Adaptive and interpretable two-sample test as a batch-level tool for AI safety

Gaia Grosso [1 2 3] **Arnau Marin-Llobet** [3] **Demba Ba** [3]

Abstract

Most safety defenses for large language models — input and output classifiers, refusal probes, per-query OOD detectors — inspect prompts and completions one at a time. Recent work shows that black-box iterative jailbreaks evade even the strongest of these (Davies et al., 2026), arguing that *"effective defence requires supplementing single-interaction methods with batch-level monitoring."* This suggests treating incoming prompts as a collection and asking whether their distribution deviates from benign traffic — a non-trivial problem, as attack queries are heavily diluted, new attacks circumvent fixed rules, and detection claims must be calibrated. We propose batch-level monitoring based on a calibrated, adaptive two-sample test applied to a model's hidden states, requiring only an attack-free reference. The test localizes the attack signal in activation space, making the geometry of the detected shift directly inspectable and providing an empirical basis for mechanistic follow-up. On `llama3-jailbreaks`, it detects attacks significantly better than standard two-sample baselines, and reveals that jailbreaks organize into two near-orthogonal groups — *scaffolded* versus *direct* prompts — visible in benign data too, suggesting that the input geometry relevant to safety is richer than a single refusal direction can capture.

1. Introduction

Mechanistic interpretability has produced a rich picture of how safety concepts are encoded in LLM activations: linear probes recover features such as truthfulness and refusal (Burns et al., 2022; Marks & Tegmark, 2023; Zou et al., 2023a; Arditi et al., 2024), sparse autoencoders decompose activations into monosemantic components (Bricken et al., 2023; Templeton, 2024), and simple probes can detect adversarial behaviors such as sleeper agents (MacDiarmid et al., 2024). A recurring theme is that single, fixed directions in activation space carry surprisingly clean behavioral signal — including a "refusal direction" whose ablation removes safety behavior on harmful prompts (Arditi et al., 2024).

This view has structural limits. Bailey et al. (2024) construct *obfuscated activations* that bypass linear probes while still eliciting harmful generations: the attack lives along directions the probe was never trained on. Davies et al. (2026) show that a fully black-box attacker can construct universal jailbreaks against state-of-the-art classifiers, noting that their attack "*is difficult to defend against in individual interactions*" but "*incurs many flags during optimization, suggesting that effective defence requires supplementing single-interaction methods with batch-level monitoring.*" An interpretability tool that reads activations along a single pre-committed direction is structurally blind to attacks that route around it. The natural alternative is to ask not where in activation space a fixed concept lives, but *whether the distribution of incoming activations differs from a benign reference at all* — a two-sample testing question that yields a test statistic, a calibrated p-value, and an explicit geometric description of the detected shift.

This paper. We instantiate this view with the Neyman–Pearson (NP) two-sample test (d'Agnolo & Wulzer, 2019; d'Agnolo et al., 2021; Grosso et al., 2024), run with SparKer (Grosso et al., 2025) — a sparse ensemble of Gaussian kernels that locally models the log-ratio between the probability density of the incoming activation stream and that of the benign reference distribution. The test is weakly-supervised (it needs an attack-free reference but no labeled attacks), calibrated (controlled false-positive rate), and adaptive (the discriminating pattern is rebuilt per window). The fitted kernels do double duty: together, they define a test statistic that aggregates evidence into a calibrated p-value, while each kernel localizes a region of activation space where the test stream is over-represented, making the geometry of the detected shift inspectable.

[1]NSF AI Institute for Artificial Intelligence and Fundamental Interactions, Cambridge, MA 02139 [2]Laboratory for Nuclear Science, Massachusetts Institute of Technology, Cambridge, MA 02139 [3]School of Engineering and Applied Sciences, Harvard University, Cambridge, MA 02138. Correspondence to: Gaia Grosso <gaiag795@mit.edu>.

Mechanistic Interpretability Workshop at the 43rd International Conference on Machine Learning , Seoul, South Korea, 2026.
Copyright 2026 by the author(s).

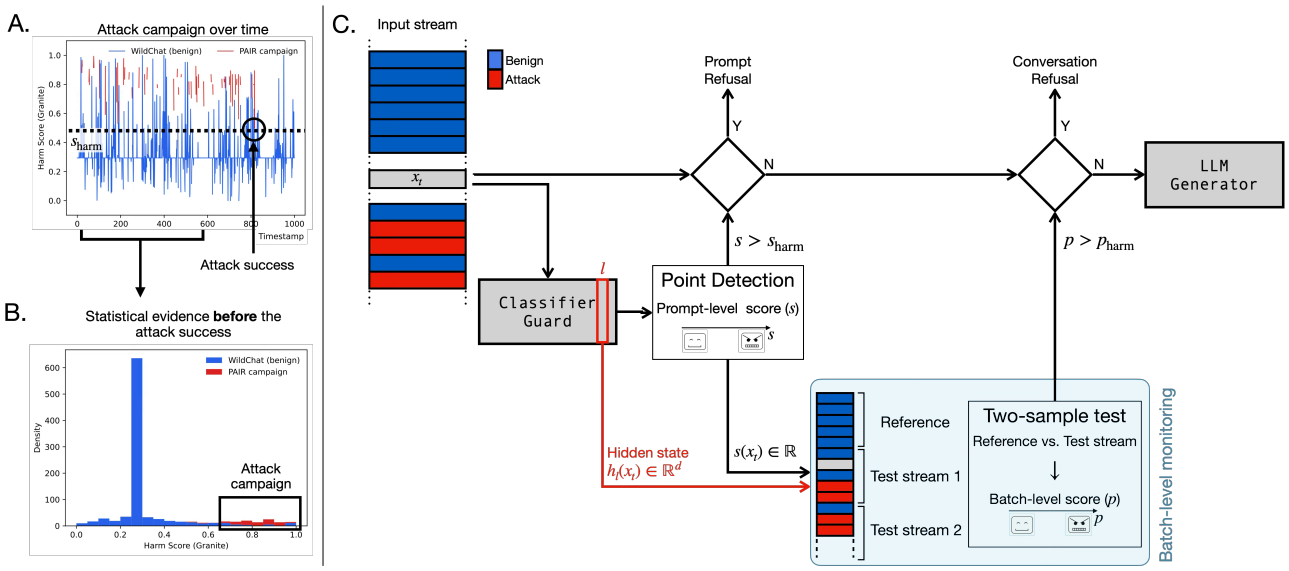

Figure 1. **Attack campaigns leave a statistical signature. Can we detect it *before* the attack succeeds?** Panel A: prompt-level harm score produced by an LLM Guard when a multi-turn attack campaign (red) is hidden among benign users (blue). Panel B: probability distribution of the harm score in the time window preceding the attack success. Panel C: supplementing single-interaction methods with batch-level monitoring. A two-sample test on sliding windows of incoming activations tells us when a multi-turn attack campaign is ongoing.

Contributions.

- **A detection method whose output is not just the detection but also a geometric characterization of the attack in activation space.** We introduce batch-level two-sample testing on hidden-state activations as a calibrated, weakly-supervised, adaptive complement to fixed-direction probes. Instantiated with the Neyman–Pearson test and SPARKER, discriminating features are fit per window from the incoming distribution, and each fitted kernel localizes a region of activation space where the test stream is over-represented — making the geometry of the detected shift directly inspectable.
- **A representational finding about Llama-3-8B-Instruct.** Applying the method to jailbreak activations, we find attacks organize into *two near-orthogonal subspaces* along a structural axis — *scaffolded* prompts (AutoDAN, many-shot, human multi-turn, prefill) versus *direct* prompts (direct-harmful, GCG, best-of-N, PAIR) — rather than a single refusal direction. The axis appears in benign WildChat data too, indicating it reflects how the model represents conversational structure in general, not a safety-specific feature. As a corollary, the benign sample in `llama3-jailbreaks` under-represents the scaffolded cluster, biasing supervised probes trained on it toward a single subspace.
- **Quantitative and operational validation.** On `llama3-jailbreaks` (Bailey et al., 2024), the NP test reaches detection power ≥ 0.95 at contamination rates 2–10$\times$ lower than MMD, MMDAgg, and classifier two-sample baselines. On live PAIR, TAP, Best-of-N, and Crescendo campaigns against Granite Guardian 3.1-8B (Padhi et al., 2024), it detects 53%–98% of campaigns that bypass the guardian *before* the breach, using only the guardian's residual stream — substantially better than the same test on its scalar harm-score output, indicating the classifier's internal harm representation carries information its surface output discards.

2. Related work.

Two-sample testing. Most two-sample tests target global distributional shifts affecting all incoming data. The Neyman–Pearson (NP) test was developed in the signal-agnostic new-physics search literature (d'Agnolo & Wulzer, 2019; d'Agnolo et al., 2021; Grosso et al., 2024) for the opposite regime: detecting deviations that affect only a small fraction of the data, on the order of a percent or less — a regime equally relevant to AI safety. Recent work by Ribero et al. (2026) corroborates this picture on a similar low-signal benchmark: regularized KL-based tests — the same family as the NP test — substantially outperform MMD when the density ratio is small, while MMD becomes competitive only as the signal grows. We compare against MMD with characteristic kernels (Gretton et al., 2012), MMDAgg (Schrab et al., 2023), and classifier two-sample tests (Lopez-Paz & Oquab, 2017).

Activation-space probes. Activation-space probes for refusal, truthfulness, and harmful intent (Zou et al., 2023a; Arditi et al., 2024; Marks & Tegmark, 2023; MacDiarmid et al., 2024) read fixed directions; sparse autoencoders (Bricken et al., 2023; Templeton, 2024) use a fixed feature dictionary learned offline. Our method is complementary: discriminating features are fit per window from the incoming distribution.

Stateful and aggregate-level monitoring. Stateful black-box defenses monitor query sequences via input-space self-similarity (Chen et al., 2020; Li et al., 2022; Juuti et al., 2019), which boundary-point attacks (Davies et al., 2026) are designed to defeat. Aggregate-level monitors include CLIO (Tamkin et al., 2024), which clusters conversation summaries; Yueh-Han et al. (2025), who use a sequential LLM judge for decomposition attacks; and Wen et al. (2025), who formalize distributed harm. These operate on summaries or per-user sequences and are largely descriptive. Ours is the weakly-supervised, distribution-level, activation-space analogue with calibrated false-positive control.

