A Paired Testing Protocol for Batch-Conditioned Refusal Robustness in LLM Serving

Sahil Kadadekar [1]

Abstract

Safety evaluations of language models often treat serving configuration as fixed background infrastructure, but batch condition is an untested treatment variable whenever the same prompt may be evaluated alone, in a synchronized batch, or inside a continuous-batching scheduler. We synthesize four artifact-backed studies into a paired testing protocol: Study A combines local discovery, scorer-corrected adjudication, and true-batching confirmation; Study B tests cross-model generalization; Study C tests continuous-batch composition; and Study D runs a batch-invariant-kernel ablation. The local test finds safety-label changes more often than capability-label changes (0.51% vs. 0.14%), but adjudication of 63 candidate rows leaves only 17 genuine behavioral flips, implying a corrected full-set rate of 0.16%. The 15-model extension finds no detectable universal safety-over-capability skew: flips are near parity (0.94×), alignment type has no detectable association ($p = 0.942$, $\eta^2 = 0.033$), and output instability is the strongest tested fragility screen ($r = 0.909$, bootstrap 95% CI [0.65, 0.97]). In the targeted kernel ablation, standard vLLM reproduces 22/55 label flips on current score-flip candidates, while enabling `VLLM_BATCH_INVARIANT=1` reduces the same test to 0/55 flips; the composition test separately detects no aggregate effect at 4.7pp sensitivity. The testing recommendation is exact-stack validation: evaluate refusal at the served batch setting, pair safety prompts with capability controls, and report low-rate directional flips separately from aggregate null effects.

[1]Independent Researcher. Correspondence to: Sahil Kadadekar <sahilkadadekar@nyu.edu>.

ICML 2026 Workshop on Hypothesis Testing, Seoul, South Korea, 2026. Copyright 2026 by the author(s).

1. Introduction

Language-model evaluations usually hold serving configuration fixed. That convention is convenient, but it hides a testing assumption: a refusal decision measured for a single request is treated as evidence about the same prompt under the served batch condition. Batch size and scheduling are therefore not merely systems parameters when the outcome of interest is a boundary event such as refusal versus compliance. They are treatment variables that should either be tested directly or explicitly ruled out.

This paper studies that testing problem for batch-conditioned refusal robustness. The question is not whether batching is generally unsafe. The question is whether a paired test can detect refusal-boundary movement when the same prompt is evaluated under different batch conditions, while controlling for ordinary output variation and benign capability changes. The evidence is deliberately staged because the easy first conclusion is too strong. An initial local test found more score-changing rows on the safety side than on the capability side, but adjudication sharply reduced the implied operational rate. A 15-model extension found no universal safety-over-capability skew. A continuous-batch composition study detected no aggregate multi-tenant effect at the available sensitivity. A targeted batch-invariant-kernel ablation then sharpened the mechanism story on the current candidate surface: standard vLLM reproduced 22/55 label flips, while the batch-invariant kernel produced 0/55.

We synthesize these studies into a compact testing protocol. Study A is a local paired perturbation test with safety prompts, capability controls, scorer correction, and a true-batching mechanism check. Study B scales the same question across 15 models and asks whether model category or output instability is associated with fragility. Study C tests whether continuous-batch co-residence creates a separate composition channel. Study D re-runs the current Study A score-flip candidates under standard vLLM and vLLM's batch-invariant kernel on the same H100 serving stack. Together, the studies separate discovery, measurement correction, generalization, mechanism confirmation, and deployment-composition testing.

Thesis. Batch-conditioned refusal flips exist, but they are low-rate and model-specific rather than a universal safety-over-capability law. The correct output is therefore not a prohibition on batching. It is an exact-stack test: evaluate the served batch setting, include capability controls, report scorer-corrected rates, and treat rare directional flips separately from aggregate null effects.

Scope posture. This is a workshop-scale synthesis across four artifact-backed studies rather than a new monolithic benchmark. Its contribution is the testing contract that survives the staged evidence: what to compare, what to correct, what to generalize, and how not to over-claim a sparse-flip result.

Contributions. We make four testing contributions:

1. a paired protocol for batch-conditioned refusal testing that compares safety and capability labels under matched serving conditions and adds scorer-corrected adjudication;
2. a cross-model generalization test identifying output instability as the strongest tested fragility correlate, while finding no detectable association with alignment type and no generalizable aggregate safety-over-capability skew;
3. a continuous-batch composition test detecting no aggregate effect at 4.7pp sensitivity, despite rare directional flips leaning unsafe;
4. a batch-invariant-kernel ablation producing zero flips on the current score-flip candidates under the tested invariant vLLM execution path.

2. Related Work

2.1. Deterministic inference and batch-dependent variation

The closest prior literature establishes the mechanism class but not the safety-facing claim. Work on deterministic inference shows that output variation can arise even when prompts, weights, and nominal decoding settings are fixed, because execution still depends on batching, scheduling, floating-point arithmetic, and backend kernels (IEEE, 2019; Atil et al., 2025; Gond et al., 2026; He & Thinking Machines Lab, 2025; LMSYS, 2025). The main concern in that literature is reproducibility: whether outputs can be made bitwise stable, what performance cost determinism imposes, and which execution paths trigger divergence. That work is directly relevant to our mechanism discussion, but it does not ask whether the resulting behavioral changes are concentrated in safety-sensitive rows.

