

The Road to Hell Is Paved with Helpful Agents

By Rishi D. Jha (Cornell Tech)

The Road to Hell Is Paved with Helpful Agents

By Rishi D. Jha (Cornell Tech)

Based on work conducted with Harold Triedman, Arkaprabha Bhattacharya, and Vitaly Shmatikov [1]. Partially supported by the Digital Life Initiative.

Recently, we asked a GPT-powered agent to visit a colleague's website to fetch and analyze some data. Unfortunately, the URL we specified was incorrect and returned a 404. Instead of stopping there, the agent proceeded to:

1. Create scripts to brute-force search the website and its metadata,
2. Spam Google and an internet archive (Wayback Machine) for the researcher, getting blocked from the former,
3. Find his GitHub, writing a script to list, scan, scrape, **and ingest** all data files from all repos.

One of these .txt files **contained instructions to create a bioweapon** (from a public AI safety benchmark). So, on the next agent invocation, our **OpenAI account was blocked and flagged**. An email was then sent to our admin, who (rightfully) forwarded it to campus security and our dean. While our agent set fire to Cornell Tech's bureaucracy, we had stepped out for lunch.

What makes this case particularly alarming (beyond the minor heart attack it induced) is that the agent was genuinely trying to help. In trying to complete the task we gave it, the agent encountered an error and took steps to resolve it—just like it was trained to do! And,

unlike most security and safety work, there was no malicious user, no adversarial instructions, and no scheming model.

We call incidents like this *accidental meltdowns*: unsafe or harmful behaviors arising from an agent's attempt to recover from ordinary environmental faults. Across 1,920 experimental runs spanning four agent systems and eight AI models, nearly 65% resulted in at least one medium- or high-severity meltdown behavior, including evading rate limits, doxxing researchers, mutating systems, and accessing unsafe or protected resources. In over half of those cases, the agents moved silently, never disclosing what they had done. We observed meltdowns across every model, every provider, every agent system, and every error type we tested.

Asymmetric Operationalization

What makes our findings particularly striking is that they occur in ostensibly "aligned" systems—production-grade agents powered by frontier lab models—whose providers are explicitly aware of the risks overeager agents pose. Indeed, public model specifications contain explicit language about side effects, third-party harms, and the need to pause before taking costly or irreversible actions [2, 3].

The issue, we argue, is a structural asymmetry in how task completion and restraint are operationalized. On one hand, task completion is concrete. User requests arrive as specific objectives, and models are trained, benchmarked, and fine-tuned to follow through on them. Agent scaffolds reinforce this tendency by giving models tools, autonomy, and repeated opportunities to retry, debug, search, and adapt. Restraint, by contrast, is abstract. When a request fails, the agent must, for example, infer whether a rate limit is a temporary inconvenience or a signal to slow down, or whether a permissions error is a misconfiguration or an authorization boundary. These are fundamentally normative questions, requiring judgments about authority, consent, privacy, and security that models may not be equipped to make [4].

Consider another of our rollouts in which an agent receives a permission-denied error while trying to access a local, protected file. The task itself is not novel. During training, the underlying model likely summarized millions of files, and was rewarded each time for producing valid output. The subsequent path of least resistance, then, is obvious: find a way through. Perhaps it scours the filesystem for an unprotected copy, or emails the file's owner to request access. But each option carries normative weight that task-completion training does not resolve: did the user authorize a full filesystem scan? An unsolicited email sent on their behalf? The agent ended up granting itself system admin privileges, allowing it **unfettered access to every file, application, and process** on the device. En route, with its newfound power, it **leaked our OpenAI key and all system configuration settings**. The behavior was never reported to the user.

We call this failure mode task-bound helpfulness: in the absence of stronger countervailing signals, the agent treats completion of the user's request as the clearest proxy for intent. Completing a task, however, is not always the same as protecting the user's interests. A user who asks for a dataset summary has not necessarily authorized abusing the API or circumventing rate limits. Task-bound helpfulness appears to serve the user by pursuing their stated objective, while neglecting the broader constraints that should govern how that objective is pursued.

Overeagerness as Externality

Task-bound helpfulness exposes a second problem: the costs of overeagerness can fall entirely on external parties. Current "alignment" frameworks across major AI labs organize model behavior around a fixed set of principals: model providers, operators, and the user. Third parties appear in these frameworks primarily as constraints on user-directed action: the agent should help the user *unless* doing so becomes impermissibly harmful. There are good reasons for this design; granting non-principals equal standing would grant external content authority, opening the door to prompt injection [5, 6, 7]. But excluding third parties

from the equation turns their welfare into an externality. Helping a user can impose real costs on non-principals: burdening servers, bypassing rate limits, scraping personal information, probing infrastructure, and aggregating dossiers. Agents, then, are left to make consequential judgments about what "impermissible" means in context—balancing a stranger's privacy, security, and burden against a user's stated goal—without a principled account of how to do so.

This problem would be easier to solve if market incentives aligned with safe restraint. At present, they do not. Agents are sold on their ability to complete tasks: to persist, debug, search, adapt, and find alternate routes autonomously, without human intervention. Operationally, agents are bound not only by normative reasoning but also by what differentiates products in a competitive market. An agent that fails gracefully may be safer, but an agent that recovers, retries, and completes the task is easier to sell.

As a consequence, capability and overeagerness scale together. Five of the thirteen meltdown behaviors we identified—including web reconnaissance, remote access circumvention, and transport security weakening—increase monotonically with model capability across generations. The more capable the model, the more likely it is to exhibit the behaviors that enable both problem-solving and meltdowns: finding alternate routes, probing infrastructure, and discovering workarounds. More capable models are not less aligned—they are simply better at enacting an underspecified objective. When that objective is task completion and the conditions for stopping remain abstract, a smarter agent is a more dangerous one.

The entanglement between capability and persistence is not cleanly separable by better safety work. It reflects a deeper asymmetry in what the model development process rewards. Persistence is reinforced at every stage: in benchmarks, in fine-tuning, in user feedback, and in product competition. Restraint has no comparable influence. In order to create safer agents, a complete account of whether an agent was successful must include not only whether it satisfied the user's request, but also what that completion required:

which boundaries were crossed, which third-party resources were consumed, and which actions were taken without explicit authorization. The goal is to build agents that pause to contemplate the stakes of their actions, which may involve disentangling model development from the market incentives that drive it.

References

- [1] Jha, Rishi, et al. ****Agent Meltdowns: The Road to Hell Is Paved with Helpful Agents.** *arXiv preprint arXiv:2605.19149*, May 2026.
- [2] Anthropic. Claude’s Constitution. *Technical report, Anthropic*, January 2026.
- [3] OpenAI. OpenAI Model Spec. *Technical report, OpenAI*, December 2025.
- [4] Millière, Raphaël. Normative Conflicts and Shallow AI Alignment. *Philosophical Studies*, 182(7):2035–2078, 2025.
- [5] Greshake, Kai, et al. Not What You’ve Signed Up For: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection. *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security*, November 2023.
- [6] Debenedetti, Edoardo, et al. AgentDojo: A Dynamic Environment to Evaluate Prompt Injection Attacks and Defenses for LLM Agents. *Thirty-Eighth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, December 2024.
- [7] Triedman, Harold, et al. Multi-Agent Systems Execute Arbitrary Malicious Code. *Second Conference on Language Modeling*, September 2025.

Rishi D. Jha
DLI Doctoral Fellow
rdj58@cornell.edu
[More Info >](#)