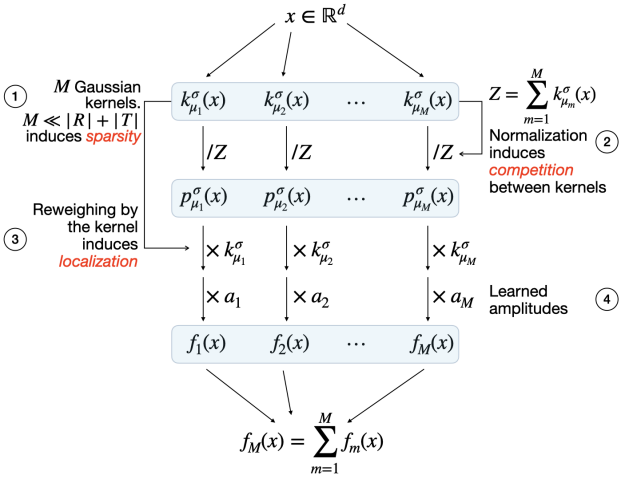

Figure 2. **SPARKER forward pass.** An input activation vector x is evaluated against M Gaussian kernels with learnable centroids $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_M)$ and per-kernel covariance scale factors σ annealed over training. $M \ll |\mathcal{R}| + |\mathcal{T}|$ induces *sparsity*. Kernel responses are normalised by their sum to induce *competition*, then element-wise reweighted by the same kernel to induce *locality* and combined via a dot product with learned amplitude vector $\boldsymbol{a} = (a_1, \ldots, a_M)$. The result, $f_M(x)$, is a scalar estimate of the log-density ratio $\log[q_T(\mathbf{x})/q_R(\mathbf{x})]$ between the test and reference distributions.

3. Method

LLM batch-level monitoring with two-sample test. The proposed monitoring workflow is shown in Panel C of Figure 1. Consider a deployed stack with one or more safety classifiers ("Classifier Guard") guarding a generator LLM. Let $\mathcal{M}$ denote the model whose activations the defender monitors — the classifier, the generator, or any intermediate model. We choose the Classifier Guard in the figure rather than the generator since black-box iterative attackers optimize against the classifier's pass/fail signal. The attack-induced distributional shift concentrates most sharply in the classifier's internal states, and the classifier is also the component the defender controls directly at deployment time. Let $h_\ell(x) \in \mathbb{R}^d$ denote its residual stream at layer ℓ when processing input x. The defender holds a reference set $\mathcal{R} = \{h_\ell(x_i)\}_{i=1}^n$ from a certified benign operating period, and observes a test stream $\mathcal{T} = \{h_\ell(x'_j)\}_{j=1}^m$ from a live deployment window in which each example is benign with probability $1 - \epsilon$ and drawn from an unknown attack distribution with probability ϵ, with ϵ unknown to the monitor.

Let $q_\mathcal{T}, q_\mathcal{R}$ be the distributions of $\mathcal{T}$ and $\mathcal{R}$. We test $H_0 : q_\mathcal{T} = q_\mathcal{R}$ against $H_1 : q_\mathcal{T} \neq q_\mathcal{R}$ without assumptions on the attack distribution. Operating in activation space makes the test agnostic to surface-level prompt diversity: text strings that look very different can produce overlapping activations, and vice versa.

3.1. Neyman–Pearson Test with SPARKER

The Neyman–Pearson (NP) test is a signal-agnostic two-sample test that quantifies the compatibility of $\mathcal{T}$ with $\mathcal{R}$ via the maximum-likelihood-ratio statistic (d'Agnolo & Wulzer, 2019; Grosso et al., 2024). The alternative is parametrized by a learnable model $f(x) \approx \log[q_\mathcal{T}(x)/q_\mathcal{R}(x)]$ of the log-density ratio, fit by minimising the extended-likelihood NP loss

$$\mathcal{L}_{\text{NP}}[f] = \sum_{x \in \mathcal{R}} w_\mathcal{R} \left(e^{f(x)} - 1\right) - \sum_{x \in \mathcal{T}} f(x). \tag{1}$$

Eq. (1) is an energy-based objective: f is a negative energy, and minimising $\mathcal{L}_{\text{NP}}$ is equivalent to maximum-likelihood fitting of the unnormalised density $q_\mathcal{R}(x)\, e^{f(x)}$ against samples from $q_\mathcal{T}$, with the exponential term acting as a sample-based partition-function estimate (Grosso et al., 2025). The test statistic is $T = -2\,\mathcal{L}_{\text{NP}}[\hat{f}]$, large when $\hat{f}$ identifies a significant departure of $\mathcal{T}$ from $\mathcal{R}$.

SPARKER. We instantiate f with SPARKER (Grosso et al., 2025), a sparse ensemble of M Gaussian kernels designed for sensitivity in the low-anomaly-fraction regime:

$$f^\sigma_M(x) = \boldsymbol{a}^\top \left[\boldsymbol{p}\left[\boldsymbol{k}^\sigma_{\boldsymbol{\mu}}\right](x) \odot \boldsymbol{k}^\sigma_{\boldsymbol{\mu}}(x)\right], \tag{2}$$

where $\boldsymbol{k}^\sigma_{\boldsymbol{\mu}}(x) \in \mathbb{R}^M$ collects the responses of M Gaussian kernels with learnable centroids $\boldsymbol{\mu}$ and diagonal covariance, $\boldsymbol{a} \in \mathbb{R}^M$ are learnable amplitudes, $\odot$ is the Hadamard product, and $\boldsymbol{p}^i = k^\sigma_{\mu_i}(x)/\sum_j k^\sigma_{\mu_j}(x)$ is a SoftMax-style normalisation inducing competition between kernels. Conversely to the isotropic kernels used in Grosso et al. (2025),

in this work each kernel has covariance

$$\Sigma_i = \left[\sigma^2 \cdot \frac{M \cdot s_i^2}{\sum_{j=1}^{M} s_j^2} + \epsilon \right] \mathbb{I}, \tag{3}$$

where σ is a global scale hyperparameter annealed over training and the learnable per-kernel factors s_i (normalised to unit mean) let individual kernels adaptively shrink or expand relative to that scale. Three inductive biases make this architecture well-suited to detecting small attack populations: *locality* (Gaussian kernels respond only in a bounded region and so individually localize the $\mathcal{R}$–$\mathcal{T}$ imbalance), *competition* (SoftMax normalization forces kernels to specialize in distinct regions), and *sparsity* ($M \ll |\mathcal{R}| + |\mathcal{T}|$ prevents memorization of reference fluctuations). Following Grosso et al. (2025), we anneal σ over τ values $\sigma_1 > \cdots > \sigma_\tau$ and aggregate evidence across scales via

$$p = -\frac{1}{2} \min_t \log p_t. \tag{4}$$

The test calibration (e.g.determination of the false positive rate) is performed empirically b y bootstrap resampling from the benign reference pool (see Appendix A.1 for additional details). A schematic view of SparKer forward pass is shown in Figure 2.

3.2. Feature discovery with SparKer

Beyond a p-value, Grosso et al. (2025) showed that SparKer's fitted kernel centroids $\{\boldsymbol{\mu}_i\}_{i=1}^M$ directly identify the regions of activation space (here in PCA coordinates) where $\mathcal{T}$ departs from $\mathcal{R}$. In other terms, a trained SparKer model, if returning a significant p-value, is itself a *feature discovery tool*. Training returns a per-point anomaly score

$$S(x) = \mathrm{Sigmoid}(f(x)) \tag{5}$$

and a geometrically interpretable decomposition into M constituents,

$$f_M(x) = \sum_{m=1}^{M} f_m(x), \tag{6}$$

each paired to a single kernel by locality. The constituent with the highest correlation to $S(x)$ concentrates the most discriminative signal; correlating its centroid with attention-head output subspaces links the statistical detection to specific model circuits, opening a path toward *geometric interpretability* of the detected shift.

4. SparKer NP test in AI safety

The batch-level monitoring approach based on SparKer's NP test has three main benefits: it is *adaptive* (the detection rule is learned directly from the incoming batch), *sensitive* to low contamination regimes (the typical setting in safety problems), and provides a *geometric characterization* of the attack relevant to understanding how the model encodes different attacks. The following controlled experiments demonstrate these properties and compare with baselines.

Data. We use the `llama3-jailbreaks` benchmark (Bailey et al., 2024) ($\sim$29,900 prompts formatted for Llama-3-8B-Instruct), extracting residual-stream activations at layer 16 of Meta-Llama-3-8B-Instruct (4096-dimensional, last 512 tokens) and projecting to $k = 64$ principal components for tractable MMD baselines and faster SparKer fits. We use $n = 10\,000$ benign activations as the reference pool $\mathcal{R}$ and construct test streams of size $m = 2000$ by mixing benign with jailbreak activations at controlled contamination rates $\varepsilon \in \{0.005, 0.01, 0.02, 0.05, 0.1, 0.2\}$, spanning the deployment-relevant low-ε regime where attack queries are heavily diluted by benign traffic. Full preparation details (benign pool, harmful-class splits, stream construction) are in Appendix A.1. Note that this is not a realistic online-monitoring scenario, since the injected signals do not originate from a single attack campaign; a more realistic setting is presented in Section 5.

Test statistic and baselines. We instantiate the NP test with $M = 5$ SparKer kernels, $\tau = 10\,000$ training epochs, and $B = 100$ bootstrap draws for null calibration. We compare against four standard two-sample tests applied to the same PCA-reduced activations: MMD with an RBF kernel (Gretton et al., 2012), MMDAgg (Schrab et al., 2023), and the classifier two-sample test (C2ST) (Lopez-Paz & Oquab, 2017) in linear (logistic regression) and non-linear (2-layer MLP) settings.