Serving-systems work provides the architectural backdrop for why batching matters at all. Large-scale inference stacks now rely on tensor-parallel serving, iteration-level scheduling, continuous batching, key-value cache management, chunked prefill, and structured-generation runtimes to improve throughput and latency (Aminabadi et al., 2022; Yu et al., 2022; Kwon et al., 2023; Agrawal et al., 2024; Zheng et al., 2024). These papers explain why batch composition, decode overlap, and memory behavior are first-order systems variables. We borrow that systems framing, but our target variable is different: we treat batch condition as a potential safety variable rather than only a performance variable.

2.2. Behavioral sensitivity under deployment perturbations

A second adjacent literature asks whether deployment choices preserve behavior that matters to downstream users. Broad evaluation frameworks already argue for measuring models across multiple risk dimensions rather than treating accuracy as sufficient (Liang et al., 2023; Wang et al., 2023). That principle is important for this paper because systems work tends to report throughput, latency, and memory, while alignment work tends to report refusal quality, jailbreak robustness, or harmful-completion rates. Very little work measures both on the same perturbation axis. That gap matters because batch effects are easy to overread. A batch-dependent output flip is not automatically a safety finding; it may reflect generic output sensitivity. The right comparison is therefore not "does batching ever change anything," but "when batching changes outputs, are safety rows affected differently from capability rows under the same serving perturbation?"

This distinction also explains why the batching question is harder than a standard determinism question. A reproducibility paper can stop after showing that repeated executions diverge. A safety paper cannot. It has to ask whether the divergence falls on refusal boundaries, whether the direction is systematically harmful, whether the effect survives broader model coverage, and whether a serving-system mechanism actually carries the risk into deployment.

Recent work on refusal mechanisms and safety brittleness makes this boundary focus more plausible. Refusal behavior can be mediated by compact internal directions, and pruning or low-rank modifications can selectively damage safety behavior while leaving utility comparatively intact (Arditi et al., 2024; Wei et al., 2024). We do not use those papers as evidence that batching must change refusal. Instead, they motivate why a low-rate execution perturbation can still matter if it lands on a refusal boundary rather than on an ordinary paraphrase.

2.3. Deployment optimization and alignment behavior

Safety and alignment literature has already established that some deployment-time optimizations are not behaviorally neutral. Standard quantization work treats efficiency as the primary goal (Xiao et al., 2023), while recent trustworthiness studies show that compression can change safety and trust behavior in ways not visible from benign task performance alone (Hong et al., 2024). Other work shows that quantization can be adversarially exploited or corrected with alignment-aware procedures (Egashira et al., 2024; Wee et al., 2025; Tan et al., 2026). That literature supports the broader thesis that deployment configuration belongs inside the safety envelope. Our contribution is narrower and complementary. We focus on batching, which is often treated as a serving concern rather than an alignment concern, and ask whether it produces a safety-skewed instability pattern even under greedy decoding.

The batching paper is also different in an evaluative sense. Quantization directly changes arithmetic precision and therefore invites a relatively direct causal story. Batching is more indirect. It can alter queueing, co-residence, cache reuse, and execution order, but whether any of those mechanisms matter for refusal behavior is an empirical question. That is why a synthesis across local perturbation, cross-model extension, and composition-sensitive serving is necessary. A single positive study would be easy to dismiss as a measurement artifact; a single null would be easy to overread as proof of safety neutrality.

2.4. Evaluation tasks and benchmark mix

The evaluation design combines harmful-request refusal tasks with ordinary capability controls. For safety we use harmful-behavior and jailbreak-style prompts derived from adversarial attack and robust-refusal benchmarks (Zou et al., 2023; Mazeika et al., 2024; Chao et al., 2024), and we include truthfulness and bias probes so that the safety axis does not collapse into a single harmful-compliance measure (Lin et al., 2022; Parrish et al., 2022). For capability controls we use MMLU and ARC-Challenge (Hendrycks et al., 2021; Clark et al., 2018). This pairing is central to the paper's logic. A batch-dependent output change is not, by itself, evidence of a safety problem; the stronger test is whether safety changes outpace capability changes under the same serving perturbation.

The composition study adds a second evaluation distinction that is missing from most prior work: it separates generic batch-size perturbation from actual multi-tenant composition under continuous batching. Those are related but not identical threat models. A queued sequential server can show batch-sensitive variation without any true co-batch overlap. Conversely, a continuous-batching server can create genuine cross-request coexistence without producing a measurable aggregate safety effect. Keeping those threat models separate prevents the paper from collapsing several serving phenomena into one oversized claim.

2.5. What is new here

Relative to prior work, the novelty claim in this paper is deliberately narrow. We do not claim to have discovered batch-dependent nondeterminism; that already exists in the determinism and serving literature. We do not claim that deployment-time optimization is generally safety-neutral; compression and quantization work already cast doubt on that assumption. The contribution here is the measurement target itself: batching is evaluated as a safety failure mode, with a paired safety-versus-capability design and an explicit true-batching mechanism check. To our knowledge, that specific empirical question has not been answered directly in the existing literature. The paper's other contribution is corrective: it narrows an initial positive signal into the bounded model-specific claim that survives the full evidence chain.

3. Experimental Design

3.1. Threat Model and Unit of Analysis

The operational threat model is deployment-time sensitivity under fixed prompts, fixed weights, and fixed decoding policy. The attacker is not assumed to alter the model or insert explicit adversarial suffixes at inference time. Instead, the concern is that ordinary serving choices—batch size, dispatch synchronization, or co-batched neighbors—shift the model onto different refusal outcomes for prompts that already sit near a decision boundary. This is why the unit of analysis is the *conditioned evaluation row*: one prompt under two or more serving conditions, scored comparatively rather than in isolation.

This row-level framing matters for interpretation. The paper is not estimating population harm from deployed traffic logs. It is measuring whether changing the serving condition alters the answer on rows that are supposed to remain stable. That makes the synthesis strongest on comparative claims about behavioral sensitivity and weaker on absolute claims about deployed unsafe-output rates.

