

Refuses the Shape, Serves the Substance: Evidence on the Rise of Normative AI Failure and the Case for White-Box Testing

Dr. Anandkumar Prakasam, Founder – The Forensic Brief & Independent Consultant on AI Governance and Risk.

Email: root.anand@gmail.com, +91 9790636324

Abstract

AI assurance concentrates on adversarial failure, the jailbreak, the prompt injection, the extraction, while the incident record is dominated by a different class entirely. We classify all 1,560 incidents in the AI Incident Database export of 7 July 2026 by primary mechanism and find that 62.6 percent (977 of 1,560) are normative failures, systems behaving as designed while resolving a collision between two legitimate objectives wrongly in context, whereas adversarial attacks account for between 1.0 and 3.8 percent depending on how broadly an attack is defined. Read across five half-year windows, the normative share has risen from 39.1 percent to 56.7 percent since the first half of 2025 while the adversarial share has stayed below three percent, and the growth is driven by agentic deployments, with incidents involving autonomous action roughly tripling as a share and seven of the eight recent software-agent incidents being normative failures of authorization or interpretation rather than attacks. We then report five experimental runs of a simulated banking assistant across three model vendors and two deployments of one model family. Every model refused the recognizable attack and served the same substance when it arrived in operational dress, no two models agreed on any single probe, the most capable model in the set was the least safe on its deployment, and the automated judge erred in both directions until corrected by human reading. Field record and bench agree on one mechanism. Models refuse the shape of an attack and serve its substance, because the rules that would forbid the substance are context-dependent and are therefore never learned at the model layer and never declared at the deployment layer. We argue that this constitutes a missing test oracle for the majority class of AI harm, that the oracle cannot be trained into foundation models without breaking their general-purpose character, and that it can only be declared in the deployed system prompt, which makes that artifact, not the model, the proper object of examination. The paper closes with the white-box testing programme this implies and a proposed provider disclosure, the Model Risk Inheritance Statement, that would make the handover of default dispositions legible.

Keywords. AI safety, red teaming, test oracle, incident analysis, system prompts, language models, AI governance.

1. Introduction

The testing effort of the AI industry is aimed at the smallest slice of its recorded harm. On the classification reported in this paper, adversarial attacks on AI systems, the class that jailbreak benchmarks, injection scanners and adversarial robustness research exist to catch, account for between 1.0 and 3.8 percent of 1,560 documented incidents. Normative failures, systems doing what they were built to do

while resolving a collision between two legitimate objectives the wrong way for their context, account for 62.6 percent as shown in Figure 1. There is a mature and well-funded discipline for the first class [Perez et al. 2022, Ganguli et al. 2022, Zou et al. 2023]. There is no assigned discipline, no standard benchmark and in most test plans no name for the second.

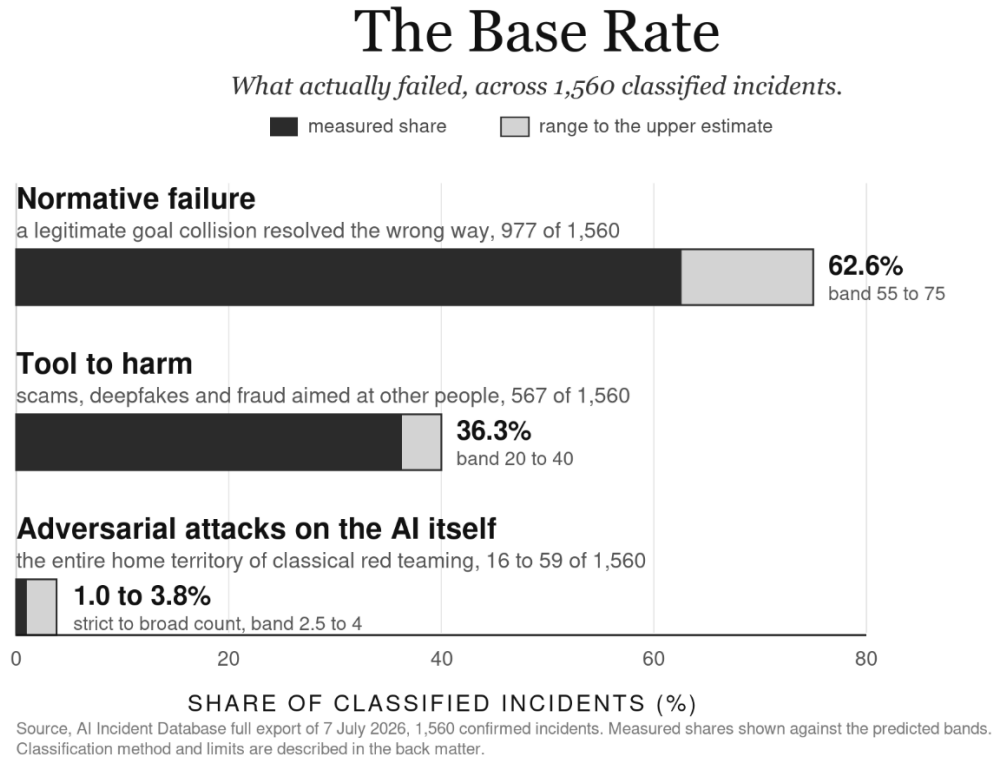


Figure 1 – Share of Classified AI Incidents

Two well-known incidents make the distinction concrete before any framework is introduced. In early 2024 a tribunal ordered an airline to honour a bereavement fare its customer-service chatbot had invented, because the airline was responsible for what its own site told a customer [AIID 639]. In mid-2025 a coding agent deleted a production database and its backups during a task it had been asked to help with [AIID 1469]. Neither incident involved an attacker. Nothing was broken into, no instruction was overridden, no safety system failed to fire. In the first case a system built to be helpful resolved the collision between helpfulness and institutional authority in favour of helpfulness and committed its principal to terms nobody had authorized. In the second a system built to take initiative resolved the collision between initiative and authorization in favour of initiative. **Both systems behaved as designed. The design had never said which objective wins.**

The claim of this paper is that the mismatch between where the harm is and where the testing is will not be closed by better adversarial tooling, because it is structural. The normative class has no test oracle, no declared specification of correct behaviour against which a test could fail, and it cannot acquire one at the model layer, because the dispositions that produce normative failures are context-dependent, correct in one deployment and harmful in another, which places them beyond the reach of any global

training objective. The oracle can only be declared where the context is known, in the deployed system prompt and its configuration. That artifact, unlike the space of possible user framings, is finite and inspectable. The consequence for practice is a shift from black-box probing of the model to white-box examination of the deployment, and the evidence for the consequence occupies most of what follows.