Evaluation. Each method's detection power at level $\alpha = 0.05$ is estimated over $R = 20$ independent draws of $\mathcal{T}$ at each ε. We report power as a function of ε, and characterize each method by the smallest ε at which power exceeds 0.95 — the threshold below which a deployment-relevant batch-level monitor cannot be considered effective.

Sensitivity to low contamination across attacks. Figure 3 shows detection power averaged over the nine attack classes. The NP test reaches power 0.95 at $\varepsilon = 1\%$, compared to 2% for MMDAgg, 5% for MMD, and 10% for C2ST — a 2–10$\times$ improvement in the contamination rate at which detection becomes reliable. At higher contamination ($\varepsilon \gtrsim 10\%$) all methods saturate at full power; the NP test's advantage is concentrated in the operationally most relevant low-ε regime. The per-class breakdown is in Figure 7, Appendix A.2. SparKer's sparsity and locality buy low-contamination sensitivity.

4.1. Inspectable attack geometry

We now turn to the more conceptual question: *why is distribution-level testing necessary at all, rather than a*

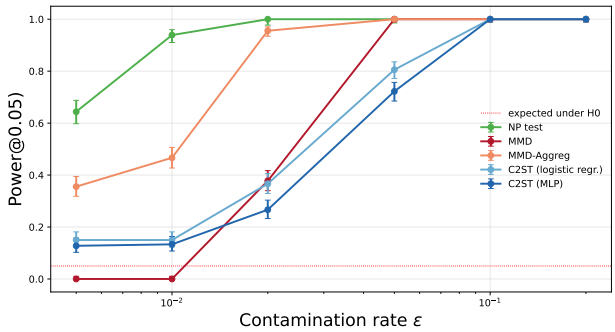

Figure 3. **Detection power at low contamination with NP test.** Detection power at 5% FPR on `llama3-jailbreaks` as a function of contamination rate ε, averaged over attack classes. The NP test (SPARKER) maintains higher than 95% power down to $\varepsilon \approx 1\%$, while all four standard two-sample tests require $\varepsilon \gtrsim$ 2–10% to reach the same power. Bands show ± 1 SD over the average.

sufficiently well-trained probe? The answer is structural and geometric.

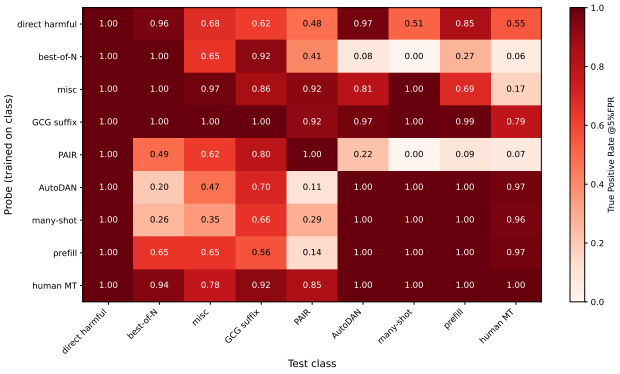

Figure 4. **Flaws of supervised probes in OOD generalization.** TPR of supervised probes at 5% FPR. Each row corresponds to a logistic regression probe trained on one attack class; each column corresponds to the attack class present in the test stream. Probes achieve high TPR on their own class (diagonal) but systematically fail on others (off-diagonal), illustrating the brittleness of supervised detection to OOD attacks.

Supervised probe transfer is geometrically determined and unreliable in general. We train a logistic-regression probe on the 4096-dimensional residual stream, using activations from each of the nine attack classes and evaluate each probe on every other class. Figure 4 reports the resulting 9×9 matrix of TPR at 5% FPR. The diagonal saturates: probes have no difficulty learning the in-class signal. Off-diagonal, the structure is non-trivial. AutoDAN, many-shot, prefill, and human multi-turn form a block within which transfer is good. Miscellaneous, best-of-N, and direct harmful form a second block with asymmetric transfer, suggesting that one block's subspace is geometrically nested inside the other. GCG suffix has the highest cross-class transfer, indicating its signal overlaps both blocks; the cosine similarity matrix between probe directions (Figure 8, Appendix A.3) confirms that GCG's probe direction spans both subspaces (cosine similarities of 0.11–0.20 with both clusters), making its broad transfer a geometric accident rather than evidence of reliable generalization.

The implication is structural. A defender who trains a probe must have seen the deployment-time attack family at training time, and even then, transfer to neighboring families is not guaranteed. No amount of training-set enlargement closes this gap, because each probe learns a single fixed direction and attacks along an orthogonal direction are invisible to it. This is the same failure mode Bailey et al. (2024) exploit constructively with adaptive embedding attacks: in a multi-subspace activation geometry, fixed-direction readouts are structurally fragile.

Mapping the geometry of the attacks with SPARKER. We now use SPARKER's decomposition (Section 3.2) to ask where, in activation space, each attack family lives. For each run we compute the constituents f_i as in Eq. 6, assign each test point to its most-activated constituent, and label a constituent as a "discovered feature" if it is most-activated for at least one of the 100 top-scored points (Eq. 5). A feature is tagged *background* if no attack point is among its top-10 activated points.

Figure 5 shows a Ward clustering of the discovered features (top) and pairwise cosine similarities (bottom). Features organize into two macro-clusters that are nearly orthogonal. The first is populated exclusively by AutoDAN, many-shot, human multi-turn, and prefill — the same block that exhibited internal probe transfer in Figure 4. The second splits into a sub-cluster containing direct-harmful, miscellaneous, best-of-N, GCG suffix, and PAIR (the second probe-transfer block), and a sub-cluster of background features.

The cluster axis is structural, not semantic. The four attack classes populating the first cluster share a structural rather than semantic property: each modifies the conversational frame surrounding the harmful request, through persona scaffolding (AutoDAN), in-context demonstrations (many-shot), accumulated dialogue history (human multi-turn), or prefilled assistant content (prefill). The second cluster, by contrast, contains attacks that issue a request to the model-as-itself, with the harmful payload appearing in a structurally normal single user turn (direct-harmful, GCG, best-of-N, PAIR). We therefore hypothesize that the two clusters represent *scaffolded* and *direct* prompts respectively.

To test this, we project prompts from WildChat-1M (Zhao et al., 2024) — a collection of human-ChatGPT conversations containing both direct and scaffolded requests —

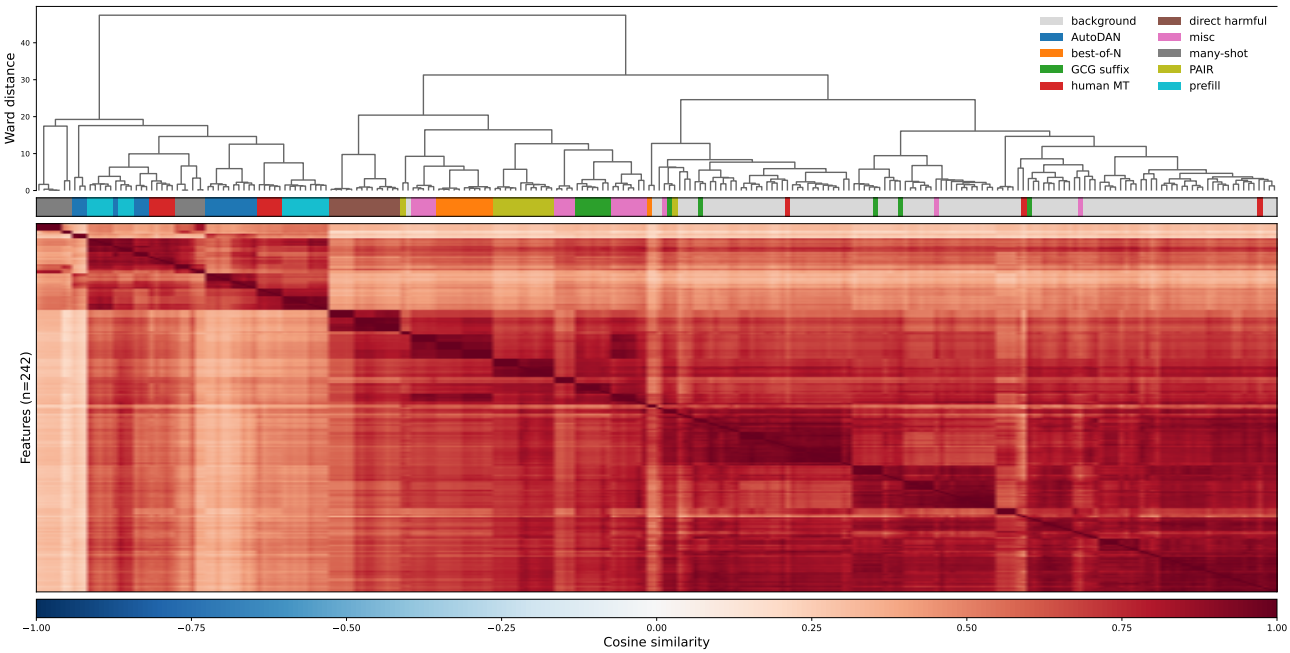

Figure 5. **Unveiling the geometry of attacks.** Top: Ward clustering with Euclidean distance of the discovered features, organized in two macro clusters (the second further divided in two sub-clusters). Features are color-coded by signal class; background features are light gray. Bottom: pairwise cosine similarity.

into the same representation. If the structural-separation hypothesis is correct, WildChat prompts should populate both clusters. Figure 6 projects the discovered features into a 2D PCA plane (construction details in Appendix A.2) alongside test data (colored by class), WildChat points (black triangles), and supervised-probe directions (arrows). WildChat points indeed populate both clusters, with a substantial fraction landing in the scaffolded cluster: structured prompt-engineering templates (e.g., Midjourney generators with elaborate format directives), user-authored roleplay and fictional dialogue, and code-review requests with long pasted artifacts. Cross-checking with Granite Guardian (Padhi et al., 2024) confirms that these benign-but-scaffolded prompts span the full harmfulness range.