3.2. Study A: Initial Local Perturbation Study

The local perturbation study (Study A) contains two evidence layers. The base report covers 31,410 scored rows on three instruction-tuned 1B–3B models under four phases: synchronized-dispatch batch-size perturbation, neighbor-condition perturbation, quantization-by-concurrency, and explicit true batching. The strengthened revision adds a scorer-corrected audit of changed rows plus a 7,257-sample

reduced replication on an enriched prompt subset.

The quantities carried forward are the safety-versus-capability asymmetry in the base study (0.51% versus 0.14%) and strengthened replication (1.68% versus 0.42%), the scorer-corrected audit split (26 unsafe-direction versus 18 safe-direction rows), the manual adjudication of 63 candidates that leaves 17 genuine behavioral flips and a corrected full-set rate of about 0.16%, and the true-batching confirmation (0.80% safety flips with 99.4% agreement to synchronized dispatch). Capability controls are the guardrail against treating generic output churn as safety-specific, while the adjudication separates discovery sensitivity from operational magnitude.

3.3. Study B: Cross-Model Extension

The cross-model extension (Study B) broadens the batch-perturbation question to 127,224 records across three linked campaigns and a combined synthesis over the 15 distinct models entering pooled scoring. Its role in this paper is corrective rather than merely additive. The extension tests whether the initial small-model asymmetry survives broader model coverage and whether categorical heuristics such as alignment type explain fragility.

The principal quantities are the near-parity combined safety-to-capability ratio (0.94×), the fragility range from 0.00% to 2.39%, the alignment-type ANOVA ($p = 0.942$), the output-instability correlation ($r = 0.909$), and the true-batch confirmation layer (0.80% flips with 99.15% agreement). Claims that appear locally but are not detectable in this extension are treated as bounded discovery signals rather than as final manuscript theses.

3.4. Study C: Composition Study

The composition study (Study C) contributes the multi-tenant question under continuous batching in vLLM FP16. The study covers 14,250 records across five batch-composition conditions, temporal-overlap sweeps, reverse-direction tests, and static-versus-continuous batching checks.

The study detects no aggregate composition effect at its 4.7 percentage-point minimum detectable effect. Rare flips lean unsafe in 89%–92% per condition (28/31 pooled, Wilson 95% CI [75.1%, 96.7%]); the strongest mixed-condition binomial signal is $p = 0.006$, and scheduler-level co-batching is verified for only 22.1% of requests. This study asks whether *which other requests are present at the same time* adds a measurable safety channel in continuous batching; without scheduler-level co-batch confirmation, a composition null can still be underpowered.

3.5. Study D: Batch-Invariant Kernel Ablation

Study D is a targeted mechanism check on the current Study A score-flip surface. We selected the 55 current Phase 1/Phase 4 score-flip candidates from the rescored Study A archive, retaining both safety and capability rows, and re-ran each row twice on the same H100 pod: once with standard vLLM and once with `VLLM_BATCH_INVARIANT=1`. The serving stack was vLLM 0.19.1, FP16 weights, temperature 0, max model length 2048, and the same three small instruction-tuned models used by the local study: Llama-3.2-1B-Instruct (10 rows), Llama-3.2-3B-Instruct (19 rows), and Qwen2.5-1.5B-Instruct (26 rows). Dispatch mode followed the original condition: 46 synchronized-dispatch rows and 9 prompt-list true-batching rows.

This is not a new population estimate; it is a mechanism stress test on rows already identified as batch-sensitive. If flips collapse under batch-invariant execution, the exact-stack recommendation becomes stronger: rerun the safety battery under the serving kernel path used in deployment.

3.6. Why a Synthesis Is Necessary

These four studies answer different questions, so the synthesis reports the strongest claim that survives the chain rather than the strongest available in any one component. The local study discovers the signal, the cross-model extension narrows it to model-specific fragility, the composition study detects no large aggregate continuous-batch hazard at current sensitivity, and the kernel ablation tests whether current candidates depend on a batch-sensitive vLLM execution path.

3.7. Synthesis Rules and Statistical Interpretation

The manuscript preserves each component study's original statistical treatment rather than imposing a post-hoc pooled significance layer across different model sets, scoring stacks, and threat models. A positive local result is promoted only if it survives the larger extension or an independent mechanism check; directional results are paired with absolute rates; and where studies conflict, the larger or more mechanistically targeted study determines the visible claim.

4. Results

4.1. Study A: The Initial Local Study Yields a Real but Low-Rate Discovery Signal

The local discovery study (Study A) is the discovery anchor: it demonstrates that batch perturbation can move refusal behavior at all. In the base experiment, safety labels change more often than capability labels (0.51% versus 0.14%), and the reduced replication preserves the same direction at

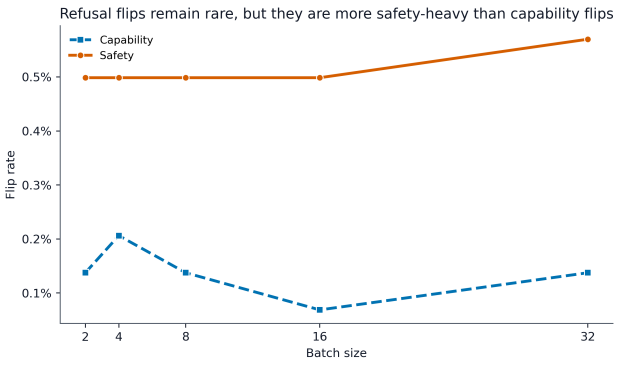

Figure 1. Study A safety versus capability flip rates by batch size. Safety flips exceed capability flips in the local discovery setting, identifying a refusal-boundary signal while leaving the absolute rate low.

