BV-SALA Savings Layer: Verifiable Measurement of Cost Savings in Large Language Model Applications

(Open-Core Edition)

Joan-Marc Fisa
Founder, BiVelio Inc
jmfisa@bivelio.com https://savings.bivelio.com

15 November 2026 (v2.3-r1; open-core release 1.2.0)
Preprint — technical report

Abstract

Cost claims for large language model (LLM) applications require an explicit workload, cost basis and counterfactual. BV-SALA is an open-core SDK and gateway integrating established optimization mechanisms with usage accounting. We specify provider usage as disjoint billable categories, distinguish observed costs from reference-valued and estimated counterfactuals, and separate mechanism diagnostics from experimental attribution. Recomputing public September 2026 artifacts yields a pooled list-price cost reduction of **31.50 %** (percentile family bootstrap interval 25.74–36.75 %) on `gpt-4o-mini`, and **75.08 %** (69.31–79.60 %) on `claude-sonnet-5`. The corresponding equal-family means are 22.88 % and 70.71 %; these answer different questions. Approximately **41.08 percentage points** of the Anthropic pooled result correspond to a descriptive net cache-pricing contrast, not an independently identified causal effect. Additional public November summaries report cost reductions from prefix reordering, response reuse and a model cascade; these are exploratory and do not establish quality non-inferiority. We identify explicit limits of line-based prefix reordering, replay equivalence and permissive answer matching. The contribution is a measurement and reproducibility framework, with results bounded to the published workloads and prices. No invoice reconciliation, general semantic-preservation guarantee or independent live replication is claimed.

Keywords: LLM cost accounting; prefix caching; response reuse; cluster bootstrap; counterfactual evidence; reproducibility.

1 Introduction

LLM applications incur provider usage charges as well as infrastructure, orchestration and verification costs. Caching, compression and model routing may reduce some of these costs, but a percentage alone does not identify what was measured, which baseline was used, or whether answer quality was preserved. A list-price usage reduction is also distinct from a reduction in a reconciled invoice.

BV-SALA combines established interventions in an SDK and local gateway. The contribution evaluated here is their integration with an auditable measurement contract, rather than the invention of response caching, prefix caching or routing. Four diagnostic ledgers record avoided calls, eliminated tokens, compressed tokens and provider-cached tokens. Their counts have different meanings and are not independently additive causal effects.

This revision makes three distinctions explicit: (i) cost aggregation versus equal-family performance; (ii) accounting arithmetic versus causal attribution; and (iii) successful software checks versus task-quality guarantees. It corrects the original estimator description to match the public analysis code. Historical SDK 0.4.31 experiments remain evidence about that version and workload, not a live revalidation of every feature of release 1.2.0.

The research questions are: what cost contrast is supported by the public artifacts; what uncertainty the recorded sampling design permits; which mechanism effects can be separately identified; and which operational and quality claims require further evidence. The November experiments are treated as exploratory. The term *preregistered* is reserved for a protocol with a verifiable record preceding the relevant data collection.

2 Related work

Token pricing is a commercial decision, not a property of the Transformer. One line of work reuses computation: PromptCache [4], CachedAttention [5], SGLang [6] and CacheBlend [7] study attention or KV-cache reuse, and Anthropic and OpenAI document provider-side prompt caching [8, 9]. These mechanisms can lower usage charges, but an experiment must distinguish caching already present in both arms from caching *enabled* by the intervention—the attribution problem of Section 4.4.

A second line reduces prompt tokens: LLMLingua [1], LLMLingua-2 [2] and gist tokens [3]. A third reuses responses: GPTCache [10], SCALM [11], MeanCache [12] and vCache [24] address freshness, privacy and per-prompt error bounds; P-CHR AUC and Operational Retention Rate evaluate deployment [27]. Routing systems such as FrugalGPT [13], RouteLLM [14] and RouteNLP [25] select among models and calibrate escalation. Serving-efficiency work—vLLM [15], SGLang [6], speculative decoding [16]—is orthogonal, as this paper does not measure throughput. OpenTelemetry's GenAI conventions [17] provide a vocabulary, not cost attribution.