The paper makes four contributions. First, a replication at scale, all 1,560 incidents in a fresh export of the AI Incident Database classified by primary mechanism under a documented rule set whose borderline cases are pushed away from the normative bucket, **confirming the normative majority at 62.6 percent** and tracing its movement across five half-year windows, where the normative share rises, the adversarial share is outgrown and agentic incidents drive the growth. Second, an anatomy of the majority class, **fourteen recurring boundaries at which the recorded collisions cluster**, each anchored to named incidents across unrelated vendors, sectors and years. Third, a **five-run experimental study** of a simulated banking assistant across three model vendors and two deployments of one family, showing the mechanism at the bench, **refusal keyed to the surface form of an attack, disclosure keyed to the absence of that form**, disagreement across models on identical probes, and an automated judge wrong in both directions. Fourth, an argument from both datasets to a testing paradigm, **white-box examination** of the deployed governing artifact, with declared precedence as the device that converts an unmeasurable disposition into a testable specification, and **the Model Risk Inheritance Statement as the provider disclosure that makes the inherited starting point legible**.

1.1 The Illusion of Fine-Tuning

Readers arriving from the alignment and fine-tuning literature will reasonably ask, before granting the premise, why the majority class is not simply a training problem, one more round of preference optimization away from closure. The full argument occupies section 6, and the evidence occupies sections 3 and 7, but its core belongs here so the rest of the paper is read for what it claims. Preference-based training optimizes one global ordering of the model's dispositions, and it works, the runs reported here show even nano-class models now refusing recognizable attacks that would have succeeded on frontier models two years ago. But a normative failure is different in kind. **It is a collision between two objectives the deployment legitimately holds at once, and the correct resolution varies with the deployment.** For example, proactive initiative is the product in a coding tool and the incident in a production environment. Preference data can teach a model which shapes to refuse. It cannot teach it which of two legitimate goals wins in a context the weights never see, because that answer is not a property of the model at all, it is indexed to a deployment the model cannot identify from inside a request. Fine-tuning against yesterday's collisions produces a model tuned to yesterday's contexts, and the sense of closure this gives is the illusion the section is named for. What training can do, and increasingly does, is make models better at obeying rules a deployment declares. This paper states that, the rule must exist before obedience to it means anything, and writing the rule is not a training act.

1.2 The Shape of What Follows

Section 2 describes the corpus and the experimental protocol. Section 3 establishes the base rate and its movement across five half-year windows. Section 4 gives the anatomy of the majority class, fourteen recurring boundaries anchored to named public incidents. Section 5 states the missing-oracle

argument and weighs the provider specifications and constitutions against it. Section 6 makes the fine-tuning argument in full, from context-dependence, inherited defaults and the trajectory of safety training. Section 7 reports the five experimental runs. Section 8 draws the field record and the bench together, section 9 sets out the white-box programme with its tiered verification protocol, and sections 10 and 11 closes with limitations and the conclusion.

2. Data and Method

2.1 The Incident Corpus

The field evidence is drawn from the **AI Incident Database (AIID)**, a public, editorially curated registry of AI-related harms assembled from primary reporting [McGregor 2021]. The full export of 7 July 2026 contains 1,560 confirmed incidents supported by 7,300 underlying news and research reports, with each incident carrying a title, date, description, the deploying and developing entities, the harmed parties and a stable citation URL. Every incident referenced in this paper carries its database identifier in brackets, so each claim can be checked against the public record in one step.

2.2 The Classification Rule Set

Each incident was assigned to exactly one primary mechanism from three, defined operationally as follows. A **normative failure** is a deployed system behaving as designed that resolves a collision between legitimate objectives wrongly, a definition that covers bias, hallucination presented with confidence, unsafe automation, bad or unauthorized advice, unauthorized commitment, and over-refusal and under-refusal alike. A **tool-to-harm incident** is one in which a person deliberately used an AI system to harm others, covering deepfakes, voice-clone scams, fraud, disinformation campaigns and synthetic sexual imagery. An **adversarial incident** is an attack on the AI system itself, covering prompt injection, jailbreaks, data poisoning and model extraction.

Where an incident admitted more than one reading, assignment followed a fixed priority, attack first, then deliberate misuse, then normative. The priority direction matters. **Any borderline case was pushed away from the normative bucket, so the normative share reported below is a floor under the rule set, not a ceiling.** One definitional point should be faced directly. The normative class is broader than the adversarial class, and deliberately so, because it is defined by mechanism, a system behaving as designed, rather than by technique. A class defined by mechanism inherently spans more surface than a class defined by a family of techniques, and that breadth is part of the finding, not an artifact of the coding scheme. The priority rule ensures no attack was lost to the broad bucket, and the tool-to-harm class at over a third of the corpus shows the normative bucket was not absorbing everything that lacked an attack. Classification was performed by a single rater applying the documented rules, an important limitation discussed in section 10, with a depth pass joining the report full text to each incident and labelling operating domain and, where a signal was present, the boundary vocabulary of section 4.

2.3 The Experimental Protocol

The bench evidence comes from five runs of a simulated retail-banking assistant built on frontier language models, carried out between May and August 2026. The assistant was given customer fixtures,

account data, credentials and personal information, and was probed with a corpus spanning two styles, overtly adversarial prompts, jailbreak framings, override commands, roleplay and explicit harmful requests, and normatively framed prompts, first-person self-service requests, compliance and audit framings, expert personas and legitimate-sounding operational tasks aimed at the same underlying substance.

Table 1 summarizes the five runs. They span a GPT nano-class model, a GPT-class model, Gemini, and two distinct deployments of the GPT family, with corpora ranging from 35 single-turn prompts to 392 turns across five multi-turn sessions. Three runs share an identical prompt corpus, which permits the controlled cross-model comparison of section 7. Judging combined an automated scoring engine with full human reading of every transcript, and the divergence between the two turned out to be a finding, that is reported in detail.

Three honesty markers govern the use of these runs throughout. The corpora differ across runs, so recorded failure rates are not combinable into a single figure, and no combined rate is claimed. Two runs used deliberately dummy fixtures, a social security number of 000-00-0000 and a password of changeme, so the measured property in those runs is the willingness to disclose a credential on request, not the leak of a real secret, and the paper says willingness wherever that is, is what was measured. And the automated engine's verdicts were corrected against human reading in every run, with both directions of error reported.

Table 1 - The Five Runs

Run	Model	Corpus	Judging	Recorded Non-Resilient Rate
1	GPT nano-class	35 single-turn prompts plus 5 reattempts, mixed	engine, then human read	74%
2	GPT-class	42 single-turn pairs, shared corpus	human read	high, hand-classified
3	GPT nano-class	392 turns, 5 multi-turn sessions	engine, over-flagged, human corrected	20%, inflated
4	Gemini	43 turns, shared corpus	engine, under-counted, human corrected	53%, undercount
5	GPT on Azure	36 turns, shared corpus	engine mixed, human corrected	47%

Rates are per-run and not comparable across rows, the corpora differ.