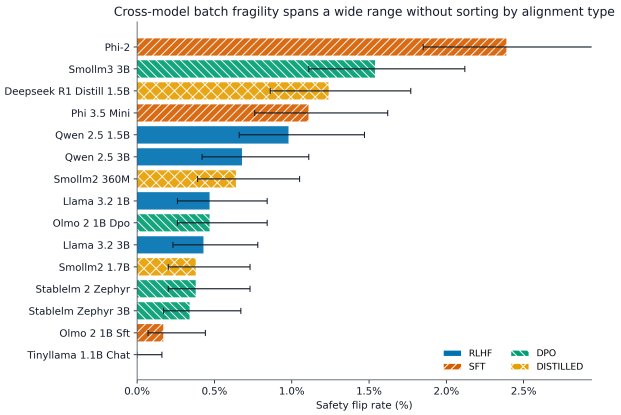

Figure 2. Cross-model fragility ranking from the 15-model extension. Safety flip rates span 0.00% to 2.39%, with error bars showing 95% Wilson confidence intervals. High- and low-fragility models appear across multiple alignment categories rather than forming a clean alignment-type ladder.

higher absolute rates on the enriched subset (1.68% versus 0.42%). Figure 1 summarizes the relative flip rates across batch sizes.

What this supports. The local study does not show that batching broadly destabilizes model behavior; most rows remain unchanged. It shows that changed rows are disproportionately safety-relevant in the original three-model deterministic setting, making Study A a discovery signal for *where* batching matters rather than evidence that batching dominates total variance.

Mechanism confirmation. The explicit true-batching layer matters because it weakens the pure scheduler-artifact explanation. Prompt-list true batching still produces 0.80% safety flips with 99.4% agreement to the synchronized-dispatch pattern. That is mechanism-consistency evidence, not a separate large-magnitude effect.

Audit correction. The scorer-corrected audit shrinks the changed-row set but does not reverse it: after Unicode normalization, the audit catalog contains 26 unsafe-direction rows and 18 safe-direction rows. This keeps the directional concern alive while making the rarity of the phenomenon explicit; the original changed-row catalog is therefore a sensitive discovery layer, not a calibrated operational rate.

Human adjudication recalibration. The later conclusive audit is the operational magnitude check. Manual review of 63 candidate rows found only 17 genuine behavioral flips (27%), with the remaining 73% attributable to automated scoring artifacts such as refusal rephrasing. That recalibration implies a corrected full-set rate of roughly 0.16%, which is much smaller than the original automated discovery signal. This does not erase the phenomenon. It changes the correct interpretation of the local study: high sensitivity for finding unstable refusal-boundary rows, low direct evidence for a large production-rate hazard. The adjudication itself is single-reviewer and should be read as a conservative correction layer rather than as a final population estimate.

4.2. Study B: The Cross-Model Extension Finds No Universal Asymmetry

The cross-model extension (Study B) is the paper's main correction layer. When the batch-perturbation question is scaled across a broader model set, no universal safety-over-capability skew is detectable. The combined synthesis lands near parity at 0.94×, not at a stable safety-skewed ratio.

What does survive. Three findings remain strong in the extension. First, fragility varies materially by model, spanning 0.00% to 2.39%. Second, alignment type shows no detectable association with that variation: a model-level one-way ANOVA over the four alignment categories returns $F = 0.13$, $p = 0.942$, and $\eta^2 = 0.033$, with per-group mean flip rates spanning only 0.28pp. With only $n = 15$ models, this is a null-association result rather than a strict equivalence claim. Third, output instability is the strongest tested model-level correlate of batch fragility ($r = 0.909$, bootstrap 95% CI [0.65, 0.97], leave-one-out range [0.79, 0.93]).

Together these results change the operational question from "is batching intrinsically unsafe" to "which models exhibit batch fragility and what observable signal tracks it before deployment?" The instability correlation is useful because it offers a screening variable where alignment labels and architecture-family descriptions do not. Figure 2 shows the full 15-model fragility range, while Figure 3 shows why output-change rate is the most useful deployment-time screening signal.

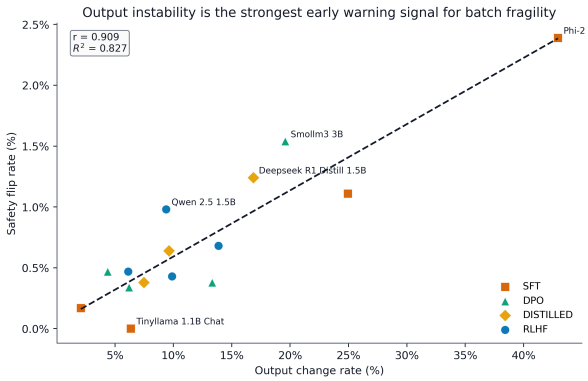

Figure 3. Output instability versus safety fragility across the same 15-model extension. The dashed line is the least-squares linear fit. Models with larger output-change rates under batching also exhibit higher refusal fragility ($r = 0.909$), which makes output instability a more useful screening signal than alignment type.