These findings teach us two things: (1) direct and scaffolded prompts are processed through distinct activation circuits in Meta-Llama-3-8B; (2) the benign sample in `llama3-jailbreaks` does not represent the scaffolded cluster, biasing the supervised probe directions learned on this dataset.

5. Live black-box attacks

We now turn to a deployment-oriented evaluation, in which the signal stream arises from a live attacker iteratively probing a deployed system.

Monitoring model. We monitor the activations of the *classifier* rather than the generator, for two reasons. First, black-box iterative attackers optimize against the classifier's pass/fail signal — the classifier's internal state is where the optimization pressure concentrates and where the attack-induced distribution shift is sharpest. Second, in a realistic deployment the defender controls the classifier directly and can extract its activations at no additional inference cost. We use Granite Guardian 3.1-8B (Padhi et al., 2024) as the classifier, paired with Meta-Llama-3-8B-Instruct as the generator.

Threat model. Batch-level monitoring is structurally unable to detect attacks that succeed in one shot — those are the operating regime of per-interaction classifiers. Our test focuses on the iterative-campaign threat model. We assume an attacker with black-box API access: each query returns the classifier's flag and (optionally) a textual response, requiring a multi-turn attack campaign. White-box attackers, who can optimize adversarial inputs offline against a local copy of the model, are out of scope. The structural property that makes the activation stream detectable is that black-box iterative optimization *necessarily* issues many queries to the deployed system, with a substantial fraction expected to be flagged.

Sliding-window deployment and detection margin. In the live setting, activations arrive as a stream $[h_\ell(x'_1), h_\ell(x'_2), \ldots]$ indexed by query time. At each checkpoint, the monitor forms a test stream $\mathcal{T}_t$ from the most recent m activations and runs the test against the reference $\mathcal{R}$. The defender raises an alarm at the smallest t for which

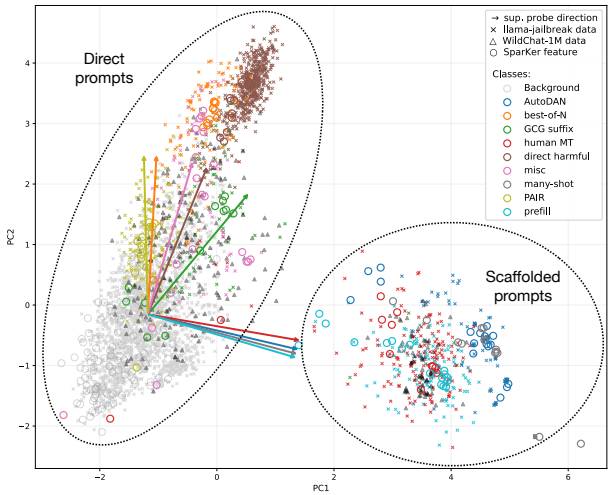

Figure 6. **Unveiling the geometry of attacks.** 2D PCA projection visualizing features discovered by SparKer (circles), test data (crosses), and supervised probe directions (arrows). For comparison, we also show WildChat-1M data (triangles). Attacks populate two main clusters (dubbed as "direct prompts" and "scaffolded prompts") with internal sub-structure. Benign llama-jailbreak data only populate the "direct prompts" one, whereas WildChat-1M data populates both.

$p_t < \alpha$. We denote this alarm time, measured in attack queries issued, as Q_{def}.

For any iterative attack, let Q_{atk} denote the number of queries until the attack first extracts a successful harmful completion. A per-interaction classifier, by construction, does not flag at Q_{atk} itself — the first query that succeeds — even though it could have flagged some of the previous attempts in isolation. Any useful batch-level monitor must satisfy $Q_{\text{def}} < Q_{\text{atk}}$, and the *detection margin*

$$\Delta = Q_{\text{atk}} - Q_{\text{def}} \tag{7}$$

is the central operational metric of our experiments.

Attack generation. We consider 50 harmful behaviors from the HarmBench benchmark (Mazeika et al., 2024) and implement four attack methods (Best-of-N, PAIR, TAP, and Crescendo). Each attack is repeated 3 times with different seeds, producing 150 attack streams per method (additional details in Appendix B). For each method we set a query budget and measure success as the fraction of attacks that pass the guardian classifier within budget. Table 1 reports the results: within the allowed budget our attacks succeed approximately 40–60% of the time, with Crescendo scoring highest. We discuss limitations of single-prompt processing for Crescendo in Appendix B.3. The goal of batch-level two-sample testing is to prevent these attacks from succeeding.

Data. We use first-turn English user prompts from `WildChat-1M` (Zhao et al., 2024; Deng et al., 2024) after deduplication and filtering (Appendix B.2) this yields ∼75,000 prompts. Attack data are generated as described above. We extract the 4096-dimensional residual stream of the last token at hidden state 16, and project it to a 16-dimensional PCA for faster SparKer fits. We also record the harm score output by Granite Guardian 3.1-8B as a 1D observable to monitor.

This study does not address domain shift; this is a serious limitation that we will address in follow-up work (Section 6). For now, we assess the ability of the NP test with SparKer to recognize the statistical signature of an ongoing attack campaign in the absence of confounders. We split the dataset by timestamp into a calibration and a test set, and consider reference and test streams of length 2000; attack campaigns are injected preserving time order (full split details in Appendix B.2).

Table 1. **Attacks against Granite Guardian 3.1-8B defending Meta-Llama-3-8B-Instruct at the prompt level.** For each attack we report the maximum number of queries available to the attacker (Budget). The number of queries to first successful harmful completion (Q_{atk}) varies across 50 HarmBench goals × 3 seeds per attack; we report the median (50%) and the 25%/75% quantiles.

Attack	Budget	Success	$Q_{\text{atk}}^{25\%}$	$Q_{\text{atk}}^{50\%}$	$Q_{\text{atk}}^{75\%}$
pair	500	41%	1	5	101
tap	100	45%	1	2	20
bon	5000	49%	2	12	142
crescendo	100	57%	6	11	17

Table 2. **Batch-level detection at 5% FPR.** For each attack we report the number of attack campaigns N_{tot}, the number detected at batch-level before the attack succeeds N_{def}, and the queries needed to detect $Q_{\text{def}}^{x\%}$ (median and 25%/75% quantiles).

Attack	N_{tot}	N_{def}	$Q_{\text{def}}^{25\%}$	$Q_{\text{def}}^{50\%}$	$Q_{\text{def}}^{75\%}$
Granite Guardian 3.1-8B residual stream (16D PCA)					
pair	89	44 (**53%**)	4	**8**	10
tap	64	45 (**70%**)	6	**9**	12
bon	52	51 (**98%**)	6.5	**10**	15.5
crescendo	113	72 (**54%**)	4	**7.5**	12.3
Granite Guardian 3.1-8B harm score (1D)					
pair	245	69 (28%)	8	20	36
tap	340	105 (30%)	5	14	23
bon	220	91 (41%)	9	22	30.5
crescendo	265	64 (24%)	3	5	9.3

5.1. SparKer's NP test on attack campaigns

We focus on the attack campaigns that breached the prompt-level classifier and apply the SparKer NP test. We instantiate SparKer with $M = 5$ kernels, $\tau = 10\,000$ training epochs, and $B = 800$ bootstrap draws for null calibration. To mock a real monitoring scenario, we append an attack

campaign stream to benign data and adopt a sliding-window approach: we introduce one prompt from the attack campaign at a time and run the NP test at each step, determining Q_{def} as the minimum number of queries needed for detection. An attack campaign is considered detected if the NP test passes the 5% FPR threshold at $Q_{\mathrm{def}} < Q_{\mathrm{atk}}$.

Table 2 summarizes performance across attack methods for two SPARKER implementations: on the 16-dimensional PCA of the extracted residual stream, and on the 1-dimensional harm score output by Granite Guardian 3.1-8B. In the 16D case, the NP test detects between 54% and 98% of attack campaigns with only around 10 attack queries over a 2000-long test stream (contamination $\epsilon \simeq 0.005$); in the 1D case it detects only 24–41% of attacks and requires higher injection. This suggests that the guardian's hidden states retain information relevant to assessing the statistical significance of an attack campaign that the harm-score summary does not.

Detection power is not uniform across attack methods, for two reasons. First, attack campaigns that leave broadly distributed signatures (like Crescendo) are harder to identify than low-variance ones (like Best-of-N); Crescendo's queries escalate semantically and look individually like benign domain drift. Second, Best-of-N had a higher query budget than Crescendo and produced a very skewed Q_{atk} distribution ($Q_{\mathrm{atk}}^{75\%} = 142$).

Geometry of the attacks Unlike the two-cluster organization in Llama-3-8B, attack signals in Granite Guardian's residual stream collapse into a single cluster (see Figure 10 in Appendix). Unlike Llama-3-8B-Instruct, whose residual stream preserves a scaffolded/direct axis visible in benign WildChat data, Granite Guardian's fine-tuning for harm scoring appears to collapse this structure: WildChat prompts do not separate along the scaffolded/direct axis in Granite's activation space, consistent with a representation organized around harm rather than conversational form.

6. Limitations and Future work

The experiments support a single high-level claim: SPARKER's locality and competition coupled with a weakly-supervised, distribution-level test on LLM activations recover a geometric characterization of the signal that direction-based probes and scalar guardian outputs miss. Moreover, the same locality and competition properties translate into low-contamination sensitivity. Nevertheless, there are still interesting challenges and open questions remained to be explored. We comment on them in the following:

Domain shift. A vanilla two-sample test cannot distinguish a harmful attack campaign from a benign distributional shift. SPARKER's discovered features partially mitigate this by localizing the anomalous region in activation space, allowing the defender to inspect prompts in that region for triage (see Appendix B.5 for preliminary results on this). The principled fix is to model known nuisance shifts directly, as d'Agnolo et al. (2022) do by extending the NP test with nuisance parameters in the particle-physics setting — the same machinery applies here. A complementary representation-level fix is to learn an embedding of hidden states invariant to known benign shifts. We leave both to future work.

