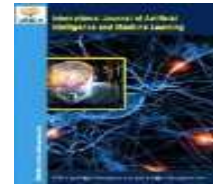




International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: <https://www.svedbergopen.com/>



Research Paper

Open Access

Validating the Intelligent Preload Engine in Production: An A/B Testing Methodology and Deployment Playbook for ML-Driven Prefetching in SPA Funnels

Althaf Khan Pattan

Independent Researcher India, E-mail: altafkhanx6@gmail.com

Abstract

The Intelligent Preload Engine (IPE), described in prior work [1], is a browser-resident machine learning framework that watches in-session behavioral signals and uses them to predict which funnel stage a user will navigate to next, then preloads the relevant route bundle and API data before the navigation occurs. Simulation results in [1] showed a 41% reduction in Largest Contentful Paint on predicted transitions and an estimated 2.1 percentage point improvement in funnel completion rate. Those numbers come from synthetic sessions, not real traffic. This paper describes how to validate them in a live environment. The framework covers A/B experiment design with session-level traffic splitting, metric selection, and sample size derivation; a 13-field OpenTelemetry instrumentation schema built across three telemetry sources; statistical test selection and multiple-comparison correction strategy; a five-stage canary rollout schedule with explicit gate criteria and automated rollback triggers; a confidence threshold sensitivity grid showing how the default 0.60 gate performs relative to alternatives; and a step-by-step deployment playbook from pre-deploy checklist through post-experiment model refresh.

Keywords: A/B testing, real user monitoring, intelligent preload engine, SPA prefetching, canary deployment, OpenTelemetry, Largest Contentful Paint, funnel completion rate, production validation, machine learning, confidence threshold tuning.

I. Introduction

Simulation is a reasonable starting point. It offers control: you can vary one thing at a time, run thousands of sessions overnight, and measure outcomes that real traffic makes unpredictable. The Intelligent Preload Engine described in [1] was built and tested this way. Ten thousand synthetic sessions, a five-stage e-commerce funnel, and a parametric behavioral model calibrated to published benchmarks produced a result set that looked promising: 81.3% next-step accuracy with an LSTM, a 58% LCP drop on correctly predicted transitions, and a weighted gain of 41% when incorrect predictions were factored in. From those numbers, a 2.1 percentage point funnel completion improvement was projected.

The texture of real traffic is harder to replicate. Users leave tabs open and walk away. They hit the back button partway through a funnel, skip stages, or switch between a Wi-Fi connection and a congested 3G signal mid-session. A parametric model approximates some of this but not all of it. The 41% LCP gain might hold in production, or it might shrink to 20% on the actual device mix, or it might split into a strong result on 4G desktop and a near-zero effect on 3G mobile. The only way to find out is to put the system in front of real sessions and measure it.

Validating a prefetching system in production is harder than validating most front-end changes. The key outcome metric, LCP on a route transition, is affected by device speed, network quality, browser cache state, and server response time, all independently of whether any preloading happened. A naive before-and-after comparison absorbs every one of those factors into the signal. A concurrent A/B experiment controls for them by running treatment and control groups at the same time, splitting by session so each user sees one consistent experience, and collecting the same telemetry from both arms. That is the foundation this paper describes.

Section II covers experiment design: traffic splitting, metric selection, and sample size. Section III covers instrumentation. Section IV covers statistical test selection and the risks of false conclusions from multiple comparisons and segment imbalance. Section V covers the canary rollout schedule. Section VI covers confidence

threshold sensitivity. Section VII gives the full deployment playbook. Section VIII covers null and negative results.

II. Experiment Design

A. Traffic Splitting

Traffic is split at the session level. Page-level splitting would let a single user land in both arms across different navigations within the same session, which contaminates the comparison. Session-level splitting assigns each anonymous session to one arm for its entire duration. The assignment is deterministic: take a SHA-256 hash of the session identifier, reduce it to a value in $[0, 1]$, and route sessions below 0.50 to control and those at or above 0.50 to treatment. Because the hash is deterministic, the same session always resolves to the same arm across page refreshes and back-navigations.

The session identifier is a server-generated random string stored in a first-party cookie, separate from any user identity. No personal data enters the routing decision. The IPE JavaScript bundle is loaded for both arms. In the control arm it initializes but does not preload anything. This is intentional: if the control arm did not load the bundle, part of any measured LCP difference could come from removing a JavaScript payload from the critical path rather than from the preloading behavior itself. Loading it in both arms isolates the preloading effect specifically.