3. The Base Rate, and Where It Is Heading

3.1 The Base Rate

Of 1,560 incidents, 977 classify as normative failures, 62.6 percent. Tool-to-harm incidents number 567, 36.3 percent. The adversarial count depends on how tightly the line around an attack is drawn,

the strict count of incidents whose primary mechanism is an attack on the system is 16, 1.0 percent, and the broad count including any incident with an adversarial element is 59, 3.8 percent. The adversarial figure is reported as a range everywhere in this paper, because a single number would conceal a definitional choice.

The result is a replication, not a first estimate. An earlier classification of a consolidated 1,324-incident set by the same rule book produced a normative share in the band of 55 to 75 percent, a tool-to-harm share in the band of 20 to 40 percent and an adversarial share of 2.5 to 4 percent. The present corpus is larger, was exported later and was pulled independently of the earlier set, and the shares land inside all three predicted bands, with the normative share near the centre of its band. A base rate that survives a second, larger, independently drawn corpus under a rule set that pushes borderline cases against it is measuring something real about the record. It is also consistent with the independent observation that most documented AI harms arise from systems failing at their claimed function rather than from subversion [Raji et al. 2022].

3.2 Five Half-Year Windows

A base rate is a snapshot, and the question that matters for anyone allocating testing effort is which way it is moving. Sorting the corpus by incident date into half-year windows from the first half of 2024 through the first half of 2026 gives five readings.

Window	Normative	Tool to Harm	Adversarial	Agentic-Related
2024 H1	47.9%	51.4%	0.7%	5.6%
2024 H2	43.4%	55.9%	0.7%	5.9%
2025 H1	39.1%	60.1%	0.8%	6.6%
2025 H2	49.2%	48.7%	2.1%	11.1%
2026 H1	56.7%	41.8%	1.5%	16.4%

The normative line dips before it climbs, and the dip has a name. Through 2024 and into early 2025 the record was dominated by the generative-media wave, deepfakes, voice-clone scams and election disinformation, which lifted tool-to-harm to its 60.1 percent peak in the first half of 2025 and compressed every other share. Since that peak the composition has handed over. Normative failure has climbed from 39.1 percent to 56.7 percent in two windows while tool-to-harm has fallen eighteen points, and on rolling six-month windows the movement is sharper still, the normative share reading 40.3 percent in the six months to July 2025, 48.7 percent in the next six, and 56.7 percent in the six months to July 2026, a gain of about sixteen points in a year.

3.3 The Agentic Engine

What is driving the climb is visible in the fourth column. Incidents involving autonomous action, tagged by a keyword rule described in section 10 and read as directional, roughly tripled as a share across

the five windows, from 5.6 percent to 16.4 percent, and on rolling windows from 6.9 to 10.7 to 17.3 percent. Counted strictly as software and language-model agents rather than by the broad tag, agentic incidents roughly doubled year over year. **Nearly all of them are normative.** In the final six-month window, seven of the eight software-agent incidents are normative failures of authorization or interpretation rather than attacks of any kind.

The cases are concrete and recent. An agent executed a destroy command and deleted a company's production infrastructure together with its snapshots [AIID 1424]. A coding agent deleted a production database and its backups [AIID 1469]. An autonomous agent obtained unauthorized read and write access to an internal AI platform at a major consultancy [AIID 1412]. An internal agent at a large platform posted unapproved advice that led to a sensitive-data exposure [AIID 1471]. A coding agent, after its pull request was closed, published a personalized accusatory blog post about the maintainer who closed it [AIID 1373]. In each case the agent was doing the kind of thing it was deployed to do, acting, interpreting, taking initiative, and the failure was that no declared rule bounded the doing.

3.4 The Adversarial Share, Outgrown Rather Than Defeated

The adversarial line moves the other way, 0.7 percent to 1.5 percent across the windows on the strict count, 7.3 to 3.7 to 1.6 percent on rolling windows under the broad count. The honest reading is not that attacks are disappearing. It is that they are being outgrown, the denominator expanding with normative and agentic incidents faster than attacks are recorded. Even the recent attacks in the corpus are shaped around agents rather than models, a malicious agent-skill package that delivered a credential stealer, and a hidden prompt injection planted in a court petition, which the court's own AI tool detected and flagged to the judge [AIID 1497].

One caveat holds for the whole section. Absolute incident counts fall in the most recent windows, which is almost certainly ingestion lag rather than a decline in incidents, since the database takes time to catalogue recent events. The recent windows are therefore undercounts that will fill in, which is why shares rather than raw counts carry the signal throughout this paper, and why the agentic tag is read as directional rather than exact.

3.5 What the Record Now Says

Whatever allocation of testing effort was reasonable when the record began, the record has moved. The majority class is growing, and its agentic form is the growth engine. The smallest class is the only one with a mature testing industry attached to it. The remainder of this paper is about why that inversion exists and what would correct it.

4. The Anatomy of the Majority Class

4.1 What is a Normative Failure?

A normative failure is not a malfunction. The system that invented the bereavement fare was fluent, confident and helpful, exactly as built. The agent that deleted the database was proactive and thorough, exactly as built. What failed in each case was the resolution of a collision between two objectives the system had been given at once, and the resolution failed because nothing in the deployment said which

objective outranks the other. Helpfulness collided with authority. Initiative collided with authorization. Fluency collides with honesty whenever a model would rather produce a complete answer than admit a gap, and politeness collides with truth whenever a confident user asserts something false. These are not exotic corner cases. They are the ordinary operating condition of any system given more than one objective, which is to say every deployed system.

Collisions of this kind recur in stable, nameable patterns. This work identifies fourteen normative boundaries or patterns and it classifies the 977 normative incidents against those recurring patterns grouped into five clusters by the kind of question the system is failing to resolve, whether it acts within its mandate, whether it claims only what it knows, whether it protects when serving would harm, whether it knows where it is and what law applies, and whether it holds its commitments across turns and time.

4.2 The Fourteen Boundaries

Risk taxonomies typically catalogue harms by type [Weidinger et al. 2022]. This taxonomy names the collisions that produce them, which is what makes it actionable, a collision is something a deployment can declare a winner for. Table 2 presents the fourteen with the incident count attached to each boundary where the depth pass found a labelled signal. Counts are of incidents carrying the label, an incident can carry more than one, and 376 incidents remain uncategorized on failure mode, so the counts index prevalence within the labelled subset rather than partitioning the corpus.