BV-SALA combines established mechanisms with request-identity guards, separate ledgers, abstention controls and economic gating; its differentiation is the integration of accounting and safety, not any single mechanism. vCache is the closest comparator for semantic caching; this work does not establish equivalent guarantees or hit rates. Structured-data rewriting requires task- and model-specific quality evaluation: Notation Matters shows token reductions can reduce tool-calling accuracy [26]. The uncertainty discussion uses established terminology [18, 19] without asserting calibration-chain traceability.

3 System model and accounting object

3.1 Notation and cost basis

We distinguish a **logical request** from a **provider execution**. A logical request may cause no execution when a response is replayed, one execution, or several executions when routing, escalation, or retries are involved. All reported monetary amounts are valued at a specified version of the list-price catalog. They are not negotiated or reconciled invoices.

For execution e, let the normalized usage vector be

$$x_e \;=\; \big(n_{u,e},\, n_{r,e},\, n_{w5,e},\, n_{w1,e},\, n_{o,e}\big), \tag{1}$$

where the components are uncached input, cache-read input, five-minute cache-write input, one-hour cache-write input, and billable output tokens. These are **non-overlapping billing categories**; write categories not offered by a provider are omitted. Let $\pi_{e,k}$ be the list price in USD per million tokens for category k, the model and endpoint used by execution e, the applicable billing mode, and the stated catalog version. Define

$$C_e \;=\; 10^{-6} \sum_{k \,\in\, \{u,r,w5,w1,o\}} \pi_{e,k}\, x_{k,e} \;+\; g_e, \tag{2}$$

where g_e contains any separately itemized charges included in the stated scope. The experiments below concern provider token charges; ancillary charges and local infrastructure costs that were not measured are outside that scope, not evidence of zero cost. Additional billable categories require an explicit extension of the usage vector and catalog. Batch and cache prices must be represented by the applicable rates, rather than by adding purported discount percentages.

The mapping from raw counters to x_e is provider-specific. If a raw input counter includes cache reads or writes, those components are subtracted before assigning $n_{u,e}$; if it excludes them, they are not subtracted again. Reasoning tokens are included exactly once in billable output according to the provider's counter semantics. A separately reported reasoning count is not an additional charge when already included in the output total. A missing counter or price is not silently replaced with zero. Partial observations must retain their incompleteness.

For an arm with execution set E, its cost is $C_{\text{arm}} = \sum_{e \in E} C_e$. This sum includes every billable attempt, including cheap-model attempts before escalation, repeated attempts, and failed attempts when they incur a charge. Known executed costs and missing or reconstructed components must be reported separately. An avoided execution has zero actual provider-execution cost, but its counterfactual cost requires separate evidence. The catalog implementation stores token prices per 1,000 tokens; this is equivalent to Equation (2) only with the corresponding unit conversion.

3.2 The layer and its report

Definition 1 (Stateful request policy)**.** *The layer is a stateful policy. Given a logical request r and current state s, it may return a stored response or initiate one or more provider executions, possibly adapting later decisions to earlier responses. Its output consists of the response, a report R, and updated state s'. State includes relevant cache contents, freshness metadata, configuration, and execution history. Provider responses and cache behavior may be stochastic or depend on state outside the layer's control.*

This model does **not** assume that identical request bodies imply identical provider responses or usage counters. Cold and warm prefix caches alone can change usage categories for otherwise identical requests. Exact request matching establishes a key match under the implemented identity policy; it does not, by itself, establish that a cached answer remains correct, authorized, or fresh. Likewise, a reversible serialization, a declared delimiter, or a valid output schema is not

a proof of task-level semantic equivalence. Each mechanism needs explicit applicability conditions and a stated quality-evaluation scope. Unsupported transforms should decline without changing the request.

