

# Layered LLM Defenses as an Ensemble: Access Tiers, Inference Cost, and the Measured Failure Correlation Between Defense Layers

Abrar Alotaibi<sup>a,b,\*</sup>, Muhammad Shahid Jabbar<sup>c</sup>, Sadam Al-Azani<sup>c</sup> and Moataz Ahmed<sup>a,c</sup>

<sup>a</sup>Information and Computer Science Department, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia

<sup>b</sup>Department of Computer Science, College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, Dammam, Saudi Arabia

<sup>c</sup>SDAIA-KFUPM Joint Research Center for Artificial Intelligence, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia

## ARTICLE INFO

### Keywords:

Large language models  
LLM security  
Classifier fusion  
Ensemble diversity  
Defense composition  
Failure correlation  
Adversarial robustness  
Blue teaming  
Threat modeling

## ABSTRACT

Practitioners defend large language models (LLMs) by stacking defenses, assuming the layers compound. A stack is an ensemble, and ensembles compound only under a condition the LLM security literature recommends but never measures: the members must fail on different inputs.

Two instruments make that measurable. The Adversary Access-Tier Model (AATM) grades an adversary by the access it holds, from system-only (A0) to influence over training data (A4). A cost model sorts defenses into five classes of inference-time overhead; because two classes require training weights or reading activations, they tier the defender as AATM tiers the adversary. From these we derive how a stack behaves, and the quantities a defender cares about diverge: coverage saturates within a tier, cost rises by class, false refusals accumulate as a union, and residual attack success falls multiplicatively only under independence.

We measure that independence. Running one adaptive adversary against a seven-layer stack, failure correlation is positive in all fifteen measurable pairs ( $\phi$  from 0.30 to 0.75), and the joint residual exceeds the multiplicative prediction by up to 0.172. Stratifying on behavior difficulty dissolves most of the association, so the dependence is predominantly common-cause, but it survives permutation inference, majority-vote grader labels, and externally calibrated thresholds. The same stack refuses four in five benign prompts while remaining statistically indistinguishable from its strongest single layer.

The dependence is architectural rather than sampling-based: members correlate through the model they all wrap, so no wider member pool weakens it. Diversity therefore selects stack members but does not predict what an assembled stack delivers, which has to be measured end to end.

## 1. Introduction

Large language models (LLMs) are being deployed into security-sensitive applications faster than the field can establish how well they are actually defended. A growing body of work proposes defenses, input and output filtering [1, 2], robust fine-tuning [3, 4], adversarial training [5, 6], runtime monitoring [7], and infrastructure hardening [8], and reports impressive reductions in attack success [9, 10]. Yet these numbers are hard to compare or trust: each is measured under a different, and frequently unstated, assumption about what the attacker knows and can do, and most are obtained against fixed attacks rather than against an adversary who adapts to the defense in place. That distinction is not academic: defenses that appeared robust against a fixed attack set have been broken repeatedly once the attacker optimizes against them [11–13]. As a result, the steady accumulation of reported gains can be difficult to interpret, and practitioners are left with limited basis for deciding which defense to deploy against which threat.

One candidate mechanism would explain much of this at once. Safety alignment as currently practiced is *shallow*: it adapts the model's output distribution over no more than the first few generated tokens [14]. That single property is why suffix [15], prefilling, decoding-parameter, and fine-tuning [16, 17] attacks all succeed by the same move, and it compounds the evaluation problem above, since rubric-based rescoring shows that many published attack-success and

\*Corresponding author.

 amotaibi@iau.edu.sa (A. Alotaibi)

ORCID(s): 0000-0003-1168-8050 (A. Alotaibi); 0000-0003-2331-876X (M.S. Jabbar); 0000-0001-7893-1196 (S. Al-Azani); 0000-0003-0042-8819 (M. Ahmed)

defense figures do not survive a stricter reading [18]. It also suggests an answer to a question the field has not asked: if defenses inherit a common point of failure from the model they all wrap, then stacking them should deliver less than the sum of their reported gains, because layers that depend on the same property fail on the same inputs. We take that as the hypothesis to test rather than as the explanation to assume. The consequence is measurable, and we measure it; the cause is harder to establish, and we report what our data can and cannot settle about it.

This paper builds the instruments that make the question answerable and then answers it. Because no deployment runs a single defense, the object of study is the stack rather than the layer, which places LLM defense composition inside a problem the information-fusion literature has studied for three decades, namely how to combine decision-makers whose errors are correlated. From that literature we inherit both the vocabulary and its central lesson: diversity between components must be measured, not assumed [19, 20]. Its adversarial branch sharpens that lesson, because dependence between members is there not merely a statistical property of the ensemble but something an attacker can act on deliberately [21, 22]. Neither lesson has reached the LLM defense literature. Section 2 states the whole argument and its result in advance; Section 4 situates it against the three literatures it draws on; and the five-layer defense review that follows is the application of these instruments rather than a free-standing catalog, its purpose being to clarify what existing results actually establish, what they cost to obtain in deployment, and which of them compound when combined.

Two distinctions organize what follows. The first separates red teaming from blue teaming. Red teaming probes a deployed model to uncover vulnerabilities, through adversarial inputs, jailbreak attempts, or misuse scenarios that test its safety boundary [23, 24], and the resulting frameworks are themselves evaluation instruments rather than one-off attack sets [25]. Blue teaming, the subject of this paper, builds the defenses: detection systems, mitigation strategies, and runtime monitoring that protect a model in deployment [26]. The second separates threats from vulnerabilities. A threat is an action an adversary takes, such as prompt injection, model extraction, or data poisoning; a vulnerability is the weakness that makes the action succeed, such as the inability to separate instructions from data [27, 28], or safety behavior concentrated in a sparse parameter subset [17, 29]. The Adversary Access-Tier Model (AATM) attaches to the first and the threat taxonomy of Section 6 to the second, which is what lets us state, for each defense, both what it stops and the access an adversary needs to get past it.

*Scope and objectives.* This study is the blue-team counterpart to the red-teaming literature cataloged by Jabbar et al. [30] and to the framework of Alotaibi et al. [25]: where those characterize how a model is probed, this one asks what the probing is defended against, and how those defenses behave together. It gives a tier-conditioned analysis of existing mitigation strategies, derives rules for combining them, and measures the one quantity those rules require. Prior work treats defenses either as isolated mechanisms or as the passive target of an attack survey [31, 32]; we treat a deployed set of them as a single system whose behavior does not follow from its parts.

### 1.1. Contributions and research significance

The key contributions of this study are as follows:

- **Two instruments, and a duality between them.** AATM defines adversaries by access tier, knowledge, capability, and goal (Section 5), and a five-class model of inference-time overhead prices what a defense costs to run (Section 9). The two are not independent: because two of the five classes require the ability to train weights or to read activations, the overhead classes tier the defender by access exactly as AATM tiers the adversary, and the intersection of the two determines which defenses a given deployment can implement at all.
- **Composition rules for layered defense.** We derive how coverage, cost, residual attack success, and false refusals behave under stacking (Section 10), and show that they behave differently: coverage saturates within a tier, cost composes by overhead class, effectiveness compounds only under failure independence, and refusals accumulate against the defender. We classify the reviewed defenses by the model property each depends on, which converts defense-in-depth from a slogan into a constraint on which combinations are worth paying for, and we read three production deployments through the result (Section 12.1).
- **A tier-conditioned reading of the evidence.** We apply these instruments across five defense layers, from input/output censorship to infrastructure hardening. Reported effectiveness figures are recast as conditional on each study's threat model rather than as comparable robustness guarantees, adaptive attacks are treated as a first-class evaluation concern (Section 8.1), and the field's metrics and benchmarks are assessed against what

a composition claim would need (Section 8). That reading isolates the one quantity the composition argument requires and no study reports: the failure correlation between two defenses under a shared adaptive adversary.

- **The first measurement of that quantity.** Failure correlation is a pairwise diversity measure classifier fusion has used for decades [19], and Section 11 is, to our knowledge, its first application to a stack of LLM defenses. We measure it for all fifteen measurable pairs of a seven-layer stack under one shared adaptive adversary, stratify it against the obvious confound, and check it against three labeling regimes and a direct attack on the assembled stack. Every pair is positive in every regime. Section 2 summarizes what follows from that.
- **A dependence structure the fusion literature has not had to model.** Classifier ensembles correlate through shared training data, and that *sampling* dependence weakens as the member pool widens. Defense stacks correlate through the target model every member wraps, and no amount of member diversity weakens that *architectural* dependence, because the members are not the thing they share (Section 10.5). Diversity creation therefore has to target the substrate rather than the pool, a case the fusion literature has not had to address, and the operational consequence is the one this paper establishes: diversity selects stack members but does not predict what the assembled stack delivers.

## 2. Overview: A Defense Stack as an Ensemble

This paper does two things: it assembles the tier-conditioned map of the defense literature that the composition analysis needs as input, and it then derives and measures how the mapped defenses behave in combination. This section states the second, which is the contribution, before the first, which is the apparatus, so that the whole argument and its result are available without first reading the review.

*The framing.* A deployed defense stack is a multiple-classifier system. Its members read the same input, each returns a block-or-admit decision, and the system combines those decisions under a unanimity veto: one blocker suffices. That is a fusion architecture, and the fusion literature settled the governing question three decades ago. Combination pays only to the extent that members fail on different inputs, the condition is measurable through pairwise diversity statistics, and it must be verified rather than inferred from the fact that two members look different [19, 20, 33]. The LLM security literature recommends stacking on every page and has never measured that condition for any pair of defenses.

*Why this ensemble is not the textbook one.* Classifier ensembles correlate because their members are trained on shared data with shared inductive biases, which is a sampling dependence: it weakens as the member pool widens and the training sets diverge. Defense stacks inherit a dependence of a different kind. Every member wraps the same target model, and safety alignment in that model is *shallow*, adapting the output distribution over no more than the first few generated tokens [14]. Mechanisms as dissimilar as a surface perplexity statistic and a mid-network linear probe can therefore share a single point of failure that neither of them contains. That is an *architectural* dependence, not a sampling one, and it does not dissolve with more data or a wider member pool. Section 10.5 develops the distinction and what it implies for diversity creation, which is the sense in which this paper is a fusion result and not only a security one.

*What we derive.* Two instruments make the derivation possible. AATM (Section 5) fixes what each reported defense result is a result *about*, by grading adversaries from system-only access (A0) to influence over training data (A4). A five-class model of inference-time overhead (Section 9) prices what each defense costs to run, and turns out to tier the defender by access much as AATM tiers the adversary, since two of the five classes require the ability to train weights or read activations. From these, Section 10 derives how a stack behaves, and the four quantities a defender cares about diverge: coverage saturates within a tier, cost composes predictably by overhead class, false refusals accumulate as a union against the defender, and residual attack success falls multiplicatively only when the layers fail independently. The last is Eq. (6), whose free parameter is the failure correlation  $\phi$  between two layers, and which no reviewed study supplies a value for.

*What we measure, and what it shows.* Section 11 measures  $\phi$  for every measurable pair of a seven-layer stack, under the design given in Section 3.2. Four findings carry the paper.

1. **Independence fails everywhere we could look.** All fifteen pairs have  $\phi > 0$  (0.30 to 0.75) and a joint residual above the multiplicative prediction, by up to 0.172. Three further diversity statistics from classifier fusion order

the pairs the same way, so the verdict is not an artifact of the statistic chosen. Multiplicative composition is a lower bound on residual risk that no stack in this experiment attains.

2. **Most of the correlation is common cause, not shared mechanism.** Holding behavior difficulty fixed dissolves almost every cross-row association. Which pair survives stratification is not stable: under the reported labels it is a same-row probe pair, under majority-of-five grader labels three pairs survive, and under externally calibrated thresholds the survivor is a cross-row pair while the probe pair's test becomes undefined. We therefore retire the mechanism-specific reading rather than report the regime that favors it (Sections 11.6 and 11.7). The composition verdict does not depend on it: common-cause dependence violates independence exactly as mechanism overlap does. No value falls below 0.300 across all fifteen pairs in all three regimes, roughly three times as many measurements as finding 1 covers.
3. **Depth is bounded by refusals, and dominated by one good layer.** The assembled stack refuses four in five benign prompts while remaining statistically indistinguishable from its single strongest member, a semantic classifier that reaches the same residual at a third of the refusal cost and that dominates every other layer pairwise, with no behavior on which it is breached while another defense holds. Remove that member and the picture inverts: a direct attack on the remaining six layers finds no feasible prompt at all, while intersecting the per-layer vectors predicts a residual of 0.190. A stack's behavior is not recoverable from its parts on either axis.
4. **Measured strength is a property of the attack class.** A perplexity filter blocks every optimized suffix and admits nearly every fluent prompt, so its residual moves from 0.00 to 0.66 with the adversary alone. A composition estimate assembled from per-defense figures obtained under different attacks is not merely imprecise but can be optimistic without bound.

*What  $\phi$  is for.* Findings 1 and 3 together fix the status of the statistic, and we state it here because it is easy to overread.  $\phi$  is a *selection* statistic, not a *prediction* statistic. It tells a designer which candidate members are near-substitutes and which are worth paying for, and Section 11.10 shows that under the reported labels, selecting on it recovers the same stack a greedy search over the full joint breach data would choose. It does not predict the residual or the refusal rate of the assembled pipeline, and we demonstrate that it does not, in both directions. The practical claim of this paper is therefore narrower and firmer than the multiplicative model it displaces: measure diversity to choose members, and measure the assembled stack to know what you have.

### 3. Methodology

This work combines two kinds of evidence, and we state the method for each separately. The framework of Sections 5 to 10 is derived analytically and grounded in a structured synthesis of the defense literature; the composition premise on which it rests is then tested experimentally in Section 11. Section 3.2 states the experimental design, the hypotheses it tests, and the analysis procedure, and Section 3.3 describes how the literature base was assembled.

#### 3.1. Research Questions

The study is organized around four research questions:

- **RQ1:** What inherent vulnerabilities are associated with LLMs across model, data, user, and infrastructure dimensions, and how do adversarial attacks exploit them?
- **RQ2:** What are the current state-of-the-art mitigation strategies and defense mechanisms, particularly those aligned with blue-teaming approaches, and how do they map to the threat landscape?
- **RQ3:** How effective are existing evaluation frameworks and metrics in assessing the robustness of LLM defenses?
- **RQ4:** How do defenses compose, and what governs whether a layered stack delivers more protection than its strongest single component?

RQ1 to RQ3 are answered from the literature base of Section 3.3. RQ4 is answered analytically in Section 10 and then empirically in Section 11, and it is the question that motivates the experiment: the analytical answer is conditional on an independence assumption that no reviewed study measures.

### 3.2. Experimental Method

The composition analysis of Section 10 shows that a stack’s residual attack success is multiplicative in its layers only when their failures are independent, and that the governing quantity is the failure correlation  $\phi$  between two defenses on a common behavior set under a common adversary. No reviewed study reports it. The experiment measures it directly.

*Hypotheses.* The experiment tests three hypotheses, each with a primary defense pair chosen to test it:

- **H1 (same-row dependence).** Two defenses drawn from the same dependency row of Table 3 fail on overlapping inputs:  $\phi > 0$ . Primary pair: perplexity filter  $\times$  token-anomaly filter (token surface).
- **H2 (cross-row independence).** Two defenses drawn from different dependency rows fail independently:  $\phi \approx 0$ . Primary pair: perplexity filter  $\times$  mid-layer probe (surface  $\times$  internal).
- **H3 (refusal budget).** The benign false-refusal rate of an assembled stack, not its inference cost, is the binding constraint on stack depth.

The two primary pairs carry H1 and H2; the remaining thirteen pairs are measured and reported as well, and are exploratory. One further pair was added during the experiment, after the primary same-row pair proved unmeasurable under the first adversary, and is identified as such in Section 11.11.

*Design.* Seven defenses, one per dependency row where a row admits an implementable instance, are evaluated independently against a shared adaptive adversary on a common behavior set, and their per-behavior breach vectors are intersected pairwise. Two attack classes are run against the same targets, a gradient-optimized suffix attack and a fluent sentence-level search, because a defense’s measured strength proved to depend on which is used. Two static baselines are run alongside. The assembled stack is additionally attacked directly, so that the pairwise estimates can be checked against the quantity a deployer actually faces.

*Targets, judge, and analysis.* The primary target is Vicuna-7B-v1.5, with a replication on the more strongly aligned Llama-2-7b-chat; a smaller Llama-3.2-3B-Instruct run is reported only where it bears on a specific claim. Behaviors are the 100 harmful prompts of JailbreakBench with its matched benign set. Breach is scored *post hoc* by the StrongREJECT rubric autograder, which never participates in the attack search, a separation enforced by a static test rather than by convention. Confidence intervals are bootstrap percentile intervals over behaviors, and multiplicity across the fifteen pairs is controlled by the Benjamini–Hochberg procedure, applied under the positive dependence the shared breach vectors induce. The pre-specified alternative explanation, that correlation reflects behavior difficulty rather than shared mechanism, is tested by a Cochran–Mantel–Haenszel stratified analysis in which the stratifier is derived from the *other* defenses, so that it is independent of the pair under test.

*Robustness protocol.* Three properties of that analysis could produce the result on their own, so each is tested rather than argued. For the bootstrap’s distributional assumptions, we run a marginal-preserving permutation test for every pair, Cochran’s  $Q$  across the seven defenses, and exact-binomial McNemar over the full pairwise matrix under Holm correction. For the single grader judgment behind every breach label, we judge each graded response five times under an unchanged rubric, rebuild every breach vector from the majority label, and report a grader-only interval. For thresholds calibrated in-sample, we re-run the primary configuration with thresholds set on 805 external instructions, disjoint from both the evaluation set and the probe training pool. Section 11.7 reports all three, and Section 11.1 gives operating points and full statistical detail. We did not repeat the experiment across attack seeds.

### 3.3. Evidence Base

The tier assignments, overhead classes, and dependency rows of Section 7 are read from published descriptions of defenses, so the literature base is an input to the framework rather than background. We searched Google Scholar, IEEE Xplore, the ACM Digital Library, Scopus, and arXiv to a cutoff of February 2026, combining LLM terms with security and defense terms, and chased citations backward and forward from influential papers. Records were screened on title and abstract, then read in full. Two properties were recorded for every included work, because the framework depends on them: whether it states an adversary model, and whether it evaluates against adaptive attacks. This is a structured critical synthesis of representative and high-impact work, not a registered systematic review, a scope we record as a limitation in Section 12.6. Search strings, screening counts, and the per-study extraction table are supplementary.

## 4. Related Work

Three bodies of work bear on this paper and each supplies a different piece of it. Surveys of LLM threats and defenses give the landscape the framework organizes; a smaller literature on defense ensembles gives the design pattern whose premise we test; and classifier fusion gives the vocabulary and the statistic for testing it. We take them in that order, from the closest to the subject matter to the most distant in field but the nearest in method.

### 4.1. Surveys of LLM threats and defenses

Several reviews have mapped this literature and we build on them. Das et al. [9] and Abdali et al. [10] catalog security and privacy risks, and Cui et al. [34] adds a risk taxonomy and assessment framework, but all three stop short of the full range of mitigations and treat dynamic monitoring and runtime response only briefly. Huang et al. [35] and Gupta et al. [36] address trustworthiness and deployment risk without systematically integrating defense mechanisms. Red-teaming surveys by Perez et al. [37], Lin et al. [38], Dong et al. [39], and Chowdhury et al. [40] map attack techniques thoroughly but remain oriented toward exposing vulnerabilities rather than preventing their exploitation.

More integrated treatments have begun to appear: Yao et al. [29] covers security and privacy broadly, Zhou et al. [41] unifies attack simulation with defensive training, and Liu and Hu [42] emphasizes evaluation benchmarks. Two surveys overlap our scope most closely: Shi et al. [43] cover LLM safety across value misalignment, robustness, misuse, and agent risks, and Li and Fung [44] review security concerns over the 2022 to 2025 window. Neither treats composition. This paper is the blue-team complement to the red-teaming program of Jabbar et al. [30], and it is narrower than the safety surveys by design: it is a security analysis organized around an explicit adversary model and an explicit confrontation with adaptive attacks, treating bias, fairness, and misinformation as adjacent concerns (Appendix A).

What none of these establish is how defenses behave in combination. Composition is recommended throughout the literature and analyzed nowhere in it: no reviewed work states rules for combining defenses, and none reports the quantity such rules require.

### 4.2. Ensembles of defenses

A smaller body of work does combine defenses, and it is the closest prior art to this paper. Lu et al. [45] propose a mixture-of-defenders framework in which two defense experts specialize in adversarial-suffix and malicious-semantics prompts respectively, motivated by the observation that individual defenses generalize poorly outside the attack class they were designed for. Guardrail ensembles that combine several classifiers follow the same intuition, among them the boosting ensemble of Huang et al. [46], the ensemble-of-experts moderation system of Ghosh et al. [47], and the logical-reasoning combination of Kang and Li [48]. These systems assume what this paper measures. Specialization is proposed as a design principle, and its premise, that the combined components fail on different inputs, is argued from the mechanisms' apparent dissimilarity rather than estimated from data. The same assumption is stated explicitly in the operational literature, where Verma et al. [49] recommend stacking guardrails on the grounds that the mistakes of different defenses are likely to be uncorrelated. Section 11 tests that premise directly and finds it does not hold for any pair we could measure. That defenses evaluated in isolation are defeated by attacks designed against the deployed configuration is also the finding of Nasr et al. [13], whose adaptive attacks defeat a range of published jailbreak and prompt-injection defenses, and which motivates the adaptive-evaluation requirement of Section 8.1.

### 4.3. Diversity in classifier fusion

The question of whether combining components pays, and under what condition, is older than the LLM literature and belongs to information fusion. Work on classifier ensembles has long held that combination pays only to the extent that members fail on different inputs, has formalized that condition through pairwise *diversity* measures including the correlation of error indicators [19, 33], and has developed methods whose explicit purpose is to create diversity rather than hope for it [20, 50], on the assumption that dependence between members comes from the data they were estimated on; Section 10.5 argues that this assumption fails for defense stacks. The adversarial specialization of that work is equally established: a multiple-classifier system is only as robust as the properties its members do not share, and an attacker who targets a shared property defeats the system at the cost of defeating one member [21, 22]. None of this is reflected in the LLM defense literature, which argues for layering without measuring diversity between layers. Section 10 states the condition for defense stacks and Section 11 measures it.

Measuring diversity between defenses presupposes something the LLM security literature also leaves implicit, namely what each defense is defending against. We therefore begin with the adversary.

## 5. The Adversary Access-Tier Model (AATM)

A defense claim is only meaningful relative to the adversary it assumes. Without it, a reported attack success rate (ASR) is a number without a referent: it cannot be compared across studies, reproduced, or trusted as a guarantee about a deployed system. Much of the LLM defense literature reports robustness without stating the adversary’s access or whether the adversary adapts to the defense, which systematically overstates security. To avoid this, we define AATM here and use it as the organizing instrument for the rest of the paper, relating each defense layer to the access tier it presupposes in the master mapping that opens Section 7 (Table 2). The tier labels, the overhead classes introduced later, and the notation used throughout are collected in Appendix B.

### 5.1. Access Tiers

We distinguish five tiers of adversary access, listed in Table 1 and ordered by what the adversary can reach rather than by how hard the attack is to mount: an A0 adversary who poisons a retrieval source never queries the model, while an A3 adversary holds its weights.

**Table 1**

Adversary access tiers. A1 to A3 are cumulative; A0 and A4 are not, and are marked accordingly.

| Tier                                 | Name            | What the adversary holds                                                              | Representative attacks                                                     |
|--------------------------------------|-----------------|---------------------------------------------------------------------------------------|----------------------------------------------------------------------------|
| <i>Outside the model</i>             |                 |                                                                                       |                                                                            |
| A0                                   | No model access | The surrounding system only: dependencies, retrieved content, the deployment pipeline | Indirect prompt injection [28], retrieval poisoning, RCE in LLM frameworks |
| <i>Through the model, cumulative</i> |                 |                                                                                       |                                                                            |
| A1                                   | Black-box query | Inputs and outputs through a public interface                                         | Direct jailbreaks, direct prompt injection, transfer attacks               |
| A2                                   | Gray-box        | + logits or token probabilities                                                       | Logprob-guided adaptive jailbreaks, some extraction                        |
| A3                                   | White-box       | + weights, and the ability to compute gradients                                       | Gradient suffix attacks [15], safety-region pruning, fine-tuning attacks   |
| <i>Before the model</i>              |                 |                                                                                       |                                                                            |
| A4                                   | Training data   | Influence over the training or fine-tuning corpus                                     | Data poisoning, backdoor and Trojan insertion                              |

AATM refines the classical white-, gray-, and black-box distinction of adversarial machine learning in two ways that matter for deployed LLM systems. First, it makes “no model access” a genuine tier (A0): indirect prompt injection through retrieved or tool-returned content attacks an LLM application without ever querying the model directly, a surface the classical taxonomy cannot express. Second, it separates influence over training data (A4) from possession of weights (A3), since poisoning a fine-tuning corpus and computing gradients require different opportunities and enable different attacks. Operational catalogs such as MITRE ATLAS [51] and the OWASP Top 10 for LLM applications [52] enumerate techniques and application-level risks; AATM is complementary to them, supplying the graded access axis that determines which defenses are implementable in a given deployment and which robustness claims transfer to it.

### 5.2. Knowledge, Capability, and Goal

Beyond access, three further axes matter. **Knowledge** captures whether the adversary knows a defense is present and how it works; this axis separates *static* evaluation (the adversary does not adapt) from *adaptive* evaluation (the adversary optimizes against the deployed defense), a distinction we develop in Section 8.1. **Capability** captures query budget, compute, and whether the adversary can fine-tune. **Goals** include eliciting prohibited content (jailbreak), hijacking behavior (injection), exfiltrating training data or parameters (extraction and inversion), degrading or backdooring the model (poisoning), and denial of service.

The practical consequence, returned to throughout, is that a defense which resists an A1 adversary may say nothing about an A3 adversary who fine-tunes the safety behavior away [17], and a filter that stops static attacks may be defeated by an A1 adversary who adapts to it [12].

## 6. LLM Security Threats and Vulnerabilities

LLMs face a broad spectrum of risks, but this review focuses specifically on *security threats*: weaknesses an adversary can exploit to compromise model integrity, data confidentiality, or system availability. Adjacent concerns such as bias, fairness, misinformation, toxicity, intellectual property, explainability, and regulatory governance are important but are not, in themselves, weaknesses an adversary exploits, so we address them in Appendix A. They enter here only where they widen the attack surface, as when the same inability to separate instructions from data that enables prompt injection is also used to mass-produce misinformation.

This section first examines the failure modes within LLM safety training that create exploitable weaknesses, then presents a taxonomy of security threats across four dimensions: model-centric, data-centric, user-centric, and infrastructure-related. Throughout, we connect each threat to the AATM tiers of Section 5, so that the taxonomy identifies not only what can go wrong but the access an adversary needs to make it go wrong.

### 6.1. Failure Modes Instigating LLM Vulnerabilities

Three failure modes in safety training recur across the attack literature [53], and they are worth separating because they arise at different points in the training pipeline. Competing objectives are a property of what the model was optimized for. Mismatched generalization is a property of the gap between the distribution the model was pretrained on and the narrower one it was aligned on. Brittleness is a property of where in the parameters the resulting safety behavior ends up living. The three are not independent, and Section 8.1 sets out the property most often proposed as connecting them, but they license different attacks and, more consequentially for this paper, they are why defenses drawn from different rows of Table 3 can still fail on the same inputs.

Each licenses a recognizable family of attack. Competing objectives are exploited by prompts that force a choice between safety and instruction following, as in the “DAN” role-play jailbreak analyzed by Wei et al. [53]. Mismatched generalization is exploited by inputs safety training never saw, among them character-level obfuscation such as ROT13 or leetspeak, synonym replacement, and translation [54]. Brittleness is exploited directly, by low-cost fine-tuning or pruning that disrupts the sparse safety region without measurably degrading general performance [55].

The consequence is an asymmetry that shapes the whole defense landscape. The capability to remove safety is cheap once the weights are in hand, requiring neither large compute nor the original alignment data, while the capability to restore it is not. This is the hinge between the A1 and A2 tiers, where an adversary must work through the model’s behavior, and the A3 and A4 tiers, where an adversary can edit the mechanism that produces it. It is also why Section 7 finds no content-level defense that addresses A3 or A4: filtering the interface cannot protect a property that no longer exists in the weights behind it.

### 6.2. Model-Centric Threats

Model-centric threats target the model itself, its outputs, its parameters, or the data those parameters encode, rather than the system around it. The heading spans the full AATM range, covering attacks a black-box user can mount through a public interface and attacks that require possession of the weights or influence over the training corpus. That spread is the reason the category is a poor unit for defense planning. No single layer covers it, and the tier a particular instance assumes, rather than the label it carries, determines what can stop it.

*Adversarial and Evasion Attacks.* LLMs are susceptible to inputs manipulated to induce harmful or incorrect outputs without changing the model itself. A black-box adversary (A1) can craft transferable adversarial prompts through the public interface, while a white-box adversary (A3) can use gradients to optimize adversarial suffixes directly [15, 31, 56]; jailbreaks are the most common A1 instance and are treated under user-centric threats below. Because these attacks exploit the failure modes above rather than implementation bugs, they call for continuously updated defenses rather than one-time patches.

*Model Inversion and Extraction.* By probing a deployed model, an adversary can reconstruct confidential inputs or approximate the model itself. Inversion and membership-style attacks typically assume query or logit access (A1 to A2), while parameter extraction targets the weights directly (A3), posing both privacy and intellectual-property risks [57, 58].

*Backdoor and Trojan Insertion.* An adversary who can influence training or fine-tuning data (A4) can embed hidden triggers that cause malicious behavior only when activated, while leaving ordinary performance intact and detection difficult [59, 60].

What makes backdoors distinctive is not their potency but their evaluation profile. A backdoored model is indistinguishable from a clean one on every input that lacks the trigger, so standard benchmarking, red teaming, and safety evaluation all return clean results, and the defect survives the entire validation pipeline that would catch a degraded model. Detection therefore has to target the training data or the parameters rather than the behavior, which is exactly the access an application developer building on a hosted model does not have.

### 6.3. Data-Centric Threats

Data-centric threats concern the corpus rather than the model: what enters training, and what leaves the model that training produced. They divide into an integrity problem and a confidentiality problem, and the two sit at opposite ends of the access ladder. Poisoning requires the ability to influence training data (A4), which makes it the harder of the two to mount and harder still to detect once mounted. Memorization leaks to any adversary who can query (A1), and costs the adversary nothing but patience. Neither is addressable at the deployment interface, which is why Section 7 locates their defenses in the training and infrastructure layers rather than in filtering.