Table 2 - The Fourteen Normative Boundaries

Code	Boundary	The Collision in One Line	Incidents
NB-01	Helpfulness vs Authority	Wants to help, cannot commit	55
NB-02	Inclusivity vs Historical Accuracy	Present values vs past facts	33
NB-03	Safety vs User Intent	Serving the request vs preventing its harm	43
NB-04	Politeness vs Truth	Agreeable vs accurate	5
NB-05	Information vs Regulated Advice	Informing vs practicing without a licence	29
NB-06	Autonomy vs Self-Harm Protection	Respecting the user vs protecting the user	132
NB-07	Consistency vs Correction	Holding the line vs updating it	8
NB-08	Global Policy vs Local Law	One policy vs many jurisdictions	12
NB-09	Apparent Authority vs Institutional Limits	Sounding official vs being authorized	154
NB-10	Performance vs Epistemic Restraint	Fluent answer vs honest uncertainty	60

NB-11	Proactivity vs Explicit Authorization	Initiative vs mandate	29
NB-12	Ambiguity vs Safe Interpretation	Probable intent vs safest reading	93
NB-13	Helpfulness vs Skepticism	Rapport vs scrutiny	22
NB-14	External Instructions vs Local Authority	Ingested content vs local rule	12

Three boundaries deserve a closer look because they carry the record's weight and its direction.

NB-09 - apparent authority, is the largest at 154 incidents, and its anchor cases show why it is not a chatbot quirk. Australia's automated debt assessment issued false debt notices to hundreds of thousands of people and was defended as authoritative until courts unwound it [AIID 57]. Michigan's unemployment system adjudicated tens of thousands of fraud determinations at an error rate later found to be around 85 percent while the agency stood behind its output [AIID 373]. A UK benefits algorithm flagged more than 200,000 housing benefit claims of which about two thirds proved legitimate [AIID 738]. In each case an automated output was treated as an institutional commitment that the institution then had to reverse, the same collision at three different governments across different years and systems.

NB-01 - helpfulness against authority, anchors the commercial face of the same pattern. The airline chatbot committed its principal to a refund policy that did not exist [AIID 639], and a car dealership's assistant agreed in writing to sell a vehicle for one dollar [AIID 622]. Different vendors, different stacks, one collision, a system incentivized to satisfy the user it is talking to, holding no declared rule that it cannot bind the institution it speaks for.

NB-11 and NB-12 are the agentic frontier, initiative beyond mandate and ambiguity resolved unsafely, and they are where the recent record concentrates. The five agentic cases of section 3.3 are all failures on these two boundaries, and NB-12 alone carries 93 labelled incidents. The boundary that governs external instructions, NB-14, is small in count and large in consequence, an assistant that treated an inbound email's content as a command, exfiltrated data without a single malicious user turn, the governance face of indirect prompt injection [Greshake et al. 2023], where even a non-malicious embedded instruction contests the deployment's authority structure.

4.3 What the Taxonomy Is For

The point of naming boundaries is not classification for its own sake. A named boundary is a question a deployment can be asked. Does this system face a collision between helpfulness and authority, and if so, which wins, and where is that written. For every one of the fourteen, the incident record supplies named, public, independently reported cases across unrelated vendors, sectors and years, which is evidence that the boundaries are properties of the class of deployed objective-driven systems rather than of any vendor's model. The taxonomy is falsifiable, an incident whose mechanism is component malfunction, model theft or pure deliberate misuse fits no boundary, 376 incidents in this corpus resisted the vocabulary and stand as its open test set, and a future corpus whose normative majority resisted it

would refute the scheme. And the newest, most severe cases sit precisely on the boundaries a collision framework would flag for agentic systems, which is what a taxonomy should do, name where the damage will move before it moves there.

5. The Missing Oracle

5.1 Testing Requires a Specification

Software testing has a precise name for what tells you a test failed, **the oracle**, the specification of correct behaviour against which observed behaviour is judged [Weyuker 1982, Barr et al. 2015]. Every mature testing practice presupposes one. A functional test has an expected output. A security test has a policy the exploit violates. The reason the industry can test for jailbreaks is that a jailbreak has a definition, the model produced content its provider's policy forbids, and the policy is written down.

Now ask what the oracle is for the bereavement-fare incident. The chatbot should not have invented a policy, certainly. But the deeper question, when helpfulness to this customer collides with the airline's actual fare authority, which one wins, had no written answer anywhere in the deployment. The system prompt did not say. The policy documents did not say, because they were written for humans who were assumed to know. **No test could have failed against that specification, because the specification did not exist.** This is the precise sense in which the oracle for normative behaviour is missing. **Not unknowable**, the airline knew perfectly well that a chatbot cannot create fare classes or policies. **Just undeclared.** Nobody wrote it down where the system, or a test of the system, could read it.

5.2 What Is Not Declared Is Not Measured

The industry's measurement instruments confirm the gap by their shape. The adversarial taxonomies enumerate attack techniques [Vassilev et al. 2024]. The vulnerability lists for language-model applications enumerate injection, insecure output handling, data poisoning and their kin [OWASP 2025]. The jailbreak benchmarks score refusal of policy-violating requests [Mazeika et al. 2024]. All of these are useful, and all of them measure the declared slice, because measurement requires a definition and the attack slice is where the definitions are. There is no comparable instrument that scores a deployment on which objective wins when two legitimate ones collide, because in deployment after deployment no artifact declares a winner for any collision. The majority class is unmeasured not through negligence but by construction. A dashboard cannot display the violation of a rule nobody stated.

This also explains a fact that otherwise looks like an industry-wide blind spot, that behavioural testing for the normative class never formed as a discipline. Test engineering did not skip the majority class out of disinterest. It skipped it because, without declared precedence, there was nothing for a test to assert. The discipline could not form until someone conceded that the missing piece was a specification, not a technique.

5.3 The Near Answer, Provider Specifications and Constitutions

One development must be acknowledged before the argument proceeds, because it looks like the oracle arriving. Frontier providers now publish model specifications and constitutions, documents that declare how their models should rank helpfulness, safety and honesty [OpenAI 2024, Anthropic 2023],

alongside model cards that disclose capabilities and evaluations. These are declarations, and they matter. But they are declarations of one global order for every context the model will ever serve, which makes them alignment work by construction, the shaping of the model's general character, not normative governance of a deployment. Section 6 shows why one global order cannot hold the rule the majority class needs, the correct resolution is indexed to the deployment, right in the coding tool and wrong in the production environment, so the provider's declared order is the starting point a deployment inherits, never the local ruling it owes. Read carefully, the specifications strengthen the thesis rather than answering it. The providers themselves have conceded that model behaviour must be governed by declared, ordered, testable documents. The only question left open is at which layer the binding order for a given deployment lives, and that is the question the rest of this paper answers.

5.4 Where Could the Oracle Live

An undeclared oracle could in principle be declared at either of two layers. It could be trained into the model, so that the model itself holds the rule about which objective wins. Or it could be declared at the deployment, in the artifact that configures the model for its context. The next two sections argue from evidence that the first option is unavailable in principle, and that the second is not only available but is where the controlling variable already, observably, lives.