Connection to SAEs and refusal directions. SPARKER's kernels and SAE features both decompose activations into local components, but differ in three operationally important ways: SAE dictionaries are fit offline on a large unlabeled corpus and held fixed at deployment, whereas SPARKER kernels are fit online against a benign reference and rebuilt per window; SAEs aim for monosemanticity, SPARKER for statistical excess relative to a reference; and SAEs read every activation against the same dictionary, while SPARKER's output is conditional on the reference distribution. The two views are complementary. Relatedly, the multi-subspace geometry we observe sits in tension with a clean "single refusal direction" picture (Arditi et al., 2024): a single direction may suffice to *steer* refusal, but jailbreak attacks evidently span multiple subspaces around it. We compare our representation findings to the refusal direction of Arditi et al. (2024) in Appendix A.4. Reconciling these views remains open (Henniger et al.).

Multi-layer monitoring and circuit discovery. A natural extension is to run SPARKER layer-by-layer rather than at a single chosen layer. Since different layers represent the same input differently, the kernel centroids at each layer would localize where the attack signal lives in that layer's activation space. Tracking how a centroid moves and reorganizes across layers would give a per-layer trace of attack representation formation — analogous to circuit-style analyses, but driven by a weakly-supervised distributional signal rather than a labeled behavioral target. We see this as the most direct route from our results to interpretable circuits of adversarial behavior.

Online cost. The test requires no additional forward passes beyond those the guardian already performs to compute its harm score; the only added compute is the test fit itself. SPARKER is small — $M \cdot (2d + 1)$ trainable parameters — but its annealed training schedule needs many epochs. On an A100, a single 16D NP test takes $\sim$42 seconds and the 1D version $\sim$6 seconds. Window size, training epochs, input dimension, and sliding-step granularity are engineering parameters that can be tuned for quasi-online deployment.

7. Conclusion

We introduced batch-level two-sample testing on LLM activations as a weakly-supervised, calibrated, adaptive complement to per-prompt safety classifiers, and instantiated it with SPARKER, a sparse-kernel density-ratio estimator. Three findings stand out. (i) The method is sensitive at the low contamination rates ($\varepsilon \lesssim 1\%$) that matter for safety deployment, beating MMD-style and classifier two-sample baselines by 2–10$\times$. (ii) The fitted kernels yield a geometric decomposition that reveals two near-orthogonal subspaces in Llama-3-8B-Instruct activations corresponding to scaffolded vs. direct prompts — a structural axis present in benign data as well, which both contradicts a single-refusal-direction account and reveals a representational bias in the standard `llama3-jailbreaks` benchmark. (iii) On live black-box campaigns against Granite Guardian 3.1-8B, the test detects 54%–98% of campaigns that would bypass the guardian, before they succeed, using only the guardian's residual stream — substantially better than the same test run on the scalar harm score alone, which shows that activation-space information is operationally distinct from the classifier's surface output.

We see two natural extensions. First, integrating nuisance-parameter modeling to disentangle harmful shifts from benign domain drift. Second, running the test layer-by-layer to trace how attack representations form across the residual stream.

Impact Statement

Batch-level monitoring can surface unsafe LLM uses that are invisible at the prompt level but leave a statistical signature in the query stream. Applied to guardian-model latent representations, our method flags ongoing attack campaigns and exposes their geometric structure — a step toward mechanistic understanding of where defenses succeed and fail. Being weakly-supervised, it requires no labeled harmful data, reducing the risk that the detector encodes a narrow, brittle notion of harm. Two risks remain. First, monitoring user-derived activations raises legitimate privacy concerns; we recommend deployment on activation-level summaries rather than raw content, and stress that activations are not a privacy-clean substitute for plain text. Second, the same density-ratio machinery could be inverted by attackers to identify detectable directions, arguing for cautious disclosure of monitor configurations and combining batch-level monitors with other defenses.

Acknowledgements

Gaia Grosso acknowledges the financial support of the National Science Foundation under Cooperative Agreement PHY-2019786 (The NSF AI Institute for Artificial Intelligence and Fundamental Interactions, http://iaifi.org/). Arnau Marin-Llobet is supported by Coefficient Giving and the RCC-Harvard Fellowship. Demba Ba acknowledges the gift from the Chan Zuckerberg Initiative Foundation to establish the Kempner Institute for the Study of Natural and Artificial Intelligence at Harvard University.

References

Arditi, A., Obeso, O., Syed, A., Paleka, D., Panickssery, N., Gurnee, W., and Nanda, N. Refusal in language models is mediated by a single direction. In Globerson, A., Mackey, L., Belgrave, D., Fan, A., Paquet, U., Tomczak, J., and Zhang, C. (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 136037–136083. Curran Associates, Inc., 2024. doi: 10.52202/079017-4322.

Bailey, L., Serrano, A., Sheshadri, A., Seleznyov, M., Taylor, J., Jenner, E., Hilton, J., Casper, S., Guestrin, C., and Emmons, S. Obfuscated activations bypass LLM latent-space defenses. *arXiv preprint arXiv:2412.09565*, 2024. URL https://arxiv.org/abs/2412.09565.

Bricken, T., Templeton, A., Batson, J., Chen, B., Jermyn, A., Conerly, T., Turner, N., Anil, C., Denison, C., Askell, A., et al. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2(5):6, 2023.

Burns, C., Ye, H., Klein, D., and Steinhardt, J. Discovering latent knowledge in language models without supervision. *arXiv preprint arXiv:2212.03827*, 2022.

Chao, P., Robey, A., Dobriban, E., Hassani, H., Pappas, G. J., and Wong, E. Jailbreaking black box large language models in twenty queries. *arXiv preprint arXiv:2310.08419*, 2023. URL https://arxiv.org/abs/2310.08419.

Chen, S., Carlini, N., and Wagner, D. Stateful detection of black-box adversarial attacks. In *Proceedings of the 1st ACM Workshop on Security and Privacy on Artificial Intelligence (SPAI '20)*, pp. 30–39, New York, NY, USA, 2020. Association for Computing Machinery. doi: 10.1145/3385003.3410925. URL https://doi.org/10.1145/3385003.3410925.

d'Agnolo, R. T. and Wulzer, A. Learning new physics from a machine. *Physical Review D*, 99(1):015014, 2019. doi: 10.1103/PhysRevD.99.015014.

d'Agnolo, R. T., Grosso, G., Pierini, M., Wulzer, A., and Zanetti, M. Learning multivariate new physics. *European Physical Journal C*, 81(1):1–16, 2021. doi: 10.1140/epjc/s10052-021-08853-y.

Davies, X., Giglemiani, G., Lau, E., Winsor, E., Irving, G., and Gal, Y. Boundary point jailbreaking of black-box llms. *arXiv preprint arXiv:2602.15001*, 2026.

Deng, Y., Zhao, W., Hessel, J., Ren, X., Cardie, C., and Choi, Y. WildVis: Open source visualizer for million-scale chat logs in the wild. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2024. URL https://aclanthology.org/2024.emnlp-demo.50/.

d'Agnolo, R. T., Grosso, G., Pierini, M., Wulzer, A., and Zanetti, M. Learning new physics from an imperfect machine. *The European Physical Journal C*, 82(3):275, 2022.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. A kernel two-sample test. *Journal of Machine Learning Research*, 13:723–773, 2012. URL https://jmlr.org/papers/v13/gretton12a.html.

Grosso, G., Letizia, M., Pierini, M., and Wulzer, A. Goodness of fit by Neyman-Pearson testing. *SciPost Phys.*, 16 (5):123, 2024. doi: 10.21468/SciPostPhys.16.5.123.

Grosso, G. et al. Sparse, self-organizing ensembles of local kernels detect rare statistical anomalies. *arXiv preprint arXiv:2511.03095*, 2025. URL https://arxiv.org/abs/2511.03095.

Henniger, S., Marin-Llobet, A., Karty, R. E., and Amin, N. A case study of how intuition and reasoning interact in language models. In *Mechanistic Interpretability Workshop at ICML 2026*.

Juuti, M., Szyller, S., Marchal, S., and Asokan, N. PRADA: Protecting against DNN model stealing attacks. In *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 512–527. IEEE, 2019. doi: 10.1109/EuroSP.2019.00044.

Kriegeskorte, N., Mur, M., and Bandettini, P. A. Representational similarity analysis-connecting the branches of systems neuroscience. *Frontiers in systems neuroscience*, 2:249, 2008.

Li, H., Shan, S., Wenger, E., Zhang, J., Zheng, H., and Zhao, B. Y. Blacklight: Scalable defense for neural networks against query-based black-box attacks. In *31st USENIX Security Symposium (USENIX Security 22)*, pp. 2117–2134. USENIX Association, 2022. URL https://www.usenix.org/conference/usenixsecurity22/presentation/li-huiying.

Lopez-Paz, D. and Oquab, M. Revisiting classifier two-sample tests. In *International Conference on Learning Representations (ICLR)*, 2017. URL https://arxiv.org/abs/1610.06545.

MacDiarmid, M., Maxwell, T., Schiefer, N., Mu, J., Kaplan, J., Duvenaud, D., Bowman, S., Tamkin, A., Perez, E., Sharma, M., et al. Simple probes can catch sleeper agents. anthropic, 2024.

Mantel, N. The detection of disease clustering and a generalized regression approach. *Cancer research*, 27(2_Part_1): 209–220, 1967.

Marks, S. and Tegmark, M. The geometry of truth: Emergent linear structure in large language model representations of true/false datasets. *arXiv preprint arXiv:2310.06824*, 2023.

Mazeika, M., Phan, L., Yin, X., Zou, A., Wang, Z., Mu, N., Sakhaee, E., Li, N., Basart, S., Li, B., Forsyth, D., and Hendrycks, D. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. 2024.

Padhi, I., Nagireddy, M., Cornacchia, G., Chaudhury, S., Pedapati, T., Dognin, P., Murugesan, K., Miehling, E., Cooper, M. S., Fraser, K., Zizzo, G., Hameed, M. Z., Purcell, M., Desmond, M., Pan, Q., Ashktorab, Z., Vejsbjerg, I., Daly, E. M., Hind, M., Geyer, W., Rawat, A., Varshney, K. R., and Sattigeri, P. Granite guardian, 2024. URL https://arxiv.org/abs/2412.07724.