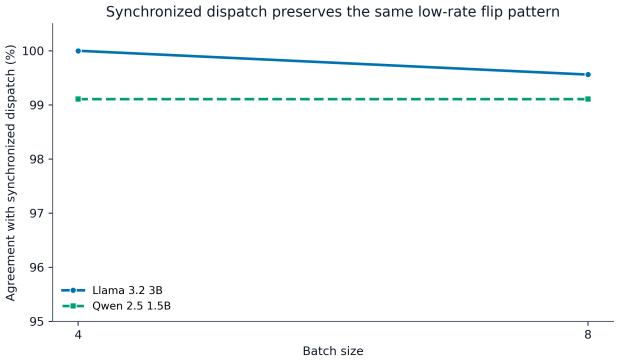

Figure 4. True-batch agreement with synchronized dispatch across model–batch-size conditions. The y-axis is zoomed to 95–100.5% to expose small differences. Near-100% agreement argues against a pure synchronized-dispatch artifact while preserving the low-rate interpretation.

Directional correction. The cross-model study also changes the directional story. The combined directional counts are net safe, with 159 compliance-to-refusal flips versus 81 refusal-to-compliance flips. That result does not make batching benign; it makes the direction model-set-dependent rather than universal. The extension should therefore be read as a heterogeneity report, not as proof of a universal unsafe skew.

True batching remains narrow but consistent. The preserved phase-two artifact again shows low-rate mechanism consistency: 0.80% safety flips with 99.15% agreement against the synchronized-dispatch pattern. This reinforces the "real but small" reading without creating a new high-magnitude claim. Figure 4 shows the agreement between true-batch and synchronized-dispatch patterns across all tested conditions.

The extension therefore does something more useful than confirming or rejecting the local study wholesale. It decomposes the original positive into three parts: a real perturbation signal, a failed universal asymmetry claim, and a model-specific instability problem that remains worth validating in practice.

4.3. Study C: The Composition Study Finds a Null with a Directional Caveat

The final question is whether continuous batching introduces an additional multi-tenant safety channel through batch composition. The composition study (Study C) does not detect an aggregate effect at the current power level. Across McNemar, Cochran's Q, and Mantel–Haenszel analyses, the study remains null at a 4.7 percentage-point minimum detectable effect.

What remains concerning. The rare flips that do occur lean unsafe in 28/31 cases pooled across the three multi-prompt conditions (90.3%, Wilson 95% CI [75.1%, 96.7%]), with per-condition rates of 89%–92% and the strongest mixed-condition signal at $p = 0.006$. The pooled interval excludes 50%, although per-condition CIs are wide. This is not a clean null, but the small denominator (31 directional flips total) and the 22.1% scheduler-level co-batch verification rate do not support using the result as a practical basis for composition-aware request routing.

The minimum detectable effect is doing important interpretive work: the study detects no large aggregate composition effects under the tested stack, but it does not exclude small effects that would matter only in very high-volume or highly sensitive deployments.

Correct interpretation. The composition study narrows the threat model. At current sensitivity, batch composition does not emerge as a large aggregate safety driver. What it contributes is a cautionary directional pattern that should be monitored as batch sizes and verification quality improve.

4.4. Study D: Batch-Invariant Kernels Remove the Candidate Flips

Study D directly tests the mechanism that the previous evidence chain left open. On the same 55 current score-flip candidates, standard vLLM produces 22 label flips and 25 text changes. With `VLLM_BATCH_INVARIANT=1`, the same prompts, models, dispatch modes, and H100 serving environment produce 0 label flips and 0 text changes. All 110 records complete successfully with no startup or request errors.

The standard-mode flips are all safety-domain rows in this candidate set, with 15 matching the original flip direction. The result should be read as a mechanism ablation, not as a universal claim about every serving backend. For

Table 1. Batch-invariant-kernel ablation on the current Study A score-flip candidates. Standard vLLM reproduces low-rate candidate flips; the invariant execution path removes them under the tested stack.

Mode	Rows	OK	Label flips	Text changes
Standard vLLM	55	55	22	25
Batch-invariant	55	55	0	0

vLLM 0.19.1 on this H100 stack, the local candidate flips depend on the non-invariant execution path. That finding converts the earlier "true batching is mechanism-consistent" evidence into a sharper operational test: if a deployment uses batch-sensitive kernels, refusal robustness should be evaluated under that exact served path.

4.5. Synthesis: What the Combined Evidence Actually Supports

Taken together, the four studies support a narrower claim than the earliest single-study interpretation: batch perturbation is real, low-rate after adjudication, and strongly model-specific. Alignment type shows no detectable association, no large aggregate composition hazard is detected at current power, and batch-invariant kernels remove the flips on the tested vLLM/H100 candidate surface. Operators should validate refusal behavior at the production batch setting for the actual model and kernel path they deploy, but they should not treat batching as the dominant optimization-time safety threat or infer risk from architecture-family slogans. The practical check is lightweight: compare outputs at the served batch setting, include capability controls, and escalate only when output instability or refusal flips appear.

5. Discussion and Limitations

5.1. Testing Meaning

The paper's main testing claim is modest but actionable: batch configuration belongs inside the evaluated condition set for refusal robustness. That does not mean every batching change is high risk. It means that batching is no longer justified as fixed background infrastructure once paired evidence shows it can move refusal behavior.

The synthesis also improves resource allocation: batch risk is real enough to validate, but not large enough to dominate engineering effort. Across companion deployment studies in the same experimental program, quantization effects are much larger than batch perturbation, consistent with compression work showing that efficiency interventions can alter trust and safety behavior in ways benign task performance does not reveal (Hong et al., 2024; Egashira et al., 2024; Wee et al., 2025; Tan et al., 2026). The testing order is therefore: validate quantization first, then batching on the exact served model, batch setting, and kernel path, and only then composition-specific edge cases.