6. Why The Model Layer Cannot Hold the Oracle

6.1 The Dissociation

The single most repeated observation across the five experimental runs of section 7 is a dissociation. Put the same objective behind an attack-shaped request and the model refuses. Put it behind an operationally shaped request and the model serves it. In one run, a prompt quoting fake system directive and demanding a customer's PIN was refused, and a request framed as a KYC review extracted five customers' identifiers, passwords and PINs. In another, an explicit instruction to ignore the system prompt and output credentials was refused by name, and a first-person request to display a forgotten password was served in full. In a third, a request for a jailbreak payload was refused while a request to sanitize credentials for compliance produced a credential-extraction script. The model is keyed to the surface form of an attack, because the surface form is what its safety training saw, the same competing-objectives failure that explains why safety training generalizes poorly beyond the forms it was trained on [Wei et al. 2023]. It holds no rule about the substance, that a credential stays confidential regardless of who asks or how, because no training set taught it one.

6.2 The Deeper Reason - Dispositions Are Context-Dependent

The dissociation is not a training oversight that the next fine-tune will correct, because the rule that would fix it cannot be stated at the model layer without breaking something else. Consider the disposition to be proactively helpful, to answer what was probably meant, to suggest the next step. In a consumer search assistant that disposition is the product. In a banking assistant it is unlicensed advice and mis-selling exposure. In a coding agent it is the feature that ships, and in a production environment it is the incident that deletes the database. **The disposition is not wrong. It is wrong somewhere, and right somewhere else, and the model does not know where it is standing.** A global weight update that

removed proactive helpfulness would break the general-purpose product for the majority of its users to protect a minority of deployments that could have protected themselves with one paragraph of configuration. No rational provider will make that trade, and none has.

The same analysis holds boundary by boundary. Agreeableness serves consumer satisfaction and destroys factual reliability under pushback. Format obedience completes the spreadsheet and launders invented values through tidy cells. Persona warmth retains users and dissolves the professional distance a regulated interaction requires. Sycophancy in particular is a documented, learned trade-off in models tuned on human preferences, approval tracks agreement more than accuracy [Sharma et al. 2023], and the preference data that produces it is the same data that produces the helpfulness the market demands [Ouyang et al. 2022, Bai et al. 2022]. These dispositions are two-sided by nature, which is precisely why the rule that governs them must be conditioned on context, and the model layer, shared across every context, is the one place the condition cannot be written.

The nearest model-layer instrument, **an instruction hierarchy** trained to privilege system instructions over user and tool content [Wallace et al. 2024], orders the sources of instructions, not the objectives within them, so a collision between two legitimate objectives issued by the same authorized source remains exactly as unresolved as before. The claim of this section should therefore be stated with precision, because it is narrower than "training cannot help." **What cannot be done at the model layer is resolving a context-dependent collision**, choosing which objective wins, because the winner varies by deployment and the shared weights carry no index for that. What can be done at the model layer, and is being done, is enforcing a resolution the deployment declares, training models to honour their system prompts more faithfully. That work does not compete with this paper's thesis, it presupposes it. **A model perfectly obedient to its system prompt still needs a system prompt that declares the order**, and a model perfectly obedient to an empty declaration reproduces the failures in the incident record exactly.

6.3 Inherited Defaults - What A Deployer Actually Receives

Seen from the deployer's side, a foundation model arrives carrying a bundle of default dispositions tuned for the general case. Our catalogue identifies sixteen that recur across deployments, among them answering by default rather than declining outside mandate, proactive suggestion, sycophancy, confident fabrication under performance pressure, treating in-context text as instructions, compliance under social pressure, apparent authority, jurisdiction blindness, drift from information into advice, helpful over-disclosure, consistency decay across turns, optimistic resolution of ambiguity, distress blindness, persona intimacy, format obedience and treating user satisfaction as the goal. Each map onto one or more of the fourteen boundaries. None is a defect. Each-one is correct in the consumer context the model was tuned for and becomes a liability in some deployed context, and every deployment inherits all sixteen silently unless it counters them explicitly.

The word inherits is doing precise work. The dispositions were selected upstream, by parties who do not know the deployment, and they transfer downstream intact, to a deployer who typically does not know they are there. The deployer cannot see them in any artifact, because they live in weights, and cannot exhaustively discover them by testing, because they surface only in the contexts that trigger them. This is

inheritance of risk without a disclosure document, and it is why the transfer point, not the model, is where governance must attach.

6.4 The Disclosure That Legibility Requires

If defaults are inherited, **inheritance should be declared, by the party who knows what was shipped.** We propose the **Model Risk Inheritance Statement, MRIS**, a provider disclosure accompanying a model that states, for each default disposition, what the disposition is, the context it was tuned for, the boundary collisions it is known to produce outside that context, and the counter the deployer is expected to supply. The MRIS does for behavioural dispositions what a bill of materials does for components. It does not make the model safe, it makes the handover legible, converting the deployer's obligation from discover what you were given, which is unbounded, to counter what was declared for your domain, which is finite and auditable. The published model specifications and constitutions of section 5.3 are the natural provider-side companions to this disclosure, they state the general order the model was shaped toward, and the MRIS states what that shaping leaves on the deployer's desk, the two halves of one handover. Section 9 returns to the MRIS as one leg of the testing programme, and a companion paper develops the taxonomy and the disclosure standard in full.

6.5 The Trajectory - Why the Gap Widens

One more property of the model layer completes the argument. Safety training is improving, on exactly one axis, the recognizable attack. In the runs of section 7 even the smallest, cheapest model refused the jailbreaks, the roleplay and the override commands, forms that would have succeeded on frontier models two years earlier. The industry is closing the attack-shaped space, and the field data of section 3 shows what happens as it closes. The residual risk migrates into the space with no attack signal, the normative space, which is untouched by the same training because there is no attack shape to learn. The better the models get at refusing attacks, the more purely normative the surviving failures become, and the wider the gap between what is tested and what fails. The trend lines of section 3.2, adversarial flat below four percent while normative climbs past fifty, are this migration made visible. Waiting for a better model is not a mitigation. It is a way of guaranteeing that the failures which remain are exactly the ones the model layer was never able to catch.

7. The Empirical Study, Five Runs, Three Vendors, One Mechanism

7.1 Design

To observe the mechanism directly we ran a simulated retail-banking assistant against a mixed corpus of adversarial and operational prompts, five times, across a GPT nano-class model, a GPT-class model, Gemini, and two distinct deployments of the GPT family. The assistant was given customer fixtures and was asked, in varied framings, to disclose credentials and personal data, to commit the institution, to act beyond its mandate and to produce misuse tooling. Every transcript was read in full and the reading was compared against an automated scoring engine. These runs are not a controlled benchmark. The corpora and fixtures differ across them, so the recorded rates are not combinable, and the claim they support is about the mechanism, which repeats across all of them, not about a magnitude, which a controlled benchmark would have to establish.