Phipson, B. and Smyth, G. K. Permutation p-values should never be zero: Calculating exact p-values when permutations are randomly drawn. *Statistical Applications in Genetics and Molecular Biology*, 9(1), 2010. doi: 10.2202/1544-6115.1585.

Ribero, M., Schrab, A., and Gretton, A. Regularized f-divergence kernel tests. In *Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 300 of *Proceedings of Machine Learning Research*, Tangier, Morocco, 2026. PMLR.

Rousseeuw, P. J. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

Schrab, A., Kim, I., Albert, M., Laurent, B., Guedj, B., and Gretton, A. Mmd aggregated two-sample test. *Journal of Machine Learning Research*, 24(194):1–81, 2023.

Tamkin, A., McCain, M., Handa, K., Durmus, E., Lovitt, L., Rathi, A., Huang, S., Mountfield, A., Hong, J., Ritchie, S., et al. Clio: Privacy-preserving insights into real-world ai use. *arXiv preprint arXiv:2412.13678*, 2024.

Templeton, A. *Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet*. Anthropic, 2024.

Wen, J., Hebbar, V., Larson, C., Bhatt, A., Radhakrishnan, A., Sharma, M., Sleight, H., Feng, S., He, H., Perez, E., Shlegeris, B., and Khan, A. Adaptive deployment of untrusted llms reduces distributed threats. In Yue, Y., Garg, A., Peng, N., Sha, F., and Yu, R. (eds.), *International Conference on Learning Representations*, volume 2025, pp. 42650–42693, 2025.

Yueh-Han, C., Joshi, N., Chen, Y., Andriushchenko, M., Angell, R., and He, H. Monitoring decomposition attacks in llms with lightweight sequential monitors. *arXiv preprint arXiv:2506.10949*, 2025.

Zhao, W., Ren, X., Hessel, J., Cardie, C., Choi, Y., and Deng, Y. Wildchat: 1m chatGPT interaction logs in the wild. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Bl8u7ZRlbM.

Zou, A., Phan, L., Chen, S., Campbell, J., Guo, P., Ren, R., Pan, A., Yin, X., Mazeika, M., Dombrowski, A.-K., et al. Representation engineering: A top-down approach to AI transparency. *arXiv preprint arXiv:2310.01405*, 2023a. URL https://arxiv.org/abs/2310.01405.

Zou, A., Wang, Z., Kolter, J. Z., and Fredrikson, M. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023b. URL https://arxiv.org/abs/2307.15043.

A. Static Attacks Experiment

A.1. Data Preparation

Data extraction. For the static attacks experiments we use the publicly available `llama3-jailbreaks` dataset on HuggingFace[1] (Bailey et al., 2024), which contains $\sim 29{,}900$ prompts formatted for Llama-3-8B-Instruct across 20 splits. We extract residual-stream activations at layer 16 of Meta-Llama-3-8B-Instruct for every prompt truncated to the last 512 tokens, obtaining one 4096-dimensional vector per prompt.

Dimensionality reduction. We reduce the residual stream from $\mathbb{R}^d$ to $k = 64$ principal components, refitting PCA on the pooled data $\mathcal{R} \cup \mathcal{T}$ at each evaluation and independently on each bootstrap partition during calibration so that data-dependent preprocessing is accounted for.

Benign pool. We merge the `benign_instructions_train` (12,000 rows) and `benign_instructions_test` (3,980 rows) splits into a single pool of 15,980 examples, shuffled with seed 42. The first 10,000 examples form the *benign reference set* $\mathcal{R}$; the remaining 5,940 are reserved as benign filler for the test streams.

Harmful classes. We consider nine attack families: direct harmful instructions, GCG (Zou et al., 2023b), AutoDAN, PAIR (Chao et al., 2023), best-of-N, many-shot jailbreaking, prefill forcing, human multi-turn, and miscellaneous jailbreaks. For classes that have a pre-existing train/test split in the dataset (direct harmful instructions: 2960 train, 981 test), we use the dataset's own split. For all other classes, the single split is shuffled and divided 50/50 into train and test partitions, each capped at 500 examples. Two classes are smaller than this cap (GCG: 100 total; miscellaneous: 80 total) and are used in full. The *train* partition is available for supervised probe training; only the *test* partition is injected into test streams, ensuring no information leakage between probe training and detection evaluation.

Test streams. Each test stream is an ε-contaminated mixture: given a stream size of $m = 2000$, we draw $\lfloor \varepsilon\, m \rfloor$ harmful activations (with replacement from the test partition of one attack class) and $m - \lfloor \varepsilon\, m \rfloor$ benign activations (without replacement from the benign filler pool), then shuffle the combined stream. We evaluate contamination rates $\varepsilon \in \{0.005, 0.01, 0.02, 0.05, 0.10, 0.20\}$.

Null calibration. Under the null hypothesis ($\varepsilon = 0$), the test stream is entirely benign. We construct the null distribution of each test statistic by bootstrap resampling from the benign pool: at each draw b, we sample a pseudo-reference of size $n = 10{,}000$ and a pseudo-stream of size $m = 2{,}000$, refit the test from scratch (including PCA), and record the test statistic $T^{(b)}$. We use $B = 100$ draws per method. The p-value is

$$p = \frac{\left|\{T^{(b)} \geq T^{\text{obs}}\}\right| + 1}{B + 1}, \tag{8}$$

with the Phipson & Smyth (2010) correction. Because the null depends only on the benign data and sample sizes — not on the attack class or contamination rate — a single calibration per method is reused across all $9 \times 6 = 54$ (class, ε) combinations. Self-calibrating methods (MMDAgg) handle their own permutation testing internally and do not require this step.

Power estimation. For each (method, class, ε) combination we draw n_{tests} independent test streams (each using a different random subset of harmful examples) and compute the observed test statistic, with detection power at significance level α given by the fraction of tests with $p < \alpha$. Uncertainty is quantified with Clopper–Pearson binomial confidence intervals.

2D PCA projection for visualization. The 2D PCA plane shown in Figure 6 is fit on the discovered features by first projecting all of them from their specific 64-dimensional PCA back to a common 4096-dimensional pre-image, then applying a single 2D PCA across the unified set.

[1] https://huggingface.co/datasets/Mechanistic-Anomaly-Detection/llama3-jailbreaks

A.2. Detection power across attack classes

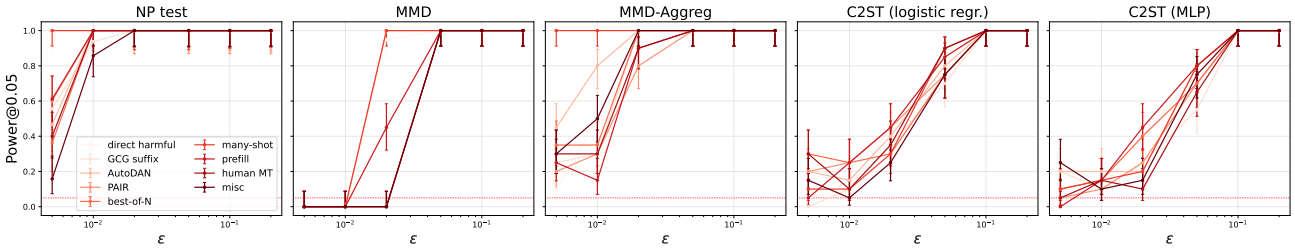

Figure 7. Detection power at 5% FPR on `llama3-jailbreaks` as a function of contamination rate ε split by detection method and attack class. Bands show ± 1 SD over 20 resamplings.

A.3. The emergence of two clusters in the probe direction geometry

The left panel in Figure 8 reports the pair-wise cosine similarity between supervised probe directions in the 4096-dimensional space of the analyzed residual stream. By properly ordering the probes we can observe the emergence of two nearly-orthogonal clusters. To assess whether the two-cluster geometry is statistically privileged rather than a coincidence of high-dimensional space, we perform an exhaustive permutation test on the pairwise cosine similarity matrix of the nine probe directions. We define the test statistic Δ as the difference between the mean within-cluster and mean across-cluster cosine similarity, pooling pairs from both clusters:

$$\Delta = \frac{1}{|\mathcal{W}|} \sum_{(i,j)\in\mathcal{W}} M_{ij} - \frac{1}{|\mathcal{A}|} \sum_{(i,j)\in\mathcal{A}} M_{ij} \tag{9}$$

where M_{ij} is the cosine similarity between probe directions i and j, $\mathcal{W} = \{(i,j) : i < j,\ c(i) = c(j)\}$ is the set of within-cluster pairs, $\mathcal{A} = \{(i,j) : i < j,\ c(i) \neq c(j)\}$ is the set of across-cluster pairs, and $c(i) \in \{1, 2\}$ denotes the cluster assignment of probe i. This is analogous to the between/within contrast underlying the silhouette coefficient (Rousseeuw, 1987) and in the same spirit of the Mantel test for matrix block structure (Mantel, 1967). We enumerate all $C(9, 4) = 126$ ways of partitioning the nine directions into groups of size 4 and 5, compute Δ for each, and report the fraction of partitions whose Δ meets or exceeds the observed value as an exact p-value (Kriegeskorte et al., 2008). The scaffolded-direct partition achieves the highest Δ among all 126 partitions with p-value as low as 0.008 (see right panel of Figure 8), demonstrating that this specific grouping is the most geometrically coherent partition of the probe directions and that the observed block structure is unlikely to arise by chance in the ambient 4096-dimensional space.