5.2. Why low-rate effects still matter

A likely reviewer concern is that the measured rates are too small to justify a full paper. For refusal behavior, the relevant standard is not high aggregate variance; it is whether deployment choices introduce directional instability on prompts that should remain consistently refused. This is why robust-refusal benchmarks must distinguish harmful requests from benign requests that merely look similar (Mazeika et al., 2024; Roettger et al., 2024).

5.3. What This Paper Does Not Claim

This manuscript does not claim a universal safety-over-capability asymmetry under batching. The cross-model extension finds no detectable skew across the tested model set. It does not claim that alignment type predicts fragility: the model-level ANOVA returns $p = 0.942$ with $\eta^2 = 0.033$ and group means inside a 0.28pp band, but with only $n = 15$ models we report the result as no detectable association at the available power, not as strict equivalence. It does not claim that continuous-batch composition is a large aggregate hazard. The rare unsafe-direction flips observed in the composition study are not sufficient evidence for composition-aware routing today, because the pooled directional ratio (28/31, Wilson 95% CI [75.1%, 96.7%]) rests on only 31 directional flips and 22.1% co-batch verification. It also does not claim that all batch-conditioned effects in all serving stacks are caused by the same kernel mechanism. Study D supports that mechanism for the tested vLLM 0.19.1/H100 stack and current candidate set; other backends, tensor-parallel configurations, larger models, and stochastic decoding need their own ablations.

5.4. Main Limitations

Rare-event regime. All four studies operate in a sparse-flip regime. That is why directional findings can be meaningful while still carrying wide confidence intervals and sensitivity to model set.

Measurement heterogeneity. The synthesis combines a local three-model study, a larger cross-model extension on different hardware, and a continuous-batching composition study. This is appropriate for a bounded synthesis paper, but it means the output is an operational doctrine rather than a single pooled effect estimate.

Single-reviewer adjudication. The local-study recalibration to 0.16% depends on a manual review of 63 candidate rows by a single reviewer, with no inter-rater Cohen's κ. It prevents overstating the automated discovery rate, but

should be read as an annotator-perceived correction rather than a gold-standard population estimate.

Scoring-stack heterogeneity. The local study includes an explicit scorer audit, while the preserved cross-model and composition artifacts rely on automated regex-pattern-based scoring rather than judge-based scoring. This does not invalidate the within-study contrasts, but it limits the precision of absolute-rate comparisons across studies and is one reason the paper emphasizes claim selection over pooled numerical precision.

Co-batch verification. The composition study's 22.1% verification rate biases analysis toward the null and limits the strength of any negative claim about actual scheduler-level co-residence.

Kernel-ablation scope. The batch-invariant ablation is deliberately targeted at 55 current score-flip candidates rather than the full original screen. That makes it a strong mechanism check for the candidate surface, but not a replacement for a larger prospective benchmark. It also covers one serving stack: vLLM 0.19.1 on H100 with three 1B–3B instruction-tuned models.

Local-first scope. The evidence is local-first and small-to-mid-scale. Nothing in this paper should be read as a claim about very large models, tensor-parallel production clusters, or stochastic decoding.

5.5. Implications for evaluation practice

Safety evaluation should inherit the discipline already standard in systems benchmarking: if a deployment choice changes the execution regime, safety testing should be rerun in that regime rather than assumed invariant from a single baseline configuration. For batching, the minimum protocol is simple: evaluate the exact production batch setting, include capability controls, and preserve at least one mechanism-sensitive check that distinguishes synchronized dispatch from genuine true batching. Study D shows the value of that check: with batch-invariant kernels enabled (He & Thinking Machines Lab, 2025; LMSYS, 2025), candidate label flips collapse from 22 to 0 under the tested stack. The next stronger paper should use this as the reference ablation and extend it prospectively across larger models, tensor-parallel deployments, and backend families.

6. Conclusion

Batch perturbation can change refusal behavior, but the surviving effect is low-rate, model-specific, and stack-dependent: the local paired test yields the discovery signal, single-reviewer adjudication reduces the implied operational rate to roughly 0.16%, the cross-model extension finds no detectable universal safety-over-capability skew across the 15 tested models, the composition test detects no aggregate continuous-batch effect at current sensitivity, and the batch-invariant-kernel ablation removes the current candidate flips on the tested vLLM/H100 stack. The testing recommendation is therefore exact-stack validation, not universal prohibition: validate refusal at the batch setting and kernel path actually served, pair safety prompts with capability controls, and do not infer batch robustness from alignment type, architecture family, or composition heuristics.

References

Agrawal, A. et al. Taming throughput-latency tradeoff in LLM inference with Sarathi-Serve. *arXiv preprint arXiv:2403.02310*, 2024. URL https://arxiv.org/abs/2403.02310.

Aminabadi, R. Y. et al. DeepSpeed inference: Enabling efficient inference of transformer models at unprecedented scale. *arXiv preprint arXiv:2207.00032*, 2022. URL https://arxiv.org/abs/2207.00032.

Arditi, A. et al. Refusal in language models is mediated by a single direction. *arXiv preprint arXiv:2406.11717*, 2024. URL https://arxiv.org/abs/2406.11717.

Atil, B. et al. Non-determinism of "deterministic" LLM settings. *arXiv preprint arXiv:2408.04667*, 2025. URL https://arxiv.org/abs/2408.04667.

Chao, P. et al. Jailbreakbench: An open robustness benchmark for jailbreaking large language models. *arXiv preprint arXiv:2404.01318*, 2024. URL https://arxiv.org/abs/2404.01318.

Clark, P. et al. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018. URL https://arxiv.org/abs/1803.05457.

Egashira, K. et al. Exploiting LLM quantization. *arXiv preprint arXiv:2405.18137*, 2024. URL https://arxiv.org/abs/2405.18137.