Diagram 1: A/B Traffic Split Architecture for IPE Production Validation

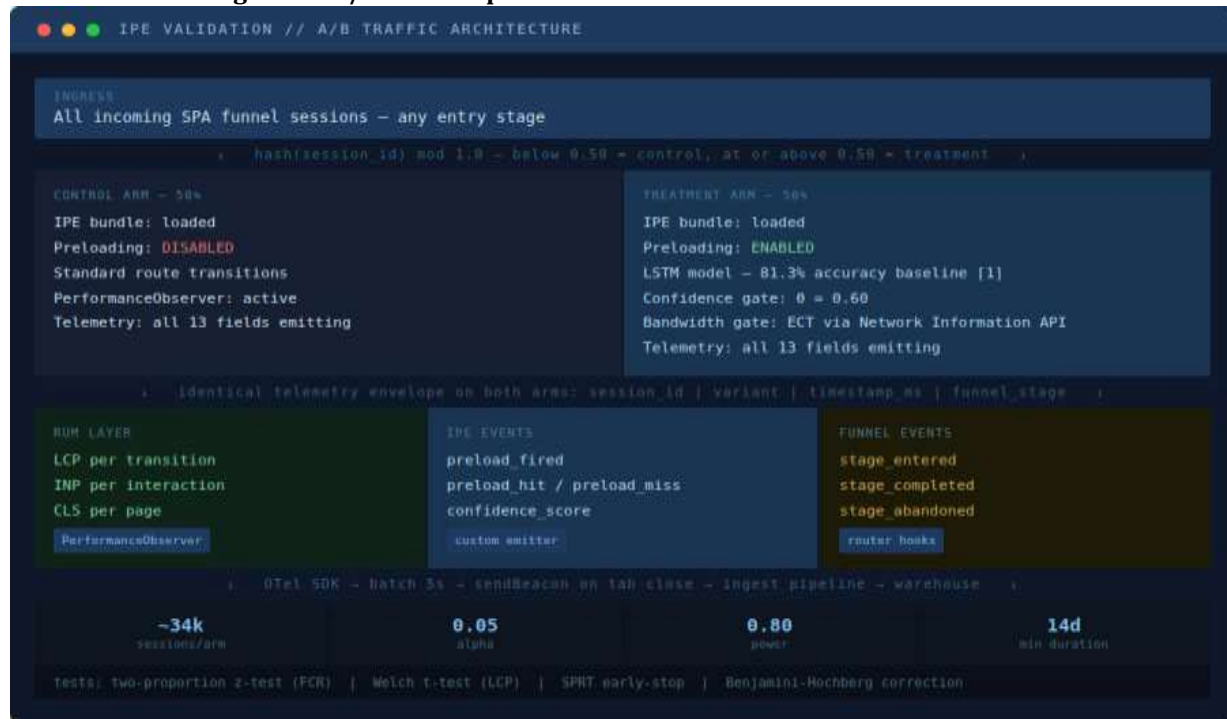


Diagram 1. Session-level 50/50 split via deterministic hash on anonymous session ID. The IPE bundle loads in both arms; only the treatment arm activates preloading. Three telemetry sources feed a 13-field OpenTelemetry schema shared identically across both arms. Statistical requirements are shown in the bottom bar.

B. Metrics

Funnel Completion Rate is the primary metric. It is the fraction of sessions that reach the checkout stage from any funnel entry point. It is chosen as primary because it captures the business outcome the IPE targets, and because it catches trade-offs that LCP alone would miss. A system that improves LCP on fast devices while consuming enough background bandwidth to slow 3G sessions would hurt FCR even if the mean LCP looked better in aggregate.

Transition LCP is the primary performance metric and a secondary outcome in the experiment, measured per route transition using the PerformanceObserver API [4]. Each transition generates an LCP entry tagged with the originating and destination funnel stage. A session-level median is computed before any cross-arm

comparison, which avoids treating correlated within-session transitions as independent observations. Mean, p95, and p99 are all reported, because improvements to the mean that come with worse tail percentiles are not automatically a good outcome for all users.

Interaction to Next Paint is tracked as a tertiary metric. Preloading dispatches dynamic import calls and background fetches. On low-end devices where the main thread is already under pressure, that background work can delay response to user input, which INP captures directly [4]. A deployment that improves LCP while degrading INP has made a trade-off that hurts the users who most need better performance.

C. Sample Size

The minimum detectable effect for FCR is set at 0.5 percentage points against a baseline of roughly 12%, which is a representative e-commerce funnel completion figure [2]. A one-tailed two-proportion z-test at alpha 0.05 with power 0.80 requires approximately 33,800 sessions per arm [13, 14]. At 5,000 sessions per day that is about 14 days, which also covers two complete weekly cycles and reduces day-of-week noise in the data.

For transition LCP, the minimum detectable effect is 150ms on a baseline mean of 2,840ms [1], with a standard deviation estimated around 800ms from published RUM distribution data [3]. A Welch t-test with the same alpha and power settings reaches significance at roughly 4,500 sessions per arm, which falls well within the FCR-driven total. The LCP comparison will be fully powered before the primary analysis concludes.

III. Instrumentation and Telemetry

A. Instrument Before the Canary Starts

Instrumentation failures in the first days of a canary produce data that cannot be used. Sessions with missing fields must either be excluded from analysis, which is acceptable if the missingness is random, or they bias the comparison if the sessions that failed to instrument are systematically different from those that succeeded. The right sequence is to build and fully validate the telemetry layer in staging, verify it across the target browser set, and only then open the canary. Adding telemetry after traffic starts is not a viable option.

Three data sources feed the experiment. The browser RUM layer collects LCP, INP, and CLS per route transition through the PerformanceObserver API. The IPE internal event layer emits a record each time the engine fires a preload, and a follow-up record when the user either navigates to the predicted stage (a hit) or somewhere else (a miss). The funnel event layer tracks each stage entry, completion, and abandonment. All three sources share a common envelope carrying the session identifier, variant label, timestamp, and current funnel stage, which makes the three streams joinable without ambiguity in the analysis warehouse.