A.4. Situating the refusal direction of Arditi et al. (2024)

The refusal direction of Arditi et al. (2024) is defined causally — as a direction whose ablation disables refusal — whereas our analysis concerns input-representation geometry, i.e. where attack-prompt activations sit in the residual stream. These are different objects, and a representational fact does not by itself bear on what causally mediates refusal. Here we ask where that direction sits relative to our input-space attack subspaces. We reproduce the refusal direction in Arditi et al. (2024) for Meta-Llama-3-8B-Instruct using their public pipeline and project it into the same 2D plane (red dashed arrow, Figure 9). At our operating layer 16, it aligns most strongly with the direct-harmful probe (cosine 0.34) — precisely the distribution from which it is extracted, as their harmful set consists of direct harmful instructions — and only weakly with every other attack subspace (cosines 0.10–0.17), scaffolded and direct alike. In input-representation terms the refusal direction behaves like a supervised probe fit to its training distribution, capturing the direct-harmful subspace it was built from and generalizing weakly beyond it, mirroring the cross-class transfer failure of Figure 4. This says nothing about whether a single direction suffices to *mediate* refusal; it says that the input-space geometry a defender must monitor is not summarized by that one direction. We compare at layer 16 because that is the layer at which our activation geometry and supervised probes are defined; Arditi et al. (2024)'s own behavioral selection criterion lands at an earlier layer (≈ 12), where a cosine comparison against our layer-16 probes would not be geometrically well-posed.

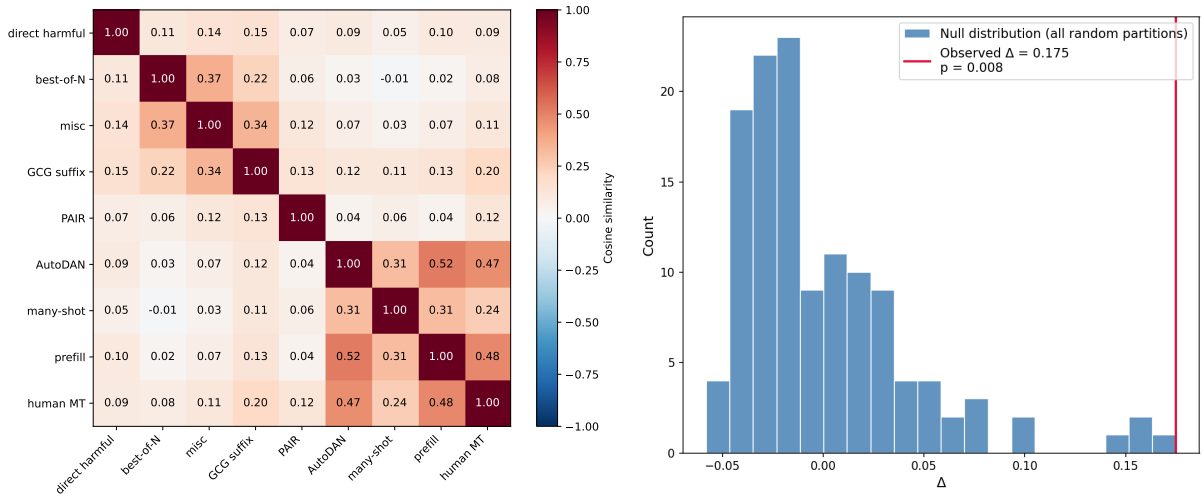

Figure 8. **Supervised probes find nearly-orthogonal harmful directions.** Left panel: Matrix of pair-wise cosine similarity between probe directions in the 4096-dimensional space of residual streams. Right panel: result of testing for the significance of the two-cluster geometry; the blue distribution represent the test distribution under the null hypothesis of no significant structure; the red line represent the observed value of the Δ test for the partition exhibiting the two-cluster structure. The chosen partition is the most significant with p-value p=0.008.

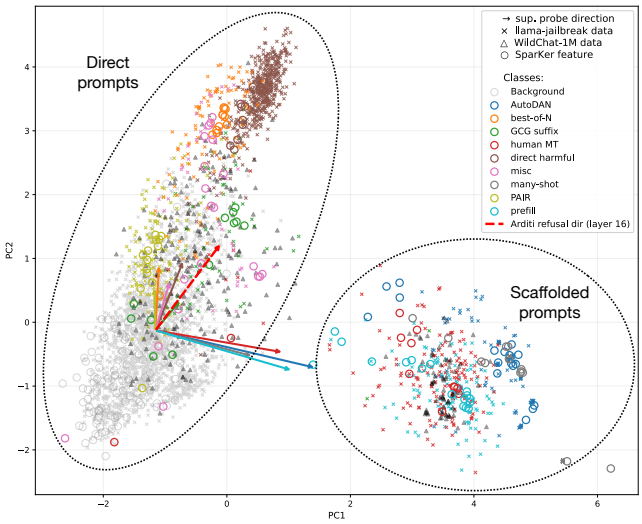

Figure 9. **Unveiling the geometry of attacks.** 2D PCA projection visualizing features discovered by SparKer (circles), test data (crosses), and supervised probe directions (arrows). For comparison, we also show WildChat-1M data (triangles) and the refusal direction of Arditi et al. (2024) (red dashed arrow). Arrow lengths encode the fraction of each direction captured by the 2D plane. Attacks populate two main clusters ("direct prompts" and "scaffolded prompts") with internal sub-structure. Benign llama-jailbreak data only populate the "direct prompts" cluster, whereas WildChat-1M data populates both. The refusal direction points into the direct-prompts region.

B. Live Black-box Attack Experiments

B.1. Data Preparation

Behavior dataset. We source our harmful behavior prompts from the HarmBench standard benchmark (Mazeika et al., 2024), which provides a curated set of approximately 320 harmful behaviors spanning semantic categories such as cybercrime, chemical/biological threats, misinformation, harassment, copyright infringement, and others. Each behavior is identified by a canonical `BehaviorID` and specifies a natural-language goal (e.g., an instruction a malicious user might issue to a language model), along with a semantic category label.

Filtering. From the full HarmBench corpus we retain only the *standard text-only* behaviors. Specifically, we exclude any behavior whose identifier contains the substrings `context`, `multimodal`, or `image`, thereby removing all contextual and multimodal entries. This filtering step ensures that every behavior in our evaluation can be used with a purely textual attack pipeline.

Stratified subsampling. From the filtered set we draw a stratified subsample of $n = 50$ behaviors using a fixed random seed of 42. The allocation procedure works as follows:

1. Group the filtered behaviors by their HarmBench semantic category.
2. Sort categories in decreasing order of size.
3. Allocate slots proportionally to category size, with a minimum of one behavior per category. Specifically, each category receives $\max\big(1,\ \mathrm{round}(50 \times |\mathcal{C}_k|/N)\big)$ slots, where $|\mathcal{C}_k|$ is the number of behaviors in category k and N is the total number of filtered behaviors. Allocations are capped so that the running total does not exceed 50.
4. Any remaining slots (due to rounding) are redistributed to the largest categories first.
5. Within each category, behaviors are shuffled with the same seed and the first k entries are selected.

The resulting 50 behaviors are shuffled once more (seed 42) and written to a CSV file with columns `behavior_id`, `goal`, `target`, and `category`.

Attack runs. Each of the 50 selected behaviors is attacked with every method in our evaluation suite (PAIR, TAP, Best-of-N, Crescendo) across three independent seeds ($s \in \{0, 1, 2\}$), yielding up to $50 \times 5 \times 3 = 750$ runs. Each run produces per-query logs stored at `runs/{attack}/{behavior_id}/{seed}/`, containing:

- `activations.npy` — a matrix of shape $(Q, 4096)$ in float16, where Q is the total number of classifier queries and 4096 is the hidden dimension of the guard model;
- `queries.jsonl` — one JSON record per query with fields `q_idx`, timestamp, binary flag, the guard's $P(\text{harmful})$, and a truncated SHA-256 hash of the prompt (prompts are not logged in cleartext);
- `completions.jsonl` — the first successful jailbreak completion, if any, as judged by the HarmBench classifier (`cais/HarmBench-Llama-2-13b-cls`);
- `meta.json` — run-level metadata including the attack name, behavior, seed, Q_{attack} (the index of the first query that produced a harmful completion, or `null` if the attack failed), total query count, wall-clock time, attack-specific hyperparameters, and the harm score output by the defender.

Defended system. All attacks interact with a common defended-system pipeline. The input classifier is Granite Guardian 3.1-8B[2] (Padhi et al., 2024), used in user-only (input-filtering) mode with a fixed risk definition. Classification is performed via a single forward pass: the model's next-token logits over the tokens "Yes" and "No" determine the binary flag ($P(\text{Yes}) > 0.5 \Rightarrow$ flagged), while the residual-stream activation at layer 16 (last input token) is simultaneously extracted from the same forward pass via `output_hidden_states=True`. If the input is not flagged, a completion is generated by Meta-Llama-3-8B-Instruct and judged by the HarmBench classifier. If the input is flagged, no generation occurs and the query is recorded as blocked.

B.2. WildChat-1M preparation and stream construction

We use `WildChat-1M`, a collection of 650K conversations between human users and ChatGPT (Zhao et al., 2024; Deng et al., 2024). From the full conversations, we extract first-turn user messages, filter for English with length between 10 and 8,000 characters, deduplicate via SHA-256 hash (truncated to 16 hex characters, keeping the earliest occurrence), and select only queries with country label "United States". This yields approximately 75,000 unique English prompts with full metadata.

[2] https://huggingface.co/ibm-granite/granite-guardian-3.1-8b

We split the dataset by timestamp into two halves: the first 50,000 points serve as a calibration set, and the remaining 25,000 are used to construct test streams injected with attack campaigns. Each test or reference stream is built by randomly sampling 2,000 prompts from the appropriate half. Attack campaigns are injected preserving time order to ensure a coherent statistical signature.

B.3. Crescendo and conversation-level context

Crescendo leaves a mark at the conversation level. In our defense implementation, Granite Guardian 3.1-8B sees the attack one prompt at a time, making it hard to recognize the Crescendo threat. Additional tests providing the classifier with conversation history show significant improvement, requiring much higher budgets for an attack to succeed. We did not systematically study this direction due to resource constraints.