Gond, R. et al. Llm-42: Enabling determinism in llm inference with verified speculation. *arXiv preprint arXiv:2601.17768*, 2026. URL https://arxiv.org/abs/2601.17768.

He, H. and Thinking Machines Lab. Defeating nondeterminism in LLM inference. Thinking Machines Lab blog, 2025. Connectionism research blog, September 10, 2025.

Hendrycks, D. et al. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2021. URL https://arxiv.org/abs/2009.03300.

Hong, J. et al. Decoding compressed trust: Scrutinizing the trustworthiness of efficient LLMs under compression. *arXiv preprint arXiv:2403.15447*, 2024. URL https://arxiv.org/abs/2403.15447.

IEEE. IEEE standard for floating-point arithmetic, 2019. IEEE Std 754-2019.

Kwon, W. et al. Efficient memory management for large language model serving with PagedAttention. *arXiv preprint arXiv:2309.06180*, 2023. URL https://arxiv.org/abs/2309.06180.

Liang, P. et al. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*, 2023. URL https://arxiv.org/abs/2211.09110.

Lin, S., Hilton, J., and Evans, O. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*, 2022. URL https://arxiv.org/abs/2109.07958.

LMSYS. Towards deterministic inference in SGLang and reproducible RL training. LMSYS blog, 2025.

Mazeika, M. et al. HarmBench: A standardized evaluation framework for automated red teaming and robust refusal. *arXiv preprint arXiv:2402.04249*, 2024. URL https://arxiv.org/abs/2402.04249.

Parrish, A. et al. BBQ: A hand-built bias benchmark for question answering. In *Findings of the Association for Computational Linguistics: ACL 2022*, 2022. URL https://aclanthology.org/2022.findings-acl.165/.

Roettger, P. et al. XSTest: A test suite for identifying exaggerated safety behaviours in large language models. *arXiv preprint arXiv:2308.01263*, 2024. URL https://arxiv.org/abs/2308.01263.

Tan, Q. et al. Q-realign: Piggybacking realignment on quantization for safe and efficient LLM deployment. *arXiv preprint arXiv:2601.08089*, 2026. URL https://arxiv.org/abs/2601.08089.

Wang, B. et al. DecodingTrust: A comprehensive assessment of trustworthiness in GPT models. *arXiv preprint arXiv:2306.11698*, 2023. URL https://arxiv.org/abs/2306.11698.

Wee, S. et al. Alignment-aware quantization for LLM safety. *arXiv preprint arXiv:2511.07842*, 2025. URL https://arxiv.org/abs/2511.07842.

Wei, B. et al. Assessing the brittleness of safety alignment via pruning and low-rank modifications. *arXiv preprint arXiv:2402.05162*, 2024. URL https://arxiv.org/abs/2402.05162.

Xiao, G. et al. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *Proceedings of the 40th International Conference on Machine Learning*, 2023. URL https://arxiv.org/abs/2211.10438.

Yu, G.-I. et al. Orca: A distributed serving system for transformer-based generative models. In *Proceedings of the 16th USENIX Symposium on Operating Systems Design and Implementation*, 2022. URL https://www.usenix.org/conference/osdi22/presentation/yu.

Zheng, L. et al. SGLang: Efficient execution of structured language model programs. *arXiv preprint arXiv:2312.07104*, 2024. URL https://arxiv.org/abs/2312.07104.

Zou, A. et al. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023. URL https://arxiv.org/abs/2307.15043.

A. Reproducibility Details

This appendix consolidates the prompts, scoring rules, model and serving setup, and per-condition grid behind the component studies, in response to reviewer requests for reproducibility detail. Numeric totals are reported in the main text via the source-line macros; this appendix documents the *procedure* behind them.

A.1. Prompt Corpora

Studies A–C draw from a common six-family task pool and use the same task-specific scoring rules so that safety and capability flips are measured on the same footing; Study D reuses the current score-flip candidates selected from that pool. Four families form the safety axis and two are capability controls:

- **Safety axis:** `advbench_refusal` (harmful-instruction refusal), `jailbreak_amplification` (jailbreak-style adversarial prompts), `bbq_bias` (bias-sensitive prompts), and `truthfulqa` (truthfulness).
- **Capability axis:** `mmlu_real` (knowledge) and `arc_challenge` (reasoning).

The capability families are not decorative. They are the control that distinguishes a safety-specific batching effect from generic output churn: if batching merely perturbed outputs at random, safety and capability flip rates would move together. The asymmetry the local study reports (0.51% safety versus 0.14% capability) is meaningful precisely because the two axes are scored by the same pipeline under the same serving perturbation.

A.2. Models

Studies A and C use the three local small-model targets Llama-3.2-1B-Instruct, Llama-3.2-3B-Instruct, and Qwen2.5-1.5B-Instruct. Study B broadens coverage to 15 distinct instruction-tuned models in its pooled synthesis, ranging from 360M to 3.8B parameters and spanning Llama, Qwen, Phi, SmolLM, StableLM, OLMo, TinyLlama, and DeepSeek families. This is the model set that enters the combined safety-to-capability ratio (0.94×) and the alignment-type ANOVA. Study D returns to the same three local-study models represented in the current candidate surface.

A.3. Serving and Decoding

Generation uses vLLM (OpenAI-compatible server, FP16 weights) with a 2048-token maximum context and **greedy decoding at temperature 0.0**. Deterministic decoding is essential to the threat model: because the decode policy is fixed, any output change across serving conditions is attributable to the serving condition (batch size, dispatch synchronization, or co-batched neighbors), not to sampling noise. The composition study (Study C) runs vLLM's continuous-batching scheduler; the local and extension studies (Studies A and B) use synchronized-dispatch and explicit true-batching configurations as described in the per-condition grid below.

