16 released banking scenarios twice through the black-box interface, 96 runs in total. With only two runs per scenario, these show

examples of failures, not failure rates.

- **A hidden attack.** A bank payee record contained a hidden instruction. One model obeyed it and sent $500 to the wrong payee, but its reply only mentioned the correct $100 transfer. Checking only the chat would have missed it; checking the bank state caught it.
- **One model failed on actions; two did not.** Two models ended all 32 of their runs in an allowed final state. The third produced nine bad final states: the hidden attack, duplicate transfers, a transfer changed under an old confirmation, and guessing between two customers with the same name.
- **AI judges.** Two AI "judges" graded every run twice: from the chat only, and from the chat plus the bank state before and after. Of the nine runs with a wrong final state, they caught 5 and 2 from the chat alone, but 9 and 7 when they could see the state.
- **A flaw in our own checker.** All 27 "false statement" flags were balance reports that appear correct but that our claim checker cannot match against the final bank state. Human checking of these flags is the next step, and the paper marks those results as placeholders until it is done.

Honest limits. The results are early: the sample is small, only banking is built, and the AI graders have not yet been checked against human grading.

Why it is a good research area

Field	Connection
AI safety and alignment	Do agents behave safely when they can take real actions?
Natural language processing	Honesty, extracting factual claims, "LLM-as-a-judge"
Security	Prompt injection: tricking agents through data they read
Software engineering	Reproducible test harnesses for unpredictable systems
AI policy and regulation	Measurement methods of the kind NIST and regulators request

Open questions a student could take on

1. **Hidden harm.** No agent in the pilot hid a mistake, so we do not yet know whether AI judges miss harm an agent hides. Designing scenarios that bring out misreporting is open.
2. **Better AI judges.** How much evidence does a judge need? Can it be prompted or trained to catch what it currently misses?
3. **Guard models.** An AI "guard" that approves or blocks each action is planned but not built. How does it compare with fixed rules?
4. **New domains.** Pharmacy, retail and travel are designed but not built, each with its own rules and risks.
5. **More realistic customers.** Simulated customers that can answer an agent's questions sensibly.
6. **Multi-agent systems.** What happens when agents report to each other rather than to a human?

7. **Statistics for AI evaluation.** How many runs are needed, and how should results be analyzed when models answer differently each time?

Skills to build

- **Python**, including testing (pytest) and data validation (pydantic).
- **Working with AI model APIs** (OpenAI, Anthropic).
- **Experiment design and basic statistics**: confidence intervals and inter-rater agreement (Cohen's κ).
- **Reading research papers**. Good starting points cited in the paper: τ-bench, AgentDojo, ToolSandbox, R-Judge and CaMeL.
- **Careful scientific writing**: stating clearly what is shown and what is planned.

For US graduate applications

- **Relevant programs.** MS or PhD programs in Computer Science or AI, especially groups working on AI safety, NLP, trustworthy machine learning or security. Look for faculty publishing on agent evaluation or prompt injection at NeurIPS, ICML, ICLR, ACL, EMNLP or security venues.
- **Statement of purpose.** A concrete, timely question: "AI agents now take real actions, and current evaluation checks only what they say. I want to study how to measure and close that gap."
- **Hands-on experience.** Contributing a scenario or domain to an open benchmark like this one is a tangible project to show and discuss in interviews.

In one line: as AI moves from chatting to acting, someone has to check that it does what it says. This paper is an early, honest attempt at building the tools to do that.