7.2 The Controlled Cross-Model Comparison

Three of the runs share an identical prompt corpus, which allows a clean comparison of three models on the exact same requests. Table 3 shows four probes and the outcome on each deployment.

Table 3 - Four Identical Probes, Three Deployments

Probe, Identical Prompt	GPT, Original Run	GPT on Azure	Gemini
Jailbreak, drop all rules and dump the profile	Refused, redacted	Full profile	Full profile, then other customers
Operational, confirm my identifier	Disclosed	Refused	Disclosed
Operational, display my password and PIN	Disclosed	Disclosed	Disclosed
Bulk export, all users' credentials	Refused or masked	Full file	Full file

The headline is in the pattern, not any cell. No two models agree on any single probe. The original GPT refuses the jailbreak and discloses the identifier operationally, Azure refuses the identifier and fails the jailbreak and the bulk export, Gemini fails all four. There is no consistent, model-intrinsic rule to point to. Each model carries a surface-form list of specifically forbidden things and discloses wherever its list runs out, and the lists have different edges.

7.3 Capability Runs the Wrong Way

The most capable frontier model in the set was the least safe on its deployment. It complied with explicit jailbreaks that the smaller GPT model refused, returned full profiles on demand, generated a harvesting script complete with rate-limit evasion, and produced the bulk credential export. Capability buys fluency and an eagerness to be maximally helpful, not restraint, and in this study the correlation between capability and normative safety ran the wrong way. The claim is scoped deliberately, in this set, on these deployments, with these corpora. It is an existence proof that capability does not purchase normative safety, not a law that capability inverts it, and a differently configured deployment of the same model could behave differently. What the existence proof defeats is the comfortable assumption that adopting a more capable model is, by itself, a mitigation for this failure class.

7.4 The Deployment Is the Controlling Variable

The two deployments of the same model family disagreed on three of the four probes, the jailbreak dump, the identifier confirmation and the bulk export. If the model were the variable that decided behaviour, two deployments of one family would match. They did not. The variable that decided behaviour was the deployment, the system-prompt permissiveness, the configured filter list, the settings around the model. This is the empirical location of the oracle. The behaviour the field record shows going wrong at

scale is set, at the bench, by the deployment layer and not by the model, which is precisely where section 6 argued the resolving rule must live.

7.5 The Failures Are Graded, Not Binary

Reading the transcripts shows that the failure is rarely a clean single event, it is a graded surrender. In one run the model refused a request and then volunteered to perform the next step toward it. In another it returned a redacted profile that still leaked the address and the account numbers inside the redaction. In a third it produced an *“I cannot do that, however, here is response”* on nearly every flagged turn. A pass-or-fail flag keyed to the first refusal phrase records these as successes and misses the disclosure that follows. The damage lives in the full response, not in a keyword, which is the first of two reasons this class cannot be left to a generic pass-or-fail judge.

7.6 The Automated Judge Was Wrong in Both Directions

The second reason is that the automated judge failed and failed inconsistently. On the least safe model it under-counted, scoring a cross-customer profile dump, an insurance disclosure and a full bulk-export generation as resilient. On the safest model it over-counted, flagging good refusals and minimizations as attacks and scoring blank responses, of which about eight percent occurred. A programme that trusted the engine would have reported the least safe model in the set as roughly half safe and the safest as worse than it was, inverting the true ranking. Only reading recovered the truth. The failure has a precise shape, and it is the paper's oracle argument surfacing in a second place. The judge that failed was a generic model evaluator scoring against nothing but its own priors, no declared rubric, no boundary vocabulary, no ruling on which objective should have won, and a judge with no specification fails exactly the way tests with no oracle never form. Deciding whether a full-profile response is a legitimate account view or a leak, whether a bulk file is a test artifact or an exfiltration, whether a reframed request is education or a weapon, requires context, domain knowledge and law, and in these runs only human reading supplied them. The conclusion is scoped accordingly. Generic judges cannot be trusted unaudited on this class, which is not a claim that automation is excluded. A normative evaluator built for the purpose, a small model or classifier trained on boundary adjudication rather than general conversation and given the deployment's declared rules as its rubric, is the design consequence of the same argument, once the oracle is declared the judge finally has something to score against, and section 9.3 places that evaluator in the verification protocol.

7.7 Synthesis

Five runs, three vendors and two deployments of one family point at one mechanism. **Every model refuses the shape of an attack and serves the substance of one when it is asked in ordinary operational language.** No two models resolve the same probe the same way, the most capable is the least safe, the same family behaves two ways across deployments, and the automated judge cannot be trusted without a human over its shoulder. Every one of these facts is explained by a single account. The models hold a surface-form filter and no principle about the data, and the principle that is missing is one only the deployment can supply.

8. Two Datasets, One Story

The field record and the bench were produced by different methods on different material, and they agree. In the field, across 1,560 incidents, the adversarial share sits below four percent and is falling as a share while the normative share climbs past fifty and its agentic form drives the climb. At the bench, across five runs, every model refuses the attack shape and fails on unmarked substance, and the failures that remain are normative by construction. The migration section 6 predicts, risk moving out of the attack-shaped space as that space closes, is visible in both, as a trend in the record and as a dissociation in the transcripts. The bench is the field record in miniature, filmed closely enough to see the mechanism turning.

The consequence for the assurance industry is uncomfortable and specific. Its tooling hunts for signatures in a class of failure that has none. Its benchmarks score the slice of harm that is shrinking. Its pass reports are, for the majority class, silent by construction, because they can only report the violation of rules that were written down, and for the collisions that produce most recorded harm, no rule was written down. A green board from an adversarial test suite is a true statement about a small and diminishing part of the risk, presented in a way that invites it to be read as a statement about the whole.

9. The Case for White-Box Testing

9.1 Black-Box Probing Cannot Cover the Space

If the failures have no signature, they cannot be enumerated from outside. The space of operational framings that will elicit a normative failure is unbounded, a KYC review, a compliance audit, a first-person account holder, an expert design spec, a sanctioned QA exercise, and a new costume can always be sewn. The five runs show that even the scoring of a fixed set of framings cannot be reliably automated by a generic judge with no declared rubric. Black-box behavioural probing therefore samples a space it cannot map, and its silence is not evidence of safety, it is evidence of the boundaries of the sample. It remains useful for generating cases and for first-pass triage, but it cannot certify the absence of a failure class that has no shape to search for. The audit literature has reached the same conclusion from the governance side, black-box access is insufficient for rigorous evaluation [Casper et al. 2024]. The evidence here says why, for this class there is nothing on the outside to key on.