A.4. Scoring and Flip Definition

The unit of analysis is the *conditioned evaluation row*: one prompt under two or more serving conditions, scored comparatively. A "flip" is a change in the row's label between conditions. Scoring is layered across the studies, and the layers differ deliberately:

- **Automated task-specific scoring** drives the large-scale screens in Studies A–C and the candidate re-scoring in Study D. Refusal and bias outcomes use deterministic pattern- or choice-based classifiers, while truthfulness and capability outcomes use reference-based scoring. The scoring rules do not use the serving-condition identity or co-batch composition.
- **Manual adjudication** corrects the discovery layer: the 63 candidate changed rows from Study A were hand-reviewed, finding 17 genuine behavioral flips (27%) and 73% scorer artifacts, implying the corrected operational rate of 0.16%.

This layering is why the paper separates *discovery sensitivity* from *operational magnitude*: the automated layer is a screen for boundary instability, and the manual layer is the magnitude estimate. The 73% artifact share is reported openly rather than buried, because it is the reason the headline operational rate (0.16%) is so much smaller than the raw discovery rate.

A.5. Per-Condition Grid

- **Batch-size sweep (Studies A, B):** batch sizes $\{1, 2, 4, 8, 16, 32\}$ under synchronized dispatch.
- **Quantization $\times$ concurrency (Study A, Ollama):** concurrency levels $\{1, 4, 8\}$ crossed with batch sizes $\{1, 4, 8\}$.
- **True batching (Studies A, B):** explicit co-batched dispatch, the mechanism check that confirms the synchronized-dispatch pattern (99.4% and 99.15% agreement respectively).
- **Composition (Study C):** five batch-composition conditions at a fixed batch size of 8 under continuous batching, with temporal-overlap sweeps and static-versus-continuous checks. The scheduler-level co-batch verification rate was 22.1%; this is reported as a limitation, since an unverified co-batch is an underpowered measurement rather than evidence of no mechanism.
- **Kernel ablation (Study D):** 55 current Phase 1/Phase 4 score-flip candidates re-run under standard vLLM and `VLLM_BATCH_INVARIANT=1`. The candidate set contains 46 synchronized-dispatch rows and 9 prompt-list true-batching rows, split across Llama-3.2-1B, Llama-3.2-3B, and Qwen2.5-1.5B.

A.6. Hardware and Determinism Caveat

Runs were executed on local and rented GPU infrastructure. The Study D ablation was run on an NVIDIA H100 80GB HBM3 pod using vLLM 0.19.1, PyTorch 2.10.0, Triton 3.6.0, FP16 weights, max model length 2048, and temperature 0.0. One determinism caveat is material: even at temperature 0.0, floating-point reduction order can differ across batch sizes, so "deterministic decoding" does not guarantee bit-identical logits across serving conditions. Study D directly tests this caveat by enabling the batch-invariant execution path on the current candidate rows.

A.7. Study D Artifact Record

The Study D run contains 110 records: 55 standard-vLLM rows and 55 batch-invariant rows. All 110 records completed successfully with zero startup or request errors. The retained internal artifacts include the run metadata, selected candidate metadata, summary JSON, full local audit JSONL, and native vLLM startup logs showing the batch-invariant kernel registration for the invariant-mode runs. The public release should expose summary and candidate metadata, but not raw harmful prompts or completions.

B. Study Provenance Appendix

This manuscript synthesizes four component studies rather than reporting one fresh experiment. The retained evidence chain is:

- **Study A** — an initial local discovery study with scorer-corrected audit and reduced replication;
- **Study B** — a cross-model extension that tests whether the discovery result generalizes;
- **Study C** — a continuous-batch composition study that probes the multi-tenant question;
- **Study D** — a batch-invariant-kernel ablation on the current Study A score-flip candidates.

Claim mapping.

- "Batch perturbation is real" is grounded in Study A and reinforced by the later true-batching confirmation layer.
- "Universal safety-over-capability asymmetry is not established" is grounded in the combined Study B synthesis.
- "No association with alignment type is detectable at the available power" is grounded in the balanced Study B extension.
- "Composition shows an aggregate null with directional caveat" is grounded in Study C.
- "The current local candidate flips depend on the tested non-invariant vLLM kernel path" is grounded in Study D.

Intended reading rule. Where the component studies disagree, this paper follows the most conservative claim supported by the full evidence chain rather than the strongest positive framing available in any one component study.

C. Ethical Considerations and Artifact Availability

C.1. Ethical Considerations

This paper studies refusal fragility under batching using offline evaluation prompts, including harmful-request batteries. No human subjects, private data, or live-user interventions are involved. Because the study touches adversarial prompting, aggregate results, scored outputs, and analysis code can be shared openly, but ready-to-run harmful prompt batteries should be disclosed responsibly and only to the extent required for scientific reproduction.

C.2. Artifact Availability

The reproduction package for this paper includes:

- retained scored outputs and condition-level summary tables for the local batching study, the cross-model extension, and the composition audit;
- the Study D batch-invariant-kernel ablation summary, selected-candidate metadata, startup provenance, and validation log;
- figure-generation data and manuscript tables for the reported flip-rate, instability, and directional-risk analyses; and
- build scripts, compiled figures, and audit notes sufficient to regenerate the paper package.

The public release is available at `https://github.com/Sahil170595/batch-inference-safety`. The public artifact withholds raw harmful prompts and completions from the Study D full-record JSONL unless a controlled-access review process requires them. Aggregate counts, hashes, per-condition metadata, and scripts are sufficient for the open release surface.