B. OpenTelemetry Schema

Events are exported as OpenTelemetry spans with custom attributes using the JavaScript SDK [5]. The SDK batches spans and sends them via HTTP every five seconds. A page visibility change handler triggers a synchronous sendBeacon call on tab close to catch events that would otherwise be lost when the page unloads. Five seconds is a deliberate balance: shorter windows generate more HTTP overhead per session on low-end devices, while longer windows risk losing data from sessions that end before the next flush.

The SDK must load as an asynchronous non-blocking script and process all spans through a microtask queue, never synchronously on the main thread. Instrumentation that runs synchronously during a route transition will inflate the LCP and INP numbers it is supposed to measure. Before the canary launches, a staging run with and without the instrumentation active must confirm that LCP and INP distributions are statistically indistinguishable between the two configurations.

Diagram 2: Instrumentation Schema and Telemetry Event Model

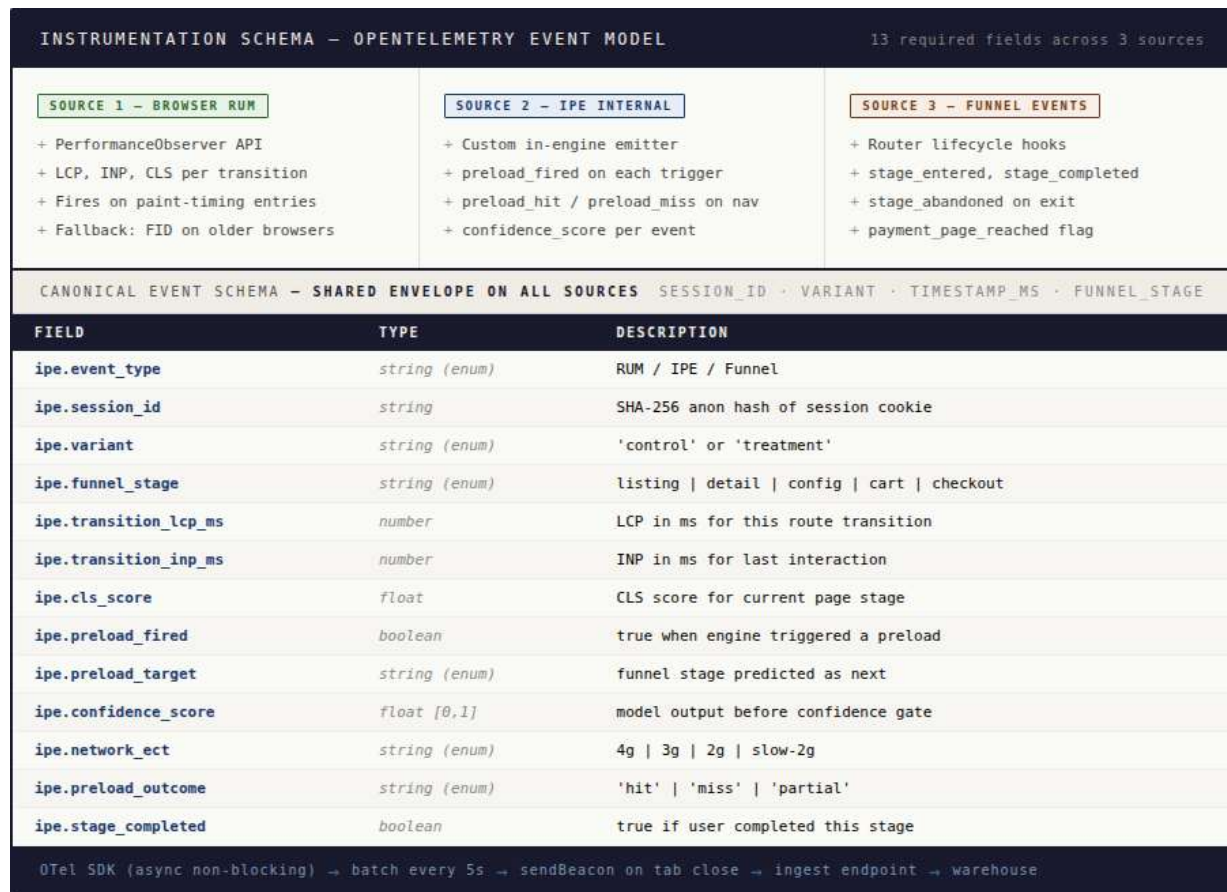


Diagram 2. Three telemetry sources share a common 13-field OpenTelemetry attribute schema. The IPE internal events (preload_fired, preload_hit, preload_miss, confidence_score) enable direct measurement of live prediction accuracy and preload hit rate, the primary diagnostics for threshold tuning after the canary period.

C. Preload Hit Rate as a Live Accuracy Signal

The LSTM model's 81.3% prediction accuracy was measured on a synthetic held-out test set [1]. That figure will not transfer directly to production. Real sessions include back-navigation, multi-tab behavior, and abandonment patterns the synthetic data did not model. The preload hit rate, defined as the fraction of preload_fired events that are followed by a preload_hit, is the live equivalent of that accuracy number. If the live hit rate falls below 60%, the model is triggering preloads for transitions that are not happening, spending bandwidth with no return [6].

Hit rate should be tracked per transition pair rather than as a single aggregate. The model may be well-calibrated on high-volume transitions such as listing to detail and detail to cart while producing poor predictions on skip-step or backward navigations that appeared rarely in training data. Per-pair breakdowns expose those gaps and allow targeted threshold adjustment without penalizing the transitions where the model is already performing well.