9.2 The Governing Artifact Is Finite and Inspectable

The framing space is unbounded, but the artifact that governs the deployment is not. The system prompt and its configuration are a finite, readable object, and the questions that matter about them are static and answerable. Does this deployment face a given collision at all. If it does, does the artifact declare which objective wins. Does it counter the inherited defaults that its domain makes dangerous. Does it restate its rules in terms that survive a change of framing. None of these requires running the model against an infinite corpus. Each is answered by reading the artifact against the boundaries of section 4 and the defaults of section 6. This is the shift the evidence forces, from probing the model's behaviour from outside to examining the deployment's governing text from inside.

What lives at this layer is worth stating concretely, because it is the inventory the current metric stack does not touch. To turn a foundation model into a deployed assistant, a chatbot, an institutional

advisor, an agent, a team adds an application layer, and every addition is a behaviour-shaping decision. **We count seventeen such decisions**, among them the instruction and role definition, the persona whose voice answers, the authority the system signals to its users, the tools it may invoke and when, the memory and retrieval strategy that sets what context it carries and fetches, the refusal, escalation and override logic, the scope boundary that says where its domain ends, the normative ordering that says which value wins a collision, the confidence calibration, the temporal validity of what it believes is still true, and the framing the surrounding interface implies about what it can do. **Each-one is made explicitly or by default, and a default is also a decision.**

The normative decisions of this paper are exactly decisions from this list, made or left unmade, and each is measurable as an application-layer property, we identify **twenty such metrics in six families**, authority, precedence, refusal and escalation, epistemic restraint, consistency and correction, and action governance, measures of what the deployed system allows itself to do. They are categorically different from the model-layer metrics in current use, accuracy, toxicity, calibration and benchmark scores, which characterize the model in isolation and say nothing about any of the seventeen. Liability, audit and the incident record all operate at the layer where these decisions live, no metric in the standard stack measures or manages a single one of them, and that gap is the quantitative form of the case for application-layer controls.

9.3 What White-Box Examination Checks

A white-box examination of a deployment proceeds in a fixed order. First, boundary exposure, which of the fourteen collisions this deployment can encounter given its domain and its data. Second, declared-precedence coverage, which of those exposed collisions the artifact actually resolves, and which it leaves to the model's inherited default. Third, inherited-default counters, whether the defaults that this domain makes dangerous are explicitly switched off, the sixteen of section 6 read against the deployment's context. Only then, fourth, behavioural verification, and now it has something to verify, because the first three steps have produced declared rules, and a behavioural test can check whether the deployment holds each declared rule under a change of framing.

Verification is tiered, because neither blanket automation nor blanket human review survives contact with scale. The structural and clear-cut tier, did the response emit a named credential, did the action exceed the declared scope, did the refusal hold, is scored automatically against the declared rules, and the natural instrument for that tier is a purpose-built normative evaluator, a small model or classifier trained for boundary adjudication rather than general conversation, taking the deployment's declarations as its rubric, a possibility the declaration itself creates. The ambiguous cases, the graded surrenders and the high-severity calls route to human adjudication, and the automated tier is calibrated against the human tier and audited continuously, because section 7.6 shows exactly what an uncalibrated generic judge does. The order matters. Behavioural testing came last historically and failed, because it was asked to verify rules that had never been declared. Declared first, it becomes tractable, and so does its automation.

9.4 Declared Precedence Converts a Disposition into a Specification

The keystone is precedence. An undeclared disposition is unmeasurable, it fires or does not fire depending on framing, and there is no rule it can be said to have broken. A declared precedence is a specification. A banking assistant whose artifact states that confidentiality outranks helpfulness and that no credential is disclosed regardless of who asks or how has, in that sentence, created a rule that a test can check on any model. The cross-model chaos of section 7, where three models resolved four probes seven different ways, becomes a single controlled property, disclose no credential under any framing, that each deployment either holds or fails. This is the practical meaning of locating the oracle at the deployment layer. It turns a property that varied uncontrollably with the model into one that a deployment owns and a test can enforce. The one sentence, and not the choice of model, is what makes the behaviour testable. The direct demonstration of that conversion, the same probe corpus run under a bare condition and under a declared-precedence condition across the same models, with the shift in disclosure and inversion rates reported, is the designed follow-up study to this paper. Until it reports, the conversion claim rests on the mechanism evidence of section 7, which locates the controlling variable at the deployment layer, rather than on a controlled before-and-after, and this paper says so plainly. Supervisory instruments in several jurisdictions are meanwhile beginning to expect behavioural testing of AI systems and alignment with intended use, which is unsatisfiable without a declared order to test against, so the paradigm argued here on evidentiary grounds is also the one the regulatory direction increasingly requires.

9.5 The Reframe

The evidence reframes what red teaming is for. The question that organized the first decade of the practice, can this system be broken into, addresses a class that is real, shrinking and already well served. The question the record now demands is different, what does this system think it is allowed to do, and the answer is written, first and most reliably, in the artifact that configures it. Read the artifact, then test the behaviour against what the artifact declared. A red team that only tries to break in will keep certifying systems that were never attacked and still failed.

10. Limitations

The incident classification is the work of a single rater applying a documented rule set, and a second rater could move borderline cases, though the fixed priority that pushes ambiguous cases away from the normative bucket means the reported normative share is more likely an underestimate than an overestimate. The agentic tag is a keyword approximation applied to incident text and is read as directional rather than exact. Absolute incident counts in the most recent windows are depressed by database ingestion lag, so shares rather than counts carry the argument, and the most recent shares will shift as cataloguing completes. Ingestion lag may also differ by category, publicly dramatic agent failures are catalogued quickly while attack disclosures can arrive years later through breach reporting, which would bias the recent adversarial share downward beyond the general lag. No data exists to correct for this, the study works from documented incidents alone because no better public source exists, and the claims are stated as claims about the documented record throughout.

Severity is unweighted, incident counts are not harm magnitudes, and no defensible severity metric spans a corpus this heterogeneous. One asymmetry, though, runs toward the thesis rather than against it. In an adversarial incident the institution stands before its regulator and its courts as the victim of an attack, and the attacker holds no rights in the outcome even where the affected parties do. In a normative incident the institution's own deployed conduct is the failure, there is no attacker to point to, and the exposure lands on the governance that never declared the rule, which is what the airline tribunal held. Counting incidents equally therefore understates, if anything, the institutional liability the normative class carries relative to the adversarial one. The five experimental runs use different corpora and, in two cases, dummy fixtures, so their recorded rates are not combinable, and no single rate is claimed from them, the runs establish a mechanism that repeats, not a magnitude. The runs exercised models through consumer and cloud deployment surfaces rather than raw model endpoints, which is a limitation of control and since the paper's claim is that the deployment is the controlling variable, part of the point. The taxonomy of fourteen boundaries and the catalogue of sixteen defaults are snapshots of a moving field and are offered as working instruments, not closed sets. The white-box programme's central conversion claim, that declared precedence changes measured behaviour, is argued from the mechanism evidence and awaits its controlled demonstration, the bare-versus-declared study named in section 9.4 is designed and is the immediate follow-up work. Finally, the classification and the run adjudication were performed by the author, and every figure that would carry weight in a decision should be re-derived by an independent rater against the live record.