*Data Poisoning.* Crafted examples injected into the pipeline can implant backdoors or skew model behavior in ways that evade standard quality control, so countering them requires validation and anomaly detection across the whole pipeline rather than at a single checkpoint [61, 62].

*Privacy Leakage and Memorization.* Targeted queries can elicit memorized training content verbatim, producing unauthorized disclosures that may violate privacy regulation [63, 64].

Two features make this harder to defend than its A1 access tier suggests. The leak is not a malfunction but the same generalization the model is valued for, operating on data that should not have been generalized from, so there is no error signal to filter on. And the exposure is set at training time while the extraction happens at inference, so a deployment inherits whatever its base model memorized, with no interface-level control able to reach it.

### 6.4. User-Centric Threats

The interface between LLMs and their users is the most exposed attack surface, because prompts fuse trusted instructions with untrusted input.

It is also the surface on which almost all deployed defense effort is spent, which makes the difference between the two threats below worth stating precisely, because they are routinely discussed as one. A jailbreak persuades the model to abandon a policy it holds. An injection exploits the model's inability to tell an instruction it should follow from data it was merely asked to process. The first is a property of alignment, the second of the input format; they occupy different rows of Table 3; and a defense that demonstrably addresses one is not evidence about the other.

*Prompt Injection.* In **direct prompt injection** (A1), a user embeds instructions that override the system prompt or safety policy. In **indirect prompt injection** (A0), the malicious instructions arrive through content the model retrieves or a tool returns, so the adversary never queries the model directly [28]. Injection is the central user-centric security threat precisely because the model cannot reliably separate instructions from data.

*Jailbreaks.* Jailbreaks (A1) use role-play framings, obfuscation, or multi-step setups to push the model past its safety boundary [53]. As Section 8.1 discusses, jailbreaks are the clearest case where static evaluation overstates safety.

They are also the threat on which the field's evidence is weakest, for a structural reason. A jailbreak is a search over phrasings, so any fixed set of jailbreak prompts measures a defense against the search that produced that set rather than against the search an adversary would run next. Reported robustness against a published jailbreak corpus therefore describes the corpus at least as much as it describes the defense.

### 6.5. Infrastructure Threats

Infrastructure threats leave the model's behavior alone and attack what surrounds it: the serving stack, the dependencies it pulls in, the retrieval path that feeds it, and the operational process that keeps all three current. This is the least studied category in the LLM security literature and the most familiar to conventional security practice, and the second fact largely explains the first. These are not new problems, and their defenses are the mature ones. They

are treated here for two reasons: AATM tier A0 is defined by them, and Section 7.7 shows that the strongest available answer to prompt injection is architectural rather than behavioral, which places it in this category rather than among the model-level defenses where the literature looks for it.

**Deployment Exploitation and Extraction.** Attackers target the serving system to steal proprietary models or manipulate outputs, often exploiting implementation flaws or weak access controls, at the system level (A0) or, where weights are exposed, directly (A3) [65].

These attacks are notable in this taxonomy for needing nothing from the model at all. They succeed or fail on the same properties that decide any other software compromise, which means the relevant defenses are the conventional ones and the relevant expertise is already present in most organizations deploying LLMs. The risk is less that they are hard to stop than that they are filed under somebody else’s responsibility while the model team attends to prompts.

**Supply-Chain and Integration Risks.** Third-party components, plugins, and data pipelines expand the attack surface beyond the core model. A system-level adversary (A0) can compromise a dependency or a retrieval source to reach the model indirectly, including remote code execution in LLM application frameworks [66].

**Operational Pressures.** The computational demands of LLMs create operational strain that can indirectly weaken security, for instance when latency or cost pressures lead teams to relax filtering or defer updates [67]. Version management compounds this, since each release can introduce or resolve different vulnerabilities and requires security regression testing [68].

This taxonomy organizes LLM security threats across the model, data, user, and infrastructure dimensions and ties each to the adversary access it requires. The recurring lesson, developed in Section 8.1, is that these threats are linked by a small number of underlying failure modes. That is why a defense effective against one adversary tier says little about another, and it is also the reason layering delivers less than the count of layers suggests: defenses that answer threats sharing a failure mode inherit that mode as a common point of failure.

## 7. LLM Defense and Mitigation Strategies

Defenses can be sorted along two axes, and the difference between them is what this section is built on. The first axis is the *intervention point*: where in the lifecycle of an LLM-based system a defense acts, from the deployment interface inward to the weights and then outward to the surrounding infrastructure. This gives the five layers used throughout the LLM security literature and retained here: Layer 1, input/output censorship; Layer 2, model training and fine-tuning; Layer 3, adversarial training; Layer 4, monitoring and response; and Layer 5, infrastructure hardening. Figure 1 presents that taxonomy, and Table 2 states, for each layer, the AATM tier it presupposes, the attacks it addresses, whether reviewed instances were evaluated adaptively, and the overhead class it incurs under the cost model of Section 9. Its “adaptive-tested” column records whether the reviewed instances were evaluated against attacks designed with knowledge of the defense, which Section 8.1 argues is the distinction that decides what any reported figure means. Each row of that table reads as a single claim: a layer presupposes an adversary of at most this tier, addresses these attacks, has been evaluated against adaptive attacks to this degree, and costs this much to run. Two entries need reading with care. Layers 2 and 3 are marked *design-time* rather than with an AATM tier, because they act before deployment and so are not defined by what a live adversary can reach; and a layer may span several overhead classes, since instances within one layer can intervene at different points.

The overhead classes label how a defense consumes compute at inference: A adds prompt tokens, B adds a full model invocation, C evaluates several perturbed copies, D is paid before deployment, and E reads activations the forward pass has already produced. The labels are all that is needed here; Section 9 derives them from a compute model and shows that D and E additionally tier the defender, since they require the ability to train weights or to read internal state. Both the tier and the overhead-class entries are assigned from each method’s published description rather than from re-implementation, so they record what a method *requires* on its authors’ account rather than what it was measured to cost, and Section 12.6 states that as a limitation.

The second axis is the *dependency*: the property of the model or of the input on which a defense relies in order to work, and therefore the condition under which it fails. This axis cuts across the first. Two defenses in the same layer can rely on unrelated properties, and defenses in different layers can rely on the same one, which is why the intervention point predicts a defense’s cost and coverage well but predicts nothing about whether two defenses fail

## Layered LLM Defenses as an Ensemble



**Figure 1:** Taxonomy of LLM blue-teaming defenses by intervention point.

**Table 2**

Master mapping of the five defense layers to AATM tiers, attack coverage, adaptive evaluation, and overhead class (Section 9: A = additive tokens, B = extra serial passes, C = multiplicative sampling, D = training-time and amortized, E = activation-derived. Classes D and E additionally require defender access to weights or activations respectively.)

| Defense layer                    | Tier assumed | Attacks addressed                                                  | Adaptive-tested                                      | Overhead class                                           | Cost profile                       |
|----------------------------------|--------------|--------------------------------------------------------------------|------------------------------------------------------|----------------------------------------------------------|------------------------------------|
| L1: Input/output censorship      | A1           | Direct jailbreaks, direct injection, harmful output                | Rarely; constitutional classifiers are the exception | B; C (smoothing); A (guard prompts); E (probe screening) | Per-query passes; false positives  |
| L2: Model training & fine-tuning | Design-time  | Alignment failures, some A1 jailbreaks, prompt injection (trained) | Rarely                                               | D                                                        | High one-time train; low per-query |
| L3: Adversarial training         | Design-time  | Optimized suffix attacks within budget                             | Partially                                            | D; A (tuned prefixes)                                    | High train; possible utility loss  |
| L4: Monitoring & response        | A0–A1        | Anomalies, some injection, operational faults                      | n/a                                                  | B (often async); E (representation monitors)             | Infrastructure; detection variance |
| L5: Infrastructure hardening     | A0, A2–A3    | Extraction, inversion, RCE; A0 injection (by isolation)            | n/a                                                  | n/a (system-level)                                       | Deployment complexity; low compute |

together. Table 3 classifies the reviewed defenses along this second axis into seven rows. It is the instrument the rest of the paper runs on, so it is worth stating how to read it: a row groups defenses that would be defeated by the *same* move, and the third column names that move. Two defenses in one row are therefore near-substitutes, and two defenses in different rows are candidates for a stack. Those assignments are interpretive, and we state them as a hypothesis for the composition literature to test rather than as established fact: Section 10 argues that stacking within a row buys cost without independence, and Section 11 measures how far that holds.

The review that follows is organized by dependency rather than by layer, for three reasons. It is the axis the composition analysis of Section 10 runs on, it is the axis the experiment of Section 11 samples from, and it groups near-substitutes together, so that a practitioner reading a row sees the alternatives to a defense rather than its neighbors in the deployment pipeline. The subsections that follow take the rows in turn, naming for each the AATM tier its instances reach, the overhead class they incur, and, where the row maps onto the layer taxonomy, the layers they occupy. A final

**Table 3**

Reviewed defenses classified by the property each relies on, with the condition under which that property stops holding.

| Dependency                  | Representative defenses                                                     | Fails when                                                                      |
|-----------------------------|-----------------------------------------------------------------------------|---------------------------------------------------------------------------------|
| Token surface               | Perplexity filtering, blacklists, retokenization, Adversarial Prompt Shield | Attack is fluent or paraphrased                                                 |
| Semantic classification     | ShieldLM, Llama Guard, WildGuard, ShieldGemma, constitutional classifiers   | Harm is context-dependent or outside the classifier's policy                    |
| First-token distribution    | SafeDecoding, standard safety fine-tuning                                   | Adversary controls or bypasses the opening tokens [14]                          |
| Internal representations    | Circuit breakers, latent adversarial training, probe monitors               | Harmful computation is not separable in the monitored subspace                  |
| Input provenance            | Spotlighting, StruQ, SecAlign, instruction hierarchy                        | Trust boundary is mislabeled, or the model overrides learned precedence [69]    |
| Perturbation stability      | SmoothLLM, erase-and-check                                                  | Attack is robust to character-level perturbation, or exceeds the certified size |
| Architecture and capability | CaMeL, IsolateGPT, deterministic routing of sensitive operations            | Required functionality cannot be expressed within the capability policy         |

subsection covers the system-level controls that occupy no row, because they do not depend on a property of the model at all.

### 7.1. Token surface

These defenses read statistics of the prompt as a string. They sit in Layer 1, reach A1 only, and are the cheapest available, from free (blacklists, retokenization) to a single small classifier pass; Jain et al. [70] evaluate this row's baselines head to head, grouping them as perplexity-based detection, input preprocessing by paraphrase or retokenization, and adversarial training, and report that filtering and preprocessing work better against LLM attacks than the vision literature would predict. The Adversarial Prompt Shield [71] is representative of the strongest form: a DistilBERT classifier trained on Bot Adversarial Noisy Dialogue data, labeling prompts safe or unsafe, reducing attack success by up to 60% at a cost low enough for real-time use. Rule-based moderation belongs here too, including the programmable rails of *NeMo Guardrails* [72] and the hybrid rule-plus-classifier guardrails examined by Dong et al. [73], which enforce ethical and regional content standards.

The row's shared failure is that surface statistics are a proxy for intent, and the proxy is only as good as the attack's fluency. Blacklists and sanitization are defeated by obfuscation, a fixed classifier is evaded by patterns outside its training distribution, and rule-based systems are predictable but inflexible and require constant updating. Section 11 measures how sharply this row's apparent strength depends on the attack class it is measured against.

### 7.2. Semantic classification

This is the largest and best-evidenced row. A separate model reads the prompt, the response, or both, and judges harm against a policy. All instances are Layer 1, reach A1, and carry class-B overhead, one auxiliary invocation per screened item.

*ShieldLM* [2] fine-tunes an LLM on a bilingual corpus of 14,387 query–response pairs to act as a detector aligned with configurable safety standards, producing an explanation for each judgment rather than matching keywords; its cost is the need for a large annotated corpus. *Llama Guard* [1] instruction-tunes Llama2-7B as an input and output classifier deployed alongside the model, reporting AUPRC of 0.945 for prompt and 0.953 for response classification on its own test set. The paradigm has since become a family: *WildGuard* [74] unifies harmful-prompt detection, harmful-response detection, and refusal judgment in one 7B model, and *ShieldGemma* [75] provides Gemma-2-based classifiers from 2B to 27B, reporting an average AUPRC gain of 10.8% over the original Llama Guard at comparable size. Deployed moderation endpoints belong to the same row. Markov et al. [76] describe one in production and locate most of the engineering effort not in the classifier but in the apparatus around it: the content taxonomy, the labeling instructions,

data quality control, and an active-learning loop to capture rare categories. That is the row's failure mode seen from the inside, since what the classifier can judge is fixed by the policy it was given.

The most heavily red-teamed instance is *Constitutional Classifiers* [77], input and output classifiers trained on synthetic data generated from a natural-language constitution, which withstood over 3,000 hours of incentivized human red teaming without a universal jailbreak being found, at a 0.38 percentage-point increase in production refusals and roughly 23.7% inference overhead. Its successor [78] cuts that cost with a two-stage cascade in which a linear probe over activations screens all traffic and escalates only flagged exchanges: classifier compute falls by more than 40× against the authors' own exchange classifier and eight-fold against a probe-free two-stage cascade, giving roughly 1% absolute overhead at a 0.05% refusal rate over a month of shadow deployment. In the terms of Section 9 this is not cheaper class-B engineering but a change of class, since the screening stage reads activations the forward pass has already computed (class E), which is what lets it run unconditionally. The open question in this row is therefore no longer whether an auxiliary check helps but which overhead class the always-on stage can be drawn from.

Judgment can also be obtained from the target model itself rather than from a dedicated classifier. *LLM Self Defense* [79] feeds a generated response to a second instance prompted to judge harm, requiring no fine-tuning and reducing attack success to near zero on GPT-3.5 and Llama-2; Intention Analysis [80] prompts the model to analyze intent before answering, cutting attack success by an average of 48.2% across jailbreak benchmarks; backtranslation [81] infers the clean prompt that would have produced the observed response and refuses the original if the model refuses the reconstruction; and self-examination techniques [82, 83] have the model evaluate and revise its own output. Zhou et al. [84] take a game-theoretic route, using Monte Carlo Tree Search to simulate an adversary against a defender and preempt a sizable share of jailbreak attempts, at a computational cost that grows steeply with search depth. Benchmarks for calibrating this row include OR-Bench [85], which evaluates 25 LLMs on 80,000 toxic prompts to balance safe rejection against utility.

The row fails where the classifier's policy does not reach: harm that is context-dependent, distributed across turns, or simply outside the constitution it was trained on. Its recurring cost is not compute but false refusals, the quantity Section 10 shows binds first under stacking.

### 7.3. First-token distribution

This row depends on the model's output distribution over the first few generated tokens, which is where safety alignment as currently practiced lives [14]. It spans Layers 1, 2, and 3, and its instances are class B, class D, or class A depending on whether they intervene at decode time, at training time, or through a prompt.

*SafeDecoding* [86] contrasts the target model's token distribution with that of a safety-tuned expert over the opening tokens, amplifying safety disclaimers and attenuating jailbreak-aligned continuations; across five models, six attacks, and four benchmarks it reduces attack success while leaving utility essentially intact (about 1% deviation on MT-Bench for Vicuna), at the cost of a parallel expert decode. Steering generation at decode time predates its safety application: PPLM controls the attributes of a generated continuation [87], and DExperts contrasts an expert against an anti-expert at each step [88], which is the construction SafeDecoding applies to safety.

Standard safety fine-tuning occupies the same row from the training side, and Bianchi et al. [89] quantify both what it buys and what it costs: adding roughly 3% safety demonstrations when fine-tuning LLaMA substantially improves safety without significant loss on standard capability benchmarks, while adding too many produces exaggerated safety, in which the model refuses safe prompts that superficially resemble unsafe ones. That second finding is the refusal budget of Section 10 appearing within a single layer. SELF-ALIGN [90] aligns LLaMA-65b through topic-guided self-instruction, principle-driven self-alignment, principle engraving, and verbose cloning, using fewer than 300 lines of human annotation; the phrase "from scratch" in its title refers to aligning without distillation from an already-aligned teacher, not to pretraining, and the verbose-cloning stage may encode redundant content. Safety-aware training adds quantitative control over the process, as in the safety basin and VISAGE metrics of Peng et al. [91], which are model-specific and may not generalize, and in the cross-modal unlearning of Chakraborty et al. [92], which lowers adversarial success at additional training overhead. The fine-tuning procedure itself has been optimized in several directions: *MoGU* [93] keeps a usable and a safe model and routes between them at inference, which is the one instance in this row that pays a per-query cost; immunization [94] combines parameter-efficient fine-tuning with adversarial training on safety-preserving tasks; and DPO [95] improves dialogue safety and summarization through preference optimization, subject to the availability of high-quality adversarially labeled data. Defensive distillation, context distillation, and RLHF sit alongside these, trading computational efficiency against robustness.

Two approaches reach the same property from outside the weights. Prompt Adversarial Tuning [96] optimizes a short guard prefix, through alternating attack and defense controls: in the white-box setting it drives GCG on Vicuna-7B from 98% to 1% while preserving benign utility and reduces attack success by roughly 80% in the multi-model setting, with negligible inference overhead but smaller gains against black-box and adaptive attacks. Purple-teaming approaches [41] train attacker and defender together through self-play, reducing attack success by roughly 66 to 71% and reaching a defense success rate near 97.6%, at the price of a complex setup that can overfit adversarial patterns.

The row's shared failure is stated precisely by Qi et al. [14]: because the safety behavior is concentrated in the opening tokens, an adversary who controls or bypasses them defeats every mechanism that reads them. This is also the row whose protection is most easily removed, since Wei et al. [17] show that minor pruning of safety-critical regions significantly compromises safety. Layer 2 and Layer 3 are nonetheless the only place where resistance to A3 and A4 adversaries can originate at all, because they are the only place where the weights and the data that produce them are under the defender's control, and their economics are attractive: cost is concentrated before deployment and marginal inference cost is near zero. The qualification is that the protection is learned rather than enforced, so the layer raises the cost of an attack without bounding it.

#### 7.4. Internal representations

These defenses read or perturb hidden state rather than tokens. They span Layers 1, 3, and 4, reach A1, and are class E at inference when they read activations and class D when they are trained in.

Short-circuiting, or circuit breakers [5], intervenes on the internal representations that drive harmful generations, interrupting them before a harmful completion is produced rather than filtering output after the fact; its cost is concentrated in training while its runtime intervention reads representations already computed, so it occupies two overhead classes at once. Latent adversarial training [6] moves the adversarial perturbation from the input to the hidden representations, motivated by evidence that safety fine-tuning suppresses rather than removes harmful capabilities; targeted LAT improves jailbreak robustness, strengthens backdoor removal, and makes unlearned behaviors harder to relearn at the 8B scale. Continuous adversarial training [97] applies perturbations directly on token embeddings, an intermediate position on the same axis, and is efficient relative to conventional adversarial training but sensitive to the perturbation magnitude. Probe monitors, including the screening stage of the production cascade above [78], are the pure class-E instance.

The trajectory across this row and the previous one, from perturbations on the prompt through the embedding space to the hidden representations, is toward properties that are harder to escape by reformulation, and it increases the independence available to a stack designer for reasons Section 10 makes precise. The row fails where harmful computation is not linearly separable in the monitored subspace.

#### 7.5. Input provenance

This row is the only one aimed squarely at the A0 to A1 boundary that content filtering handles poorly. Its defenses do not ask whether text is harmful but where it came from, and whether instructions arriving in data may override instructions arriving from the developer.

Spotlighting [98] marks untrusted content with provenance-encoding transformations, cutting indirect-injection success from above 50% to below 2% on GPT-family models at class-A cost. *StruQ* [99] separates the trusted prompt from untrusted data with structured queries and fine-tunes on simulated injections, driving the success of optimization-free injection attacks to near zero with little utility loss, and *SecAlign* [3] strengthens this with preference optimization over paired responses to the intended and the injected instruction, holding even optimization-based attacks below 15% success across five models. The instruction hierarchy [4] trains models to prioritize privileged system instructions over user and tool content on the same principle. These are class-D defenses, but the protection is learned rather than enforced: models trained with an instruction hierarchy still fail against a portion of agentic injection attacks [69], which is what motivates the architectural row below.

#### 7.6. Perturbation stability

These defenses do not inspect the prompt at all. They perturb it, observe whether the model's behavior is stable, and infer an attack from instability. They sit in Layer 1, reach A1, and are the only row that is class C, since every term of the cost scales with the number of variants evaluated.

*SmoothLLM* [100] perturbs each prompt into several randomized copies and aggregates the responses, exploiting the brittleness of optimized adversarial suffixes to character-level perturbation. It requires no retraining and is model-agnostic, but multiplies per-query cost by the number of copies, and its protection is strongest against precisely the

brittle-suffix attacks it was designed for, weakening against adversaries who add a fluency or robustness constraint to their optimization. *SelfDenoise* [101] varies the same idea: rather than scoring the perturbed copies directly, it masks words at random, has the target model fill the masked spans back in, and evaluates the denoised versions, using the model's own multitasking ability in place of a separately trained denoiser. *Erase-and-check* [102] offers a different kind of evidence: it erases tokens and inspects every resulting subsequence with a safety filter, yielding a certificate that no harmful prompt is mislabeled safe under adversarial suffixes, insertions, or infusions up to a bounded size. Certified schemes remain narrow, and randomized or gradient-guided variants trade certification for speed, but within their regime their claims are the only ones in this review that survive an unrestricted adaptive adversary. The row inherits the classical robust-optimization lineage: adversarial examples generated by PGD [103] and FGSM, which assume an adversary bounded by the chosen perturbation budget; adversarial data augmentation through ensemble methods [104–106], which lengthen training; adversarial training carried out in the embedding space for language models [107]; and min-max and certification-based formulations such as TRADES [108], convex-relaxation defenses [109] and distributionally robust training [110], which provide certifiable robustness under conditions but can be conservative on clean data.

The row's boundary is explicit in a way most are not: it holds as far as the certified perturbation size and says nothing beyond it.

### 7.7. Architecture and capability

The last row does not ask the model to behave correctly at all. It constrains what a compromised or manipulated model is able to do, which is why its guarantees are the most durable surveyed here and why they are stated as properties of the system rather than as reductions in attack success.

*CaMeL* [69] wraps the LLM in a protective system layer that extracts control and data flow from the trusted user query, so that untrusted retrieved content can never alter program flow, and enforces capability-based policies before each tool call to prevent unauthorized exfiltration. On AgentDojo this yields 77% task completion with provable security, against 84% for an undefended agent. Its ingredients are borrowed rather than invented: control-flow integrity, access control, and information flow control, applied to a language model instead of a program.

The row is newer than the others and is filling in quickly. Wu et al. [111] isolate execution between components so that a compromised one cannot reach the rest of the system, Wu et al. [112] give the same problem an information-flow-control treatment, and Beurer-Kellner et al. [113] generalize from single systems to a set of design patterns, of which *CaMeL*'s is one. Deterministic routing of sensitive operations out of the model's action space belongs here too, and Section 12.1 finds it deployed. This is the design-level answer to the A0 tier: rather than asking the model to resist injected instructions, the architecture makes following them inconsequential. The price is paid in engineering rather than compute, and the utility cost appears as functionality that cannot be expressed within the capability policy rather than as added latency.

### 7.8. System-level controls outside the dependency taxonomy

Three groups of Layer 4 and Layer 5 mechanisms occupy no row, because they do not depend on a property of the model. They are included because a deployment needs them and because Table 2 would otherwise misrepresent what those layers contain.

*Confidentiality of the weights.* For edge-deployed transformers, *TransLinkGuard* [8] combines weight-matrix permutation with a lightweight trusted-execution-environment authorization module integrated into MLP blocks, at 0.1% computational overhead, though its effectiveness may diminish under complex multi-step extraction. Removing information already encoded in weights is harder than it appears: Patil et al. [114] edit weights with methods such as ROME to delete specific facts and find that whitebox and blackbox attacks still recover the supposedly deleted content roughly 38% of the time, because traces persist in intermediate hidden states and edits fail to generalize across rephrasings. Weight editing alone is therefore not yet a reliable extraction defense.

*Security of the surrounding software.* *LLMSMITH* [66] uses static analysis and dynamic scanning to detect remote code execution vulnerabilities in LLM-integrating frameworks, outperforming standard scanners while generating false positives and missing zero-day exploits. Pan et al. [115] review LLM-powered frameworks that automate vulnerability discovery, exploitation, and validation, and in the Capture-the-Flag domain Zou et al. [116] propose an automated LLM agent framework while Zou et al. [117] introduce CTFAgent, whose plan-and-execute paradigm and stateful task tree outperformed 88% of human participants in evaluated competitions.

*Operational integrity.* *TRANSOM* [7] combines automated fault tolerance through a finite state machine, ensemble anomaly detection reaching around 70% accuracy, and asynchronous checkpointing; on a GPU cluster running GPT3-175B it cut training time by 28%, improved checkpointing from 255 to 16 seconds, and reduced task restart times from hours to 12 minutes, with detection accuracy that varies across domains and requires recalibration. This class of mechanism defends availability rather than integrity, and it is the least studied in the blue-team literature.

Monitoring is the only layer that assumes defense will sometimes fail, and its value is measured in what happens afterward: how quickly a novel attack pattern is detected, attributed, and fed back into the layers above. That value is real but poorly evidenced, since reviewed instances report detection accuracy on their own deployments and nothing about detection under an adversary aware of the monitor.

## 7.9. Defense Trade-offs and Selection

No single defense provides complete protection. Three trade-offs recur across the rows above and motivate the cost model of Section 9 and the composition analysis of Section 10. The first is between *effectiveness and usability*, which is where the token-surface and semantic rows spend most of their budget: aggressive filtering reduces attack success but increases false positives, and practitioners must calibrate sensitivity to their risk tolerance. The second is between *security and performance*, which is where the first-token and internal-representation rows spend theirs, and which Section 9 converts from a qualitative concern into arithmetic. The third is between *generalization and specificity*: baseline defenses such as blacklisting are predictable and inexpensive but fail against adaptive adversaries, while specialized guardrails such as MedSafetyBench for healthcare [118] achieve high precision within their domain and may not transfer. The three are not independent, since a defense bought cheaply on the second axis usually pays for it on the first or the third, which is what makes defense selection a decision problem rather than a ranking. Table 4 takes a different cut. It does not tabulate the three trade-offs; it summarizes, for each *layer* rather than each row, what that intervention point offers, what it costs, and the deployment it suits. The switch of axis is deliberate: budgets, teams and deployment decisions are organized by intervention point, so a practitioner asking what a layer is good for is asking a layer question, even though the question of which defenses fail together is a row question.

**Table 4**

Strengths, limitations, and deployment fit by intervention layer. Resource requirements are given by overhead class in Table 2.

| Defense layer                    | Primary Strengths                                     | Key Limitations                                                      | Best Suited For                                 |
|----------------------------------|-------------------------------------------------------|----------------------------------------------------------------------|-------------------------------------------------|
| L1: Input/output censorship      | Immediate protection; easy deployment; model-agnostic | False positives; latency overhead; bypass via obfuscation            | User-facing apps; content moderation            |
| L2: Model training & fine-tuning | Embedded security; improved robustness                | High computational cost; requires retraining; potential utility loss | Safety-critical systems; long-term deployments  |
| L3: Adversarial training         | Strong attack resistance; adaptive learning           | Computationally expensive; may not generalize to novel attacks       | Research environments; high-security apps       |
| L4: Monitoring & response        | Real-time detection; adaptive responses               | Detection accuracy varies; infrastructure complexity                 | Production environments; enterprise deployments |
| L5: Infrastructure hardening     | System-level protection; prevents extraction attacks  | Implementation complexity; hardware dependencies                     | Edge deployments; proprietary model protection  |

The rows also make the coverage structure visible in a way the layers do not. Five of the seven reach A1 and no further; only the provenance and architecture rows reach A0, and no content-level row reaches A3 or A4, for the reason given in Section 6: filtering an interface cannot protect a property that no longer exists in the weights behind it. Section 10 makes the conditions under which such layering actually pays precise, and Section 11 measures whether they hold.

One qualification applies to every effectiveness figure quoted in this section, and it is severe enough to determine how the rest of the paper is built. Almost all of them were obtained against attacks fixed in advance of the defense.

What those numbers mean under an adversary who adapts is the first question the next section takes up, and the answer governs what the cost model, the composition rules and the measurement can each be built on.

## 8. Evaluation Methods and Benchmarks

Sound measurement is what lets a defense result be compared and trusted rather than taken in isolation, so the choice of adversary, metric and benchmark determines what a security claim can establish. We begin with the choice that matters most and is made least deliberately, namely whether the attacker is allowed to adapt, and then turn to the pipelines, metrics and benchmarks the field uses to report what it finds.

### 8.1. The limits of static evaluation

This is the central weakness in current evidence: robustness that holds only against attacks fixed in advance does not establish robustness against the adaptive adversary a real deployment faces. The defenses reviewed above are almost always evaluated against fixed attack sets, which measures robustness against an adversary who does not know the defense is present. This has a long precedent: in the adversarial-example literature, defenses that appeared robust were repeatedly broken once attackers adapted to them, for example by circumventing obfuscated gradients [11]. The same pattern now holds for LLMs. Andriushchenko et al. [12] show that simple adaptive attacks, using a hand-designed prompt template plus random search over a suffix to maximize a target logprob, reach near-complete attack success against a wide range of safety-aligned models, including one that was adversarially trained against a static attack. Separately, Souly et al. [18] demonstrate that many published attack-success and defense figures do not survive a stricter, rubric-based evaluation, because jailbreaks that bypass safety fine-tuning often degrade the very capabilities needed to produce a genuinely harmful answer.

A unifying explanation for why so many defenses prove brittle is *shallow safety alignment*: current alignment adapts the model’s output distribution mostly over the first few generated tokens, so suffix attacks, prefilling attacks, decoding-parameter exploits, and fine-tuning attacks all succeed by perturbing those initial tokens [14]. This single property links several otherwise separate vulnerabilities, motivates defenses whose safety behavior is deeper and more persistent, and supplies the composition analysis of Section 10 with its candidate mechanism for correlated failure across stacked defenses. Section 11 tests that candidate and does not confirm it: the dependence it measures is present in every pair, but conditioning on behavior difficulty attributes most of it to a common gradient rather than to a shared mechanism.