IV. Statistical Analysis

A. Test Selection

FCR is a binary outcome per session: each session either reached checkout or it did not. Comparing two independent proportions calls for a two-proportion z-test [7]. The hypothesis is directional, meaning the prediction is that IPE improves FCR rather than merely changing it, so a one-tailed test is used. This concentrates power against the directional alternative at the cost of providing no information about whether IPE might reduce FCR under some conditions. If there is any prior concern that the system could hurt completion rates on constrained devices, switching to a two-tailed test before launch is the more cautious path. Transition LCP is continuous with a within-session correlation structure. Every transition in a session generates an LCP observation, and fast devices produce fast transitions across all funnel stages. Treating each transition

as independent would be pseudo-replication. The analysis instead computes a session-level median LCP and runs Welch t-test on those medians. Welch's variant is preferred over Student's because it does not assume equal variance across the two arms, which is not guaranteed when one arm is loading additional background resources the other arm is not.

Multiple secondary metrics are tested alongside the primaries: INP, CLS, and per-stage completion rates for each of the five funnel stages. Running ten comparisons at alpha 0.05 produces roughly a 40% chance of at least one false positive when the treatment has no real effect. The Benjamini-Hochberg procedure [8] controls the false discovery rate across secondary metrics. The primary FCR and LCP tests keep their unadjusted alpha of 0.05.

Diagram 4: Statistical Analysis Decision Tree for A/B Results

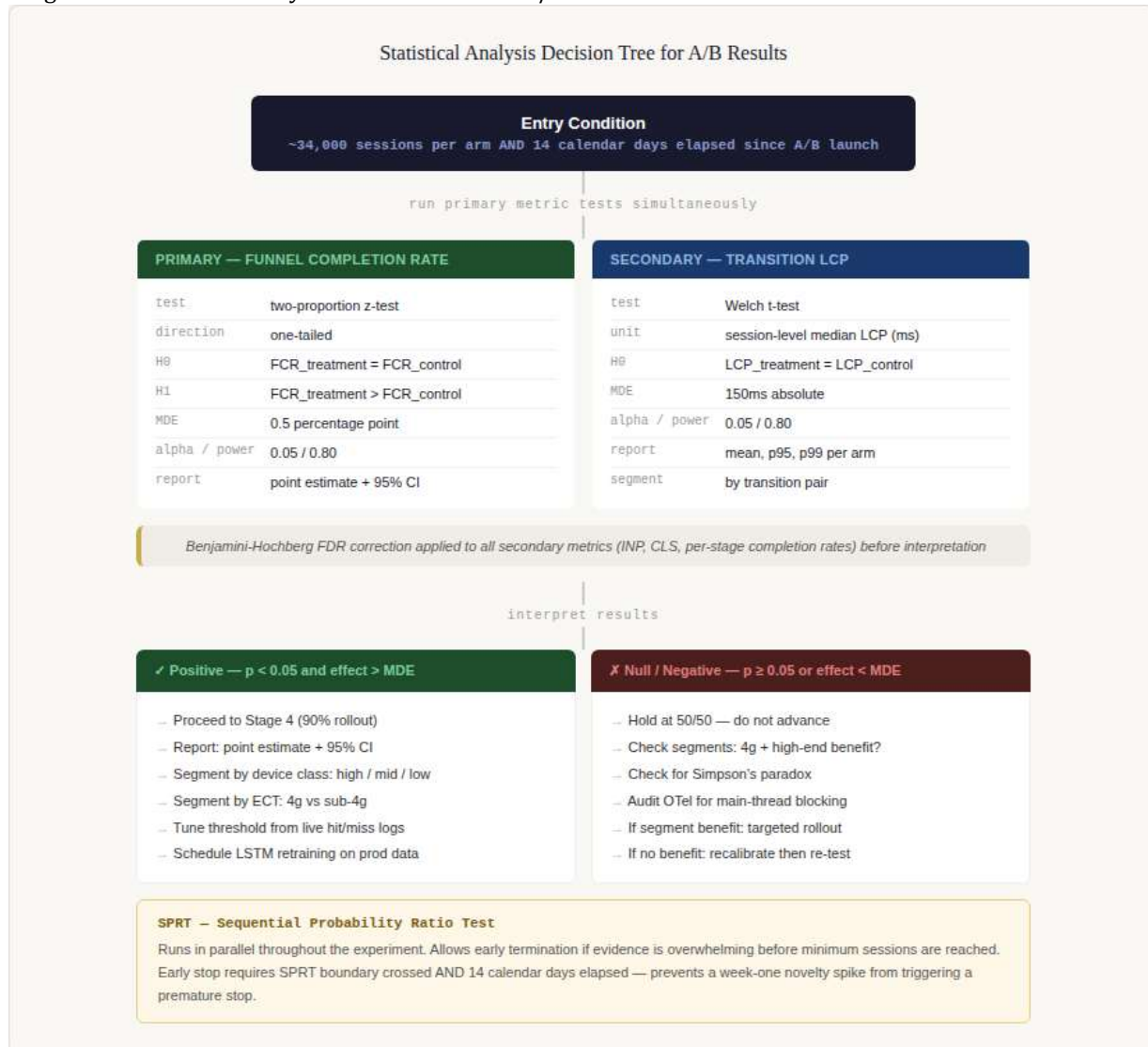


Diagram 4. Decision flow for interpreting A/B results after the minimum session count and calendar duration are both satisfied. Primary tests run first; secondary metrics receive Benjamini-Hochberg correction. Both positive and null/negative result paths lead to specific, actionable next steps.