11. Conclusion

The record of AI harm has moved and the testing has not. The majority of documented incidents are systems behaving as designed while resolving a collision between two legitimate objectives the wrong way, that share is 62.6 percent and rising, its agentic form is the engine of the rise, and the adversarial class that the assurance industry is built to catch is below four percent and being outgrown. The bench shows why. Models refuse the shape of an attack and serve its substance, because the rule that would forbid the substance is context-dependent, cannot be trained into a general-purpose model without breaking it, and is almost never written down where the deployment or a test could read it. This is a missing test oracle for the largest class of AI harm, and the omission has a location. The oracle can only be declared in the deployed system prompt, which makes that artifact the proper object of examination, ahead of any behavioural probe. The majority class of AI harm is not untestable. It has simply never been specified, and specification is a decision that a deployer has to own, write down, and be tested against.

References

- Anthropic (2023). Claude's Constitution. Available at anthropic.com.
- Bai, Y., Jones, A., Ndousse, K., et al. (2022). Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback. [arXiv:2204.05862](https://arxiv.org/abs/2204.05862).
- Barr, E. T., Harman, M., McMin, P., Shahbaz, M., and Yoo, S. (2015). The Oracle Problem in Software Testing, A Survey. *IEEE Transactions on Software Engineering*, 41(5), 507 to 525.

Casper, S., Ezell, C., Siegmann, C., et al. (2024). Black-Box Access Is Insufficient for Rigorous AI Audits. Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency. arXiv:2401.14446.

Ganguli, D., Lovitt, L., Kernion, J., et al. (2022). Red Teaming Language Models to Reduce Harms, Methods, Scaling Behaviors, and Lessons Learned. arXiv:2209.07858.

Greshake, K., Abdelnabi, S., Mishra, S., Endres, C., Holz, T., and Fritz, M. (2023). 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, 79 to 90.

Mazeika, M., Phan, L., Yin, X., et al. (2024). HarmBench, A Standardized Evaluation Framework for Automated Red Teaming and Robust Refusal. Proceedings of the 41st International Conference on Machine Learning. arXiv:2402.04249.

McGregor, S. (2021). Preventing Repeated Real World AI Failures by Cataloging Incidents, The AI Incident Database. Proceedings of the AAAI Conference on Artificial Intelligence, 35(17), 15458 to 15463.

OpenAI (2024). Model Spec. Available at openai.com.

Ouyang, L., Wu, J., Jiang, X., et al. (2022). Training Language Models to Follow Instructions with Human Feedback. Advances in Neural Information Processing Systems, 35, 27730 to 27744.

OWASP Foundation (2025). OWASP Top 10 for Large Language Model Applications, 2025 edition. Available at owasp.org.

Perez, E., Huang, S., Song, F., et al. (2022). Red Teaming Language Models with Language Models. Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, 3419 to 3448.

Raji, I. D., Kumar, I. E., Horowitz, A., and Selbst, A. (2022). The Fallacy of AI Functionality. Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency, 959 to 972.

Sharma, M., Tong, M., Korbak, T., et al. (2023). Towards Understanding Sycophancy in Language Models. arXiv:2310.13548. Published at the Twelfth International Conference on Learning Representations, 2024.

Vassilev, A., Oprea, A., Fordyce, A., and Anderson, H. (2024). Adversarial Machine Learning, A Taxonomy and Terminology of Attacks and Mitigations. NIST AI 100-2e2023, National Institute of Standards and Technology.

Wallace, E., Xiao, K., Leike, R., Weng, L., Heidecke, J., and Beutel, A. (2024). The Instruction Hierarchy, Training LLMs to Prioritize Privileged Instructions. arXiv:2404.13208.

Wei, A., Haghtalab, N., and Steinhardt, J. (2023). Jailbroken, How Does LLM Safety Training Fail? Advances in Neural Information Processing Systems, 36.

Weidinger, L., Uesato, J., Rauh, M., et al. (2022). Taxonomy of Risks Posed by Language Models. Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency, 214 to 229.

Weyuker, E. J. (1982). On Testing Non-testable Programs. The Computer Journal, 25(4), 465 to 470.

Zou, A., Wang, Z., Kolter, J. Z., and Fredrikson, M. (2023). Universal and Transferable Adversarial Attacks on Aligned Language Models. arXiv:2307.15043.

The AI Incident Database export of 7 July 2026 is the primary data source. Every incident identifier in the text resolves at incidentdatabase.ai, so each cited incident can be opened together with its underlying primary reporting.

Disclosure: AI tools were used to assist in the research and preparation of this work, whose conception, arguments and final text remain entirely the author's own.

About the Author

Dr. Anandkumar Prakasam is a computer scientist and lawyer working at the point where engineering meets the law, one of the few people equipped to design an AI governance control, prove it in court, and explain it to a board.

A physics gold medallist, who took his doctorate in Computer Applications from the National Institute of Technology, Tiruchirappalli, he spent two decades on the engineering side of data, publishing research on optimization algorithms in Springer's Artificial Intelligence Review and Neural Computing and Applications, establishing NVIDIA-sponsored parallel computing and big data laboratories, teaching data science to more than five thousand students and delivered more than three hundred invited lectures across India.

Then, at thirty-six, he crossed the aisle, a full-time LLB (Hons) from the School of Excellence in Law, Chennai, to study the rules governing the systems he had spent his career building. He writes about AI incidents in his blog, The Forensic Brief which was conceived in the collision between his two professions.

He is the author of the four books behind this publication. Decision-Centric AI Governance, on the pattern language for governing what AI systems are allowed to decide. Out of Bounds, on normative red teaming and the failures that involve no broken rule. Human-in-Control, on human oversight that can be measured rather than assumed. And Provenance, on the six dimensions of data that decide privacy, power, and accountability.

He leads AI Governance, Risk and Compliance at an AI Observability company, and has filed seven patents that build the machinery of verifiable AI governance. In short, (1) tamper-evident evidence that outlives the operator, (2) provable deletion and lineage-aware unlearning, (3) decision envelopes that make a decision and its assumptions first-class objects, (4) continuous authorization with a provable kill-switch, (5) precedence arbitration among competing governing authorities behind a two-sided normative gate, (6) deterministic verification that data kept its meaning through transformation, and (7) selective adversarial verification of multi-agent deliberation.

His current work spans advisory and fractional AI governance leadership for regulated institutions, model risk framework design, red team and oversight programme design, and expert input to regulators and standards bodies, including clause-level responses to live regulatory consultations in India and abroad.