Definition 2 (Report)**.** *A report R records logical requests, provider executions, normalized usage, activity ledgers, costs, provenance, and traces. The four activity ledgers are avoided calls, eliminated tokens, compressed tokens, and provider-cached tokens. They describe different activities and units. They are not assumed to be four additive monetary or causal contributions. Cost reporting distinguishes baseline, actual, ratio, catalog version, and evidence basis.*

Definition 3 (Evidence provenance)**.** *For each reported quantity,* `measured` *denotes an observed counter, or a cost obtained from observed usage and specified catalog prices;* `reference` *denotes a value borrowed from an explicitly identified prior execution under a stated matching rule;* `estimated` *denotes a reconstructed value; and* `unmeasured` *denotes a required observation that is absent.*

These labels are a reporting convention, not probabilities of correctness. Baseline cost, actual cost, and any attributed effect retain separate provenance. A derived quantity cannot be assigned stronger provenance than the weakest necessary input without additional evidence. Mixed observed and reconstructed components must be disclosed. Missing values must not be presented as complete totals. In particular, `measured` usage valued at list prices does not imply invoice reconciliation, semantic correctness, or causal attribution.

3.3 Accounting and factorial attribution

For positive baseline cost, the observed aggregate reduction is

$$R \;=\; 1 - \frac{C_{\text{opt}}}{C_{\text{base}}}. \tag{3}$$

If $C_{\text{base}} = 0$, this ratio is undefined and is reported as unavailable. Negative reductions are permitted: an intervention can increase cost. This identity depends on complete, consistently scoped arm costs, not on a partition of savings among activity ledgers.

For attribution, consider a specified two-factor experiment. Let C_{ab} denote the cost for the same workload and price basis with factor states $a, b \in \{0, 1\}$. The factors may, for example, be response deduplication and explicit prefix-cache markers. Observed finite-run cell costs and population expected cell costs must not be interchanged without specifying which is intended.

Proposition 1 (Two-factor cost identity)**.** *Define the cost interaction*

$$IC \;=\; C_{11} - C_{10} - C_{01} + C_{00}. \tag{4}$$

Then the combined cost reduction satisfies

$$C_{00} - C_{11} \;=\; (C_{00} - C_{10}) + (C_{00} - C_{01}) - IC. \tag{5}$$

If $C_{00} > 0$, the corresponding reduction ratios satisfy $R_{11} = R_{10} + R_{01} - IC/C_{00}$, where $R_{ab} = (C_{00} - C_{ab})/C_{00}$.

Proof. Substitution of the definition of IC cancels C_{10} and C_{01} and leaves $C_{00} - C_{11}$. Division by a positive C_{00} gives the ratio identity. □

This is an exact algebraic contrast, not a claim of algorithmic novelty or a guarantee of causal identification. Positive IC means that the combined saving is smaller than the sum of the two baseline-relative standalone savings. Its sign is therefore opposite to the residual in a sum-of-savings equation. Estimating a causal interaction requires an appropriate assignment

and state-control protocol; uncertainty requires independent replicated units or other justified inference. Traces showing that two mechanisms ran do not observe their absent-intervention counterfactuals. Neither those traces nor the largest interaction observed in a finite sample establish a bound for future workloads. Routing, output-length changes, retries, and other effects also need explicit contrasts if attributed.

3.4 The open-core contract

In the described distribution, community mode grants local feature entitlements without a license key or a control-plane dependency. Availability of a feature is distinct from whether it is enabled by its default configuration: opt-in mechanisms remain opt-in. With no cloud key, the local layer sends no BiVelio telemetry; provider API traffic needed to serve requests is separate. A supplied invalid, expired, or revoked key fails the license gate. With a valid cloud key, the additional telemetry surface is the per-request metadata specified by the Cloud Services Terms, excluding prompt content, completions, and provider credentials. Community mode has no cloud revocation channel by design (Section 9). These are distribution and configuration properties, not experimentally measured cost reductions.

3.5 Metering the three protocols

The gateway observes usage from Anthropic Messages, OpenAI Chat Completions/Responses, and the Gemini protocol. For Gemini, `:generateContent` and `:streamGenerateContent`, including their supported streaming transports, are forwarded while the observer maps `usageMetadata`. The mapping separates cache reads from uncached input and accounts for billable thinking output according to the native protocol's counter semantics. Other protocol adapters require their own mapping; similarly named fields must not be assumed to have identical inclusion rules. A stream without the required final usage remains `unmeasured`, or partially measured where specific counters are available. Protocol-fidelity tests validate the tested formats and forwarding behavior; they do not establish savings or invoice accuracy.