B. Early Stop and Novelty Effects

A sequential probability ratio test [9] runs alongside the fixed-horizon primary test throughout the experiment. The SPRT allows stopping early if the evidence for a treatment effect becomes overwhelming before the minimum session count is reached. Early stopping on a strongly positive signal is legitimate when the effect is real and large; it is not appropriate when the early positive signal reflects a novelty effect in the opening week rather than a durable improvement.

Novelty effects arise in prefetching systems when users notice unusually fast transitions and complete more sessions simply because the interaction feels different. This effect tends to decay within one to two weeks. The SPRT early-stop criterion is conditioned on two requirements: the SPRT boundary must be crossed and 14 calendar days must have elapsed. A practical check is to compare week-one and week-two FCR within the treatment arm; if week-two is materially lower, the novelty effect is still decaying and the experiment should be extended before any conclusions are drawn.

C. Segment Reporting and Simpson's Paradox

Aggregate results can mislead when the treatment and control arms differ in composition along a dimension that correlates with the outcome. Device class is the most likely source of imbalance. High-end devices produce faster transitions and higher completion rates regardless of any intervention. If the hash-based split by chance assigns more high-end sessions to one arm, the aggregate FCR comparison will favor that arm. This is an instance of Simpson's paradox [10], where an apparent aggregate relationship inverts or disappears when the data is partitioned by a relevant subgroup.

The mitigation is segment reporting. FCR and LCP are reported separately for three device tiers, classified by deviceMemory and hardwareConcurrency from the Navigator API, and two network tiers, 4G and sub-4G based on ECT. A treatment effect that is directionally consistent across all segments supports the aggregate conclusion. An effect that is positive on 4G high-end sessions and negative on 3G low-end sessions should not be reported as an aggregate win; it represents a benefit for one population and a potential harm for another.

V. Canary Rollout Schedule

A. Why Canary Before A/B

Canary deployment and A/B experimentation answer different questions. The A/B experiment tests whether IPE improves user outcomes. The canary tests whether IPE runs correctly in production at all. Running a full 50/50 split on day one conflates the two. A bug in the preloading logic that throws a JavaScript error in a specific browser version will show up in the A/B data as a confounded FCR drop, making the system appear to hurt completion rates when the actual problem is a runtime error. A low-volume canary gives the team time to catch those issues before they reach a significant share of sessions.

Gate criteria and rollback triggers are different mechanisms and should never be confused. Gate criteria are forward-looking; they define what must be true for the rollout to advance to the next stage. A stage that fails to meet its gate stays put until the criteria are satisfied. Rollback triggers are independent and fire automatically the moment a threshold is breached; they do not wait for a gate check and do not require any manual action.

B. Rollback Mechanism

The rollback sets the IPE feature flag to disabled for all sessions and clears any preloaded resources from memory. Four conditions trigger it automatically: a JavaScript error rate above 0.1%; an LCP regression exceeding 10% relative to the control arm; FCR dropping more than 0.5 percentage points below control; or any spike in payment-processing errors. Payment errors are included because a preloading defect that generates a duplicate API call could in edge cases interact with payment intent creation on the back end. The rollback must complete within 60 seconds of the trigger firing, and this timing must be confirmed with a test revert in staging before any production traffic is sent.

Diagram 3: Phased Canary Rollout Schedule and Gate Criteria



Diagram 3. Five rollout stages from 2% canary to 100% full deployment. Gate criteria must all be satisfied before advancing. Rollback triggers are independent of gate criteria and fire automatically on JS error rate, LCP regression, FCR drop, or payment error spike at any stage.

VI. Confidence Threshold Sensitivity

A. Why the Default Threshold May Not Hold

The 0.60 confidence gate used in [1] was grounded in the finding by Domenech et al. [6] that prefetching yields net benefit only when prediction accuracy clears 60%. That figure came from population-level prefetching research and is a sensible starting point, but it may not be the right value for every funnel and user population. If the live preload hit rate at 0.60 falls well below 60%, the model is less accurate on real traffic than it was on the synthetic test set, and raising the threshold will cut wasted bandwidth even as it reduces total preload volume. If the live hit rate is running above 80%, there may be room to lower the threshold and capture more preloading opportunities without excessive misses.

The sensitivity grid in Diagram 5 shows how five threshold values compare across preload rate, hit rate, wasted bandwidth, and net LCP gain, using the simulation from [1] with threshold as the independent variable. These are directional estimates, not production measurements. After two weeks of canary data, the same analysis should be rerun from real preload_hit and preload_miss logs. That production-calibrated version will show where the actual optimum sits for the specific funnel and user base.

B. Network-Stratified Thresholds

Wasted bandwidth carries a higher cost at lower connection speeds. A missed preload on a 4G connection wastes a short background download. The same missed preload on a 3G connection can consume enough of the available capacity to noticeably delay the current page while the unwanted resource loads. A two-tier threshold scheme addresses this: 0.60 for ECT 4G sessions and 0.70 for ECT 3G sessions. The bandwidth gate already blocks preloading below 3G; the two-tier approach adds a tighter confidence requirement within the 3G tier. Whether this adjustment actually helps for a given funnel should be confirmed from A/B segment data rather than assumed.