Two consequences follow for how this survey should be read. First, every effectiveness figure reported from the literature in the supplementary extraction tables is conditional on the original study’s attacks, models, and datasets, and the figures are therefore *not comparable across studies*. Second, a reported attack-success reduction is a meaningful lower bound on residual risk only if the evaluation was adaptive; most of the reviewed evaluations were not, which we treat as a material limitation rather than a footnote. Section 11 puts numbers on both consequences. The adaptive-versus-static gap there reaches 0.66 for a single filter, so a static figure can understate residual risk by roughly an order of magnitude, and by an unbounded factor where the static attack happens to be one the filter blocks outright; and the same experiment shows that a defense’s measured strength depends on the attack *class* as well as on whether the attacker adapts, since a perplexity filter scores near-perfect against optimized suffixes and near-useless against fluent prompts of comparable effectiveness.

### 8.2. Evaluation Framework for Mitigation Strategies

Automated evaluation pipelines simulate a range of adversarial scenarios while applying various mitigation techniques to quantify improvements in safety and robustness. The typical workflow generates adversarial inputs from benchmark datasets, applies the defense under test, runs inference on the defended inputs, and scores the results across multiple metrics. Algorithm 1 states it as five phases.

Algorithm 1 abstracts the practice common to the reviewed evaluations rather than reproducing any single published pipeline. Two design points deserve emphasis. Phase 3 is what separates a meaningful robustness estimate from one that merely reflects an adversary unaware of the defense, and a large gap between the static and adaptive ASR is itself a warning sign (Section 8.1). Phase 4 is the step no reviewed study performs: by composition rule 6, a stack whose components were each evaluated separately has no measured robustness, since the adversary optimizes against the weakest shared dependency of Table 3 rather than against any single layer. Phase 5 retains a stratified human assessment because automated classifiers miss context-dependent harms that expert evaluation catches [119].

**Algorithm 1** Automated evaluation pipeline for LLM defense strategies

---

**Require:** Adversarial dataset  $\mathcal{D}_{adv}$ , benign dataset  $\mathcal{D}_{ben}$ , target LLM  $\mathcal{M}$ , defense mechanisms  $\mathcal{F} = \{f_1, \dots, f_k\}$ , evaluation metrics  $\mathcal{E} = \{e_1, \dots, e_m\}$

**Ensure:** Defense effectiveness scores, trade-off analysis

- 1: **Phase 1: Baseline evaluation (no defense)**
- 2: **for** each input  $x \in \mathcal{D}_{adv} \cup \mathcal{D}_{ben}$  **do**
- 3:    $y_{base} \leftarrow \mathcal{M}(x)$
- 4:    $S_{base}[x] \leftarrow \text{EvaluateMetrics}(x, y_{base}, \mathcal{E})$
- 5: **end for**
- 6: Compute baseline ASR over  $\mathcal{D}_{adv}$
- 7: **Phase 2: Defense application and evaluation**
- 8: **for** each defense mechanism  $f_i \in \mathcal{F}$  **do**
- 9:    $\mathcal{M}_{def} \leftarrow \text{ApplyDefense}(\mathcal{M}, f_i)$
- 10:   **for** each input  $x \in \mathcal{D}_{adv} \cup \mathcal{D}_{ben}$  **do**
- 11:      $x_{filt} \leftarrow f_i.\text{FilterInput}(x)$
- 12:     **if**  $x_{filt} = \text{BLOCKED}$  **then**
- 13:        $y_{def} \leftarrow \text{"Request blocked by safety filter"}$
- 14:     **else**
- 15:        $y_{def} \leftarrow f_i.\text{FilterOutput}(\mathcal{M}_{def}(x_{filt}))$
- 16:     **end if**
- 17:      $S_{def}[f_i][x] \leftarrow \text{EvaluateMetrics}(x, y_{def}, \mathcal{E})$
- 18:      $L[f_i][x] \leftarrow \text{MeasureLatency}()$
- 19:   **end for**
- 20:   Compute defended ASR, FPR on benign inputs, and utility preservation
- 21: **end for**
- 22: **Phase 3: Adaptive evaluation**
- 23: **for** each defense  $f_i \in \mathcal{F}$  **do**
- 24:    $\mathcal{A}_{adapt} \leftarrow \text{AdaptiveAttacker}(\mathcal{M}_{def}, f_i)$
- 25:    $\mathcal{D}_{adapt} \leftarrow \mathcal{A}_{adapt}.\text{Optimize}(\mathcal{D}_{adv})$
- 26:   Compute  $ASR_{adapt}[f_i]$  over  $\mathcal{D}_{adapt}$
- 27: **end for**
- 28: **Phase 4: Stack evaluation**
- 29: Repeat Phases 2 and 3 with  $\mathcal{F}$  replaced by the composed stack  $\mathcal{S}$
- 30: **Phase 5: Human-in-the-loop validation (optional)**
- 31:  $\mathcal{D}_{samp} \leftarrow \text{StratifiedSample}(\mathcal{D}_{adv}, n = 100)$
- 32: Compute human-automated correlation  $\rho$
- 33: **return**  $\{ASR_{def}, ASR_{adapt}, FPR, UP, \text{TradeOff}, \rho\}$

---

The adversarial inputs at stage one are rarely purpose-built: safety datasets such as SafetyBench [120] and Real-ToxicityPrompts [121] are repurposed, AdvBench [15] is extended, and defense papers curate their own variants [1, 2], while system-level work substitutes operational measures such as overhead and recovery latency [8, 66].

### 8.3. Evaluation Metrics and Benchmarks

Comparing defenses requires agreeing on what is being measured, and the literature does not. The distinction that matters is between quantities defined independently of any one defense, which support comparison, and quantities a paper defines for its own system, which do not.

*Key Metrics.* It is useful to separate two kinds of quantity. *Cross-cutting metrics* are defined independently of any single defense and therefore support comparison across methods; these are summarized in Table 5. *Method-specific quantities* are figures that individual papers define for their own systems; these appear in the supplementary extraction tables and, as discussed in Section 8.1, are not comparable across studies. The last two rows of Table 5 are cross-cutting

in a further sense: they are defined over a pair of defenses rather than over one, which is why no reviewed study reports them.

**Table 5**

Cross-cutting metrics used to evaluate LLM defenses. The last two are defined over a *pair* of defenses.

| Metric                                | Definition                                                                                                                    | Range (direction)       |
|---------------------------------------|-------------------------------------------------------------------------------------------------------------------------------|-------------------------|
| Attack Success Rate (ASR)             | Fraction of adversarial inputs that elicit the targeted unsafe behavior                                                       | 0–1 (lower better)      |
| Defense Success Rate (DSR)            | Fraction of adversarial inputs the defense correctly blocks or neutralizes                                                    | 0–1 (higher better)     |
| False Positive Rate (FPR)             | Fraction of benign inputs incorrectly flagged or refused                                                                      | 0–1 (lower better)      |
| AUPRC                                 | Area under the precision–recall curve for safety classification                                                               | 0–1 (higher better)     |
| F1                                    | Harmonic mean of precision and recall for harmful-content detection                                                           | 0–1 (higher better)     |
| Utility preservation                  | Retained task performance under the defense                                                                                   | higher better           |
| Latency / overhead                    | Added inference or deployment cost                                                                                            | $\geq 0$ (lower better) |
| Failure correlation ( $\phi$ )        | Correlation of the breach indicators of two defenses on a common behavior set under a common adversary; the $\rho$ of Eq. (6) | –1 to 1 (lower better)  |
| Excess over independence ( $\Delta$ ) | Joint residual minus the multiplicative prediction, $ASR_{d_1 d_2} - \hat{p}_1 \hat{p}_2$                                     | –1 to 1 (lower better)  |

*Inter-metric correlations.* These metrics are not independent, and a defense that improves one routinely degrades another, which matters for composition because a stack inherits the trade. ASR reduction typically moves with DSR, since both measure resistance to adversarial inputs from complementary perspectives, while security metrics move against utility: aggressive input filtering raises the false positive rate and degrades performance on legitimate queries. These are the field’s working assumptions rather than measured quantities, and we report them as such.

*Predictive validity.* ASR measured on diverse, representative adversarial datasets is the strongest single indicator of real-world attack resistance [122], but no single metric predicts deployment success on its own, because each of the others carries a qualification that is deployment-specific: a false positive rate tracks user experience only against that deployment’s traffic, and utility preservation matters only in proportion to the task.

*Metric limitations.* ASR and DSR depend heavily on the definition of a successful attack, which varies across frameworks, and binary classification fails to capture severity gradations. Latency measurements depend on hardware, batch size, and input characteristics, so laboratory figures often underestimate production latency. Human evaluation is expensive and subject to inter-annotator variability, with human-automated correlation typically in the 0.6 to 0.8 range [119]. Most consequentially for Section 10, none of these metrics is defined over a stack: the literature has no standard quantity expressing the failure correlation between two defenses, which is precisely what the multiplicative assumption requires. Section 11 adopts  $\phi$  and the excess over independence  $\Delta$  of Eq. (6) for this purpose, and we propose them as the pair of cross-cutting metrics Table 5 currently lacks.

*Benchmarks for Defense Evaluation.* A benchmark comprises not only a dataset but a standardized evaluation methodology, metrics, and often comparison baselines. Table 6 surveys the established ones. It is a map of what the field measures against rather than a list of the materials this paper uses, which is why it includes benchmarks our experiment does not run: the gaps below are claims about the landscape, and a landscape claim needs the whole landscape.

These benchmarks cluster around a few evaluation goals: most target harmful-content elicitation under jailbreaks, while OR-Bench measures the opposite failure of over-refusal and StrongREJECT addresses the problem that many successful jailbreaks do not actually elicit useful harmful content. Three gaps are visible even across this set: coverage is predominantly English, only JailbreakBench and StrongREJECT are designed with the adaptive-evaluation concerns

**Table 6**

Established benchmarks used to evaluate LLM defenses and safety.

| Benchmark      | Ref   | Year | Focus                                                 | Composition and notes                                                                                          |
|----------------|-------|------|-------------------------------------------------------|----------------------------------------------------------------------------------------------------------------|
| AdvBench       | [15]  | 2023 | Harmful-behavior elicitation; suffix attacks          | 520 harmful behaviors and strings; the standard target set for GCG-style attacks and many defense evaluations  |
| HarmBench      | [122] | 2024 | Standardized automated red teaming and robust refusal | Common set of harmful behaviors scored by a shared classifier; compares many attacks and defenses head-to-head |
| SafetyBench    | [120] | 2023 | Multiple-choice safety knowledge                      | Thousands of questions across seven safety categories, in English and Chinese                                  |
| OR-Bench       | [85]  | 2025 | Over-refusal measurement                              | Around 80,000 prompts that appear toxic but are safe to answer, with a hard subset                             |
| JailbreakBench | [123] | 2024 | Open jailbreak robustness with a leaderboard          | 100 behaviors; standardized artifacts and evaluation for both attacks and defenses                             |
| StrongREJECT   | [18]  | 2024 | Rigorous jailbreak scoring                            | Rubric-based autograder that checks whether a jailbreak yields genuinely useful harmful content                |
| AgentDojo      | [124] | 2024 | Prompt injection against LLM agents                   | Dynamic agentic environment; the standard benchmark for A0 injection defenses such as CaMeL                    |

of Section 8.1 in mind, and none evaluates a composed stack rather than a single defense. The first gap is narrower on the red-team side, where cross-lingual evaluation has been carried out directly [25], which makes its absence on the defense side a property of the benchmarks rather than of the problem.

#### 8.4. Comparative Analysis

Read across the benchmarks above and the per-category results in the supplementary extraction tables, one imbalance dominates: most defenses are evaluated on harmful-content elicitation, so resistance to jailbreaks is comparatively well characterized while complementary failure modes are not. Combining defensive layers generally improves reported safety, but the underlying figures are not comparable across studies and the gaps above leave the composition question unasked, so these combinations are better read as qualitative evidence for defense-in-depth than as additive gains. Section 11 measures the missing correlation on one stack and finds it positive for every pair, which suggests the qualitative reading is the one to keep.

Effectiveness is only half of what a deployment decision depends on. The other half is what a defense costs to run, which the literature reports even less consistently than it reports security, and which the next section puts on the same footing for every defense reviewed here.

### 9. Quantifying Inference-Time Overhead and Defense Selection

The trade-offs of Section 7.9 are stated qualitatively, and qualitative statements do not support the comparisons a practitioner has to make: whether a second classifier pass costs more than randomized smoothing, whether a guard prefix is affordable at all, whether an always-on monitor is even possible on a given deployment. This section makes the cost side precise. It develops a compute model for a decoder-only transformer, reduces inference-time defenses to five overhead classes, shows that those classes also tier the *defender* by the access they hold, and closes by turning the result, together with the AATM tiers of Section 5, into a decision procedure and a worked example.

#### 9.1. The Cost Model

The cost profiles in Table 2 can be made precise. Consider a decoder-only transformer with  $L$  layers and model width  $d$ , serving a prompt of  $n$  tokens and generating  $m$  tokens. Counting two floating-point operations per multiply-accumulate, with multi-head attention and a  $4d$  feed-forward expansion, the forward cost of processing a sequence of

length  $s$  during prefill is

$$F_{\text{pre}}(s) \approx L (24 s d^2 + 4 s^2 d), \quad (1)$$

where the linear term collects the query, key, and value projections ( $6sd^2$ ), the output projection ( $2sd^2$ ), and the feed-forward block ( $16sd^2$ ), and the quadratic term collects the attention scores ( $2s^2d$ ) and the mixing product ( $2s^2d$ ). Grouped-query attention and mixture-of-experts routing change the constants, not the structure. With key-value caching, each generated token then costs

$$F_{\text{dec}}(s) \approx L (24 d^2 + 4 d s) \quad (2)$$

at context length  $s$ , and the cache itself occupies  $2 b L d s$  bytes per sequence at  $b$  bytes per element, which bounds batch size and hence serving throughput.

One caveat governs every figure derived below. Equations (1) and (2) count arithmetic, and arithmetic is a good proxy for the compute a defense consumes but a poor one for the latency a user experiences. Prefill is compute-bound and tracks the FLOP count closely; decoding is dominated by memory traffic, since each generated token reads the full weight matrix and the accumulated key-value cache for a single column of arithmetic, so wall-clock time there scales with bytes moved rather than with operations performed. The practical consequences run in the same direction throughout this paper, which is why we report overhead as a compute ratio and flag latency separately: a class-B defense that doubles compute may more than double perceived latency because its passes are serial, whereas a class-A defense that adds prompt tokens is absorbed almost entirely by a throughput-bound prefill. Treat the percentages below as compute budgets, not as response times.

Two consequences follow. First, the quadratic term in Eq. (1) overtakes the linear term only when  $s > 6d$ , that is 24,576 tokens at  $d = 4096$  and 49,152 at  $d = 8192$ ; typical guarded prompts in deployment are one to two orders of magnitude shorter, so within deployed regimes the marginal compute of an extra prompt token is nearly constant, and it would be misleading to describe prompt-level defenses as imposing quadratic cost. Second, commercial APIs price tokens linearly and discount cached prefixes, so the billed cost of a static defensive prefix is linear in its length and amortizable, independent of the compute accounting above.

A defense that adds  $k$  wrapper tokens to every prompt therefore adds

$$\Delta F = \underbrace{L [24 d^2 k + 4d (2nk + k^2)]}_{\text{prefill}} + \underbrace{4 d L k m}_{\text{decode}} \quad (3)$$

operations per query, linear in  $k$  throughout the deployed regime. This accounting motivates a five-class taxonomy of inference-time overhead, used as a column of Table 2 and as an input to the decision framework and composition rules below:

#### Class A (additive tokens)

The defense enlarges the prompt by  $k$  tokens; cost follows Eq. (3). Guard prefixes such as PAT's tuned prefix [96], prompt-level safety instructions, and spotlighting's input-marking transformations [98] are class A; the prefix is short and static, so its cost is small and amortizable by caching.

#### Class B (extra serial passes)

The defense adds one or more full model invocations per query: input or output classifiers such as ShieldLM [2] and Llama Guard [1], judge passes such as LLM Self Defense [79], intent-analysis prompting [80], backtranslation [81], and decoding-time contrasts such as SafeDecoding [86]. Each pass contributes its own  $F_{\text{pre}} + m' F_{\text{dec}}$  on the auxiliary model's dimensions and adds serially to latency unless run asynchronously.

#### Class C (multiplicative sampling)

The defense evaluates  $q$  perturbed variants of each query and aggregates, as in SmoothLLM's randomized copies [100] and erase-and-check's per-subsequence filter calls [102]; every term of the cost scales by  $q$ .

#### Class D (training-time, amortized)

The defense is paid for before deployment: safety fine-tuning, adversarial training such as CAT/CAPO [97], immunization [94], and circuit breakers [5]. Marginal inference cost is near zero, which is why Table 2 records their cost profile as a high one-time training cost with a low per-query cost.

### Class E (activation-derived)

The defense reads internal state that the forward pass has already computed, adding no tokens, no additional model invocation, and no sampling. A linear probe over hidden states costs  $2d$  operations per monitored position per layer, so probing all layers of an 8B-class model ( $L = 32$ ,  $d = 4096$ ) costs  $2dL = 262,144$  operations against a per-token decode cost of  $1.45 \times 10^{10}$  by Eq. (2), or 0.002% of baseline. Representation-level monitors, the runtime component of circuit breakers, and the screening stage of the production cascade of Section 7.2 belong here. Class E is four orders of magnitude cheaper than class B on the same query, which is the difference between a monitor that can run on every request and one that must be budgeted.

## 9.2. Overhead Classes Tier the Defender

The classes divide along a second axis that the cost accounting alone does not expose. Classes A, B, and C operate entirely over a black-box interface: they wrap the prompt, invoke the model again, or resample, and a practitioner building on a third-party API can deploy all three. Classes D and E cannot be deployed that way at all. Class D requires the ability to train or fine-tune the weights, and class E requires read access to internal activations. Just as AATM tiers an adversary by the access they hold, the overhead classes tier a *defender* by the access they hold, and the two axes are independent: an operator serving their own weights can reach every class, while an application developer building on a hosted API is confined to A, B, and C no matter what their compute budget allows. This is a selection constraint the decision framework must respect before cost enters the calculation, and it explains a pattern in Section 12.1 that the tier analysis alone leaves unexplained: all three reviewed deployments select from classes A and B not only because those are affordable but because two of the three sit above a model they do not train.

## 9.3. Decision Framework for Defense Selection

The two instruments combine into a selection procedure. Steps 1 and 2 fix what is *possible* for a given deployment, steps 3 and 4 what is *affordable*, and steps 5 and 6 what is *defensible*:

1. **Assess the exposed AATM tiers:** Identify the adversary tiers the deployment surface exposes: a public query interface exposes A1, retrieval and tool use add A0, released weights add A3, and third-party training data adds A4. Only layers that cover the exposed tiers (Table 2) are candidates.
2. **Establish which overhead classes are reachable:** Determine whether the deployment can train the weights (class D) and read activations (class E). A deployment on a hosted API is confined to classes A, B, and C regardless of budget.
3. **Evaluate resource constraints by overhead class:** Determine the computational budget and the acceptable latency separately, since Section 9.1 shows they are not the same constraint, then price the candidates by Eq. (5).
4. **Set the refusal budget by risk profile:** Decide what fraction of benign traffic the deployment can afford to lose before choosing layers. Section 11 finds refusals, not compute, to be the binding constraint on depth, so this budget rather than the FLOP budget is what determines how many layers a stack can carry.
5. **Compose by the rules of Section 10:** Cover tiers before deepening layers, buy what independence is available rather than redundancy, order by cost class, and budget refusals globally.
6. **Establish continuous evaluation:** Implement ongoing monitoring and regular red-teaming exercises against the composed stack rather than its components.

## 9.4. A Worked Example

Consider a customer-support assistant: a self-hosted 8B-class model ( $L = 32$ ,  $d = 4096$ ) behind a public chat interface, answering over retrieved knowledge-base content, with a typical prompt of  $n = 3,000$  tokens and  $m = 300$  generated tokens.

*Step 1, exposed tiers.* The public interface exposes A1; retrieval exposes A0. Weights are not released and training data is internal, so A3 and A4 lie outside the deployment's primary surface. By Table 2, the candidate layers are Layer 1 filtering, Layer 2 injection-robust training, Layer 4 monitoring, and Layer 5 isolation of the retrieval path.

*Step 2, reachable classes.* The model is self-hosted, so all five classes are available. An otherwise identical deployment on a hosted API would lose classes D and E, and would have to obtain its A0 to A1 coverage from spotlighting (class A) and classifiers (class B) instead of from injection-robust fine-tuning.

*Step 3, overhead budget.* Equations (1) and (2) give a baseline of about 43.4 TFLOPs of prefill and 14.5 GFLOPs per generated token, roughly 47.7 TFLOPs per query. A static 200-token guard prefix (class A) adds 3.26 TFLOPs by Eq. (3), about 6.8% per query, approaching zero marginal cost under prefix caching. Screening input and output with a same-size 8B classifier (class B, twice) roughly triples per-query compute, since each pass costs about one full prefill. Substituting a 1B-class guard model ( $L = 16$ ,  $d = 2048$ ) cuts each pass to 0.125 of that, because per-pass cost scales with  $Ld^2$ , so two 1B passes add roughly 22.7% rather than 182%; running the output pass over the generated tokens rather than the full context brings the pair to about 12.6%. Randomized smoothing with  $q = 4$  (class C), applied only to the 5% of traffic flagged as suspicious, raises mean compute by 15% while quadrupling worst-case latency on flagged queries. Injection-robust fine-tuning in the style of SecAlign (class D) is a one-time training cost with near-zero marginal inference cost.

*Steps 4 to 6, selection.* Step 4 binds first. A customer-support surface tolerates little benign refusal, which caps how many thresholded layers the stack can carry. Within that cap the framework prices a defensible stack: class-D injection-robust fine-tuning covering the A0 to A1 boundary at zero marginal cost, a 1B-class output classifier run asynchronously where policy allows, and the cached class-A prefix, for a marginal overhead near 12% with at most one added serial pass, plus Layer 4 logging feeding periodic red-team review. Only the guard model carries a tunable threshold, so the whole refusal budget is available to it rather than divided among layers, and Section 10.7 returns to the same deployment to spend it. A naive alternative, two same-size classifier passes plus smoothing on all traffic, costs more than four times the compute for coverage of the same tiers. The framework does not choose for the practitioner, but it converts the choice into arithmetic on stated assumptions, and it makes explicit what remains uncovered: an A3 or A4 adversary is out of scope for this surface and must be handled contractually and organizationally.

## 10. Composing Defenses

Everything to this point selects defenses one at a time. No production system works that way: all three deployments in Section 12.1 run stacks, and the defense-in-depth principle invoked throughout the literature is a claim about combinations rather than components. Yet the reviewed work evaluates almost exclusively in isolation, so a practitioner assembling a stack has no basis for predicting what it will deliver. This section supplies that basis. The instruments already developed are sufficient to derive how a stack behaves, and the derivation yields a result that is not obvious: the quantities a defender cares about compose in different, and partly opposing, ways.

Write  $S = \{\ell_1, \dots, \ell_r\}$  for a stack of  $r$  deployed defenses. We treat coverage, cost, residual attack success, and false refusal in turn.

### 10.1. Coverage composes as a saturating union

Tier coverage is the easy case. A stack covers the union of the tiers its components cover,

$$T(S) = \bigcup_{\ell \in S} T(\ell), \quad (4)$$

read off Table 2. Two consequences follow, and the second is the one practitioners get wrong. First, coverage is monotone: adding a layer never reduces the set of tiers defended. Second, and decisively, it saturates within a tier. Adding a second, third, or fourth A1 defense enlarges  $T(S)$  not at all. A stack of three guard models, a perplexity filter, and randomized smoothing still covers exactly  $\{A1\}$ , and an adversary who obtains weights (A3) or influences fine-tuning data (A4) walks past all five. This is the structural reason the cross-case analysis of Section 12.1 finds every reviewed deployment defending A0 to A1 and none defending A3 or A4: content-level defense is a tier-bounded resource, and no amount of investment inside the boundary crosses it. The first composition rule follows directly. Establish  $T(S) \supseteq T_{\text{exposed}}$  before deepening any single tier, and where a tier cannot be covered technically, record it as an organizational or contractual control rather than leaving it silently uncovered.

### 10.2. Cost composes by overhead class

Cost composes predictably, but the classes of Section 9 do not combine in the same way, and the difference determines what a stack can afford. Let  $F_{\text{base}}$  be the undefended per-query cost. Classes A and B add; class C multiplies; class D contributes nothing at inference; class E contributes a term that is nonzero but negligible at any realistic probe count. Writing  $\Delta F_a$  for the wrapper cost of class-A defense  $a$  under Eq. (3),  $F_b$  for the full auxiliary invocation of

class-B defense  $b$ ,  $F_e$  for the activation read of class-E defense  $e$ , and letting a class-C defense evaluate  $q$  variants on a fraction  $g$  of traffic,

$$\mathbb{E}[F(S)] \approx [1 + g(q - 1)] \left( F_{\text{base}} + \sum_a \Delta F_a \right) + \sum_b F_b + \sum_e F_e, \quad \sum_e F_e \ll F_{\text{base}}. \quad (5)$$

Three planning consequences follow. Class-E and class-A defenses are effectively free in a stack, so a defender should exhaust them before spending anywhere else. Class-B defenses add compute additively but latency serially, so the binding constraint is usually the latency floor rather than the FLOP budget; the mitigation is a smaller auxiliary model, since per-pass cost scales with  $Ld^2$ , or asynchronous execution where policy permits. Class C is where stacks explode. Two ungated multiplicative defenses compose as  $q_1 q_2$ , so a defender who smooths with four copies and certifies with erase-and-check over a twenty-token window is paying close to two orders of magnitude over baseline. Gating is what makes class C tractable: at  $g = 0.05$  and  $q = 4$ , Eq. (5) gives a mean overhead of 15% rather than 300%.

Ordering therefore matters even though Eq. (5) is symmetric in its terms. A cheap stage that screens all traffic and escalates only suspicious exchanges converts a multiplicative cost into a gated one, which is precisely the cascade design of Section 7.2.

### 10.3. Effectiveness composes only under failure independence

The quantity practitioners most want to compose is the one that composes worst. The tempting assumption is multiplicative: if two defenses each reduce attack success by 90%, the stack should leave 1% residual. That holds only if the two fail on disjoint inputs. Where failures are correlated, the stack is bounded below by the correlated component, and in the limit of perfect correlation the second defense contributes nothing at all.

Write  $b_\ell(i) \in \{0, 1\}$  for the event that layer  $\ell$  is breached on input  $i$ , and  $p_\ell = \Pr[b_\ell = 1]$  for its marginal attack success rate. For two layers the residual of the stack is the probability that both are breached, which for binary variables is exactly

$$\text{ASR}_{\ell_1 \ell_2} = p_1 p_2 + \rho \sqrt{p_1(1 - p_1) p_2(1 - p_2)}, \quad (6)$$

where  $\rho$  is the Pearson correlation of the two breach indicators, equal to their Matthews coefficient  $\phi$ . The multiplicative assumption is the special case  $\rho = 0$ , and the excess  $\Delta = \text{ASR}_{\ell_1 \ell_2} - p_1 p_2$  measures how far a stack falls short of it. This makes the missing quantity explicit:  $\rho$  is a single number, estimable from the breach vectors of two defenses evaluated on the same behaviors under the same adversary, and Section 11 estimates it.

Two properties of Eq. (6) bound what it can be used for, and we state them here rather than discover them later. First, it is an identity for two indicators evaluated *in parallel* on the same input. A deployed stack runs its members in sequence, and a member that perturbs the prompt or prepends a prefix changes what the next member sees, so the deployed pipeline is not the system Eq. (6) describes. Section 11.10 measures the size of that discrepancy and finds it large on both axes, which is why composition rule 5 below demands an end-to-end measurement rather than an assembled estimate. Second,  $\rho$  on binary indicators is bounded away from  $\pm 1$  whenever the two marginals differ, so  $\phi$  is not directly comparable across pairs with different  $p_1, p_2$ ; Section 11.1 states the bound and reports the normalization. Neither property weakens the central point, which is that  $\rho = 0$  is an assumption and not a default, but both constrain the arithmetic that follows.

*This is the fusion problem, restated for defenses.* Equation (6) is not new mathematics, and it is important to say so. Combining decision-makers whose errors are correlated is the founding problem of classifier fusion, where the correlation of two members' error indicators is one of the standard pairwise *diversity* measures, cataloged alongside the  $Q$ -statistic and the disagreement measure and shown to track ensemble accuracy only imperfectly [19, 33, 125]. An entire subfield is devoted to *creating* diversity rather than assuming it [20, 50], precisely because independent members are hard to obtain, and the adversarial case was posed thirty years later than the statistical one but with the same structure: an attacker who targets what a set of classifiers has in common defeats the set at the cost of defeating one [21, 22].

Two things follow immediately. First, our contribution is not the statistic.  $\phi$  is a diversity measure with a long history, and we adopt it rather than propose it. What is new is that no study of LLM defenses reports it, so a literature that argues for defense-in-depth on every page has never measured the quantity defense-in-depth depends on. Second, the fusion literature supplies the corrective this domain lacks: diversity must be engineered and verified, not inferred from the fact that two components look different. Composition rule 2 is that corrective stated for defense stacks, and

Section 11 is its verification. A third consequence concerns what kind of ensemble a defense stack is, and Section 10.5 takes it up once the size of the error is on the table.

Figure 2 plots both quantities. Neither panel shows measured data: both trace the behavior of the model derived in this section under its stated assumptions, with the dotted lines in (a) marking the multiplicative prediction that holds only at  $\rho = 0$ , and the union bound of Eq. (7) dashed against the independent form solid in (b). The failure correlation panel (a) is drawn over is the quantity no reviewed study reports, and Section 11 measures where a real seven-layer stack falls on that axis. Panel (a) shows the size of the error. At a marginal attack success of 0.20 per layer, the multiplicative assumption predicts a residual of 0.04, while a pair of layers correlated at  $\rho = 0.5$  leaves 0.12, three times as much, and a perfectly correlated pair leaves the full 0.20 that one layer alone would have left. There is a standing reason to expect correlation rather than independence here, developed in Section 8.1. Shallow safety alignment [14] shows that a single property, the model's output distribution over the first few generated tokens, underlies suffix attacks, prefilling attacks, decoding-parameter exploits, and fine-tuning attacks alike, so two defenses that both depend on that property would fail together. Whether that is why the measured pairs fail together is a separate question, and Section 11 finds the evidence for it weaker than the argument suggests. This gives a sharper design criterion than defense-in-depth as usually stated. What a stack purchases is not redundancy but *independence*, and independence is scarce.

Table 3, introduced in Section 7, is the instrument this argument needs. Defenses in the same row are near-substitutes, and stacking them compounds cost without compounding protection. Defenses in different rows are complements, and stacking those is what defense-in-depth was meant to describe.