3.6 The cache-break audit

The deterministic, local audit scans the cacheable `system` header for ISO 8601 and RFC 2822 timestamps, UUID v4 values, and epoch timestamps, and records a `cache-audit` trace and a warning keyed by the detected pattern. Such content can indicate a variable prefix that reduces cache reuse. A pattern match does not prove that the value changes between requests or that moving it preserves meaning. A changed prefix can invalidate reuse from the affected position onward; the actual outcome depends on the provider's cache rules and eligible stable prefix. The audit itself is diagnostic and does not rewrite the prompt. No general claim that this is the dominant production failure mode is made without workload data.

3.7 Levers inside the batch

The batch surface shares elision by supersession and optional tool-output compaction with the generation pipeline. This parity claim is limited to the transformations and options common to both paths; it does not cover every generation-only lever. Byte-level parity tests can establish that the two paths produce the same transformed payload for tested inputs and options. They do not establish task-level losslessness of either path.

Opt-in intra-batch deduplication submits one execution for matching final provider bodies and replays its result to the matching logical requests. Its applicability requires that the workload's contract permits response replay under the relevant identity, authorization, sampling, and freshness conditions. It is not equivalent to obtaining independent fresh samples, and identical bodies do not guarantee identical fresh provider responses. The deterministic adapter benchmark

validates execution counting under its fixture. Counts, input tokens, and monetary savings are reported separately; equal percentage reductions require additional workload and pricing conditions. Each submitted execution is counted once in actual cost.

3.8 The cacheable-prefix reorder

With explicit opt-in through `savings.cachePrefix`, the described transform moves matching lines from a leading string `system` message to the end of the header, splits the stable and moved portions, and marks only the stable portion as the cacheable prefix. It declines when there is no movable content or no stable content. The reviewed trace records the number of moved lines and detected patterns, not the moved text or exact offsets.

The resulting stable prefix can enable provider-cache reuse when length, TTL, placement, request compatibility, and provider state permit it. Byte stability is a mechanical precondition, not a guarantee of a hit. Keeping the same lines in a different order does not guarantee equivalent model behavior, so task-level quality requires evaluation. The provider supplies the cache tariff; the intervention may enable its use. Those two facts must be kept distinct when interpreting savings.

4 Measurement methodology

4.1 Design

The live SDK comparisons execute baseline and instrumented arms on paired constructed content. Families group related requests; the gateway analysis instead uses sessions and a reconstructed counterfactual. The two designs have different evidence bases. Both arms of a live comparison use a common price-catalog version, while execution-specific model and cache rates are retained. Repeated calls are affected by state. A reproducible protocol must specify arm order, cache initialization and warming, TTL, isolation or sharing of cache keys, model identifiers, sampling parameters, scheduling, and failed-attempt handling. Pairing alone does not remove temporal confounding or cache spillover. Where such controls are not established, the comparison is an observed paired contrast on the executed workload rather than an isolated causal effect of one mechanism.

4.2 Estimator

Let b_f and o_f be total baseline and optimized costs for family f, including all executions in the respective arm. The primary economic estimator implemented by the paired benchmark is the pooled ratio of cost sums,

$$\widehat{R}_{\text{pooled}} \;=\; 1 - \frac{\sum_{f=1}^{F} o_f}{\sum_{f=1}^{F} b_f}, \qquad \sum_f b_f > 0. \tag{6}$$

It measures reduction in the aggregate cost of that workload. When every $b_f > 0$, it is a baseline-cost-weighted average of family reductions,

$$\widehat{R}_{\text{pooled}} \;=\; \sum_f w_f \, R_f, \qquad w_f \;=\; \frac{b_f}{\sum_j b_j}, \qquad R_f \;=\; 1 - \frac{o_f}{b_f}. \tag{7}$$

A distinct secondary descriptive statistic is $\widehat{R}_{\text{macro}} = F^{-1} \sum_f R_f$, which gives each family equal weight. The two statistics answer different questions and need not agree. The macro statistic is undefined for a family with zero baseline cost unless an explicit exclusion policy is stated; it is not substituted for the pooled estimator. Neither statistic automatically predicts the reduction for a future workload with a different mix of requests. Financial break-even arithmetic additionally

assumes that the reduction applies to the priced bill and that all relevant incremental charges are included.