Diagram 5: Confidence Threshold Sensitivity Grid

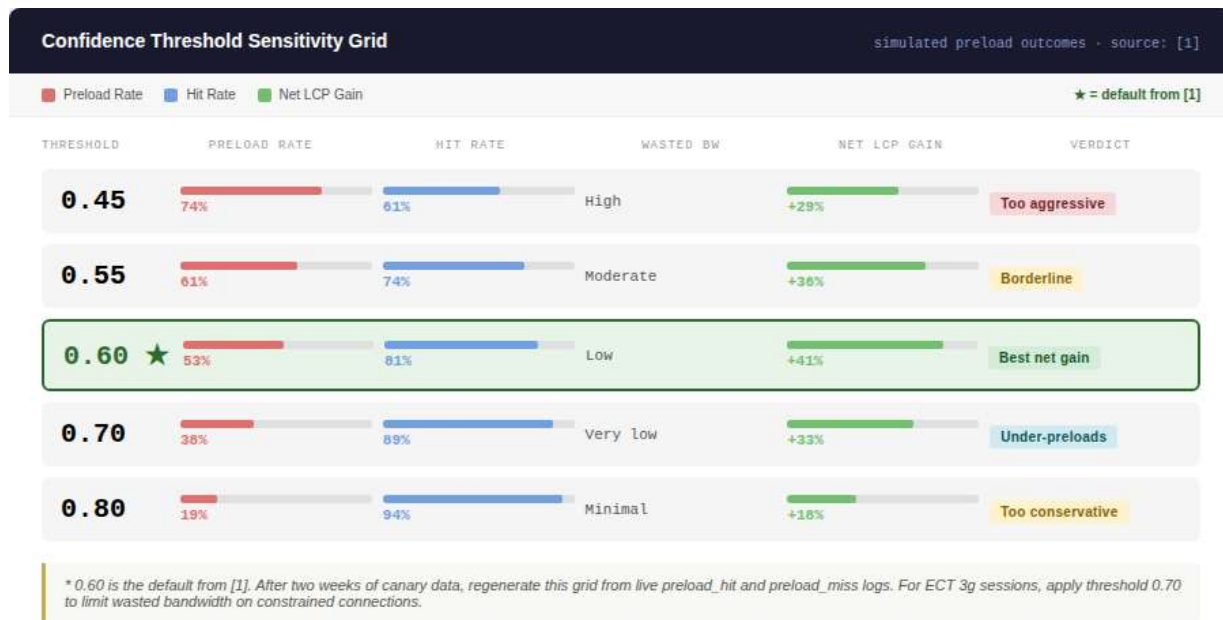


Diagram 5. Sensitivity of simulated preload outcomes to the confidence threshold, recomputed from the simulation in [1] with threshold as the independent variable. The 0.60 default (marked with a star) produces the best net LCP gain overall. After two weeks of canary data, regenerate this analysis from live logs and adjust accordingly.

VII. Deployment Playbook

A. Model Artifact Deployment

The IPE ships two model artifacts: the Markov transition matrix at roughly 2.4KB as a JSON file, and the LSTM weights at roughly 186KB as a binary. Both are deployed to the CDN with one-year immutable cache headers. Artifact filenames include a content hash, so a model update produces a new filename and requires no cache invalidation. The bundle references artifact URLs by version string and is itself cache-busted on every deploy. The LSTM artifact loads lazily, after the session is confirmed in the treatment arm and the user has entered the first funnel stage. Loading 186KB for sessions that abandon at the landing page wastes bandwidth, and the lazy load costs nothing on sessions where preloading never runs. The Markov matrix loads eagerly because at 2.4KB the overhead is negligible and deferring it adds latency to the very first behavioral event.

B. Post-Experiment Model Refresh

The LSTM in [1] was trained on synthetic data. After the A/B experiment, a real session dataset exists containing genuine behavioral patterns, actual transition outcomes, and the full range of user behavior the synthetic model could not anticipate. Retraining on that production data is the highest-leverage action available after a successful result. A retrained model should be validated offline against a held-out session window before deployment, and then launched as a new A/B variant rather than a direct swap into the production slot. For funnels with more than 10,000 sessions per day, monthly retraining keeps the model current with seasonal shifts in user behavior. For lower-traffic funnels, quarterly retraining is more practical because the model needs sufficient new sessions to converge on stable transition probabilities. In both cases, the deployment threshold is that the retrained model must match or exceed the current production model's accuracy on the held-out window before it goes live.