The table explains results that otherwise look anomalous. Constitutional classifiers survived thousands of hours of incentivized red teaming without a universal jailbreak [77] in part because the input and output classifiers fail on different inputs: an attack engineered to read as benign on submission must still produce harmful content that the output stage inspects. CaMeL [69] attains provable security against A0 injection not by detecting injected instructions more accurately but by moving to a row no detector occupies, making compliance with them inconsequential. Conversely, a stack of three semantic classifiers occupies one row, and an adaptive adversary who defeats the shared dependency defeats all three at once, which is exactly the failure that simple adaptive attacks exploit against models hardened only against static attacks [12].

The argument is not ours alone. The CaMeL authors observe that their design is directly compatible with defenses that harden the model itself, and that combining the two would yield a stronger security argument than either alone [69]. That is precisely the cross-row composition Table 3 recommends, stated by the designers of one of the rows. What is missing is not the intuition but the measurement: no reviewed study reports the failure correlation between two defenses, which is the quantity the multiplicative assumption requires and the one a composition literature would need to establish. Section 11 supplies it for a seven-layer stack, and the result qualifies the reading of Table 3 above in two ways. Cross-row pairs are correlated too, so cross-row selection buys more independence than same-row selection without buying the full multiplicative gain. And the correlation the table attributes to shared mechanism is, for most pairs, attributable instead to a common difficulty gradient once that is held fixed (Section 11.6); which association survives stratification, if any, turns out to depend on the labeling regime (Section 11.7). The table therefore remains useful as a selection heuristic, and Section 11.10 shows that under the reported labels it recovers the same members a search over the joint breach data would choose, though that agreement does not survive every labeling regime. Its causal reading is weaker than stated here and should be treated as a hypothesis rather than a finding.

#### 10.4. False refusals compose against the defender

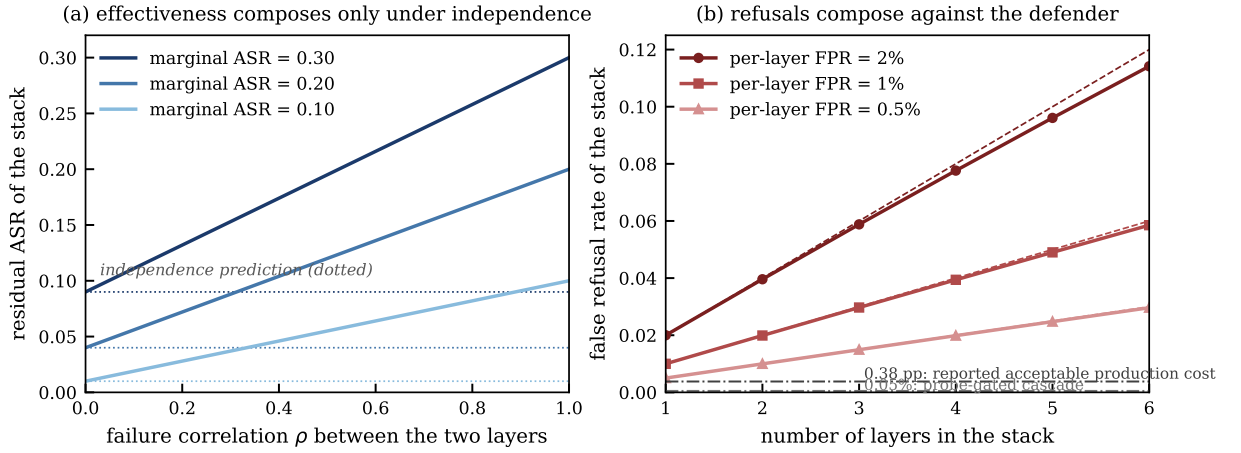
The asymmetry that bounds stack depth is that security composes sublinearly while the usability cost composes in the defender's disfavor. A query is refused if *any* layer refuses it, so false-positive rates take a union rather than an intersection. For layers with false-positive rates  $f_i$ ,

$$\text{FPR}(S) \leq \sum_i f_i, \quad \text{FPR}(S) \approx 1 - \prod_i (1 - f_i) \text{ under independence,} \quad (7)$$

and in either form  $\text{FPR}(S) \geq \max_i f_i$ , strictly increasing in depth. Worse, the same independence that makes a stack effective makes its refusals accumulate fastest: perfectly correlated layers refuse the same queries and add nothing to the refusal rate, while perfectly independent layers add the most protection and the most refusals together. There is no configuration that escapes the trade, only configurations that price it favorably.

Figure 2(b) makes the arithmetic concrete. The magnitude is operationally serious. Three layers at half a percent each yield roughly 1.5% of legitimate traffic refused, about four times the 0.38 percentage-point increase reported as

an acceptable production cost [77] and thirty times the 0.05% achieved by a probe-gated cascade [78]. Refusal budget, not compute, is therefore the binding constraint on stack depth in most deployments, and it should be allocated as a global quantity rather than negotiated layer by layer.



**Figure 2:** How the two quantities that trade against each other behave under stacking. (a) Residual attack success of a two-layer stack against the failure correlation  $\rho$ , at three marginal attack success rates. (b) False refusal rate against stack depth, at three per-layer rates.

### 10.5. Architectural dependence: how a defense stack differs from a classifier ensemble

The fusion literature’s account of why members correlate is a *sampling* account. Ensemble members are estimated from overlapping draws of a common training distribution under related inductive biases, so their errors covary, and the remedies follow the diagnosis: resample the data, perturb the representation, vary the hypothesis class, penalize agreement during training [20, 50]. Each remedy works by widening the pool from which members are drawn. Sampling dependence is attenuable in principle, and much of the subfield consists of methods for attenuating it.

Defense stacks carry a dependence with a different origin, and it is not attenuable that way. The members here are not estimated from a shared sample; several of them are not estimated at all, since a perplexity threshold and a randomized-smoothing wrapper have no training set in common with a semantic classifier or with each other. What they share instead is the object they operate on. Every member wraps the same target model, and the safety behavior of that model is concentrated in its output distribution over the first few generated tokens [14]. An input that moves the model off that narrow behavior is, simultaneously, an input on which a filter tuned to recognize such inputs is more likely to be wrong, a probe trained on representations of such inputs is more likely to be wrong, and a smoothing wrapper relying on the instability of such inputs is more likely to be wrong. The shared point of failure is located in the wrapped model rather than in any member, and no amount of member diversity removes it, because the members are not the thing that is shared.

We call this *architectural* dependence, and three consequences follow that the sampling account does not predict.

- **Diversity creation has to target the substrate, not the pool.** Adding a fifth mechanistically distinct filter cannot reduce a dependence carried by the model all five wrap. The only interventions that reach it either deepen the alignment of the target model, so that safety is not concentrated where every member’s blind spot is, or move a member off the substrate entirely, which is what the architecture row of Table 3 does by constraining what a compromised model can *do* rather than judging what it says. That is a design principle the fusion literature has no analogue for, because a classifier ensemble has no substrate to leave.
- **The adversary chooses the correlation.** In the statistical case the dependence structure is a property of the data-generating process and is fixed once the members are trained. Here it is partly under the attacker’s control: an adversary who targets the shared property induces correlated failure on demand, which is the adversarial-fusion

observation of Biggio et al. [21, 22] sharpened by the fact that the shared property is now nameable. Section 11.2 shows the practical form of this. A defense’s measured worth, and therefore the measured diversity of any pair containing it, changes with the attack class used, so diversity here is a property of a (stack, adversary) pair rather than of the stack alone.

- **A common-cause term is present by construction.** Because every member is evaluated on outputs of one model, behaviors that are intrinsically easier to elicit raise every member’s failure probability together, before any mechanism overlap. Section 11.6 separates this term from mechanism overlap by stratification and finds that it accounts for most of the measured association. This matters for interpretation and not for arithmetic: Eq. (6) is agnostic about why  $\rho$  is nonzero.

The practical upshot is that a defense stack is an ensemble whose achievable diversity is capped by a property none of its members controls. Composition rule 2 should be read in that light: selecting across dependency rows spends a stack budget well, and it does not, and cannot, restore the independence the multiplicative model assumes.

## 10.6. Composition rules

The analysis reduces to six rules, which together take a practitioner from the per-layer map of Table 2 to a defensible stack.

1. **Cover tiers before deepening layers.** Satisfy  $T(S) \supseteq T_{\text{exposed}}$  by Eq. (4) first. A tier left uncovered is not mitigated by additional depth elsewhere, and should be recorded as an organizational control.
2. **Buy independence, not redundancy, and expect less of it than the rows suggest.** Select at most one defense per row of Table 3: a second defense from the same row adds cost and refusals while adding little residual protection. Prefer, among cross-row candidates, layers whose blind spots lie in different *attack classes*, since that is the dimension along which complementarity was actually observed. Cross-row selection improves the joint residual but does not restore multiplicativity: measured cross-row correlation is positive throughout, and below the largest within-row value though not below every one of them (Section 11). Treat Eq. (6) at  $\rho = 0$  as a lower bound on residual risk that no real stack attains, not as a design target. Use the measured correlation to *choose* members and not to *forecast* the result of choosing them; rule 6 covers the second.
3. **Order by cost class, and gate with the cheapest.** Spend class E first, then class A, then class B, then class C, in ascending order of the contribution each makes to Eq. (5). A class-E screen is the cheapest available gate: because it reads state the forward pass already produced, it can run on every request and set the gating fraction  $g$  for the classes behind it at effectively no cost. This is why a probe-gated cascade reaches roughly 1% overhead where an unconditional classifier stage costs 23.7% [77, 78]. A deployment without activation access cannot open with class E and must gate with a small class-B model instead.
4. **Gate multiplicative defenses on a cheap signal.** Reduce  $g$  rather than  $q$ ; a class-C defense applied to all traffic is almost never the right configuration.
5. **Measure the stack’s refusal rate; do not assemble it from per-layer rates.** A per-layer rate that looks tolerable in isolation is not informative about the stack, and neither form of Eq. (7) recovers the deployed rate from the parts. Section 11.10 measures both failure directions: positively correlated refusals put the union of independently measured decisions *below* the independence prediction, while sequential execution, in which each layer alters the input the next one sees, puts the deployed pipeline *above* it. The union bound remains valid as a bound and is useful for rejecting a configuration early, but a refusal budget must be allocated against a measured rate for the assembled pipeline.
6. **Evaluate the stack, not the components.** An adaptive adversary attacks the composition. A stack whose components were each evaluated separately has no measured robustness, because the adversary optimizes against the weakest shared dependency rather than against any single layer. The error this avoids is not a matter of a few points either. Intersecting per-layer breach vectors and attacking the assembled stack directly disagreed by the full size of the estimate on the one configuration where the comparison could discriminate (Section 11.2), and it disagreed in the conservative direction, so a stack estimated from parts can be worse than it looks on refusals and better than it looks on residual risk. Neither error is acceptable in a deployment decision, and only an end-to-end

measurement removes both. Two further conditions make such a measurement meaningful: the adversary must be adaptive, and at least one fluent and one optimized attack class must be run, since a defense's measured worth is a property of the class it faces (Section 11.1).

### 10.7. Composition in the worked example

Return to the customer-support assistant of Section 9.4. Rule 1 fixes coverage: the exposed tiers are A0 and A1, so the stack needs the provenance row for A0 and at least one content-level defense for A1. Rule 2 then forbids the common configuration of an input classifier plus an output classifier plus a perplexity filter, since the first two share the semantic-classification row and the third adds a token-surface defense that any fluent attack defeats. The second half of rule 2 sharpens the same verdict: the perplexity filter and the classifiers are not merely same-row or adjacent-row neighbors, they cover overlapping attack classes, and the measured behavior of exactly this pairing is reported in Section 11.5.

The rules instead select one defense from each of three rows: injection-robust fine-tuning in the style of SecAlign from the provenance row (class D, zero marginal inference cost), a 1B-class output guard from the semantic row (class B, roughly 12% of baseline, asynchronous where policy allows), and a cached guard prefix (class A, 3.26 TFLOPs, approaching zero under prefix caching). Marginal overhead is near 12% with at most one added serial pass. By Eq. (7), only the guard model contributes materially to refusals, so the full refusal budget can be allocated to tuning its threshold rather than split three ways.

Compare the naive alternative: two same-size classifier passes plus ungated smoothing at  $q = 4$ . Equation (5) prices it above four times the compute of the selected stack, Eq. (4) shows it covers no additional tier, Table 3 shows both classifiers occupy one row so the second adds little independence, and Eq. (7) shows its refusal rate is the sum of three contributions rather than one. It is worse on every axis the framework measures. That the comparison is decidable at all, before any deployment, is the point of the framework.

## 11. Measuring Failure Correlation Between Stacked Defenses

Section 10 argued that layered defenses compound only when their failures are independent, and identified the failure correlation between two defenses as the quantity no reviewed study reports. This section supplies it. We measure  $\phi$  directly for every pair drawn from a seven-layer stack, under a shared adaptive adversary and a gold judge, and report what the measurement implies for the composition rules of Section 10.

### 11.1. Measurement protocol

For each defense  $d$  and behavior  $i$  we record a binary breach  $b_d(i) \in \{0, 1\}$  under a gold judge, and for each pair report the marginals  $\hat{p}_1, \hat{p}_2$ , the joint  $\text{ASR}_{d_1 d_2} = \frac{1}{n} \sum_i b_{d_1}(i) b_{d_2}(i)$ , the excess over independence  $\Delta = \text{ASR}_{d_1 d_2} - \hat{p}_1 \hat{p}_2$ , and the failure correlation  $\phi$ , the Matthews coefficient of the two breach vectors [126], which is exactly the  $\rho$  of Eq. (6) and the axis over which Figure 2(a) is drawn. As a pairwise diversity measure  $\phi$  is standard in classifier fusion [19]; what is novel here is its target, not its form.

*Setup.* Behaviors are the  $n = 100$  harmful prompts of JailbreakBench [123] with its matched benign set. Seven defenses span the dependency rows of Table 3: a windowed perplexity filter in the form used by Jain et al. [70] and a surface token-anomaly filter (token surface); Llama Guard 3 8B [1] (semantic classification); a refusal-priming prefix (first-token distribution, a SafeDecoding stand-in); SmoothLLM [100] at  $q=4$  with 10% character perturbation (perturbation stability); and linear probes on hidden states at layers 16 and 8 (internal representations). This stack is a measurement instrument, not a recommended deployment, and it deliberately violates composition rule 2: it contains two token-surface filters and two probes, precisely so that within-row pairs exist to test H1 against. Figures reported for the assembled seven-layer configuration, in particular its benign refusal rate, should therefore be read as an upper bound on what depth costs rather than as the cost of a well-selected stack. Section 11.10 prices the rule-compliant four-layer subset separately, and that is the configuration a practitioner should compare against. Every thresholded filter has its threshold set so that it *blocks* 1% of the benign prompts, a 1% filter false-positive rate; the realized block rate is 0.010 for all four. This is a different quantity from the system-level false-refusal rate reported under H3, which also counts prompts the base model declines of its own accord, and the two should not be confused (Section 11.2). Thresholds are set in-sample on the same benign prompts later used to report refusal rates, a limitation we return to in Section 11.11. The in-loop attack signal is the target logprob of the affirmative prefix; the StrongREJECT autograder [18] scores

breach *post hoc* at its standard binarization threshold and never participates in the search. Confidence intervals are 10,000-resample bootstrap percentile intervals over behaviors, and  $q$ -values are Benjamini–Hochberg adjusted [127] over the fifteen pairs tested.

*Attack, target and defense configuration.* Table 7 states the attack budgets and decoding settings in full, and Table 8 the operating point of every defense, so that the residual attack success rates below can be read against the search effort that produced them. Two points are load-bearing for what follows. The in-loop signal is the target logprob of the affirmative prefix over its first 12 tokens; the StrongREJECT autograder scores breach *post hoc* and is unreachable from the search, which is enforced by a static test rather than by convention. And every filter threshold is set so that it blocks 1% of the benign prompts, which at  $n = 100$  is a single prompt, a calibration limitation we quantify in Section 11.11.

**Table 7**

Attack configuration. Both adversaries run at a fixed budget with no restarts, against the same targets and the same behavior set. Only the block matching the adversary in use is live in a given run.

|                        | fluent (AutoDAN-style)                                           | GCG                           |
|------------------------|------------------------------------------------------------------|-------------------------------|
| generations / steps    | 30 generations                                                   | 60 steps                      |
| population             | 24                                                               | –                             |
| elite fraction         | 0.25                                                             | –                             |
| mutation rate          | 0.15                                                             | –                             |
| suffix length          | –                                                                | 20 tokens                     |
| top- $k$               | –                                                                | 128                           |
| candidate batch        | –                                                                | 64                            |
| restarts               | none                                                             | none                          |
| stopping               | fixed budget                                                     | loss < 0.4 or steps exhausted |
| evaluations / behavior | $30 \times 24 = 720$                                             | $\leq 60 \times 64$           |
| <i>Shared</i>          |                                                                  |                               |
| in-loop objective      | target logprob of the affirmative prefix, first 12 tokens        |                               |
| target decoding        | greedy (no sampling), max_new_tokens 256, fp16, no system prompt |                               |
| seed                   | a single seed; see Section 11.11                                 |                               |

**Table 8**

Defense operating points. Thresholded filters are calibrated to block 1% of the benign prompts; the realised block rate is 0.010 for all four. Llama Guard, the refusal-priming prefix and SmoothLLM have no scalar threshold.

| Defense                         | Dependency row | Threshold     | Benign block rate |
|---------------------------------|----------------|---------------|-------------------|
| perplexity (window 16)          | token surface  | 4.849077      | 0.010             |
| token-anomaly                   | token surface  | 0.272857      | 0.010             |
| probe <sub>16</sub>             | internal repr. | 28.924215     | 0.010             |
| probe <sub>8</sub>              | internal repr. | 110.767708    | 0.010             |
| Llama Guard 3 8B                | semantic       | categorical   | –                 |
| refusal-prime (prefix)          | first-token    | n/a           | –                 |
| SmoothLLM ( $q = 4$ , 10% swap) | perturbation   | majority vote | –                 |

Llama Guard is meta-llama/Llama-Guard-3-8B in fp16, resident on GPU, screening both input and output. The two probes are linear, trained on a pool of 520 training and 130 validation examples deduplicated at Jaccard 0.6, reaching validation AUROC 1.0; that near-ceiling figure is itself discussed in Section 11.11. All runs used a single NVIDIA H100 per job and consumed roughly 60 GPU-hours in total, of which about half was spent on jobs that terminated without producing reported results.

*Comparability across pairs.*  $\phi$  on two binary indicators cannot reach 1 unless the marginals agree. Writing  $p_1 \leq p_2$  without loss of generality, the attainable maximum is

$$\phi_{\max}(p_1, p_2) = \sqrt{\frac{p_1(1-p_2)}{p_2(1-p_1)}}, \quad (8)$$

attained when breaching the less permissive defense implies breaching the more permissive one. Our marginals span 0.35 to 0.68, so  $\phi_{\max}$  ranges from 0.503 to 1.000 across the fifteen pairs, a factor of two, and a raw ordering by  $\phi$  is therefore not an ordering by dependence. We report both: raw  $\phi$  for continuity with the fusion literature, which tabulates it unnormalized [19], and the normalized  $\phi/\phi_{\max}$  in the adjacent column of Table 10 wherever pairs are ranked. Figure 5 plots raw  $\phi$ , since the bootstrap intervals are on that scale.

Normalization does not change the verdict, because  $\phi_{\max} > 0$  for every pair and every measured  $\phi$  is positive, and it does not change the top of the ordering: the probe pair remains the largest effect by a clear margin (0.893 against a next-highest 0.733). It does change the middle, and in one place the change matters. The primary same-row pair, perplexity  $\times$  token-anomaly, has the second-lowest  $\phi_{\max}$  in the grid at 0.527, because its marginals are the most unequal of any pair. Ranked raw it is twelfth of fifteen; ranked normalized it is fourth. Both same-row pairs therefore sit in the upper third of the normalized ordering, which is what the dependency taxonomy predicts and what the raw ordering obscures. We flag this because the raw ranking, read on its own, understates the same-row effect rather than overstating it. The same bound is why the  $Q$ -statistic saturates for the one defense with a near-degenerate marginal, a point Section 11.9 returns to.

*Multiplicity under dependence.* The fifteen pairs are drawn from seven defenses and therefore share members, so the hypotheses are not independent. Benjamini–Hochberg controls the false discovery rate under positive regression dependence, which is the direction the shared breach vectors induce, so the correction is valid here but is not conservative in the way an independent-tests reading would suggest. All fifteen  $q$ -values fall below 0.005, so the conclusion does not depend on the choice of correction.

*The attack class determines what can be measured.* We ran two adaptive adversaries. GCG [15] optimizes an adversarial suffix of arbitrary tokens; AutoDAN [128] performs a sentence-level genetic search over fluent prompts. Their prompts differ sharply in surface statistics, and the perplexity filter is calibrated at a windowed negative-log-likelihood threshold of 4.85. Placing four prompt styles on that scale, as Table 9 does, puts the two attack classes on opposite sides of it.

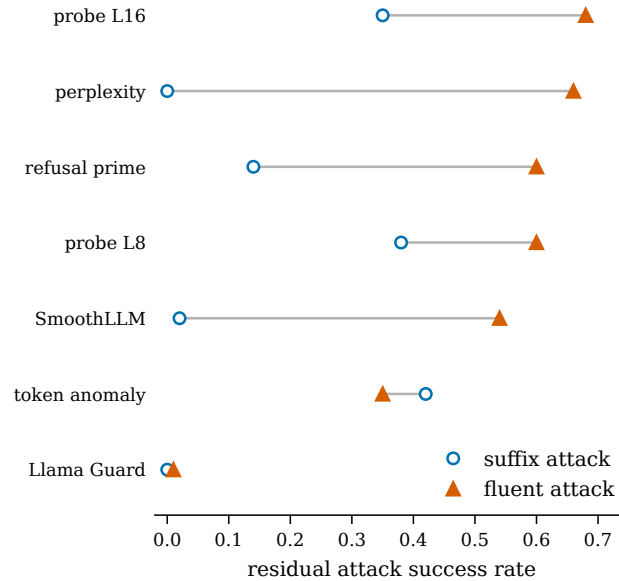
**Table 9**

Fluency of four prompt styles and whether the perplexity filter admits them. Mean windowed negative log-likelihood over five probe prompts per style; the admission column counts how many of the five pass the filter at its calibrated threshold.

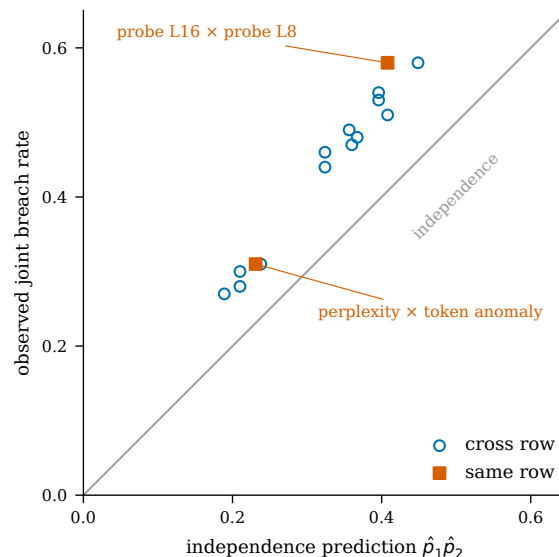
| Prompt style        | mean windowed NLL | admitted by the perplexity filter |
|---------------------|-------------------|-----------------------------------|
| Plain behavior      | 3.18              | 5/5                               |
| AutoDAN (fluent)    | 4.47              | 5/5                               |
| Compliance template | 5.15              | 0/5                               |
| GCG suffix          | 13.56             | 0/5                               |

Under GCG the perplexity filter blocks 100/100 attack prompts and its residual ASR is exactly 0, so every pair containing it has an *undefined*  $\phi$ . Under AutoDAN it blocks 0/100 and its residual ASR is 0.66. **The filter’s apparent invulnerability is a property of the attack class, not of the defense.** This is the central methodological caution of this section, and it is the concrete form of the warning issued abstractly in Section 7.6: a defense evaluated only against high-perplexity suffix attacks will be scored as near-perfect and will contribute a spurious zero to any composition estimate. All results below therefore use the fluent adversary, which admits every defense into the measurement. Figure 3 shows the effect across the whole stack. Both columns there re-optimize against each defense, so neither is the weaker static transfer suffix tabulated in Section 11.4; and the suffix column is a floor rather than a worst case for the two defenses that neutralize the suffix class, since an adversary free to choose between an optimized suffix and a plain prompt reaches 0.07 against the perplexity filter and 0.06 against SmoothLLM (Section 11.4). The reversal the figure shows is not confined to the perplexity filter: SmoothLLM holds the adaptive suffix attack to 0.02 and the fluent

and a scalar signal (A1 to A2), so the adversary that defeats more of this stack is also the cheaper one to mount.



**Figure 3:** Residual attack success rate of each defense under the two adaptive adversaries. Vicuna-7B-v1.5,  $n = 100$ .



**Figure 4:** Observed joint breach rate against the independence prediction  $\hat{p}_1\hat{p}_2$  for all fifteen measurable pairs. Vicuna-7B-v1.5 under the fluent adversary,  $n = 100$ .

**Table 10**

Measured failure correlation, Vicuna-7B-v1.5 [129] under the fluent (AutoDAN-style) adversary,  $n = 100$ , StrongREJECT gold.  $\Delta$  is the excess over the independence prediction. Llama Guard pairs are omitted (Section 11.5).

| Defense pair                             | Rows                   | $\hat{p}_1$ | $\hat{p}_2$ | $\Delta$ | $\phi$ (95% CI)          | $\phi/\phi_{\max}$ | $q$     |
|------------------------------------------|------------------------|-------------|-------------|----------|--------------------------|--------------------|---------|
| <i>Same row (H1)</i>                     |                        |             |             |          |                          |                    |         |
| probe <sub>16</sub> × probe <sub>8</sub> | internal repr.         | 0.68        | 0.60        | 0.172    | <b>0.75</b> [0.62, 0.88] | 0.89               | < 0.001 |
| perplexity × token-anomaly               | token surface          | 0.66        | 0.35        | 0.079    | <b>0.35</b> [0.18, 0.50] | 0.66               | < 0.001 |
| <i>Cross row (H2)</i>                    |                        |             |             |          |                          |                    |         |
| perplexity × probe <sub>8</sub>          | surface × internal     | 0.66        | 0.60        | 0.144    | 0.62 [0.45, 0.77]        | 0.71               | < 0.001 |
| perplexity × probe <sub>16</sub>         | surface × internal     | 0.66        | 0.68        | 0.131    | 0.59 [0.42, 0.76]        | 0.62               | < 0.001 |
| perplexity × refusal-prime               | surface × first-token  | 0.66        | 0.60        | 0.134    | 0.58 [0.41, 0.73]        | 0.66               | < 0.001 |
| perplexity × SmoothLLM                   | surface × perturb.     | 0.66        | 0.54        | 0.134    | 0.57 [0.40, 0.72]        | 0.73               | < 0.001 |
| refusal-prime × SmoothLLM                | first-token × perturb. | 0.60        | 0.54        | 0.136    | 0.56 [0.39, 0.71]        | 0.63               | < 0.001 |
| probe <sub>16</sub> × SmoothLLM          | internal × perturb.    | 0.68        | 0.54        | 0.113    | 0.49 [0.31, 0.65]        | 0.66               | < 0.001 |
| probe <sub>8</sub> × SmoothLLM           | internal × perturb.    | 0.60        | 0.54        | 0.116    | 0.48 [0.30, 0.64]        | 0.54               | < 0.001 |
| probe <sub>8</sub> × refusal-prime       | internal × first-token | 0.60        | 0.60        | 0.110    | 0.46 [0.28, 0.63]        | 0.46               | < 0.001 |
| probe <sub>16</sub> × refusal-prime      | internal × first-token | 0.68        | 0.60        | 0.102    | 0.45 [0.26, 0.62]        | 0.54               | < 0.001 |
| probe <sub>8</sub> × token-anomaly       | internal × surface     | 0.60        | 0.35        | 0.090    | 0.39 [0.22, 0.54]        | 0.65               | < 0.001 |
| SmoothLLM × token-anomaly                | perturb. × surface     | 0.54        | 0.35        | 0.081    | 0.34 [0.16, 0.52]        | 0.50               | 0.001   |
| probe <sub>16</sub> × token-anomaly      | internal × surface     | 0.68        | 0.35        | 0.072    | 0.32 [0.16, 0.48]        | 0.64               | < 0.001 |
| refusal-prime × token-anomaly            | first-token × surface  | 0.60        | 0.35        | 0.070    | 0.30 [0.12, 0.47]        | 0.50               | 0.004   |

## 11.2. Results: the three hypotheses

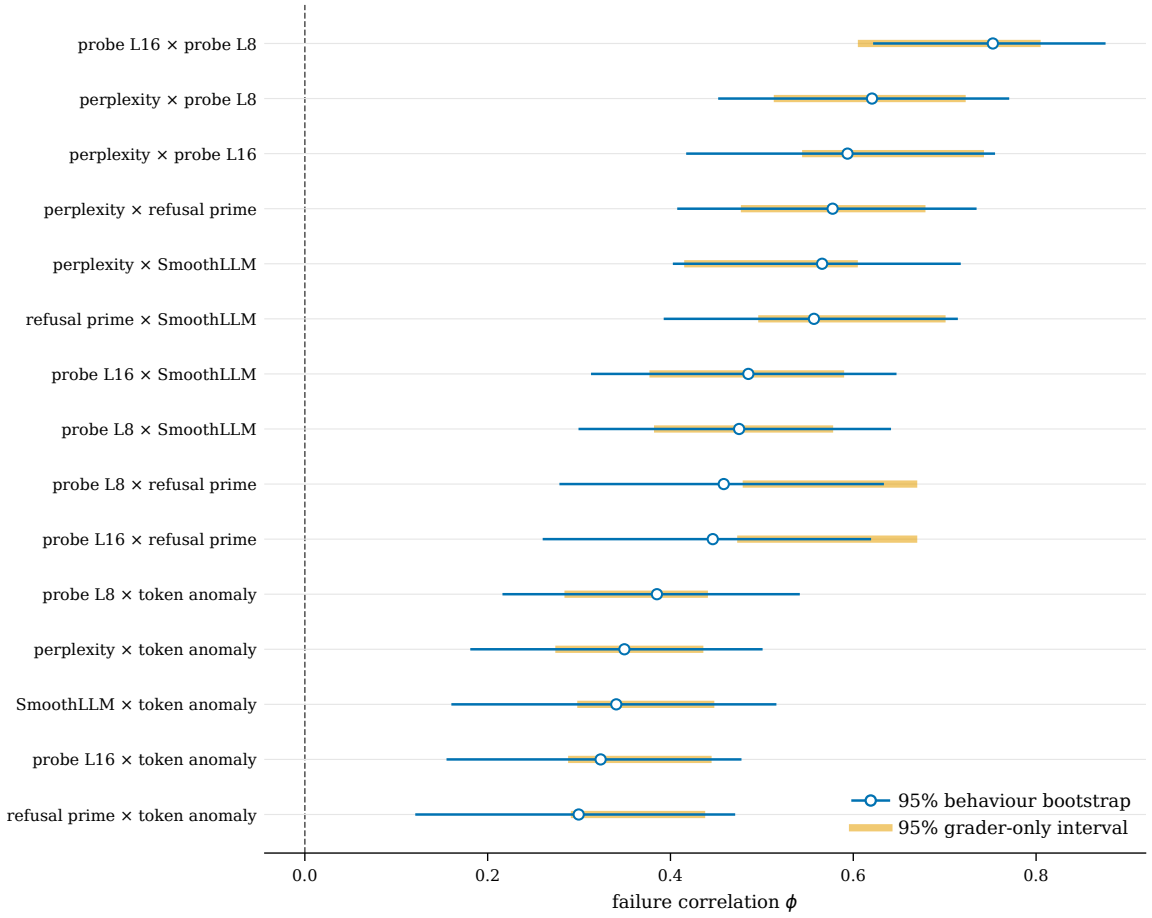
Table 10 reports the fifteen measurable pairs. One row reads as follows: two defenses are breached on  $\hat{p}_1$  and  $\hat{p}_2$  of the behaviors individually; if they failed independently they would both be breached on  $\hat{p}_1\hat{p}_2$ ; they are in fact both breached on  $\hat{p}_1\hat{p}_2 + \Delta$ ; and  $\phi$  expresses that excess as a correlation, with  $\phi/\phi_{\max}$  correcting for the ceiling that unequal marginals impose. A row with  $\Delta > 0$  is a pair that delivers less than the multiplicative model promises, and every row does. We take the three pre-specified hypotheses in turn here; Section 11.3 then places the pairwise measurement against the assembled stack, and Section 11.4 contrasts adaptive with static attackers, which determines how much of any of it transfers.