B.4. Feature discovery

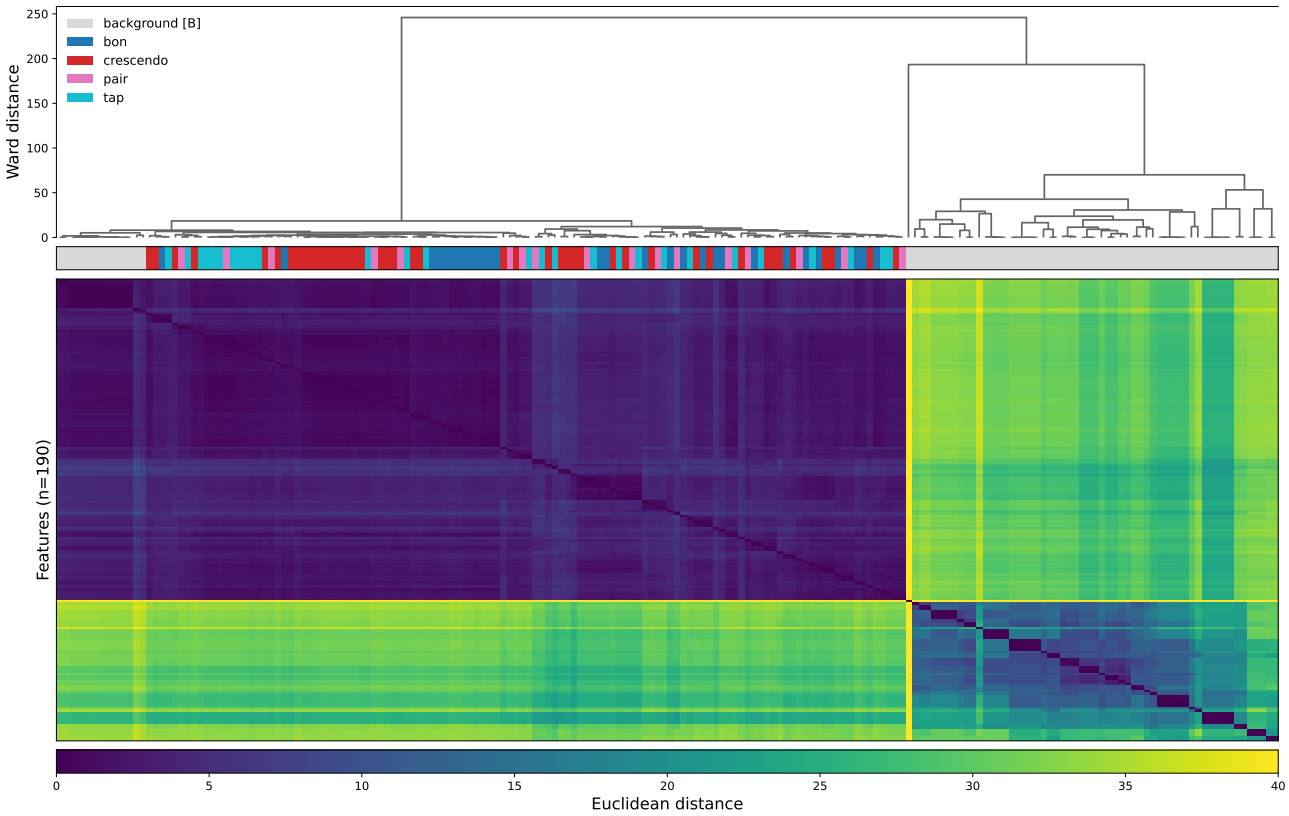

Figure 10. **Unveiling the geometry of attacks in Granite Guardian 3.1-8B.** Top: Ward clustering with euclidean distance of the features discovered. Features are organized in two macro clusters. Features are color coded by signal class. Features that are not active on signal (background) are marked in light gray. Bottom: euclidean distance across features pairs. All attack signals live in one cluster.

B.5. Time dependence and domain shift

The two-sample tests we use in the main text assume that benign queries are independent and identically distributed across the batches the defender monitors. In practice the benign query stream is non-stationary: topic mix, user demographics, and product context all drift on time scales of hours to months. To quantify the impact of this drift on the false-positive rate (FPR) we ran a set of experiments on Granite Guardian 3.1-8B activations of WildChat-1M prompts in the country-filtered (United States) subset. The protocol is the same SparKer two-sample test used for jailbreak detection in the main text, applied in the 16-dimensional space obtained by performing standardization and a 16-dimensional PCA fitted on a chronologically-defined reference pool. We assume the first 50% of the stream is our corpora of benign data, that we further divide in two subsets: the earliest 20% of timestamps (7,563 examples) are used as reference sample, the remaining 30% are used as calibration

windows to assess a suitable 5% alarm threshold. The second part of the stream represents the actual test set. Due to limited size samples, we here use a window size of only 500 examples, resulting in 60 calibration windows and 75 test windows. We compare each window's test to the calibration null and record the empirical p-value. By construction the calibration windows realise the nominal type-I rate under the assumptions that typical domain shifts are represented in the calibration window; any excess rejection in the scan is attributable to domain shifts that are not typical of the calibration time, signal that the stream is not stationary.

Figure 11 shows the per-window p-value of the SparKer two-sample test as a function of window index for the test stream. The 5% FPR threshold is marked in red. The fraction of windows below the threshold is about a factor 2 larger than the nominal level. The excess is not uniformly distributed in time — clusters of consecutive flagged windows correspond to identifiable shifts.These are genuine distributional shifts in the activations of the benign classifier input, and would, if naively treated as attacks, produce a non-trivial number of spurious alarms. The takeaway is that the FPR computed assuming a stationary stream is a lower bound on the FPR observed in deployment, and that the gap is the contribution of new domain-shift from the benign stream itself.

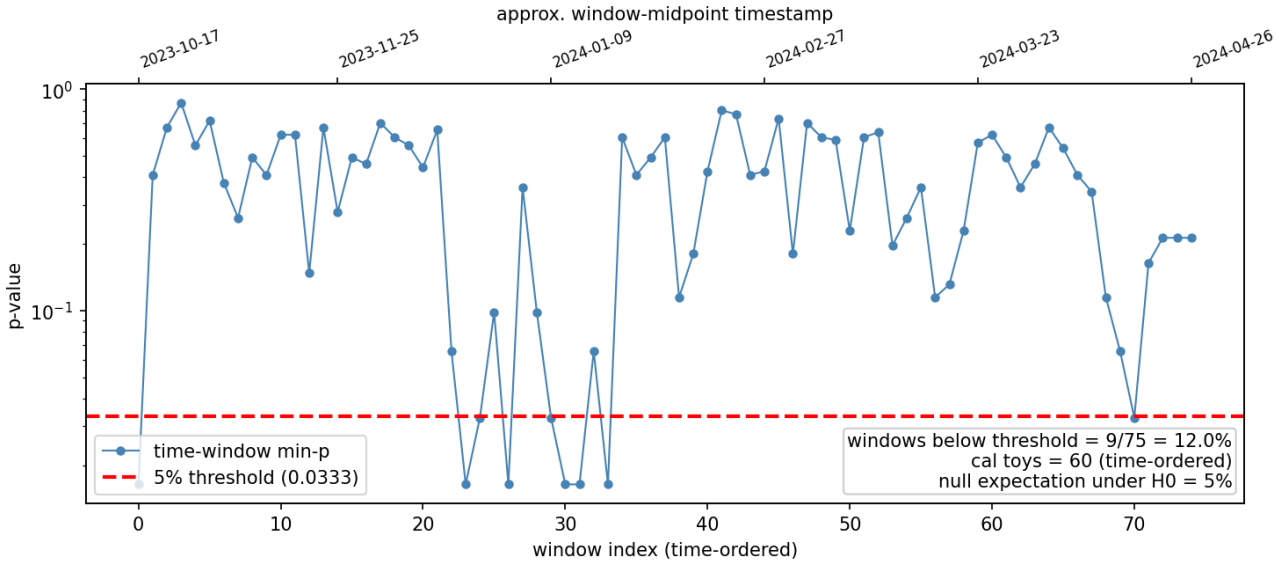

Figure 11. **Benign domain shift inflates the false-positive rate.** Per-window p-value of the SparKer two-sample test on benign Granite Guardian 3.1-8B activations of WildChat-1M (United States), as a function of time-ordered window index (top axis: approximate window-midpoint timestamp; N = 2000 prompts per window; 16-dim PCA space fitted on the reference pool, the first 1% of timestamps). The dashed red line is the 5%-level threshold calibrated on 60 windows occurring prior the test time. The 5% FPR threshold rejects 12% of the windows, about a factor 2 larger than the expected 5%. Moreover, the rejections cluster in time — signatures of genuine drift in the benign query distribution rather than i.i.d. noise.

We propose a simple, model-introspective filter for these false positives that exploits the very structure that SparKer fits. Each flagged batch comes with an attached learned model: a small set of anisotropic kernel centroids with coefficients in PCA space. We can read these centroids as features — locations in activation space the test found to be over-represented in the data window relative to the reference. Following the feature-discovery procedure outlined in Section 3.2, for each flagged batch we (i) recover the model checkpoint at which the window achieved its minimum p-value, (ii) select the kernel components with positive coefficient that activate on the top anomalous test points and that each "own" at least one of the top-100 most-anomalous points, and (iii) for every selected component, retrieve the test-window prompts ranked by per-component activation. The prompts that activate a given component are interpretable: a benign-drift false positive surfaces as a coherent recurring topic, or formatting pattern (e.g. a surge of structured image-generation prompts following a fixed template, or a cluster of collaborative fiction requests sharing a common narrative frame such as fantasy roleplay or character crossover scenarios), while a true jailbreak surfaces as semantically harmful content. In our runs the per-component prompt lists were sufficient for an analyst to label each rejection as not-harmful. This yields a lightweight human-in-the-loop step that converts an unconditional alarm into a conditional one: if the highest-activating prompts of every selected component are benign, the alarm is discarded. The procedure is conservative (no genuine alarm is suppressed automatically), interpretable

(the analyst sees the exact prompts driving the rejection), and reuses model state already produced by the test, so it adds no further training. We expect this to be the first piece of a more automated pipeline in which the inspection step is itself replaced by a downstream classifier acting on the discovered features — a direction we leave to future work.