Diagram 6: Full Deployment Playbook

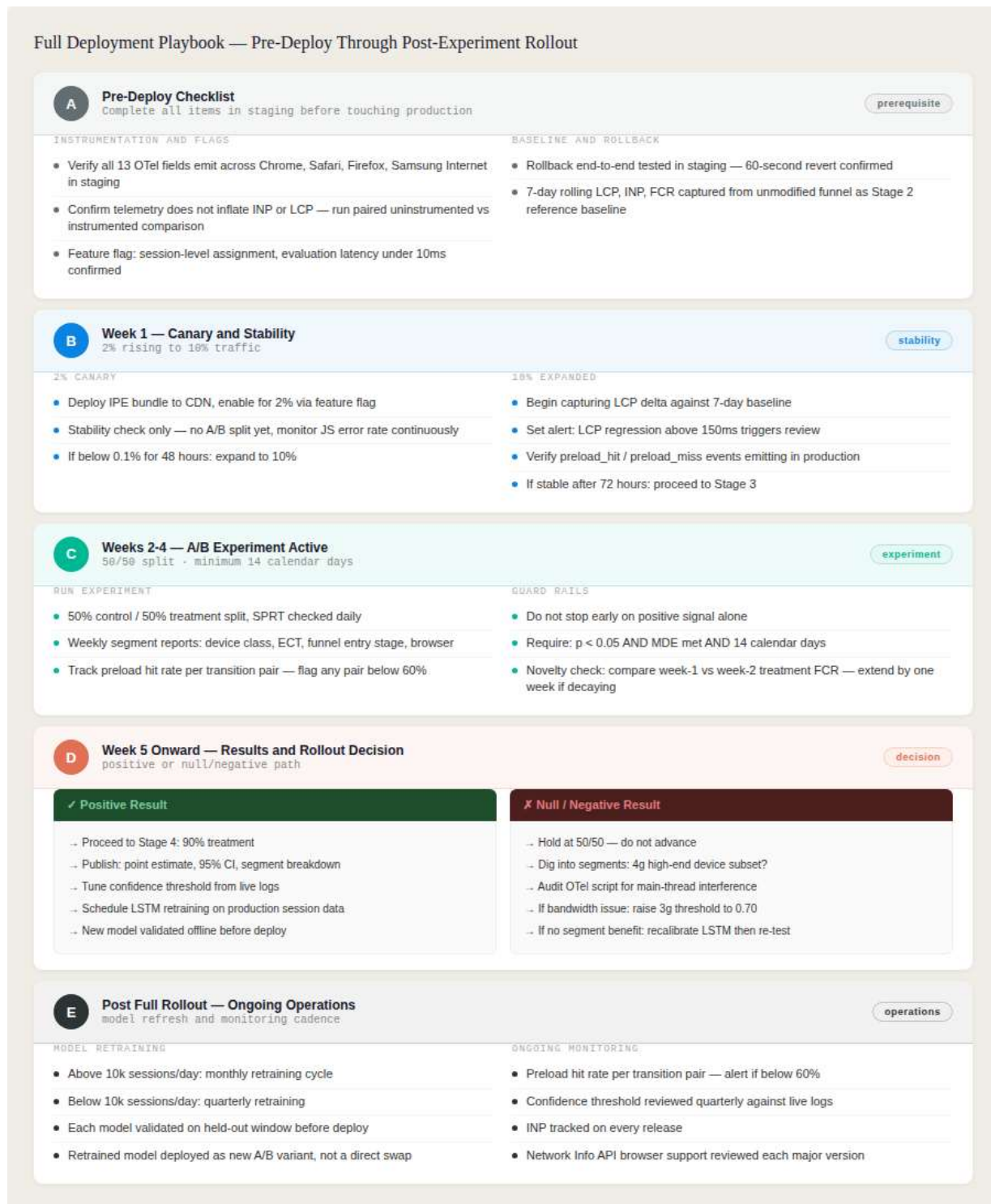


Diagram 6. End-to-end playbook across five phases from pre-deploy verification through post-experiment ongoing operations. Phase D presents both the positive result and the null/negative result paths with explicit, actionable next steps for each.

VIII. Interpreting Null and Negative Results

A null result on the primary FCR metric does not mean the system failed. Several conditions produce a null result even when the IPE runs correctly. If the baseline FCR is already high, above 25%, the marginal conversion return from further LCP improvements tends to be small [11], and the 0.5pp MDE may simply be below what

this funnel can detect at the chosen sample size. Extending the experiment or reporting the LCP improvement separately from the FCR null are both reasonable responses in that situation.

Segment disaggregation often reveals more than the aggregate. An experiment that shows no FCR movement at the aggregate level but a statistically significant LCP improvement on high-end 4G sessions is telling you something specific: LCP on fast devices was already below the threshold where further improvement converts additional sessions. The productive follow-up is to check whether a slower-device or slower-network segment exists where LCP was high enough that the improvement does affect FCR. Reporting both the aggregate null and the segment-level LCP result is more honest and more useful than collapsing everything to a single binary pass or fail.

A genuinely negative result, where treatment FCR lands significantly below control, warrants careful understanding before any action is taken. The most common cause is bandwidth interference on constrained connections: preloading the wrong stage on a 3G session consumes capacity the current page needs, slowing the ongoing render rather than accelerating the next one. That pattern shows up in segment data as a negative treatment effect on 3G sessions and a neutral or positive effect on 4G sessions. If that is what the data shows, the fix is to tighten the bandwidth gate or raise the 3G threshold, not to shut down the system.

A uniform negative result across all device and network segments is unusual. Before drawing any architectural conclusion, audit the instrumentation. Telemetry code that runs synchronously on the main thread during a route transition can inflate INP and block the LCP paint event, producing an apparent LCP regression that has nothing to do with preloading. Ruling out that possibility is the right first step.

IX. Limitations

The sample size derivation assumes a stable funnel population across the experiment window. Promotional events, seasonal traffic surges, or product launches during the run shift the user mix and can break the assumption that control and treatment sessions come from the same distribution. Running this experiment during a major promotional period is generally inadvisable; if it cannot be avoided, a promotional-period covariate should be included in the analysis and the results treated with extra caution.