*H1: same-row defenses fail together. Supported as correlation, not adjudicable as mechanism.* Both within-row pairs show strong positive correlation. Two probes reading the same residual stream at different depths reach  $\phi = 0.75$  ([0.62, 0.88]), with a joint residual of 0.58 against an independence prediction of 0.41. The primary token-surface pair, which could not be evaluated under GCG because the perplexity filter admitted nothing, is measurable under the fluent adversary and gives  $\phi = 0.35$  ([0.18, 0.50],  $q < 0.001$ ). So the correlation half of H1 holds on both pairs.

The causal half does not follow, and we do not claim it. Under the labels reported here the probe pair is the one association that survives the stringent difficulty-stratified test of Section 11.6 (common odds ratio 159,  $q < 0.001$ ) while the primary token-surface pair does not (common odds ratio 1.39,  $p = 0.907$ ), which would read as mechanism-specific failure on the first and a shared difficulty gradient on the second. That reading is not stable. The surviving pair is the exploratory one added by amendment (Section 11.11); under majority-of-five grader labels its common odds ratio falls to 11.8 and two cross-row pairs begin surviving as well; and under externally calibrated thresholds its stratified test becomes undefined while the sole survivor is a cross-row pair. Section 11.7 reports all three regimes. We therefore retire the mechanism-specific reading of H1 rather than report the regime that favors it, and record that on this data the question is not identifiable. Nothing in the composition argument depends on the answer, because Eq. (6) is agnostic about why  $\rho$  is nonzero.

Figure 5 places these two results in the context of the whole grid. It draws two intervals per pair: the thin one is the 10,000-resample bootstrap over behaviors used throughout this section, and the thick band behind it is the grader-only interval of Section 11.7. No interval of either kind covers the independence value  $\phi = 0$ . Ordering the fifteen pairs by effect size shows that the probe pair stands clearly apart from the rest, at the bottom of the figure, while the token-surface pair, though within-row, sits among the cross-row pairs rather than above them. That last reading is an

## Layered LLM Defenses as an Ensemble



**Figure 5:** Failure correlation  $\phi$  for all fifteen measurable pairs, ordered by effect size, with the behavior bootstrap and the grader-only interval shown separately.

artifact of the scale, and the correction runs in H1's favor. The token-surface pair has the most unequal marginals in the grid and therefore the second-lowest attainable  $\phi_{\max}$  (Eq. (8)), so on the normalized column of Table 10 it moves from twelfth to fourth and both same-row pairs sit in the upper third. Same-row membership raises correlation reliably once the bound is accounted for, though the stratified analysis of Section 11.6 shows that raising correlation and sharing a mechanism are not the same thing. No interval in the figure covers the independence value  $\phi = 0$ .

**H2: cross-row defenses are not independent. Rejected.** Every one of the thirteen non-degenerate cross-row pairs shows a positive failure correlation, all significant after multiplicity correction, with  $\phi$  ranging from 0.30 to 0.62. Defenses that share no mechanism, namely a surface statistic, a mid-network linear probe, a first-token prior, and a randomized-smoothing wrapper, nonetheless fail on overlapping inputs. The effect replicates across targets and attack classes: the probe  $\times$  refusal-prime correlation is 0.21 on Llama-3.2-3B, 0.26 on Llama-2-7B and 0.25 on Vicuna under GCG, and 0.45 under the fluent attack.

The practical consequence is immediate: the multiplicative model systematically overstates a stack's protection. For perplexity  $\times$  probe<sub>16</sub> the independence prediction is 0.449 while the measured joint residual is 0.580, and the pattern holds for every pair we measured, with  $\Delta > 0$  throughout and  $\Delta$  up to 0.172. Figure 4 shows the shortfall directly. Its diagonal is the multiplicative prediction of Eq. (6) at  $\rho = 0$ , and the vertical distance from that diagonal is  $\Delta$ . Plotting each pair's observed joint breach rate against its independence prediction puts every one of the fifteen points above the diagonal, and the vertical gap is the protection the multiplicative model assigns to a stack but the stack does not deliver. The gap widens with the prediction itself, so the error is largest exactly where a defender would expect a stack

to be doing the most work. **Layered defenses compound only under independence, and independence does not hold in practice.**

One caveat belongs here rather than only in the limitations, because it bears on how individual  $\Delta$  values should be read. Breach is scored by a non-deterministic rubric grader, and re-judging an identical response set moved the undefended ASR by 0.06 (Section 11.11). Several of the smaller  $\Delta$  values in Table 10 are of that order, and the bootstrap intervals resample behaviors rather than grader verdicts, so no interval in the table covers grader variation. Section 11.7 measures that variation directly by judging every graded response five times, and the caution is confirmed rather than dissolved: all fifteen pairs stay positive and significant under majority labels, so the direction of the result holds, but among pairs of pairs whose  $\Delta$  differs by less than 0.05, 91% have overlapping grader intervals. Individual  $\Delta$  values separated by less than about 0.05 should therefore not be ordered against one another.

Section 11.6 shows that most of this cross-row association is attributable to a common cause, variation in how jailbreakable individual behaviors are, rather than to mechanism overlap. That distinction matters for interpreting Table 10, but it does not soften the composition result: whether layers fail together because they share a blind spot or because they face a common difficulty gradient, the joint residual exceeds  $\hat{p}_1\hat{p}_2$  and the stack under-delivers.

*H3: the refusal budget binds. Supported.* The quantity  $f_i$  here is the *system-level* false-refusal rate: the fraction of benign prompts on which the deployed configuration returns a refusal, whether because the filter blocked the prompt or because the base model declined it unprompted. It is therefore larger than the 1% filter block rate the thresholds were calibrated to. Measured directly, the undefended model refuses 8 of the 100 benign prompts, and every defended configuration refuses a superset of exactly those eight. Measured  $f_i$  are refusal-prime 0.39, Llama Guard 0.26, SmoothLLM 0.26, probe<sub>g</sub> 0.09, token-anomaly 0.09, perplexity 0.08 and probe<sub>16</sub> 0.08.

Reporting those rates raw overstates what the defenses cost. Netting out the floor, the rate at which a configuration refuses a prompt the bare model did *not* refuse, taken over the 92 benign prompts the bare model admits rather than over all 100, gives an attributable burden of 0.337 for refusal-prime, 0.196 for Llama Guard and SmoothLLM, 0.011 for probe<sub>g</sub> and token-anomaly, and **0.000** for perplexity and probe<sub>16</sub>: those two refuse on precisely the floor and nothing else, so their entire apparent false-refusal cost belongs to the base model. The assembled stack refuses 0.810 of benign prompts, against an independence prediction of  $1 - \prod_i (1 - f_i) = 0.766$  and a union bound of  $\sum_i f_i = 1.25$  by Eq. (7). The observed rate lies between the two and comfortably inside the bound.

Neither form of Eq. (7) accounts for this figure, and Section 11.10 shows why. Because every  $f_i$  contains the shared base-model floor, the layers' refusals are positively correlated by construction, and the union of their independently measured refusals is accordingly *below* the independence prediction, at 0.520. The deployed pipeline exceeds both, because its members run in sequence and each alters the input the next one sees, so refusals compound rather than merely union. The union bound of Eq. (7) still holds, but no form of that equation predicts a deployed stack's refusal rate from per-layer rates, which is a stronger statement than the one this paragraph originally made and is developed in Section 11.10. Refusal budget is therefore exhausted long before inference cost becomes the binding constraint: a seven-layer stack that admits at most 1% of adaptive attacks also refuses four in five benign requests, which is not a deployable operating point. This is the strongest available evidence for composition rule 5, and it is what makes rule 2 a practical necessity rather than an efficiency preference: every layer that fails to add independence still adds refusals.

### 11.3. Pairwise estimates versus the assembled stack

*Intersection versus a direct attack.* Failure correlation is estimated by intersecting per-defense breach vectors, but the quantity a deployer cares about is the residual of the assembled stack under an adversary that attacks *it*. We measure both. Intersecting the seven per-defense vectors predicts a joint residual of 0.010; running the same fluent adversary directly against the assembled stack yields 0.000, and the two agree on 99 of 100 behaviors. The single behavior that survives the intersection is the same one Llama Guard fails on, and the direct attack breaches none. That agreement establishes that the pairwise analysis is not an artifact of never having attacked the real system. It is weak evidence for anything more, because both quantities lie on the floor, where any method returning near-zero would agree.

To give the comparison resolution we repeated it with Llama Guard removed, since its residual of 0.01 is what pins the intersection to the floor. Over the remaining six defenses the intersection predicts a residual of 0.190. A direct fluent attack on that assembled six-layer stack yields 0.000, disagreeing with the intersection by the full 0.190. The mechanism is joint infeasibility rather than the stack defeating successful attacks: every attempt was rejected at the input stage, and the search failed to find a prompt satisfying all six constraints simultaneously on 100 of 100 behaviors.

Optimizing against each layer separately and intersecting implicitly assumes that a prompt defeating one layer survives the others when they are present, and the joint feasible region is far smaller than that assumption allows.

**The intersection estimate is therefore not a faithful proxy for an assembled stack, and its error is conservative for the deployer.** On the one configuration where the check can discriminate, the error is as large as the quantity being estimated. We accordingly report the pairwise analysis as a measurement of correlation between defenses, which is what it is, rather than as a prediction of assembled-stack residual. The claim this licenses is bounded by the attack budget: under a search sufficient to produce a 0.190 per-layer intersection residual, a direct attack finds no feasible prompt on any behavior. That is a statement about that search, not a proof that the feasible region is empty.

Two comparisons against single layers are worth stating together, because they say different things. Against the mid-layer probe, which admits most of the attack ( $\hat{p} = 0.68$ ), the stack is decisive: 68 behaviors breached by the probe and held by the stack, none in the reverse direction (McNemar,  $p \approx 7 \times 10^{-21}$ ). Against Llama Guard, the strongest layer in the stack at  $\hat{p} = 0.01$ , it is not: the two differ on a single behavior ( $n_{01} = 1$ ,  $n_{10} = 0$ , exact binomial  $p = 1.0$ ), which no test can distinguish from chance. That comparison is the informative one, and Section 11.7 extends it to the full pairwise matrix, where Llama Guard dominates every other member outright: there is no behavior on which it is breached while another defense holds. **The assembled stack is, on this target and this adversary, statistically indistinguishable from its best single layer.** The comparison is bounded below as well as above, and we state the limit rather than lean on it: with Llama Guard admitting one behavior in 100 there is at most one behavior of headroom, so no experiment at this sample size could have shown the stack improving on it, and the result is an absence of measurable improvement rather than evidence that none exists. What the sample does support is the asymmetry between the two axes. The six additional layers bought no measurable residual protection over Llama Guard alone, while raising the benign refusal rate from 0.26 to 0.810 and adding the cost of two class-B passes, a class-C wrapper and two class-E probes by Eq. (5). On this configuration, depth was paid for and not delivered.

That verdict does not generalize, and the six-layer measurement above is the reason. Remove Llama Guard and the best remaining member is the token-anomaly filter at  $\hat{p} = 0.35$ , while the assembled six-layer stack admits nothing at all. Depth delivered there, and by a wide margin, at a refusal cost the selection analysis of Section 11.10 prices at 0.45 for the four rule-compliant layers. The two configurations therefore say different things: where one member already dominates, additional layers buy refusals and no measurable security; where no member dominates, the joint constraint is far stronger than any member and far stronger than the correlation analysis predicts. What both share is that the assembled residual could not have been derived from the per-layer numbers. Composition has to be measured end to end, on the security axis as well as the refusal axis, which is the practical content of composition rules 5 and 6.

*What  $\phi$  is for, given that it does not predict the stack.* Taken together, the two comparisons above raise an obvious objection to the whole enterprise: if the pairwise measurement mispredicts the assembled stack by the full size of the estimate on the security axis, and by 0.29 in the other direction on the refusal axis, why measure  $\phi$  at all? We state the answer plainly rather than leave it to the reader, because it narrows the paper's claim.  $\phi$  is a *selection* statistic, not a *prediction* statistic. Its job is to tell a designer which candidate members are near-substitutes, so that a stack budget is not spent twice on the same blind spot, and Section 11.10 shows it does that job: under the reported labels, selecting members by minimizing double-fault, which is  $\phi$  restated, recovers the same stack a greedy search with full access to the joint breach data would build, and the same stack the dependency taxonomy nominates. Its job is not to forecast what the assembled pipeline will deliver, and Eq. (6) should not be used that way, for the reason given when it was introduced: it is an identity for parallel evaluation and a deployed stack runs in sequence. The multiplicative model this paper displaces made the stronger claim, that per-layer numbers predict stack behavior. We are showing that the weaker claim is the only one the data support, and that the correct workflow is therefore two-stage: select on measured diversity, then measure the assembled configuration before deploying it.

#### 11.4. Adaptive versus static attackers

*Two static baselines.* “Static” is ambiguous in this literature and the ambiguity is not harmless here, so we run two static baselines against every defense: the unmodified behavior prompt, and a single precomputed GCG suffix reused unchanged for all behaviors. Both are gold-judged on the same  $n = 100$  behaviors. Neither is the same thing as the *adaptive* GCG adversary of Figure 3, which re-optimizes a suffix against each defense and each behavior, and which is correspondingly stronger. Setting the adaptive attack against both baselines, as in Table 11, shows that the size of the adaptive gap depends on which static attack is meant.

**Table 11**

Residual attack success under one adaptive and two static adversaries, for the five defenses measurable against all three. The adaptive column is the fluent sentence-level search; the two static columns are a single precomputed GCG suffix reused unchanged, and the unmodified behavior prompt. All gold-judged on the same  $n = 100$  behaviors.

| Defense             | adaptive (fluent) | static: transfer suffix | static: plain prompt |
|---------------------|-------------------|-------------------------|----------------------|
| perplexity          | 0.66              | 0.00                    | 0.07                 |
| probe <sub>16</sub> | 0.68              | 0.28                    | 0.08                 |
| refusal-prime       | 0.60              | 0.11                    | 0.03                 |
| SmoothLLM           | 0.54              | 0.04                    | 0.06                 |
| token-anomaly       | 0.35              | 0.28                    | 0.07                 |

Against the transfer suffix the adaptive gap is 0.66 for the perplexity filter, 0.50 for SmoothLLM, 0.49 for refusal-prime and 0.40 for the mid-layer probe. Which of the two static baselines is the harder one is not constant across defenses, and the exception is instructive. For every defense that reads the residual stream the transfer suffix is the stronger static attack, as one would expect. For the perplexity filter the ordering inverts, at 0.07 for the plain prompt against 0.00 for the suffix, for the reason given in Section 11.1: the transfer suffix is high-perplexity and the filter blocks it outright, whereas an unmodified prompt passes. That is the attack-class effect appearing inside the static baseline itself, and it is a warning about the quantity as much as about the defense. An adaptive-versus-static gap is interpretable only once the static attack is named, since for this one defense the two baselines put the headline gap at 0.66 and 0.59 respectively.

On the strongly aligned Llama-2-7b-chat target the undefended gold ASR is 0.37 under GCG ( $n = 100$ ) and 0.40 under the fluent attack ( $n = 50$ ), against 0.02 for the plain prompt in both cases, a factor of twenty. Read against the weaker of the two baselines on Vicuna the same comparison gives roughly a factor of ten. On either target and either baseline a static evaluation understates residual risk by about an order of magnitude, which is the concern of Section 8.1 measured rather than argued.

*Where the optimizer underperforms a fixed suffix.* In the *suffix*-attack runs the optimized attack is occasionally less successful than a static baseline. On Vicuna the perplexity filter records 0.00 adaptive against 0.07 for the plain prompt, and SmoothLLM 0.02 adaptive against 0.04 for the transfer suffix and 0.06 for the plain prompt; on Llama-2 the same inversion appears for four defenses. We checked that this is not a labeling error: adaptive rows carry per-behavior optimized suffixes, plain rows are byte-identical to the behavior text, and transfer rows carry the shared fixed suffix. The mechanism is the one in Section 11.1. Every inverted case is a defense that neutralizes the high-perplexity suffix class, so the constrained optimizer is driven into a region where its output is blocked or destabilized while an unmodified prompt passes unflagged; for the perplexity filter this is literal, since no feasible suffix was located at all. Taken individually each inversion is two to six breaches at  $n = 100$  and well within sampling noise; taken together the direction is systematic.

Two consequences follow. First, the adaptive column reports one optimizer's output rather than a best-of-strategies adversary, so a deployer should read  $\max(\text{adaptive}, \text{static})$ , which in the suffix runs raises the perplexity filter to 0.07 and SmoothLLM to 0.06. This is a caveat about the threat model rather than about these defenses: an adversary who simply tries both is stronger than either column alone, and nothing in Section 8.1 entitles a defender to assume otherwise. Second, and more important for everything above, **no inversion occurs in either fluent run**. The primary Vicuna results and the Llama-2 replication are free of it, so Table 10, H1 to H3, the stratified analysis of Section 11.6 and the intersection comparison are unaffected.

### 11.5. Complementary coverage: what survives a fluent attack

Llama Guard 3 is the one defense that holds under both adversaries, with residual ASR 0.00 under GCG and 0.01 under AutoDAN, at the cost of one of the highest false-refusal rates in the stack (0.26). The perplexity filter shows the opposite profile: near-perfect against high-perplexity suffixes and largely ineffective (0.66 residual) against fluent prompts. The two are complementary in *attack-class coverage* rather than in per-input failure, which is the practical reading of the correlations above: a useful stack pairs layers whose blind spots lie in different attack classes, not merely in different dependency rows. This is the second half of composition rule 2, a refinement the derivation of Section 10

could not have produced on its own, and the main way the measurement changes the framework rather than merely confirming it.

### 11.6. Is the correlation just behavior difficulty?

A positive  $\phi$  between two defenses does not by itself establish a shared blind spot. Behaviors differ in how readily they are jailbroken, and if every defense fails on the same easy behaviors the pairwise correlations follow from that common cause alone. Because this is the most consequential alternative explanation of the results above, we test it explicitly rather than leaving it to interpretation.

For each pair ( $d_1, d_2$ ) we stratify the behaviors by a difficulty score computed from the *other* defenses only, namely the number of the remaining live defenses that were breached on that behavior, and compute the Cochran–Mantel–Haenszel (CMH) common odds ratio [130], which measures the  $d_1$ – $d_2$  association *within* strata of equal difficulty. Deriving the stratifier from the other defenses keeps it independent of the pair under test. Odds ratios carry the Haldane–Anscombe +0.5 continuity correction, applied uniformly so that crude and stratified values stay comparable, and  $q$  is Benjamini–Hochberg adjusted over the fifteen pairs, the same correction applied to  $\phi$  in Table 10. Table 12 reports the two same-row pairs and a representative selection of cross-row pairs.

**Table 12**

Difficulty-stratified association for the two same-row pairs and a representative selection of cross-row pairs; all fifteen are in the supplementary material.

| Pair                                     | Rows  | $\phi$ | crude OR | CMH OR       | $p$     | $q$     |
|------------------------------------------|-------|--------|----------|--------------|---------|---------|
| probe <sub>16</sub> × probe <sub>8</sub> | same  | 0.75   | 68.0     | <b>158.7</b> | < 0.001 | < 0.001 |
| perplexity × token-anomaly               | same  | 0.35   | 6.0      | 1.39         | 0.907   | 0.943   |
| refusal-prime × SmoothLLM                | cross | 0.56   | 12.3     | 4.34         | 0.023   | 0.173   |
| perplexity × refusal-prime               | cross | 0.58   | 14.5     | 3.95         | 0.063   | 0.314   |
| perplexity × probe <sub>8</sub>          | cross | 0.62   | 19.1     | 3.07         | 0.209   | 0.562   |
| probe <sub>16</sub> × refusal-prime      | cross | 0.45   | 7.3      | 0.72         | 0.943   | 0.943   |
| probe <sub>8</sub> × SmoothLLM           | cross | 0.48   | 7.8      | 0.91         | 0.835   | 0.943   |

The pattern sharpens the reading of Table 10 and it also qualifies it. Of fifteen pairs, only two retain a positive within-stratum association at the nominal level, and after Benjamini–Hochberg correction only one does: the same-row probe pair, whose common odds ratio of 159 is more than twice its crude value, so holding difficulty fixed *strengthens* rather than explains away its association. Three cross-row pairs have CMH odds ratios at or below unity, two of which are shown, and the remainder are underpowered once the sample is divided into strata of roughly fifteen behaviors each.

The pair that does not survive is the one the design nominated. Perplexity × token-anomaly is the primary H1 pair, and its within-stratum association does not reach the nominal level, so the same-row claim rests on the probe pair alone. We report this rather than the aggregate because the two same-row pairs are not equivalent evidence: the probes share an input representation and a training pipeline, which is why Section 11.11 records the pair as bounding the same-row effect from the easy side. Under the other two label regimes the picture differs again, which is the subject of Section 11.7.

**Interpretation.** Before drawing anything from the pattern, note that Section 11.7 shows which pair survives this test is not stable across grader draws or calibration corpora, so the reading below describes the reported labels and is retired as a standing claim there. The two same-row pairs behave differently, and the difference is instructive. The probe pair retains a large within-stratum association; the pre-registered token-surface pair does not, its crude odds ratio of 6.0 falling to 1.39 within strata ( $p = 0.907$ ), which is what a pure difficulty effect looks like. Same-row membership is therefore not sufficient on its own for correlated failure that survives conditioning. What the surviving pair has in addition is a shared *input representation*, two probes reading the same residual stream at different depths, which is a stronger relation than sharing a dependency row and would suggest the taxonomy is coarser than the phenomenon. We record that as the natural reading of these labels and not as a finding, because it does not hold across regimes: under majority grader labels three pairs survive and under external thresholds the survivor is cross-row (Section 11.7). What does hold across every regime is the negative half. Cross-row defenses fail together predominantly because they face a common difficulty gradient rather than because they share a mechanism, and that common-cause dependence violates

the independence assumption multiplicative composition requires exactly as mechanism overlap would. The practical conclusion of Section 11.2 is therefore unchanged, and composition rule 2 survives in its revised form: a same-row pair is a near-substitute in practice, and a cross-row pair still fails to deliver the multiplicative residual, whatever the reason turns out to be.

**Two caveats, stated plainly.** The test is deliberately conservative: conditioning on the outcomes of other defenses removes shared variance that is arguably part of the effect of interest, so it under-credits genuine mechanism overlap. And with  $n = 100$  split across about six strata it is underpowered, since ten of the fifteen pairs return wide intervals, so “not significant within strata” should be read as insufficient evidence rather than as evidence of no effect. A larger behavior set, or pooling across targets, would sharpen this analysis, and we flag it as the most valuable extension of this experiment.

### 11.7. Robustness: inference, grader, and calibration

Three properties of the analysis above could in principle be doing the work rather than the data: the bootstrap’s distributional assumptions, the single grader judgment behind every breach label, and thresholds calibrated on the same benign prompts used to report false refusals. We tested all three. The composition result survives all of them; two narrower claims do not, and we say which.

*Inference is not doing the work.* Permuting one member of each pair 10,000 times, which holds both marginals fixed by construction, makes all fifteen associations significant, and the permutation verdict agrees with the bootstrap interval on significance for 15/15 pairs. Cochran’s  $Q$  across the seven defenses is 193.4 ( $df = 6$ ,  $p = 4.8 \times 10^{-39}$ ), so they are not interchangeable, and exact-binomial McNemar separates 11 of the 21 pairs after Holm correction. The pairs that do *not* separate are the mid-strength layers among themselves, which is what one expects if they are close in strength.

Two of those comparisons are load-bearing elsewhere in the paper. First, Llama Guard dominates every other member pairwise: there is no behavior on which Llama Guard is breached while another defense holds. Against the assembled seven-layer stack it is indistinguishable ( $n_{01} = 1$ ,  $n_{10} = 0$ , exact  $p = 1.0$ ), which is the proper form of the comparison reported in Section 11.3. Second, probe<sub>16</sub> and probe<sub>8</sub> do not differ in marginal ASR after correction ( $p = 0.039$  uncorrected, 0.309 after Holm) even though they carry the highest failure correlation in the study. Indistinguishable marginals and correlated failures are different claims, and the first does not follow from the second.

*The grader is not deterministic, and its variance is not small.* We judged all 1,887 graded responses five times under an unchanged rubric, model and temperature. 90.5% of responses are unanimous, 9.5% split, and 4.2% rest on a bare 3–2 majority. Rebuilding every breach vector from the majority label leaves all fifteen pairs positive and significant ( $q \leq 1.5 \times 10^{-3}$ ), so the correlation result is not an artifact of one judgment draw.

The individual values move, however, and predominantly upward: for thirteen of the fifteen pairs the published value is the smaller of the two, by as much as +0.16 under majority labels. The direction is not uniform, and we state the exceptions rather than round them off. Two pairs fall, perplexity  $\times$  SmoothLLM from 0.566 to 0.515 and token-anomaly  $\times$  probe<sub>8</sub> from 0.385 to 0.369, and probe<sub>16</sub>’s marginal ASR drops from 0.68 to 0.64, so for those three the value tabulated in Table 10 overstates rather than understates.

We nonetheless retain the single-judgment values throughout, for a reason stronger than aggregate conservatism. Those breach vectors are not confined to Table 10: the stratified analysis of Section 11.6, the diversity recast of Section 11.9, the McNemar and intersection comparisons of Section 11.3, and every figure in this section derive from the same vectors. Substituting majority labels in one place would desynchronize it from the rest, and substituting them everywhere would move the stratified result from one surviving pair to three, which is a different analysis rather than a re-estimate of this one. We therefore report one labeling regime consistently and carry the comparison here. Resampling one judgment per response gives a grader-only interval averaging 0.185 wide against 0.312 for the behavior bootstrap, about 60% of it. Figure 5 shows both, reported side by side rather than combined, since one covers a different sample of behaviors and the other a different draw from the grader, and combining them would assume an independence nothing here establishes. The practical consequence is the one anticipated in Section 11.2:  $\Delta$  values within roughly 0.05 of each other cannot be ordered, and among pairs of pairs separated by less than that, 91% have overlapping grader intervals.

Two marginals also swap rank under majority labels,  $\text{probe}_8$  rising above  $\text{probe}_{16}$ , which is enough to break the agreement between the row heuristic and the diversity-greedy search reported in Section 11.10 from stack size three onward. We mark that agreement as labeling-dependent there.

*Threshold transfer.* Every threshold in the primary run is set on the same 100 benign prompts later used to report false-refusal rates, where a 1% target is a single prompt. We re-ran the entire primary configuration with thresholds set instead on 805 external instructions, disjoint from both the evaluation set and the probe’s training pool.

| Defense             | thr. in-sample | thr. external | FRR in $\rightarrow$ ext                  | ASR in $\rightarrow$ ext                  |
|---------------------|----------------|---------------|-------------------------------------------|-------------------------------------------|
| perplexity          | 4.849          | 6.516         | 0.08 $\rightarrow$ 0.08                   | 0.66 $\rightarrow$ 0.67                   |
| token-anomaly       | 0.273          | 0.523         | 0.09 $\rightarrow$ 0.08                   | <b>0.35 <math>\rightarrow</math> 0.64</b> |
| $\text{probe}_{16}$ | 28.924         | -5.210        | <b>0.08 <math>\rightarrow</math> 0.26</b> | 0.68 $\rightarrow$ 0.70                   |
| $\text{probe}_8$    | 110.768        | 100.593       | 0.09 $\rightarrow$ 0.15                   | 0.60 $\rightarrow$ 0.66                   |

Two things follow. The probe’s 1% operating point does not transfer: calibrated out of sample it refuses 26% of the held-out benign set, because the evaluation benign prompts are matched counterparts to the harmful behaviors and sit far closer to the probe’s decision boundary than ordinary instructions do. And the token-anomaly filter’s in-sample threshold was carrying most of its measured effectiveness, not merely flattering its refusal rate: relaxed to an out-of-sample operating point its residual ASR nearly doubles, from 0.35 to 0.64.

All fifteen pairs remain positive and significant under external thresholds ( $q \leq 5 \times 10^{-4}$ ), several more strongly than in-sample. Figure 6 shows  $\phi$  for every pair under all three regimes: the single judgment with in-sample thresholds reported in Table 10, the majority of five grader judgments, and the external thresholds above. Every pair is positive in every regime, the minimum across the whole grid being 0.300, while individual values move by as much as 0.4. That spread is why we report the direction of these associations rather than their precise magnitudes.

*What does not survive.* The mechanism-specific reading of H1 does not, and we retire it rather than report the regime that favors it. Under majority labels the probe pair still clears the stratified test but its common odds ratio falls from 159 to 11.8, and two *cross-row* pairs begin clearing it as well. Under external thresholds the pair’s stratified test becomes undefined rather than negative: every difficulty stratum is saturated, since in the easy strata neither probe is breached and in the hard strata both are, so the common odds ratio has an empty concordant cell in every stratum and is identically zero rather than small. It is the same degeneracy that makes the semantic-classification row untestable (Section 11.5), and among the fifteen pairs it strikes only this one. The single pair that does clear the stratified test under external thresholds is perplexity  $\times$   $\text{probe}_8$ , which is cross-row.

So the taxonomy of Table 3 remains a useful organizing device for choosing which defenses to combine, and the composition result holds under every perturbation we applied, but we do not claim that row membership predicts mechanism-specific correlated failure. On this data that question cannot be settled.

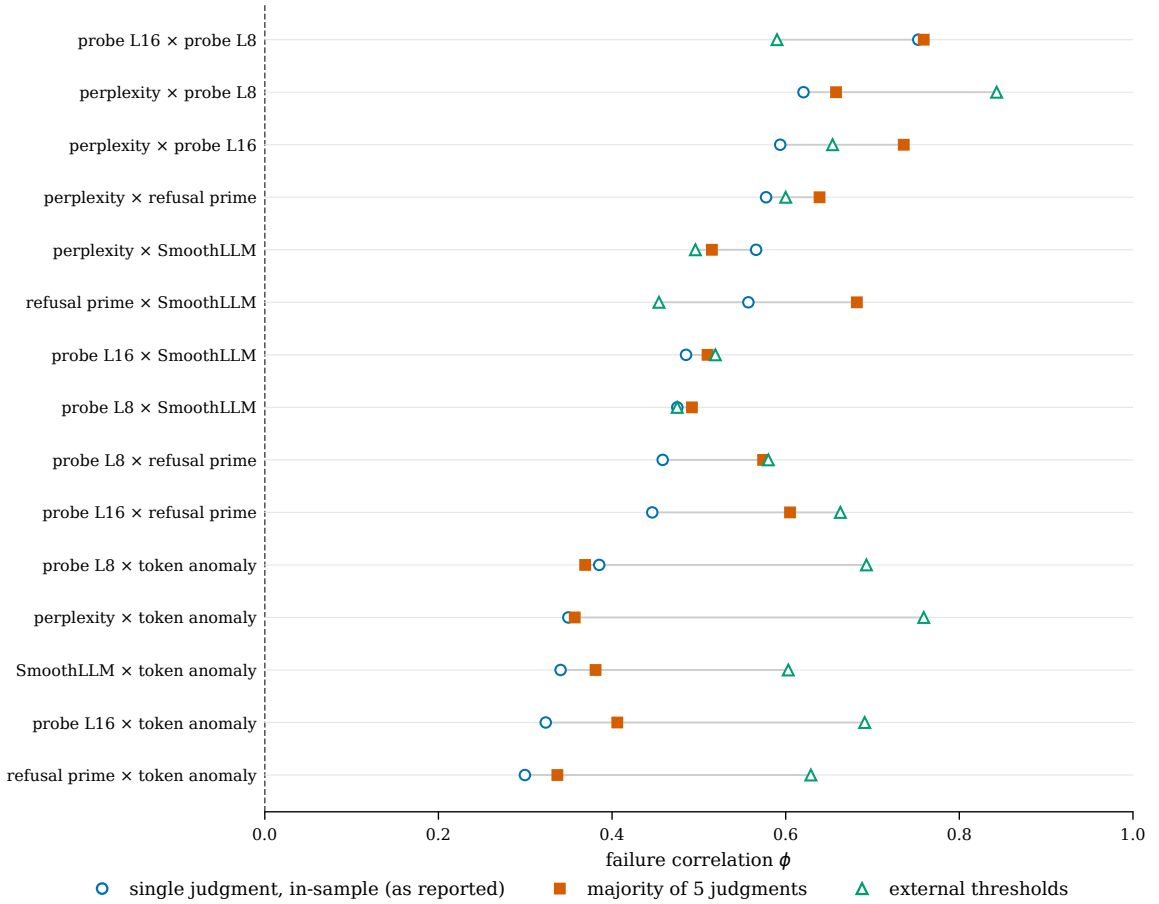
## 11.8. Replication on a strongly aligned target

Vicuna-7B [129] is weakly aligned by construction, so we repeat the experiment on Llama-2-7b-chat [131] with the *same* fluent adversary and the *same* attack budget, varying only the target. The replication evaluates the four defenses the headline pairs require on the first 50 behaviors.

Two findings replicate cleanly and one does not.

*The attack-class result replicates.* On Llama-2 the perplexity filter admits the fluent attack exactly as it did on Vicuna: its residual ASR is 0.44, against 0.00 for the same defense under GCG on the same target (Section 11.1). The adaptive-versus-static gap replicates at the same magnitude against the transfer-suffix baseline (0.44 adaptive versus 0.00 for the perplexity filter; 0.42 for the early-layer probe), with the same caveat about which static attack is meant. The undefended residual is 0.40, so the fluent adversary is effective against a refusal-robust target, not only against a weakly aligned one. Note that the filter’s defended residual of 0.44 sits slightly *above* the undefended 0.40: at  $n = 50$  that is 22 breaches against 20, a difference of two behaviors and well inside sampling noise, and it should be read as the filter contributing nothing measurable against this attack class rather than as the filter assisting the adversary. It is the same inversion, and the same mechanism, as the one analyzed for the suffix runs in Section 11.4. This is the finding we regard as established across targets.

### Layered LLM Defenses as an Ensemble



**Figure 6:** Failure correlation for every measurable pair under three labeling regimes: as reported, majority of five grader judgments, and external thresholds.

*The correlation structure is directionally consistent but underpowered.* Every pair that is measurable on both targets keeps its sign, and the two probes remain positively associated ( $\phi = 0.25$ , 95% CI  $[-0.03, 0.52]$ , versus 0.75 on Vicuna), but at  $n = 50$  with lower marginals the intervals are wide and only perplexity × probe<sub>8</sub> reaches nominal significance ( $\phi = 0.35$ ,  $[0.08, 0.61]$ ,  $p = 0.015$ ,  $q = 0.089$ ). We therefore report the replication as *consistent in direction and not independently confirmatory*: it does not contradict the Vicuna result, and it does not on its own establish it. A full-size replication is the natural next step.

*The primary same-row pair remains untestable.* On Llama-2 the token-anomaly filter admits almost nothing under the fluent attack (residual ASR 0.02, one breach in fifty), so perplexity × token-anomaly is again degenerate, this time because the *second* member of the pair is near-constant rather than the first. The primary same-row test is thus measurable on the weakly aligned target only. That a row becomes untestable exactly when one of its members is highly effective is a structural limitation of correlation-based composition analysis, not a property of these two filters (Section 11.11).

### 11.9. The same result in the vocabulary of classifier fusion

Stacking defenses is an ensemble construction, and the ensemble literature has measured error dependence for two decades. Restating our result in that vocabulary does two things: it checks that the effect is not an artifact of the particular statistic we chose, and it lets us ask the design question  $\phi$  alone cannot answer, namely which combination rule and which membership one should actually deploy.

Following Kuncheva and Whitaker [19] we compute diversity over *correctness*: a defense is correct on a behavior when it blocks the attack, so  $c_d(i) = 1 - b_d(i)$ . On that convention the  $Q$ -statistic, the disagreement measure and the double-fault rate carry their standard signs. Two correspondences are worth stating because they are definitional rather than empirical. First,  $\phi$  is invariant under flipping both variables, so the  $\phi$  computed on correctness is exactly the  $\phi$  of Table 10; no number changes. Second, the double-fault rate is by construction the rate at which both members are wrong, which is the joint breach rate  $ASR_{d_1 d_2}$  of Eq. (6). Our  $\Delta$  is therefore the excess double-fault over independence, and Table 13 is a re-presentation of Table 10 rather than a new measurement.

**Table 13**

Pairwise diversity for the fifteen measurable pairs, in the vocabulary of Kuncheva and Whitaker [19], computed on the correctness indicator and ordered by  $DF$ . Llama Guard is excluded.

| Defense pair                                    | $Q$   | dis. | $DF$  | $DF_{ind}$ | $\phi$ | $\phi/\phi_{max}$ |
|-------------------------------------------------|-------|------|-------|------------|--------|-------------------|
| perplexity $\times$ probe <sub>16</sub>         | 0.891 | 0.18 | 0.580 | 0.449      | 0.59   | 0.62              |
| probe <sub>16</sub> $\times$ probe <sub>8</sub> | 0.977 | 0.12 | 0.580 | 0.408      | 0.75   | 0.89              |
| perplexity $\times$ probe <sub>8</sub>          | 0.909 | 0.18 | 0.540 | 0.396      | 0.62   | 0.71              |
| perplexity $\times$ refusal-prime               | 0.880 | 0.20 | 0.530 | 0.396      | 0.58   | 0.66              |
| probe <sub>16</sub> $\times$ refusal-prime      | 0.769 | 0.26 | 0.510 | 0.408      | 0.45   | 0.54              |
| perplexity $\times$ SmoothLLM                   | 0.887 | 0.22 | 0.490 | 0.356      | 0.57   | 0.73              |
| probe <sub>16</sub> $\times$ SmoothLLM          | 0.825 | 0.26 | 0.480 | 0.367      | 0.49   | 0.66              |
| probe <sub>8</sub> $\times$ refusal-prime       | 0.765 | 0.26 | 0.470 | 0.360      | 0.46   | 0.46              |
| refusal-prime $\times$ SmoothLLM                | 0.859 | 0.22 | 0.460 | 0.324      | 0.56   | 0.63              |
| probe <sub>8</sub> $\times$ SmoothLLM           | 0.784 | 0.26 | 0.440 | 0.324      | 0.48   | 0.54              |
| perplexity $\times$ token-anomaly               | 0.738 | 0.39 | 0.310 | 0.231      | 0.35   | 0.66              |
| probe <sub>16</sub> $\times$ token-anomaly      | 0.709 | 0.41 | 0.310 | 0.238      | 0.32   | 0.64              |
| probe <sub>8</sub> $\times$ token-anomaly       | 0.750 | 0.35 | 0.300 | 0.210      | 0.39   | 0.65              |
| refusal-prime $\times$ token-anomaly            | 0.610 | 0.39 | 0.280 | 0.210      | 0.30   | 0.50              |
| SmoothLLM $\times$ token-anomaly                | 0.652 | 0.35 | 0.270 | 0.189      | 0.34   | 0.50              |

*The verdict is measure-independent.* Every one of the fifteen pairs has  $Q > 0$  and a double-fault rate above its independence prediction, with  $Q$  ranging from 0.610 to 0.977. Across pairs the rank agreement between measures is high: Spearman  $\rho = 0.99$  between  $Q$  and  $\phi$ ,  $-0.97$  between disagreement and  $\phi$ , which is the expected sign since disagreement rises as agreement falls, and 0.89 between double-fault and  $\phi$ . The conclusion of Section 11.2 is therefore not an artifact of choosing  $\phi$ ; three further statistics, developed independently for ensemble analysis, order the pairs the same way and reach the same sign. We regard this as a robustness check rather than a new finding, and report it as such. Table 13 gives point estimates for legibility; 10,000-resample bootstrap intervals for  $Q$ , disagreement and double-fault, on the same footing as those reported for  $\phi$ , are in the supplementary material.

*Why Llama Guard is excluded here.* Table 13 omits Llama Guard for a sharper reason than in Table 10, and the exclusion is now overdetermined. With a residual ASR of 0.01 the product  $n_{01}n_{10}$  underflows and  $Q$  saturates at exactly 1.000 for all six of its pairs, its algebraic maximum, irrespective of the underlying association. Including those six pairs drags the  $Q$ - $\phi$  rank correlation from 0.99 to  $-0.26$ . The same marginal breaks two further diagnostics independently: the stratified analysis of Section 11.6 returns an infinite common odds ratio for all six of its pairs and an undefined  $p$  for one of them, and the selection comparison earlier in this section is floor-limited while it is in the pool. Three separate procedures degenerate on the same member, which is a general point about evaluating stacks rather than a quirk of one statistic: a defense that the available adversary cannot stress is not merely hard to assess, it breaks the standard diagnostics outright, and  $\phi$  is preferable here only because its bootstrap interval widens instead of saturating. This is the measurement-side counterpart of Section 11.1.

### 11.10. Design consequences: combination rule, membership, and the refusal axis

Having checked that the dependence verdict is measure-independent, we use the same vocabulary to ask the two design questions  $\phi$  alone cannot answer: how the members' decisions should be combined, and which members to include.

*Combination rules.* The stack of Section 11.2 is a unanimity veto: one blocker suffices, so an attack succeeds only when every member is breached. That is the  $k = 1$  case of a  $k$ -of- $n$  rule, and sweeping  $k$  traces the operating curve of Table 14, whose two blocks answer different questions. The upper block combines the members’ *independently measured* decisions, so its residual ASR inherits the intersection assumption discussed in Section 11.11 and its refusal column is the union of per-layer refusals on the benign set; the lower block gives the two configurations run end to end, and the deployed pipeline there refuses more than that union, which we analyze below. The curve is steep and asymmetric. Relaxing from  $k = 1$  to  $k = 2$  buys 22 points of benign false refusal at a cost of 18 points of attack success, and by  $k = 3$  residual ASR has passed 0.4. The  $k = 2$  residual of 0.190 is the same figure the six-defense intersection of Section 11.3 produces, and on the labels reported here it is the same nineteen behaviors rather than a coincidence. Containment holds by construction, since a behavior that breaches all six non-Llama-Guard defenses necessarily breaches at least six of seven. The two sets coincide because the converse gap is empty: a behavior could enter the  $k = 2$  set without entering the intersection only by breaching Llama Guard while some other member holds, and there is no such behavior in this data. Eighteen of the nineteen breach exactly six of seven, and the nineteenth breaches all seven, which is the single behavior Llama Guard fails on. At this operating point the veto is the only defensible corner, which is worth stating because majority voting, a default combiner across much of the ensemble literature, is actively harmful when the members’ errors are as correlated as they are here. The asymmetry has a simple source: raising  $k$  requires more members to agree before an input is blocked, and members that fail together cannot supply that agreement on the inputs where it is needed. False refusal is flat at 0.080 from  $k = 4$  onward because no benign prompt is refused by four or more members at once, so the tail is the base model’s own floor rather than a contribution from any layer.

The last two rows of Table 14 are the more uncomfortable ones. Llama Guard alone attains the same 0.010 residual ASR as the union of all seven members at half its false-refusal cost, and at a third of the cost of the pipeline as actually deployed. Six of the seven layers, on this target and this adversary, purchase no additional security and a great deal of benign refusal. This is the intersection-versus-direct observation of Section 11.2 restated as a deployment decision.

**Table 14**

Combination rules over the seven defenses. The stack blocks an input when at least  $k$  members block it.

| Rule                                                  | residual ASR | benign FRR |
|-------------------------------------------------------|--------------|------------|
| <i>Combined from independently measured decisions</i> |              |            |
| $k = 1$ of 7 (veto)                                   | 0.010        | 0.520      |
| $k = 2$ of 7                                          | 0.190        | 0.300      |
| $k = 3$ of 7                                          | 0.430        | 0.110      |
| $k = 4$ of 7                                          | 0.590        | 0.080      |
| $k = 5$ of 7                                          | 0.680        | 0.080      |
| $k = 6$ of 7                                          | 0.750        | 0.080      |
| $k = 7$ of 7 (unanimous flag)                         | 0.790        | 0.080      |
| <i>Run end to end</i>                                 |              |            |
| deployed seven-layer pipeline (veto)                  | 0.000        | 0.810      |
| strongest single layer (Llama Guard)                  | 0.010        | 0.260      |

*Selecting members: the row heuristic is not leaving anything on the table.* Composition rule 2 recommends at most one defense per dependency row. An obvious objection is that this is a coarse proxy for what one actually wants, which is low double-fault. We therefore compare it against a greedy search that selects members by directly minimizing double-fault with the current set. Excluding Llama Guard, whose floor makes every rule look identical, the two procedures select the *same* members in the *same* order, with identical ASR and FRR at every stack size. That agreement is labeling-dependent and should not be carried further than the data supports: under majority-of-five grader labels probe<sub>8</sub> overtakes probe<sub>16</sub> on marginal ASR, which breaks it from stack size three onward (Section 11.7). Writing each stack’s cumulative (ASR, FRR) after the member that joins it: token-anomaly (0.35, 0.09), then SmoothLLM (0.27, 0.27), then probe<sub>8</sub> (0.23, 0.28), then refusal-prime (0.20, 0.45). The row heuristic, which requires only the taxonomy of Table 3 and no measurement at all, recovers the choice made by a search with full access to the joint breach data. That is a useful property for practitioners, who will have the taxonomy long before they have the correlation

matrix. The agreement should not be over-read: with six candidates spanning four rows the two procedures have limited room to diverge, so this is a consistency check on rule 2 rather than evidence that a taxonomy is sufficient in general.

The comparison stops at size four, and the reason is structural rather than empirical: the six non-Llama-Guard defenses span exactly four dependency rows, so rule 2 admits no size-five stack. The greedy search continues, adding probe<sub>16</sub> and then perplexity, and reaches 0.190 against the row rule's 0.200, so the taxonomy is not strictly optimal, merely capped by itself, and the last two additions buy nothing (ASR 0.190 twice, false refusal flat at 0.450).

Read against the single-layer reference, the same selection is a caution. Four rule-compliant layers reach a residual of 0.20 at a refusal rate of 0.45, while Llama Guard alone reaches 0.01 at 0.26. Selecting across rows improves a stack over its own members, exactly as rule 2 predicts, and still does not approach what one strong layer delivers. Cross-row selection is a rule for spending a stack budget well, not an argument for having one.

*False refusals are correlated too, and compose worse than any rule predicts.* Section 11.2 reported that the assembled stack refuses 0.810 of benign prompts against an independence prediction of 0.766. The per-prompt refusal vectors let us decompose that, and the decomposition changes the reading twice. Pairwise  $\phi$  on benign refusals is positive for all 21 pairs, mean 0.573, reaching 1.000 for perplexity  $\times$  probe<sub>16</sub>. In consequence the *empirical* union, the fraction of benign prompts that at least one member refuses, is only 0.520, far below the 0.766 that independence predicts. That is the direction positive correlation implies, and it is the opposite of the inference drawn in Section 11.2 from the deployed figure alone.

That dependence is almost entirely the base model's own refusal floor, and reporting it as a shared blind spot would be a mistake. Measured directly, the undefended model refuses 8 of the 100 benign prompts, and every one of the seven defended configurations refuses a superset of exactly those eight; perplexity and probe<sub>16</sub> refuse on precisely the floor and nothing else, which is why their  $\phi$  is 1.000. Recomputing on the 92 non-floor prompts collapses the effect: eleven of the twenty-one pairs become degenerate because a member adds no refusal beyond the floor, and across the ten that remain the mean  $\phi$  falls from 0.573 to 0.056, with four of ten positive. The benign side therefore does *not* independently reproduce the harmful-side dependence. What it shows is a shared common cause, the same shape that difficulty stratification gives for most of the cross-row breach correlation in Section 11.6.

The deployed stack nonetheless refuses 0.810, more than either quantity. No union over independently measured decisions can produce that value, and the gap is not noise: it is 0.29 above the empirical union. The mechanism is composition itself. Members run in sequence and each alters the response the next one sees, since SmoothLLM perturbs the prompt and refusal-priming prepends a refusal-shaped prefix, so refusals compound rather than merely intersect. The practical consequence is that independence fails on the benign side as well as the harmful side, and it fails in *both* directions: it overestimates the refusal burden of a hypothetical union and underestimates that of the pipeline actually deployed. A designer who budgets false refusals by assuming independence will be wrong either way, and the sign of the error depends on a modeling choice they may not realise they are making. Composition rule 5 should therefore be read as requiring a measured refusal rate for the assembled pipeline rather than a budget assembled from per-layer rates by any formula.

One qualification bounds the refusal correlations specifically. Every  $f_i$  contains the shared base-model floor of roughly 7 to 8%, and for the two lowest-refusal filters that floor is most of the rate: perplexity and probe<sub>16</sub> sit at  $f = 0.08$  and refuse an identical set of prompts, so their  $\phi = 1.000$  reflects a common component neither layer contributes rather than a shared blind spot. Refusal correlation among the low-rate filters is, to that extent, correlation of the model underneath them.

### 11.11. Limitations of the correlation measurement

**Added pair.** The probe<sub>16</sub> $\times$ probe<sub>8</sub> pair was not part of the original design. It was added after the primary same-row pair proved unmeasurable under GCG; it is reported as exploratory, and the two instances share an input representation, so it bounds the same-row effect from the easy side. The primary token-surface pair is unaffected and is reported above.

**Adversary definition.** Each reported ASR is the output of a single attack strategy, not of an adversary free to choose among strategies. In the suffix runs this understates risk for defenses that neutralize the suffix class (Section 11.4); a best-of-strategies adversary is the quantity a deployer should assume, and we report the components from which it can be formed rather than the maximum itself.

**Attack implementation.** Our fluent adversary is an AutoDAN-*style* sentence-level genetic search over a fixed pool of framings; the released AutoDAN additionally uses an LLM for paraphrase mutation [128], which would likely raise ASR further and can only strengthen the direction of these findings.

**Target strength and replication power.** The primary target is weakly aligned, giving high residual ASRs and well-powered correlations. The Llama-2-7B replication (Section 11.8) reproduces the attack-class result and the adaptive-versus-static gap, but at  $n = 50$  its correlation estimates are directionally consistent rather than independently confirmatory; the correlation claims therefore rest on the single primary target, and a full-size replication remains outstanding.

**Untestable rows.** H1 is evaluated in two of the five rows the stack occupies. The semantic-classification row admits no within-row test on this data for a substantive reason rather than a practical one: Llama Guard’s residual ASR is 0.01, so any second instance in that row would face a near-constant breach vector and an undefined  $\phi$ . A row that is highly effective against the available adversary cannot be assessed for internal correlation, which is itself a limitation of correlation-based composition analysis. Input-provenance and architecture rows are out of scope because they target A0 injection rather than A1 jailbreak.

**Statistical power and seeds.** Results come from a single attack seed at  $n = 100$ ; the bootstrap intervals quantify sampling over behaviors but not the stochasticity of the genetic search itself. The stratified analysis of Section 11.6 is underpowered by construction.

**Probe validity.** The internal-representations defense is a probe we trained ourselves; its validation AUROC is near-ceiling on a held-out split of the training pools, which may reflect topical separability between the harmful and benign corpora as much as a representation of harmfulness. Its operating threshold is calibrated on the *same* 100 benign prompts later used to report false-refusal rates, so every threshold in this study is set in-sample, and at  $n = 100$  a 1% target rate is a single prompt. We have since measured that margin rather than leaving it unknown. Recalibrating on an external corpus of 805 general instructions, disjoint from both the evaluation set and the probe’s training pool, moves the surface filters only slightly and in the permissive direction, but moves  $\text{probe}_{16}$  so far that at its external threshold it would refuse 26% of the held-out benign set instead of 1%. The evaluation benign prompts are matched counterparts to the harmful behaviors and therefore sit far closer to the probe’s decision boundary than ordinary instructions do. The probe’s 1% operating point is thus an artifact of the calibration corpus rather than a transferable property, which is direct evidence for the topical-separability concern raised above. Because recalibrating changes which prompts are blocked and therefore every downstream breach outcome, we re-ran the entire primary configuration under external thresholds rather than reasoning about the margin: Section 11.7 reports that run beside the one tabulated here. All fifteen correlations remain positive and significant, so the composition result does not depend on the calibration corpus, but the token-anomaly filter turns out to share the probe’s problem in a different form, its residual ASR nearly doubling once its threshold is set out of sample. The results tabulated throughout this section retain the in-sample thresholds and are labeled as such.

**Judging.** Breach is scored by a single automated rubric grader at the standard binarization threshold, without human adjudication on this corpus. Of the 1,887 responses actually sent to the grader, since blocked inputs are never judged, 66, or 3.5%, did not parse into the rubric format under the single judgment reported here and are counted as non-breach, so the reported ASRs are very slightly conservative. That count is specific to this run and sits at the favorable end of its own distribution: across the five repetitions of Section 11.7 the unparsed count ranged from 68 to 79 (mean 72.6), and the majority-vote equivalent is 72. We report the figure for the run tabulated here and state the range rather than the single draw.

**Judge reproducibility.** The grader is not deterministic. We measured this rather than leaving it as a caveat: Section 11.7 reports five judgments of all 1,887 graded responses and draws the resulting grader interval alongside the behavior bootstrap in Figure 5. The pilot below motivated that measurement and is retained for the record: re-judging an identical set of 300 undefended responses with the same rubric, model and temperature produced different verdicts on 19 of them (6.3%), only one of which involved a parse failure, and the undefended adaptive ASR moved from 0.58 to 0.64 on that re-judge alone. Under the five-judgment majority the undefended adaptive ASR is 0.63. The bootstrap intervals tabulated in this section resample behaviors and therefore quantify sampling variation, not grader variation; Figure 5 reports the two side by side rather than combined. Correlations do prove steadier in sign than levels, since all fifteen stay positive and significant under majority labels, but not steadier in magnitude: thirteen of fifteen rise, by up to 0.16.

**Stack composition.** The direct-stack measurement uses one fixed layer ordering; we do not vary the order, and with Llama Guard present the assembled residual is near zero, which limits what the intersection-versus-direct comparison can discriminate.

**Scope.** Single-turn, A1 jailbreak only. All reported analyses run on two 7B-class open-weight targets; a smaller Llama-3.2-3B-Instruct run contributes one comparison in Section 11.2 and nothing else, and we do not treat it as a

third target. Nothing here establishes that the same correlation structure holds at frontier scale or on the proprietary stacks of Section 12.1.

That last caveat invites the obvious question of whether the framework says anything about systems built at a scale we cannot measure. Section 12.1 answers it in the only way available without access to those systems: by reading three published production deployments through the framework's categories and asking whether they resolve into it.

## 12. Discussion

We first read three deployed systems through the framework, then answer the four research questions of Section 3 in turn. RQ1 to RQ3 draw on the literature base of Section 3.3; RQ4 draws on the derivation of Section 10 and the measurement of Section 11. Limitations and the work they leave open close the section.

### 12.1. Applying the framework to production deployments

The instruments developed so far, the AATM tiers of Section 5, the overhead classes of Section 9, and the composition rules of Section 10, are meant to describe systems that exist, not only systems a practitioner might design. We apply them here to three deployed LLM defense systems, selected because their engineering teams published enough architectural and quantitative detail to support the analysis. Reading published deployments through the framework tests it in a way a constructed example cannot: the deployments were designed without it, so any regularity the framework exposes in them is a property of the deployments rather than of our exposition.

*Case 1: DoorDash customer support.* DoorDash deployed an LLM-based support system for delivery contractors using a retrieval-augmented architecture with multi-layered quality control [132]. A two-tiered guardrail checks each generated response in real time for accuracy and policy compliance, while an offline judge monitors quality across retrieval correctness, response accuracy, grammar, coherence, and relevance. The deployment reports a 90% reduction in hallucinated responses and a 99% reduction in severe compliance issues against the initial unguarded deployment. In framework terms, it defends an A0 to A1 surface with Layer 1 and Layer 4 mechanisms and accepts class-B overhead: the real-time guardrail adds serial passes on the critical path while the offline judge adds asynchronous ones.

*Case 2: Airbnb conversational AI.* Airbnb upgraded their platform to a hybrid architecture integrating LLMs for natural language understanding while retaining rule-based workflows for sensitive operations [133]. The guardrails framework operates at three levels: pre-processing filters that sanitize inputs, inference-time constraints that limit the model's action space for sensitive operations, and post-processing validators. In framework terms, the platform narrows the A1 attack surface by construction, routing sensitive operations out of the model's action space, and guards the remainder with class-B filters at both ends, so no single layer is load-bearing. By Table 3 the deterministic routing occupies the architecture row while the filters occupy the semantic row, which is a cross-row composition and therefore the configuration rule 2 recommends.

*Case 3: OneShield privacy guard.* OneShield was deployed within a large enterprise data and model factory, addressing personally identifiable information across user inputs and model outputs in a multilingual environment spanning 26 languages [134]. It combines named-entity recognition models fine-tuned for PII detection with rule-based validators for structured formats, operating as an intermediary layer that intercepts and anonymizes sensitive content in real time. It achieved a 0.95 F1 score in detecting sensitive entities, outperforming comparable tools by up to 12%. In framework terms this is a Layer 1 defense against privacy leakage at the A1 interface with class-B overhead, and its 26-language scope addresses the cross-lingual evaluation gap identified in Section 8.

*Cross-case analysis.* All three deployments implement multiple defensive layers rather than relying on a single mechanism, maintain pathways for human intervention, require ongoing monitoring, and involve explicit thresholds for acceptable performance degradation. Read through the framework, three further observations emerge. First, all three defend the A0 to A1 surface and none addresses A3 or A4, which matches the coverage gap Table 2 exposes and which by Eq. (4) no amount of additional content-level defense would close. Second, all three select from classes A and B, and two of the three do so partly because they sit above a model they do not train, so classes D and E were never available to them. Third, all three compose across rows of Table 3 rather than within, which is consistent with the first half of rule 2 and may explain why their reported results held in production, though Section 11 shows that cross-row selection reduces rather than eliminates correlated failure, so the residual these stacks deliver should

still be expected to exceed the multiplicative prediction. These are published third-party accounts read through our framework, not controlled experiments, so we treat them as illustrative application rather than empirical proof. What they do establish is that the framework's categories are the ones production teams are already reasoning in, without having had a vocabulary for them: each deployment had chosen an exposed surface, a reachable cost class, and a set of mechanisms drawn from different dependency rows, and none had a way to say so.

## 12.2. RQ1: What vulnerabilities are inherent to LLMs, and how do adversarial attacks exploit them?

The threat taxonomy of Section 6 and AATM (Section 5) together answer both halves. At the model level the decisive weakness is that safety is enforced by a small, brittle subset of parameters layered over competing training objectives, which is what lets low-cost fine-tuning or pruning strip safety without harming general performance (A3 to A4) and lets role-play prompts pit instruction-following against safety at the input level (A1) [53, 55]. At the data level, uncurated corpora carry biases and sensitive content, and the gap between broad pretraining and narrow safety tuning is what poisoning, backdoor insertion (A4), and obfuscation exploit [54, 64]. At the user level, prompts fuse trusted instructions with untrusted input, the structural opening behind direct and indirect injection (A0 to A1) [28]. At the infrastructure level, weak access controls and exposed deployment surfaces enable theft, extraction, and remote code execution [66]. The recurring pattern is that many of these exploits succeed by perturbing the first few generated tokens, which is why shallow safety alignment links vulnerabilities that otherwise look unrelated. It is also the natural candidate for why stacked defenses fail together, though Section 11 finds the measured dependence better explained by a common difficulty gradient.