The Network Information API that the bandwidth gate depends on has no support in Safari or Firefox as of 2024 [12]. Sessions on those browsers fall back to link prefetch only, regardless of confidence level. Browser is therefore a required segment dimension in the reporting plan. Safari and Firefox sessions should be analyzed separately from Chrome sessions because their treatment behavior differs materially, and combining them in an aggregate result dilutes the Chrome-specific effect where the full IPE behavior actually runs.

This paper describes a methodology, not a production result. The outcome of running this framework on any specific funnel will depend on that funnel's user population, device mix, network profile, and stage structure. The numbers used for sample size calculations and threshold analysis come from published benchmarks and the simulation in [1] and should be recalibrated against funnel-specific baseline metrics before the experiment launches.

X. Conclusion

Prior work [1] showed in simulation that the IPE can reduce transition LCP by 41% and project a 2.1 percentage point gain in funnel completion rate. This paper provides the framework for testing those claims against real user traffic. The framework covers six areas: session-level A/B traffic splitting with deterministic hash assignment; a 13-field OpenTelemetry instrumentation schema across three telemetry sources; test selection and multiple-comparison correction suited to the data structure each metric produces; a five-stage canary rollout with explicit gate criteria and automated rollback triggers; a confidence threshold sensitivity analysis to guide post-canary tuning; and a deployment playbook that runs from pre-deploy verification through ongoing model refresh.

The design choices that matter most are the ones that control confounding: splitting at the session level, loading the bundle in both arms, validating instrumentation before the canary starts, and reporting segment results before drawing any aggregate conclusion. These are not choices specific to the IPE. They apply to any front-end performance intervention where the outcome metric is noisy, the treatment acts in the background, and the user population varies across device class and network quality.

A null result from this framework is not a failure. It is a structured answer that identifies which segments the system helps, which it does not, and what the probable causes of the difference are. That answer feeds the next iteration, whether that means recalibrating the confidence threshold, retraining the model on production

session data, tightening the bandwidth gate, or extending the IPE to operate at the component level rather than the route level.

Acknowledgment

The author thanks the web performance, statistics, and machine learning research communities whose published work on A/B experiment methodology, real user monitoring, and sequential inference provided the background this paper builds on.

References

- [1] Pattan AK. Intelligent UI Rendering: Leveraging Machine Learning to Predict and Preload User Journey Steps in E-Commerce Sales Funnels. IJAIDSML [Internet]. 2026 Mar. 23 [cited 2026 Apr. 22];7(1):384-9. Available from: <https://ijaidsm.org/index.php/ijaidsm/article/view/517>
- [2] Baymard Institute, "E-Commerce Checkout Usability Research," 2023. Available: <https://baymard.com/research>
- [3] Google Chrome Developers, "Core Web Vitals Field Data and RUM Distributions," web.dev, 2024. Available: <https://web.dev/vitals/>
- [4] W3C, "Performance Timeline Level 2 (PerformanceObserver)," W3C Specification, 2023. Available: <https://www.w3.org/TR/performance-timeline/>
- [5] OpenTelemetry Contributors, "OpenTelemetry JavaScript SDK," 2024. Available: <https://opentelemetry.io/docs/languages/js/>
- [6] J. Domenech, J. A. Gil, J. Sahuquillo, and A. Pont, "Web Prefetching Performance Metrics: A Survey," Performance Evaluation, vol. 63, no. 9-10, pp. 988-1004, 2006. Available: <https://doi.org/10.1016/j.peva.2005.07.006>
- [7] D. C. Montgomery and G. C. Runger, Applied Statistics and Probability for Engineers, 7th ed. Hoboken, NJ: Wiley, 2018. Available: <https://www.wiley.com/en-us/Applied+Statistics+and+Probability+for+Engineers>
- [8] Y. Benjamini and Y. Hochberg, "Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing," Journal of the Royal Statistical Society: Series B, vol. 57, no. 1, pp. 289-300, 1995. Available: <https://doi.org/10.1111/j.2517-6161.1995.tb02031.x>
- [9] A. Wald, Sequential Analysis. New York: Wiley, 1947. Reprinted by Dover, 2004. Available: <https://store.doverpublications.com/0486615790.html>
- [10] C. R. Blyth, "On Simpson's Paradox and the Sure-Thing Principle," Journal of the American Statistical Association, vol. 67, no. 338, pp. 364-366, 1972. Available: <https://doi.org/10.1080/01621459.1972.10482387>
- [11] Deloitte Digital, "Milliseconds Make Millions: The Impact of Mobile Performance on Retail Consumer Behavior," 2020. Available: <https://www2.deloitte.com/content/dam/Deloitte/ie/Documents/Consulting/milliseconds-make-millions-report.pdf>
- [12] MDN Web Docs, "Network Information API - Browser Compatibility," Mozilla Developer Network, 2024. Available: https://developer.mozilla.org/en-US/docs/Web/API/NetworkInformation#browser_compatibility
- [13] R. Kohavi, D. Tang, and Y. Xu, Trustworthy Online Controlled Experiments: A Practical Guide to A/B Testing. Cambridge: Cambridge University Press, 2020. Available: <https://doi.org/10.1017/9781108653985>
- [14] Geetanjali, & Singh, A. R. (2025). Precision and smart farming: An approach to efficient resource management in agriculture. Journal of Artificial Intelligence in Agriculture and Applied Research, 1(1), 25-33. <https://agrpjournals.com/journals/journal/index.php/jaiaar/article/view/6>