## 12.3. RQ2: What are the current state-of-the-art defenses, and how do they map to the threat landscape?

Section 7 reviews the defenses; the informative comparison is structural rather than numerical, since reported reductions in attack success cannot be compared directly (Section 8.1). Table 15 maps the defense categories onto the threat dimensions, and a per-study extraction table covering every reviewed work, with its evaluation setting, reported metric, adaptive-testing status, and resource class, is provided as supplementary material.

**Table 15**  
Defensive strategies mapped to threat dimensions.

| Defense Category         | Primary Threat | Secondary Threat | Key Target Areas                                                 |
|--------------------------|----------------|------------------|------------------------------------------------------------------|
| Input/Output Censorship  | User-centric   | Model-centric    | Input validation; response filtering; content moderation         |
| Model Training           | Model-centric  | Data-centric     | Architecture security; safety training; fine-tuning optimization |
| Adversarial Training     | Model-centric  | Infrastructure   | Defense generation; attack resistance; model hardening           |
| Monitoring Systems       | Infrastructure | User-centric     | Threat detection; incident response; system monitoring           |
| Infrastructure Hardening | Infrastructure | Data-centric     | System security; deployment protection; access control           |

Three patterns emerge that the per-category figures obscure. First, almost all content-level defenses assume an A1 adversary, while the attacks that most reliably remove safety, fine-tuning and pruning, require A3 or A4 and are addressed by no content-level class; by Eq. (4) that gap cannot be closed by adding more A1 defenses. Second, adaptive evaluation remains the exception: adversarial training targets adaptive attacks by construction, bounded by its perturbation model, and among inference-time defenses only the constitutional-classifier line has faced sustained adaptive human red teaming [77, 78], while certified schemes such as erase-and-check [102] occupy the complementary extreme, guaranteeing safety against any adversary but only within a bounded attack size. Third, the dependency rows are unevenly populated: the semantic-classification row holds most of the reviewed work and most of the reported effectiveness, which is precisely the concentration that makes independence scarce when these defenses are stacked.

*Research Gaps.* **1) Effectiveness versus usability.** Input/output censorship systems struggle with false positive rates and utility preservation, and the field lacks standardized approaches for measuring the balance between security and performance degradation. Equation (7) formalizes why this worsens under stacking.

**2) Safety mechanism persistence.** Current architectures remain vulnerable to targeted attacks despite sophisticated training methods, and robust methodologies for ensuring the persistence of safety across model updates and fine-tuning iterations are lacking.

**3) Computational scalability and generalization.** Adversarial training is computationally intensive and may not generalize to novel adaptive attacks, and many approaches focus on known attack patterns.

**4) Real-time response.** Achieving high-accuracy anomaly detection without unacceptable latency remains open, though class-E monitoring (Section 9) suggests a cheaper path than the auxiliary-model approaches that dominate the literature.

**5) Systematic incident analysis.** Unlike traditional cybersecurity, LLM security lacks standardized frameworks for documenting, analyzing, and sharing security incidents, which limits understanding of real-world attack patterns.

**6) Infrastructure deployment.** The field lacks unified frameworks for assessing system-wide security across diverse deployment environments.

**7) Measured composition.** Section 10 derives how coverage, cost, residual risk, and refusals compose, but the derivation rests on one quantity no reviewed study reports: the failure correlation between two defenses. Section 11 closes the first part of this gap by measuring it for all fifteen pairs of a seven-layer stack, which converts the same-row half of Table 3 from a hypothesis into a supported claim and refutes the cross-row half's implicit independence. What remains open is scale and generality: one primary target, one behavior set of 100, two attack classes, and a single-turn A1 threat model. The correlation structure at frontier scale, across multi-turn and agentic settings, and for the rows this experiment could not test, is still unmeasured, and it remains in our view the single most valuable open problem in blue-team LLM security, because it is the precondition for predicting what any deployed stack actually delivers.

Section 12.1 reads three deployed systems through the framework and finds the same pattern from the outside: every one of them defends the A0 to A1 surface, none addresses A3 or A4, and all three compose across rows of Table 3 rather than within.

## 12.4. RQ3: How effective are existing evaluation frameworks and metrics?

Section 8 surveys the landscape; the short answer is that current frameworks are reproducible and convenient but systematically optimistic. Automated pipelines make large-scale comparison easy, yet four limitations recur. First, the headline figures are mostly method-specific quantities measured against static attack sets, so they overstate robustness against an adversary who adapts (Section 8.1). Second, many widely cited scores are tuned to particular architectures and do not transfer across model families, while automated classifiers miss context-dependent harms that human assessment catches [119]. Third, coverage is predominantly English and concentrated on known attack patterns. Fourth, and newly visible from Section 10, no benchmark evaluates a composed stack, and no metric expresses the failure correlation between defenses. The benchmarks that begin to close the first three gaps, such as StrongREJECT and JailbreakBench, remain the exception; nothing yet addresses the fourth.

## 12.5. RQ4: How do defenses compose?

Section 10 answers this in full. In summary: coverage composes as a union that saturates within a tier, so stacking cannot reach an uncovered adversary tier; cost composes predictably by overhead class, with class C the term that makes stacks unaffordable unless gated; effectiveness composes multiplicatively only when the stacked defenses depend on independent model properties, and shallow safety alignment makes such independence scarce; and false refusals compose as a union against the defender, placing a usability ceiling on depth that binds before the compute budget does. What a layered defense purchases is therefore independence rather than redundancy, which converts defense-in-depth from a slogan into a constraint on which combinations are worth paying for. Stated in the vocabulary of classifier fusion, a defense stack is an ensemble and independence is diversity, so the conclusion that field reached about combining classifiers applies here: diversity has to be engineered and verified [20]. The rules that follow are stated in Section 10.

Section 11 then measures the quantity the rules most need. Five results bear directly on this research question. First, independence fails everywhere we could look: all fifteen measured pairs have  $\phi > 0$  and  $\Delta > 0$ , so a stack always delivers less than the multiplicative prediction, and the shortfall reaches 0.172 in joint residual on this target. Second, the dependence is predominantly common-cause: conditioning on behavior difficulty dissolves almost every cross-row association. Which pair, if any, retains a within-stratum association is not identifiable on this data, since the

survivor changes with the grader draw and with the calibration corpus, so we retire the mechanism-specific reading rather than report the regime that favors it. Rule 2 survives as a rule that improves the residual rather than one that restores multiplicativity. The composition verdict itself is robust: every pair stays positive under permutation inference, majority-of-five grader labels and externally calibrated thresholds, the minimum over the whole grid being 0.300. Third, the refusal ceiling is real and low: the stack refuses 0.810 of benign prompts, so depth is bounded by usability long before it is bounded by compute.

Fourth, and least expected, the per-defense figures do not predict the assembled stack in either direction. Where one member already dominates, the stack is statistically indistinguishable from it and the remaining layers buy only refusals; where none dominates, a direct attack on the assembled stack found no feasible prompt at all, against an intersection prediction of 0.190, because the joint constraint is far stronger than any member. On the refusal axis the error runs the other way, with the deployed pipeline exceeding the union of its members' measured refusals by 0.29. Composition therefore has to be measured end to end on both axes; it cannot be derived. A fifth result changes how the question should be asked at all: because a defense's measured strength is a property of the attack class it faces, a composition estimate assembled from per-defense figures obtained under different attacks is not merely imprecise but can be arbitrarily optimistic.

Taken together these five results narrow the claim the framework can support. Diversity between defenses is measurable, it is never zero, and measuring it is the right way to choose which layers to buy. It is not a way to forecast what the resulting stack will do, on either axis. The adjacent concerns of Appendix A inherit the same precondition: risk assessment covering bias, alignment, and trustworthiness feeds defense selection only to the extent that the underlying evaluation is adaptive and cross-lingual, which is precisely what the field currently lacks.

## 12.6. Limitations

The limitations specific to the correlation experiment are stated where the measurement is reported (Section 11.11). The five below apply to the paper as a whole, and the first two bound how far its conclusions travel beyond the configuration studied.

- **Scope of the literature synthesis.** The evidence base of Section 3.3 is a structured, critical synthesis rather than a registered, exhaustive systematic review. We applied a staged screening process to make selection transparent, but among eligible works we prioritized representative and influential studies by author judgment. Coverage is therefore representative rather than complete.
- **Corpus recency.** The defense literature moves quickly. Although our cutoff is February 2026, classifications made here, particularly overhead-class and adaptive-testing assignments for the newest systems, will need revisiting as those systems are independently evaluated.
- **Tier, overhead, and dependency assignments.** AATM tiers, overhead classes, and the dependency assignments of Table 3 were made from published descriptions by author judgment, not by re-implementing the defenses. Borderline mechanisms exist and are noted where they occur: routing designs such as MoGU sit in Layer 2 but add runtime cost, prompt-tuned prefixes such as PAT are trained adversarially but deploy as class-A tokens, and circuit breakers are paid for in training (class D) while their runtime intervention reads representations (class E), so a single defense can occupy two classes. The dependency assignments remain an interpretive claim about why each defense fails. Section 11 tests that claim for two of the five rows and one instance per row, and supports it there; the assignments for the remaining rows, and for the many defenses in each row we did not instantiate, stay a hypothesis for the composition literature to test.
- **Scope of the correlation experiment.** The measurement of Section 11 covers a single behavior set of  $n = 100$ , two 7B-class open-weight targets, one attack seed, and a single-turn A1 threat model, with breach scored by one automated rubric grader. The replication on a strongly aligned target is directionally consistent but underpowered at  $n = 50$ , so the correlation claims rest on the primary target. Section 11.11 states these and the remaining measurement-specific caveats in full.
- **Modeling assumptions in the composition analysis.** Equation (4) treats tier coverage as binary rather than graded. Equation (5) assumes class-C gating is independent of class-B screening; if one signal gates both, a joint term is required. In Eq. (7) the union bound holds unconditionally, but the product form assumes layer-independent false positives, so the bound is the load-bearing claim.

- **Secondary deployment evidence.** The framework application of Section 12.1 relies on self-reported industry engineering accounts. These were selected for architectural and quantitative detail, but they are not controlled experiments, and reported figures could not be independently verified.

## 12.7. Future Research Directions

The limitations above are not all equally tractable, and the directions below are ordered by how directly they follow from what this paper leaves undone. The first three extend the measurement itself; the remainder concern what the field would have to change for composition to become reportable at all.

- **Scaling the correlation measurement.** Section 11 establishes that  $\phi$  is measurable and that it is positive across a seven-layer stack, but on one primary target and one behavior set. The natural continuations are a full-size replication on a strongly aligned target, pooling across targets to power the difficulty-stratified analysis, extension to the rows this experiment could not instantiate, and repetition across attack seeds, which we did not do and which the grader-variance measurement of Section 11.7 does not substitute for. Pooling would also be the way to settle the question this study leaves open, namely whether any pair's correlation survives conditioning for mechanism-specific reasons; at  $n = 100$  split across six strata that question is underdetermined, and the answer we obtain changes with the labeling regime. Reporting the joint rather than the marginal attack success rate should become routine in defense evaluation, since the marginals alone determine nothing about a stack: as Section 11.3 shows, even the joint rate over pairs does not predict the assembled configuration, which has to be attacked directly.
- **Diversity creation for defense stacks.** Classifier fusion responded to the scarcity of independent members by developing methods that create diversity deliberately [20, 50]. No counterpart exists for LLM defenses, where layer selection is still a matter of picking components that look different. Adapting diversity-creation methods, and defining what a diversity-creating objective would mean when the shared dependency is the wrapped model rather than the training sample, is in our view the most promising route from measurement to design.
- **Attack-class coverage as a design axis.** Because a defense's measured strength depends on the attack class it faces (Section 11.1), stack design needs a notion of complementarity defined over attack classes rather than over per-input failures alone. Characterizing which defenses cover which classes, and reporting defense results against at least one fluent and one optimized adversary, would make that axis usable.
- **Universal evaluation frameworks.** Standardized, cross-model protocols that account for multilingual and domain-specific adversarial challenges are needed [122, 135], together with benchmarks defined over composed stacks rather than single defenses [123].
- **Resource-efficient and real-time defenses.** Class-E monitoring suggests that always-on detection need not carry the cost the literature assumes. SafeDecoding confines its intervention to the first generated tokens [86], and probe-gated cascades cut classifier compute by more than an order of magnitude [78]; systematic study of what can be detected from activations alone is a natural next step.
- **Deepening alignment.** If shallow safety alignment is a source of correlated failure, which our data leaves open, then defenses whose safety behavior extends beyond the first few tokens would increase the independence available to stack designers rather than merely the strength of a single layer. Deciding this is one motivation for the larger correlation measurement above: the prediction is that deepening alignment lowers  $\phi$ , and that is testable.
- **Incident reporting.** Standardized frameworks for incident reporting, following models such as CVE databases, would improve collective understanding of real-world attack patterns [136].
- **Cross-domain and multi-modal attacks.** As LLMs are applied to multi-modal and domain-specific tasks [137], systematic cross-domain analysis and broader benchmarks are needed [138].

### 13. Conclusion

This article introduced the Adversary Access-Tier Model (AATM) and a five-class model of inference-time overhead, and used them to convert a scattered defense literature into a map that states, for each defense layer, the adversary access it presupposes, the attacks it addresses, the evidence behind its reported effectiveness, and the cost of running it. We reviewed five layers of defensive mechanisms, from input/output censorship and robust model training to adversarial training, monitoring, and infrastructure hardening, mapping each to the AATM tier it presupposes and the overhead class it incurs (Table 2).

Because no deployment runs a single defense, we then derived how these quantities behave under composition. Coverage composes as a union that saturates within a tier; cost composes predictably by overhead class; effectiveness composes multiplicatively only when the stacked defenses fail for independent reasons; and false refusals compose against the defender. The consequence is that a layered defense purchases independence rather than redundancy, and that the property most worth knowing about a candidate stack is the failure correlation between its components: a defense stack is an ensemble, and what an ensemble buys is diversity.

We then measured that property. Across all fifteen pairs of a seven-layer stack evaluated under a shared adaptive adversary, failure correlation was positive everywhere, so no pair delivered the residual the multiplicative model predicts. Holding behavior difficulty fixed dissolves most of that association: cross-row defenses fail together mainly because they face a common difficulty gradient. Which pair, if any, retains a within-stratum association proved not to be identifiable on this data. Under the reported labels it is a same-row probe pair; under majority-of-five grader labels three pairs survive and that pair's odds ratio falls by more than an order of magnitude; under externally calibrated thresholds the survivor is cross-row and the probe pair's test is undefined. We therefore retire the mechanism-specific reading rather than report the regime that favors it. This does not weaken the composition verdict, since common-cause dependence violates independence exactly as mechanism overlap would, and every pair stayed positive under permutation inference, majority-of-five grader labels, and external thresholds, the minimum over the whole grid being 0.300. Selecting across rows remains the right rule for spending a stack budget, and it improves the residual rather than restoring multiplicativity, but we do not claim that row membership predicts mechanism-specific correlated failure.

The assembled stack proved harder to predict than the pairwise analysis suggests, in both directions. It refused four in five benign prompts, more than the union of its members' own measured refusals, while remaining statistically indistinguishable from its single strongest layer, so where one member dominates, depth buys refusals and compute without measurable residual protection. Remove that member and the picture inverts: a direct attack on the remaining six-layer stack found no feasible prompt on any behavior, far below what intersecting the per-defense vectors predicts. The lesson common to both is that a stack's behavior is not recoverable from its parts and has to be measured as assembled. And because a perplexity filter blocked every optimized suffix while admitting nearly every fluent prompt, what a defense appears to be worth turned out to depend on the attack class it was measured against, which means composition estimates assembled from figures obtained under different attacks can be optimistic without limit.

Stated for information fusion, the result identifies a dependence structure the ensemble literature has not had to model. Classifier ensembles correlate through the data their members were estimated on, and that dependence weakens as the member pool widens; defense stacks correlate through the model every member wraps, and no amount of member diversity reaches it. Diversity creation for a stack therefore has to target the substrate rather than the pool, which is a design problem the field's existing methods were not built for.

The result also fixes what the statistic is good for, and the answer is narrower than the model it replaces. Failure correlation selects stack members: it identifies near-substitutes, and under the reported labels choosing on it recovers the same configuration a search over the full joint breach data would choose. It does not predict what the assembled stack delivers, and we show that it does not, in opposite directions on the two axes a deployer cares about. The correct workflow is therefore two-stage rather than one: measure diversity to decide what to buy, then measure the assembled configuration to learn what you bought.

Two limitations of the field follow from this analysis and frame the work ahead. Robustness figures obtained against fixed attacks do not bound the risk a deployment faces, and figures obtained against components do not bound the risk a stack faces. Closing the first requires adaptive evaluation; closing the second requires evaluating compositions, and reporting the joint rather than the marginal attack success rate is the smallest change that would make that possible. Until that becomes routine, a deployed stack's security is not a quantity the field currently knows how to state.

## CRedit authorship contribution statement

**Abrar Alotaibi:** Conceptualization, Methodology, Software, Investigation, Formal analysis, Data curation, Validation, Visualization, Writing – original draft, Writing – review & editing. **Muhammad Shahid Jabbar:** Writing – review & editing. **Sadam Al-Azani:** Writing – review & editing. **Moataz Ahmed:** Supervision, Funding acquisition, Project administration, Writing – review & editing.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgements

The authors acknowledge the support received from the Saudi Data and AI Authority (SDAIA) and King Fahd University of Petroleum & Minerals (KFUPM) under the SDAIA-KFUPM Joint Research Center for Artificial Intelligence Grant JRC-AI-RFP-20. This research is also supported by grant CRPG-25-2057 under the Cybersecurity Research and Innovation Pioneers Initiative, provided by the National Cybersecurity Authority (NCA) in the Kingdom of Saudi Arabia.

## Data availability

The code and data supporting the experiment of Section 11 are available at <https://github.com/AbrarAlotaibi/defense-correlation>. The repository contains the defense implementations and calibrated thresholds, the attack harnesses for both adversaries, the per-behavior breach vectors from which every  $\phi$ ,  $\Delta$  and confidence interval in Table 10 is computed, the StrongREJECT grading prompt with its SHA-256 digest, and the analysis scripts that regenerate Table 10 and Figures 3, 4 and 5. The behavior set and its matched benign set are JailbreakBench [123] and are redistributed under their original license. Model weights are obtained from their respective public releases and are not redistributed.

## A. Adjacent Concerns and Responsible Deployment

Section 6 placed a set of concerns outside the security scope: bias and fairness, misinformation, toxicity, ethical alignment, explainability, and regulatory compliance. They are not weaknesses an adversary exploits to break confidentiality, integrity, or availability, so folding them into the threat taxonomy would blur the boundary that makes AATM useful. They remain relevant to deployment, and they couple back to security in one specific way: a model that is opaque or non-compliant is harder to defend and harder to hold accountable once an attack succeeds. This section records the coupling and points to the assessment literature, rather than attempting a second review inside the first.

Risk assessment for LLMs is conventionally organized along four dimensions. *Bias and fairness* covers gender, racial, and ideological skew inherited from imbalanced corpora, surfaced by instruction tuning on unaligned data [139] and measured with StereoSet, CrowS-Pairs, and related instruments. *Ethical alignment* concerns whether model behavior tracks stated values, with RLHF and Constitutional AI [140] the dominant techniques and truthfulness, safety, fairness, and transparency the usual axes [35]. *Robustness* spans both adversarial robustness, which overlaps our security scope directly, and out-of-distribution robustness, which does not. *Trustworthiness* aggregates these with privacy and transparency into composite frameworks [35], of which the European Commission’s lawfulness, ethicality, and robustness criteria are the best known regulatory instance.

The datasets serving these dimensions form an ecosystem parallel to the defense benchmarks of Section 8. Toxicity and social bias draw on ToxiGen [141], RealToxicityPrompts [121], CrowS-Pairs [142], and StereoSet [143]; conversational safety on TRUSTGPT [144], SafetyBench [120], the Anthropic red-teaming corpus [145], and BeaverTails [146]; robustness on AdvGLUE [147], PromptBench [148], GLUE-X [149], and BOSS [150]; and truthfulness on TruthfulQA [151]. The coverage caveats of Section 8 apply here with equal force: predominantly English, predominantly static.

Responsible deployment practice reduces to a short list that the security analysis above should be read alongside rather than in place of: guardrails that monitor and filter interactions, documented and externally auditable evaluation,

compliance with data-protection regulation, and post-training alignment through RLHF or Constitutional AI. The dependency running through this paper applies here too. An assessment is only as trustworthy as its evaluation protocol, so the adaptive and cross-lingual protocols called for in Sections 8.1 and 8 are a precondition for risk assessment that reflects deployment rather than a static benchmark.

## B. Glossary of Terms

Table 16 lists the acronyms used throughout, including the access tiers of AATM and the overhead classes of Section 9, and Table 17 defines the technical terms the composition argument relies on.

**Table 16**

Acronyms used in this paper.

| Acronym | Expansion                             | Acronym | Expansion                                  |
|---------|---------------------------------------|---------|--------------------------------------------|
| AATM    | Adversary Access-Tier Model           | GCG     | Greedy Coordinate Gradient                 |
| ASR     | Attack Success Rate                   | JBB     | JailbreakBench                             |
| AUPRC   | Area Under the Precision–Recall Curve | LAT     | Latent Adversarial Training                |
| AUROC   | Area Under the ROC Curve              | LLM     | Large Language Model                       |
| CMH     | Cochran–Mantel–Haenszel               | OR      | Odds Ratio                                 |
| DPO     | Direct Preference Optimization        | PII     | Personally Identifiable Information        |
| DSR     | Defense Success Rate                  | RAG     | Retrieval-Augmented Generation             |
| FPR     | False Positive Rate                   | RLHF    | Reinforcement Learning from Human Feedback |
| FRR     | False Refusal Rate                    | TEE     | Trusted Execution Environment              |

**Table 17**

Technical terms used in this paper.

| Term                           | Definition                                                                                                                                                                |
|--------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Backdoor attack                | Attack inserting triggers during training that cause malicious outputs when activated.                                                                                    |
| Blue teaming                   | Defensive security practices protecting systems from attacks.                                                                                                             |
| Data poisoning                 | Attack manipulating training data to compromise model behavior.                                                                                                           |
| Diversity                      | In classifier fusion, the extent to which combined components fail on different inputs; failure correlation is one of its standard pairwise measures.                     |
| Failure correlation ( $\phi$ ) | Correlation between the breach indicators of two defenses on a common behavior set under a common adversary; the $\rho$ of Eq. (6).                                       |
| Guardrails                     | Safety mechanisms constraining LLM behavior through filtering or validation.                                                                                              |
| Jailbreak                      | Prompt designed to bypass safety measures and produce harmful content.                                                                                                    |
| Failure independence           | Property of two defenses whose breach indicators are uncorrelated ( $\rho = 0$ in Eq. (6)); the condition under which residual attack success composes multiplicatively.  |
| Mechanism independence         | Property of two defenses that depend on disjoint model properties. Sufficient for failure independence but not necessary, and not established for any pair in this study. |
| Prompt injection               | Malicious instructions embedded in inputs to manipulate LLM behavior.                                                                                                     |
| Red teaming                    | Offensive security practices simulating attacks to find vulnerabilities.                                                                                                  |
| Safety alignment               | Training LLMs to behave according to human values.                                                                                                                        |
| Shallow safety alignment       | Alignment that adapts the output distribution only over the first few generated tokens.                                                                                   |
| Transfer attack                | Attack developed against one model that affects other models.                                                                                                             |

## References

- [1] H. Inan, K. Upasani, J. Chi, R. Rungta, K. Iyer, Y. Mao, M. Tontchev, Q. Hu, B. Fuller, D. Testuggine, et al., Llama guard: Llm-based input-output safeguard for human-ai conversations, arXiv preprint arXiv:2312.06674 (2023).
- [2] Z. Zhang, Y. Lu, J. Ma, D. Zhang, R. Li, P. Ke, H. Sun, L. Sha, Z. Sui, H. Wang, et al., Shieldlm: Empowering llms as aligned, customizable and explainable safety detectors, in: Findings of the Association for Computational Linguistics: EMNLP 2024, 2024, pp. 10420–10438. doi:10.18653/v1/2024.findings-emnlp.610, arXiv:2402.16444.
- [3] S. Chen, A. Zharmagambetov, S. Mahloujifar, K. Chaudhuri, D. Wagner, C. Guo, SecAlign: Defending against prompt injection with preference optimization, in: ACM SIGSAC Conference on Computer and Communications Security (CCS), 2025.
- [4] E. Wallace, K. Xiao, R. Leike, L. Weng, J. Heidecke, A. Beutel, The instruction hierarchy: Training llms to prioritize privileged instructions, arXiv preprint arXiv:2404.13208 (2024).

- [5] A. Zou, L. Phan, J. Wang, D. Duenas, M. Lin, M. Andriushchenko, J. Z. Kolter, M. Fredrikson, D. Hendrycks, Improving alignment and robustness with circuit breakers, in: *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024*, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024, 2024. ArXiv:2406.04313.
- [6] A. Sheshadri, A. Ewart, P. Guo, A. Lynch, C. Wu, V. Hebbar, H. Sleight, A. C. Stickland, E. Perez, D. Hadfield-Menell, S. Casper, Latent adversarial training improves robustness to persistent harmful behaviors in LLMs, *Transactions on Machine Learning Research* (2024). ArXiv:2407.15549.
- [7] B. Wu, L. Xia, Q. Li, K. Li, X. Chen, Y. Guo, T. Xiang, Y. Chen, S. Li, Transom: An efficient fault-tolerant system for training llms, arXiv preprint arXiv:2310.10046 (2023).
- [8] Q. Li, Z. Shen, Z. Qin, Y. Xie, X. Zhang, T. Du, S. Cheng, X. Wang, J. Yin, Translinkguard: Safeguarding transformer models against model stealing in edge deployment, in: *Proceedings of the 32nd ACM International Conference on Multimedia*, 2024, pp. 3479–3488.
- [9] B. C. Das, M. H. Amini, Y. Wu, Security and privacy challenges of large language models: A survey, *ACM Computing Surveys* 57 (2025) 1–39. ArXiv:2402.00888.
- [10] S. Abdali, R. Anarfi, C. Barberan, J. He, Securing large language models: Threats, vulnerabilities and responsible practices, arXiv preprint arXiv:2403.12503 (2024).
- [11] A. Athalye, N. Carlini, D. Wagner, Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples, in: *International conference on machine learning*, PMLR, 2018, pp. 274–283.
- [12] M. Andriushchenko, F. Croce, N. Flammarion, Jailbreaking leading safety-aligned LLMs with simple adaptive attacks, in: *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025. ArXiv:2404.02151.
- [13] M. Nasr, N. Carlini, C. Sitawarin, S. V. Schulhoff, J. Hayes, M. Ilie, J. Pluto, S. Song, H. Chaudhari, I. Shumailov, A. Thakurta, K. Y. Xiao, A. Terzis, F. Tramèr, The attacker moves second: Stronger adaptive attacks bypass defenses against LLM jailbreaks and prompt injections, arXiv preprint arXiv:2510.09023 (2025).
- [14] X. Qi, A. Panda, K. Lyu, X. Ma, S. Roy, A. Beirami, P. Mittal, P. Henderson, Safety alignment should be made more than just a few tokens deep, in: *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025. Outstanding Paper Award; arXiv:2406.05946.
- [15] A. Zou, Z. Wang, N. Carlini, M. Nasr, J. Z. Kolter, M. Fredrikson, Universal and transferable adversarial attacks on aligned language models, arXiv preprint arXiv:2307.15043 (2023).
- [16] X. Qi, Y. Zeng, T. Xie, P.-Y. Chen, R. Jia, P. Mittal, P. Henderson, Fine-tuning aligned language models compromises safety, even when users do not intend to!, arXiv preprint arXiv:2310.03693 (2023).
- [17] B. Wei, K. Huang, Y. Huang, T. Xie, X. Qi, M. Xia, P. Mittal, M. Wang, P. Henderson, Assessing the brittleness of safety alignment via pruning and low-rank modifications, in: *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21–27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, PMLR / OpenReview.net, 2024, pp. 52588–52610. ArXiv:2402.05162.
- [18] A. Souly, Q. Lu, D. Bowen, T. Trinh, E. Hsieh, S. Pandey, P. Abbeel, J. Svegliato, S. Emmons, O. Watkins, S. Toyer, A StrongREJECT for empty jailbreaks, in: *Advances in Neural Information Processing Systems 37 (NeurIPS)*, Datasets and Benchmarks Track, 2024. ArXiv:2402.10260.
- [19] L. I. Kuncheva, C. J. Whitaker, Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy, *Machine Learning* 51 (2003) 181–207.
- [20] G. Brown, J. Wyatt, R. Harris, X. Yao, Diversity creation methods: a survey and categorisation, *Information Fusion* 6 (2005) 5–20.
- [21] B. Biggio, G. Fumera, F. Roli, Multiple classifier systems for robust classifier design in adversarial environments, *International Journal of Machine Learning and Cybernetics* 1 (2010) 27–41.
- [22] B. Biggio, G. Fumera, F. Roli, Security evaluation of pattern classifiers under attack, *IEEE Transactions on Knowledge and Data Engineering* 26 (2014) 984–996.
- [23] Y. Zhou, T. Ni, W.-B. Lee, Q. Zhao, A survey on backdoor threats in large language models (llms): Attacks, defenses, and evaluations, arXiv preprint arXiv:2502.05224 (2025).
- [24] M. Feffer, A. Sinha, W. H. Deng, Z. C. Lipton, H. Heidari, Red-teaming for generative ai: Silver bullet or security theater?, in: *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, volume 7, 2024, pp. 421–437.
- [25] A. Alotaibi, R. Mughus, M. Ahmed, A red teaming framework for large language models: a case study on faithfulness evaluation, *Software Quality Journal* 34 (2026) 42.
- [26] M. Tan, K. Yamaguchi, A. Raney, V. Nockles, M. Leblanc, S. Bendelac, An ai blue team playbook, in: *Assurance and Security for AI-enabled Systems*, volume 13054, SPIE, 2024, pp. 259–279.
- [27] F. Perez, I. Ribeiro, Ignore previous prompt: Attack techniques for language models, arXiv preprint arXiv:2211.09527 (2022).
- [28] K. Greshake, S. Abdelnabi, S. Mishra, C. Endres, T. Holz, M. Fritz, Not what you’ve signed up for: Compromising real-world llm-integrated applications with indirect prompt injection, in: *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security*, 2023, pp. 79–90.
- [29] Y. Yao, J. Duan, K. Xu, Y. Cai, Z. Sun, Y. Zhang, A survey on large language model (llm) security and privacy: The good, the bad, and the ugly, *High-Confidence Computing* (2024) 100211.
- [30] M. S. Jabbar, S. Al-Azani, A. Alotaibi, M. Ahmed, Red teaming large language models: A comprehensive review and critical analysis, *Information Processing & Management* 62 (2025) 104239.
- [31] R. Jia, P. Liang, Adversarial examples for evaluating reading comprehension systems, in: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 2021–2031.
- [32] T. Orekondy, B. Schiele, M. Fritz, Knockoff nets: Stealing functionality of black-box models, in: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4958–4967.
- [33] E. K. Tang, P. N. Suganthan, X. Yao, An analysis of diversity measures, *Machine Learning* 65 (2006) 247–271.
- [34] T. Cui, Y. Wang, C. Fu, Y. Xiao, S. Li, X. Deng, Y. Liu, Q. Zhang, Z. Qiu, P. Li, et al., Risk taxonomy, mitigation, and assessment benchmarks of large language model systems, arXiv preprint arXiv:2401.05778 (2024).

- [35] Y. Huang, L. Sun, H. Wang, S. Wu, Q. Zhang, Y. Li, C. Gao, Y. Huang, W. Lyu, Y. Zhang, et al., Position: Trustllm: Trustworthiness in large language models, in: Forty-first International Conference on Machine Learning, 2024.
- [36] M. Gupta, C. Akiri, K. Aryal, E. Parker, L. Praharaj, From chatgpt to threatgpt: Impact of generative ai in cybersecurity and privacy, IEEE Access (2023).
- [37] E. Perez, S. Huang, F. Song, T. Cai, R. Ring, J. Aslanides, A. Glaese, N. McAleese, G. Irving, Red teaming language models with language models, in: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2022. doi:10.18653/v1/2022.emnlp-main.225, arXiv:2202.03286.
- [38] L. Lin, H. Mu, Z. Zhai, M. Wang, Y. Wang, R. Wang, J. Gao, Y. Zhang, W. Che, T. Baldwin, X. Han, H. Li, Against the achilles' heel: A survey on red teaming for generative models, J. Artif. Int. Res. 82 (2025).
- [39] Z. Dong, Z. Zhou, C. Yang, J. Shao, Y. Qiao, Attacks, defenses and evaluations for llm conversation safety: A survey, in: Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), 2024, pp. 6734–6747. doi:10.18653/v1/2024.naacl-long.375, arXiv:2402.09283.
- [40] A. G. Chowdhury, M. M. Islam, V. Kumar, F. H. Shezan, V. Jain, A. Chadha, Breaking down the defenses: A comparative survey of attacks on large language models, arXiv preprint arXiv:2403.04786 (2024).
- [41] J. Zhou, K. Li, J. Li, J. Kang, M. Hu, X. Wu, H. Meng, Purple-teaming llms with adversarial defender training, arXiv preprint arXiv:2407.01850 (2024).
- [42] F. W. Liu, C. Hu, Exploring vulnerabilities and protections in large language models: A survey, arXiv preprint arXiv:2406.00240 (2024).
- [43] D. Shi, T. Shen, Y. Huang, Z. Li, Y. Leng, R. Jin, C. Liu, X. Wu, Z. Guo, L. Yu, L. Shi, B. Jiang, D. Xiong, Large language model safety: A holistic survey, arXiv preprint arXiv:2412.17686 (2024).
- [44] M. Q. Li, B. C. M. Fung, Security concerns for large language models: A survey, Journal of Information Security and Applications 95 (2025) 104284. ArXiv:2505.18889.
- [45] L. Lu, H. Yan, Z. Yuan, J. Shi, W. Wei, P.-Y. Chen, P. Zhou, AutoJailbreak: Exploring jailbreak attacks and defenses through a dependency lens, arXiv preprint arXiv:2406.03805 (2024).
- [46] X. Huang, K. Xiu, T. Zheng, C. Zeng, W. Ni, Z. Qin, K. Ren, C. Chen, DualBreach: Efficient dual-jailbreaking via target-driven initialization and multi-target optimization, arXiv preprint arXiv:2504.18564 (2025).
- [47] S. Ghosh, P. Varshney, E. Galinkin, C. Parisien, AEGIS: Online adaptive AI content safety moderation with ensemble of LLM experts, arXiv preprint arXiv:2404.05993 (2024).
- [48] M. Kang, B. Li, R<sup>2</sup>-Guard: Robust reasoning enabled LLM guardrail via knowledge-enhanced logical reasoning, in: The Thirteenth International Conference on Learning Representations, 2025. URL: <https://openreview.net/forum?id=CkgKSqZbuC>.
- [49] A. Verma, S. Krishna, S. Gehrmann, M. Seshadri, A. Pradhan, T. Ault, L. Barrett, D. Rabinowitz, J. Doucette, N. Phan, Operationalizing a threat model for red-teaming large language models (LLMs), Transactions on Machine Learning Research (2025).
- [50] O. Sagi, L. Rokach, Ensemble learning: A survey, WIREs Data Mining and Knowledge Discovery 8 (2018) e1249.
- [51] MITRE Corporation, MITRE ATLAS: Adversarial threat landscape for artificial-intelligence systems, <https://atlas.mitre.org>, ??? Accessed 2026.
- [52] OWASP Foundation, OWASP top 10 for large language model applications, <https://owasp.org/www-project-top-10-for-large-language-model-applications/>, ??? Accessed 2026.
- [53] A. Wei, N. Haghtalab, J. Steinhardt, Jailbroken: How does llm safety training fail?, Advances in Neural Information Processing Systems 36 (2024).
- [54] X. Guo, H. Yu, On the domain adaptation and generalization of pretrained language models: A survey, arXiv preprint arXiv:2211.03154 (2022).
- [55] Y. Dong, R. Mu, Y. Zhang, S. Sun, T. Zhang, C. Wu, G. Jin, Y. Qi, J. Hu, J. Meng, et al., Safeguarding large language models: A survey, Artificial Intelligence Review 58 (2025). ArXiv:2406.02622.
- [56] J. Ebrahimi, A. Rao, D. Lowd, D. Dou, Hotflip: White-box adversarial examples for text classification, arXiv preprint arXiv:1712.06751 (2017).
- [57] M. Fredrikson, S. Jha, T. Ristenpart, Model inversion attacks that exploit confidence information and basic countermeasures, in: Proceedings of the 22nd ACM SIGSAC conference on computer and communications security, 2015, pp. 1322–1333.
- [58] Y. Zhang, R. Jia, H. Pei, W. Wang, B. Li, D. Song, The secret revealer: Generative model-inversion attacks against deep neural networks, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 253–261.
- [59] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, J. Zhai, W. Wang, X. Zhang, Trojaning attack on neural networks, in: Proceedings of the 25th Annual Network and Distributed System Security Symposium (NDSS), 2018.
- [60] T. Gu, B. Dolan-Gavitt, S. Garg, Badnets: Identifying vulnerabilities in the machine learning model supply chain, arXiv preprint arXiv:1708.06733 (2017).
- [61] J. Steinhardt, P. W. Koh, P. S. Liang, Certified defenses for data poisoning attacks, Advances in neural information processing systems 30 (2017).
- [62] B. Biggio, B. Nelson, P. Laskov, Poisoning attacks against support vector machines, in: Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012, icml.cc / Omnipress, 2012. ArXiv:1206.6389.
- [63] F. Tramèr, D. Boneh, Differentially private learning needs better features (or much more data), in: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021, OpenReview.net, 2021. ArXiv:2011.11660.
- [64] N. Carlini, F. Tramèr, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, U. Erlingsson, et al., Extracting training data from large language models, in: 30th USENIX Security Symposium (USENIX Security 21), 2021, pp. 2633–2650.
- [65] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, T. Ristenpart, Stealing machine learning models via prediction {APIs}, in: 25th USENIX security symposium (USENIX Security 16), 2016, pp. 601–618.

- [66] T. Liu, Z. Deng, G. Meng, Y. Li, K. Chen, Demystifying rce vulnerabilities in llm-integrated apps, in: Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, 2024, pp. 1716–1730. doi:10.1145/3658644.3690338, arXiv:2309.02926.
- [67] E. Strubell, A. Ganesh, A. McCallum, Energy and policy considerations for modern deep learning research, in: Proceedings of the AAAI conference on artificial intelligence, volume 34, 2020, pp. 13693–13696.
- [68] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, K. Keutzer, A survey of quantization methods for efficient neural network inference, in: Low-Power Computer Vision, Chapman and Hall/CRC, 2022, pp. 291–326.
- [69] E. Debenedetti, I. Shumailov, T. Fan, J. Hayes, N. Carlini, D. Fabian, C. Kern, C. Shi, A. Terzis, F. Tramèr, Defeating prompt injections by design, arXiv preprint arXiv:2503.18813 (2025).
- [70] N. Jain, A. Schwarzschild, Y. Wen, G. Somepalli, J. Kirchenbauer, P.-y. Chiang, M. Goldblum, A. Saha, J. Geiping, T. Goldstein, Baseline defenses for adversarial attacks against aligned language models, arXiv preprint arXiv:2309.00614 (2023).
- [71] J. Kim, A. Derakhshan, I. G. Harris, Robust safety classifier for large language models: Adversarial prompt shield, arXiv preprint arXiv:2311.00172 (2023).
- [72] T. Rebedea, R. Dinu, M. N. Sreedhar, C. Parisien, J. Cohen, NeMo guardrails: A toolkit for controllable and safe LLM applications with programmable rails, in: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Association for Computational Linguistics, Singapore, 2023, pp. 431–445. doi:10.18653/v1/2023.emnlp-demo.40.
- [73] Y. Dong, R. Mu, G. Jin, Y. Qi, J. Hu, X. Zhao, J. Meng, W. Ruan, X. Huang, Building guardrails for large language models, 2024. URL: <https://arxiv.org/abs/2402.01822>. arXiv:2402.01822.
- [74] S. Han, K. Rao, A. Ettinger, L. Jiang, B. Y. Lin, N. Lambert, Y. Choi, N. Dziri, WildGuard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of LLMs, in: Advances in Neural Information Processing Systems 37, 2024, pp. 8093–8131. doi:10.52202/079017-0261, arXiv:2406.18495.
- [75] W. Zeng, Y. Liu, R. Mullins, L. Peran, J. Fernandez, H. Harkous, K. Narasimhan, D. Proud, P. Kumar, B. Radharapu, et al., ShieldGemma: Generative AI content moderation based on Gemma, arXiv preprint arXiv:2407.21772 (2024).
- [76] T. Markov, C. Zhang, S. Agarwal, F. E. Nekoul, T. Lee, S. Adler, A. Jiang, L. Weng, A holistic approach to undesired content detection in the real world, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 37, 2023, pp. 15009–15018.
- [77] M. Sharma, M. Tong, J. Mu, J. Wei, J. Kruthoff, S. Goodfriend, E. Ong, A. Peng, R. Agarwal, C. Anil, et al., Constitutional classifiers: Defending against universal jailbreaks across thousands of hours of red teaming, arXiv preprint arXiv:2501.18837 (2025).
- [78] H. Cunningham, et al., Constitutional classifiers++: Efficient production-grade defenses against universal jailbreaks, arXiv preprint arXiv:2601.04603 (2026).
- [79] M. Phute, A. Helbling, M. Hull, S. Peng, S. Szyller, C. Cornelius, D. H. Chau, LLM self defense: By self examination, llms know they are being tricked, in: The Second Tiny Papers Track at ICLR 2024, Tiny Papers @ ICLR 2024, Vienna, Austria, May 11, 2024, OpenReview.net, 2024. ArXiv:2308.07308.
- [80] Y. Zhang, L. Ding, L. Zhang, D. Tao, Intention analysis makes llms A good jailbreak defender, in: Proceedings of the 31st International Conference on Computational Linguistics, COLING 2025, Abu Dhabi, UAE, January 19-24, 2025, Association for Computational Linguistics, 2025, pp. 2947–2968. ArXiv:2401.06561.
- [81] Y. Wang, Z. Shi, A. Bai, C.-J. Hsieh, Defending llms against jailbreaking attacks via backtranslation, in: Findings of the Association for Computational Linguistics ACL 2024, 2024, pp. 16031–16046. doi:10.18653/v1/2024.findings-acl.948, arXiv:2402.16459.
- [82] Y. Wang, Y. Wu, Z. Wei, S. Jegelka, Y. Wang, A theoretical understanding of self-correction through in-context alignment, in: Advances in Neural Information Processing Systems 37, 2024, pp. 89869–89912. doi:10.52202/079017-2854, arXiv:2405.18634.
- [83] H. Brown, L. Lin, K. Kawaguchi, M. Shieh, Self-evaluation as a defense against adversarial attacks on llms, arXiv preprint arXiv:2407.03234 (2024).
- [84] Y. Zhou, Y. Han, H. Zhuang, K. Guo, Z. Liang, H. Bao, X. Zhang, Defending jailbreak prompts via in-context adversarial game, in: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, 2024, pp. 20084–20105. doi:10.18653/v1/2024.emnlp-main.1121, arXiv:2402.13148.
- [85] J. Cui, W. Chiang, I. Stoica, C. Hsieh, Or-bench: An over-refusal benchmark for large language models, in: Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025, volume 267 of *Proceedings of Machine Learning Research*, PMLR / OpenReview.net, 2025. ArXiv:2405.20947.
- [86] Z. Xu, F. Jiang, L. Niu, J. Jia, B. Y. Lin, R. Poovendran, SafeDecoding: Defending against jailbreak attacks via safety-aware decoding, in: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2024, pp. 5587–5605.
- [87] S. Dathathri, A. Madotto, J. Lan, J. Hung, E. Frank, P. Molino, J. Yosinski, R. Liu, Plug and play language models: A simple approach to controlled text generation, in: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020, OpenReview.net, 2020. ArXiv:1912.02164.
- [88] A. Liu, M. Sap, X. Lu, S. Swayamdipta, C. Bhagavatula, N. A. Smith, Y. Choi, Dexperts: Decoding-time controlled text generation with experts and anti-experts, in: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), 2021, pp. 6691–6706. doi:10.18653/v1/2021.acl-long.522, arXiv:2105.03023.
- [89] F. Bianchi, M. Suzgun, G. Attanasio, P. Röttger, D. Jurafsky, T. Hashimoto, J. Zou, Safety-tuned llamas: Lessons from improving the safety of large language models that follow instructions, arXiv preprint arXiv:2309.07875 (2023).
- [90] Z. Sun, Y. Shen, Q. Zhou, H. Zhang, Z. Chen, D. Cox, Y. Yang, C. Gan, Principle-driven self-alignment of language models from scratch with minimal human supervision, Advances in Neural Information Processing Systems 36 (2024).
- [91] S. Peng, P.-Y. Chen, M. Hull, D. H. Chau, Navigating the safety landscape: Measuring risks in finetuning large language models, in: Advances in Neural Information Processing Systems 37, 2024, pp. 95692–95715. doi:10.52202/079017-3032, arXiv:2405.17374.

- [92] T. Chakraborty, E. Shayegani, Z. Cai, N. Abu-Ghazaleh, M. S. Asif, Y. Dong, A. K. Roy-Chowdhury, C. Song, Cross-modal safety alignment: Is textual unlearning all you need?, 2024. URL: <https://arxiv.org/abs/2406.02575>. arXiv:2406.02575.
- [93] Y. Du, S. Zhao, D. Zhao, M. Ma, Y. Chen, L. Huo, Q. Yang, D. Xu, B. Qin, Mogu: A framework for enhancing safety of open-sourced llms while preserving their usability, arXiv preprint arXiv:2405.14488 (2024).
- [94] D. Rosati, J. Wehner, K. Williams, Ł. Bartoszcze, J. Batzner, H. Sajjad, F. Rudzicz, Immunization against harmful fine-tuning attacks, in: Findings of the Association for Computational Linguistics: EMNLP 2024, 2024, pp. 5234–5247. doi:10.18653/v1/2024.findings-emnlp.301, arXiv:2402.16382.
- [95] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, C. Finn, Direct preference optimization: Your language model is secretly a reward model, Advances in Neural Information Processing Systems 36 (2024).
- [96] Y. Mo, Y. Wang, Z. Wei, Y. Wang, Studious bob fight back against jailbreaking via prompt adversarial tuning, in: Advances in Neural Information Processing Systems 37, 2024, pp. 64242–64272. doi:10.52202/079017-2049, arXiv:2402.06255.
- [97] S. Xhonneux, A. Sordoni, S. Günnemann, G. Gidel, L. Schwinn, Efficient adversarial training in llms with continuous attacks, in: Advances in Neural Information Processing Systems 37, 2024, pp. 1502–1530. doi:10.52202/079017-0048, arXiv:2405.15589.
- [98] K. Hines, G. Lopez, M. Hall, F. Zarfati, Y. Zunger, E. Kiciman, Defending against indirect prompt injection attacks with spotlighting, in: Proceedings of the Conference on Applied Machine Learning in Information Security (CAMLIS), CEUR Workshop Proceedings Vol. 3920, 2024, pp. 48–62. ArXiv:2403.14720.
- [99] S. Chen, J. Piet, C. Sitawarin, D. Wagner, StruQ: Defending against prompt injection with structured queries, in: USENIX Security Symposium, 2025. ArXiv:2402.06363.
- [100] A. Robey, E. Wong, H. Hassani, G. J. Pappas, Smoothllm: Defending large language models against jailbreaking attacks, Trans. Mach. Learn. Res. 2025 (2025). ArXiv:2310.03684.
- [101] J. Ji, B. Hou, Z. Zhang, G. Zhang, W. Fan, Q. Li, Y. Zhang, G. Liu, S. Liu, S. Chang, Advancing the robustness of large language models through self-denoised smoothing, in: Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers), 2024, pp. 246–257. doi:10.18653/v1/2024.naacl-short.23, arXiv:2404.12274.
- [102] A. Kumar, C. Agarwal, S. Srinivas, A. J. Li, S. Feizi, H. Lakkaraju, Certifying LLM safety against adversarial prompting, arXiv preprint arXiv:2309.02705 (2023).
- [103] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, A. Vladu, Towards deep learning models resistant to adversarial attacks, in: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings, OpenReview.net, 2018. ArXiv:1706.06083.
- [104] F. Tramèr, A. Kurakin, N. Papernot, I. J. Goodfellow, D. Boneh, P. D. McDaniel, Ensemble adversarial training: Attacks and defenses, in: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings, OpenReview.net, 2018. ArXiv:1705.07204.
- [105] C. Xie, Y. Wu, L. v. d. Maaten, A. L. Yuille, K. He, Feature denoising for improving adversarial robustness, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 501–509.
- [106] Y. Zhou, X. Zheng, C.-J. Hsieh, K.-W. Chang, X. Huang, Defense against synonym substitution-based adversarial attacks via dirichlet neighborhood ensemble, in: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), 2021, pp. 5482–5492. doi:10.18653/v1/2021.acl-long.426, arXiv:2006.11627.
- [107] C. Zhu, Y. Cheng, Z. Gan, S. Sun, T. Goldstein, J. Liu, FreeLb: Enhanced adversarial training for natural language understanding, in: 8th International Conference on Learning Representations, ICLR 2020, 2020. ArXiv:1909.11764.
- [108] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. El Ghaoui, M. Jordan, Theoretically principled trade-off between robustness and accuracy, in: International conference on machine learning, PMLR, 2019, pp. 7472–7482.
- [109] E. Wong, Z. Kolter, Provable defenses against adversarial examples via the convex outer adversarial polytope, in: International conference on machine learning, PMLR, 2018, pp. 5286–5295.
- [110] A. Sinha, H. Namkoong, R. Volpi, J. Duchi, Certifying some distributional robustness with principled adversarial training, arXiv preprint arXiv:1710.10571 (2017).
- [111] Y. Wu, F. Roesner, T. Kohno, N. Zhang, U. Iqbal, Isolategpt: An execution isolation architecture for LLM-based systems, in: Network and Distributed System Security Symposium (NDSS), 2025.
- [112] F. Wu, E. Cecchetti, C. Xiao, System-level defense against indirect prompt injection attacks: An information flow control perspective, arXiv preprint arXiv:2409.19091 (2024).
- [113] L. Beurer-Kellner, B. Buesser, A.-M. Crețu, E. DeBenedetti, D. Dobos, D. Fabian, M. Fischer, D. Froelicher, K. Grosse, D. Naeff, E. Ozoani, A. Pavard, F. Tramèr, V. Volhejn, Design patterns for securing LLM agents against prompt injections, arXiv preprint arXiv:2506.08837 (2025).
- [114] V. Patil, P. Hase, M. Bansal, Can sensitive information be deleted from llms? objectives for defending against extraction attacks, in: The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024, OpenReview.net, 2024. ArXiv:2309.17410.
- [115] Z. Pan, J. Liu, Y. Dai, W. Fan, Large language model-enabled vulnerability investigation: A review, in: Proceedings of the 3rd International Conference on Intelligent Computing and Next Generation Networks (ICNGN), IEEE, 2024, pp. 1–5. doi:10.1109/ICNGN63705.2024.10871716.
- [116] Y. Zou, Y. Hong, J. Xu, L. Liu, W. Fan, Leveraging large language models for challenge solving in capture-the-flag, in: Proceedings of the 23rd IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), IEEE, 2024, pp. 1541–1550. doi:10.1109/TrustCom63139.2024.00213.

- [117] Y. Zou, J. Liu, W. Fan, CTFAgent: An LLM-powered agent for CTF challenge solving, *Journal of Information Security and Applications* 96 (2025) 104305.
- [118] T. Han, A. Kumar, C. Agarwal, H. Lakkaraju, Medsafetybench: Evaluating and improving the medical safety of large language models, in: *Advances in Neural Information Processing Systems* 37, 2024, pp. 33423–33454. doi:10.52202/079017-1054, arXiv:2403.03744.
- [119] A. Esmradi, D. W. Yip, C. F. Chan, A comprehensive survey of attack techniques, implementation, and mitigation strategies in large language models, in: *International Conference on Ubiquitous Security*, Springer, 2023, pp. 76–95.
- [120] Z. Zhang, L. Lei, L. Wu, R. Sun, Y. Huang, C. Long, X. Liu, X. Lei, J. Tang, M. Huang, SafetyBench: Evaluating the safety of large language models, in: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024, pp. 15537–15553. doi:10.18653/v1/2024.acl-long.830, arXiv:2309.07045.
- [121] S. Gehman, S. Gururangan, M. Sap, Y. Choi, N. A. Smith, Realtotoxicityprompts: Evaluating neural toxic degeneration in language models, in: *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020, pp. 3356–3369. doi:10.18653/v1/2020.findings-emnlp.301, arXiv:2009.11462.
- [122] M. Mazeika, L. Phan, X. Yin, A. Zou, Z. Wang, N. Mu, E. Sakhaee, N. Li, S. Basart, B. Li, et al., HarmBench: A standardized evaluation framework for automated red teaming and robust refusal, in: *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024. ArXiv:2402.04249.
- [123] P. Chao, E. Debenedetti, A. Robey, M. Andriushchenko, F. Croce, et al., JailbreakBench: An open robustness benchmark for jailbreaking large language models, in: *Advances in Neural Information Processing Systems* 37, 2024, pp. 55005–55029. doi:10.52202/079017-1745, arXiv:2404.01318.
- [124] E. Debenedetti, J. Zhang, M. Balunović, L. Beurer-Kellner, M. Fischer, F. Tramèr, AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents, in: *Advances in Neural Information Processing Systems (NeurIPS)*, Datasets and Benchmarks Track, 2024.
- [125] L. I. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms*, 2nd ed., John Wiley & Sons, 2014. doi:10.1002/9781118914564.
- [126] B. W. Matthews, Comparison of the predicted and observed secondary structure of T4 phage lysozyme, *Biochimica et Biophysica Acta (BBA) – Protein Structure* 405 (1975) 442–451.
- [127] Y. Benjamini, Y. Hochberg, Controlling the false discovery rate: a practical and powerful approach to multiple testing, *Journal of the Royal Statistical Society: Series B (Methodological)* 57 (1995) 289–300.
- [128] X. Liu, N. Xu, M. Chen, C. Xiao, AutoDAN: Generating stealthy jailbreak prompts on aligned large language models, in: *The Twelfth International Conference on Learning Representations (ICLR)*, 2024. ArXiv:2310.04451.
- [129] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. E. Gonzalez, I. Stoica, Judging LLM-as-a-judge with MT-Bench and Chatbot Arena, *Advances in Neural Information Processing Systems (NeurIPS)* 36 (2023). Vicuna model card; arXiv:2306.05685.
- [130] N. Mantel, W. Haenszel, Statistical aspects of the analysis of data from retrospective studies of disease, *Journal of the National Cancer Institute* 22 (1959) 719–748.
- [131] H. Touvron, L. Martin, K. Stone, et al., Llama 2: Open foundation and fine-tuned chat models, arXiv preprint arXiv:2307.09288 (2023).
- [132] DoorDash Engineering, Building a RAG-based LLM support system: Lessons from production deployment, DoorDash Engineering Blog, 2024. Accessed: 2024-12-01.
- [133] Airbnb Engineering, Upgrading conversational AI: A hybrid approach to LLM integration, Airbnb Tech Blog, 2024. Accessed: 2024-12-01.
- [134] S. Asthana, B. Zhang, R. Mahindru, C. DeLuca, A. L. Gentile, S. Gopisetty, Deploying privacy guardrails for LLMs: A comparative analysis of real-world applications, arXiv preprint arXiv:2501.12456 (2025).
- [135] S. Xu, W. Dong, Z. Guo, X. Wu, D. Xiong, Exploring multilingual human value concepts in large language models: Is value alignment consistent, transferable and controllable across languages?, in: *Findings of the Association for Computational Linguistics: EMNLP 2024*, 2024, pp. 1771–1793. doi:10.18653/v1/2024.findings-emnlp.96, arXiv:2402.18120.
- [136] R. Pankajakshan, S. Biswal, Y. Govindarajulu, G. Gressel, Mapping llm security landscapes: A comprehensive stakeholder risk assessment proposal, arXiv preprint arXiv:2403.13309 (2024).
- [137] Y. Zhao, T. Pang, C. Du, X. Yang, C. Li, N.-M. M. Cheung, M. Lin, On evaluating adversarial robustness of large vision-language models, *Advances in Neural Information Processing Systems* 36 (2024).
- [138] S. Tedeschi, F. Friedrich, P. Schramowski, K. Kersting, R. Navigli, H. Nguyen, B. Li, Alert: A comprehensive benchmark for assessing large language models’ safety through red teaming, arXiv preprint arXiv:2404.08676 (2024).
- [139] R. Bhardwaj, S. Poria, Language model unalignment: Parametric red-teaming to expose hidden harms and biases, arXiv preprint arXiv:2310.14303 (2023).
- [140] Y. Bai, S. Kadavath, S. Kundu, A. Askell, J. Kernion, A. Jones, A. Chen, A. Goldie, A. Mirhoseini, C. McKinnon, et al., Constitutional AI: Harmlessness from AI feedback, arXiv preprint arXiv:2212.08073 (2022).
- [141] T. Hartvigsen, S. Gabriel, H. Palangi, M. Sap, D. Ray, E. Kamar, ToxiGen: A large-scale machine-generated dataset for adversarial and implicit hate speech detection, in: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2022, pp. 3309–3326. doi:10.18653/v1/2022.acl-long.234, arXiv:2203.09509.
- [142] N. Nangia, C. Vania, R. Bhalerao, S. R. Bowman, Crows-pairs: A challenge dataset for measuring social biases in masked language models, in: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, 2020, pp. 1953–1967.
- [143] M. Nadeem, A. Bethke, S. Reddy, Stereoset: Measuring stereotypical bias in pretrained language models, in: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, 2021, pp. 5356–5371.
- [144] Y. Huang, Q. Zhang, P. S. Yu, L. Sun, TrustGPT: A benchmark for trustworthy and responsible large language models, arXiv preprint arXiv:2306.11507 (2023).

- [145] D. Ganguli, L. Lovitt, J. Kernion, A. Askell, Y. Bai, S. Kadavath, B. Mann, E. Perez, N. Schiefer, K. Ndousse, et al., Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned, arXiv preprint arXiv:2209.07858 (2022).
- [146] J. Ji, M. Liu, J. Dai, X. Pan, C. Zhang, C. Bian, C. Zhang, R. Sun, Y. Wang, Y. Yang, BeaverTails: Towards improved safety alignment of LLM via a human-preference dataset, in: Advances in Neural Information Processing Systems 36, 2023, pp. 24678–24704. doi:10.52202/075280-1072, arXiv:2307.04657.
- [147] B. Wang, C. Xu, S. Wang, Z. Gan, Y. Cheng, J. Gao, A. H. Awadallah, B. Li, Adversarial GLUE: A multi-task benchmark for robustness evaluation of language models, in: Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual, 2021. ArXiv:2111.02840.
- [148] K. Zhu, J. Wang, J. Zhou, Z. Wang, H. Chen, Y. Wang, L. Yang, W. Ye, Y. Zhang, N. Z. Gong, X. Xie, PromptRobust: Towards evaluating the robustness of large language models on adversarial prompts, in: Proceedings of the 1st ACM Workshop on Large AI Systems and Models with Privacy and Safety Analysis, 2023, pp. 57–68. doi:10.1145/3689217.3690621, arXiv:2306.04528.
- [149] L. Yang, S. Zhang, L. Qin, Y. Li, Y. Wang, H. Liu, J. Wang, X. Xie, Y. Zhang, GLUE-X: Evaluating natural language understanding models from an out-of-distribution generalization perspective, in: Findings of the Association for Computational Linguistics: ACL 2023, 2023.
- [150] L. Yuan, Y. Chen, G. Cui, H. Gao, F. Zou, X. Cheng, H. Ji, Z. Liu, M. Sun, Revisiting out-of-distribution robustness in NLP: Benchmark, analysis, and LLMs evaluations, in: Advances in Neural Information Processing Systems (NeurIPS), 2023. BOSS benchmark; arXiv:2306.04618.
- [151] S. Lin, J. Hilton, O. Evans, Truthfulqa: Measuring how models mimic human falsehoods, in: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2022, pp. 3214–3252. doi:10.18653/v1/2022.acl-long.229, arXiv:2109.07958.