4.3 Uncertainty

The percentile cluster bootstrap samples F family indices with replacement for each of $B = 10{,}000$ resamples, retains the complete paired records of each sampled family, and recomputes Equation (6) from the resampled cost sums. The interval consists of the empirical 2.5th and 97.5th percentiles. The historical gateway report describes session-level resampling; its interval is not independently recomputed here because the required session-level costs were not located in the public artifacts. A zero-denominator resample is undefined and must be disclosed rather than assigned an artificial ratio. The seed, quantile convention, resample count, and input-artifact identifier are required for exact numerical reproduction.

Resampling families preserves observed within-family dependence. Its inferential interpretation requires a population of exchangeable independent family units, or another explicit justification for the resampling model. Shared provider-cache state, execution days, accounts, or scheduling can induce dependence between families that this procedure does not capture. The interval does not incorporate uncertainty from unmeasured costs, reconstructed counterfactuals, price errors, or uncontrolled state. It is a percentile interval, not BCa [22, 23], and does not correct an unsupported design or a mismatch of estimands. No universal $O(1/F)$ bias claim is made for the mean of family ratios.

4.4 Interpreting the Anthropic comparison

The instrumented arm enables explicit `cache_control` markers and reports more cache-read usage. This supports a descriptive account of how its observed usage differs from the baseline. Comparing cache-read counts alone does not isolate a causal saving: writes, uncached input, output length, avoided executions, and cache state may also change. The previously reported approximately 41-percentage-point attribution must therefore be treated as a descriptive calculation under its specified counterfactual assumptions, not as a separately identified causal effect. A defensible numerical attribution requires its formula, price basis, and contributing usage records. A factorial comparison on another model or workload does not retrospectively identify this effect.

4.5 What the method cannot do

These comparisons do not establish invoice reconciliation, generalization beyond the stated workload and resampling model, or quality and latency non-inferiority. Observed agreement on a small answer-key test is a useful exploratory result, but non-inferiority additionally requires a prespecified endpoint, margin, sampling design, and appropriate one-sided interval or test. Activity counters, cost contrasts, semantic quality, and operational correctness are separate claims with separate evidence requirements (Section 8).

5 Experimental design

The primary artifacts are dated 28 September 2026 and identify SDK 0.4.31. The OpenAI workload contains 200 logical requests in 100 families: 100 original GSM8K-derived questions, 60 exact repetitions, 10 surface paraphrases, 10 lexical paraphrases, 10 numeric variants and 10 word variants. The latter two categories change the expected answer; they are not answer-preserving paraphrases. Thus the realized mix is 50 % original, 30 % exact-repeat, 10 % paraphrase and 10 % answer-changing variation across the workload, not within every family. Family sizes vary from one to six requests.

The Anthropic artifact contains 62 requests in 31 families: 31 originals, 19 exact repetitions, and three examples of each of the other four categories. Family sizes vary from one to five. For both models, logical requests must be distinguished from provider executions: a served cache hit remains a logical request but does not execute a new provider call. The paired families are the units resampled in the published bootstrap.

The data permit computational reproduction of the reported aggregate analysis. They do not authenticate the provider origin of every stored counter, independently rerun the APIs, or establish population representativeness. Shared provider-cache state may introduce dependence across families; the reported bootstrap conditions on the observed run and does not reconstruct cache warm-up under resampled request orders.

6 Results

6.1 Recomputed historical SDK experiments

Table 1 reports the ratio of aggregate costs used by the public code. Costs are USD at the recorded list-price basis. Bootstrap intervals use 10,000 family resamples and seed `0xa11`; they are percentile intervals, not BCa intervals.

Model	Baseline USD	Optimized USD	Reduction [95 % interval]
`gpt-4o-mini`	0.0412380	0.0282477	**31.50 %** [25.74; 36.75]
`claude-sonnet-5`	0.3152840	0.0785669	**75.08 %** [69.31; 79.60]

Table 1: Offline recomputation of published September 2026 artifacts; no new provider calls.

As a secondary estimand, the equal-family mean is 22.88 % [17.56; 28.10] for OpenAI and 70.71 % [65.69; 75.40] for Anthropic, using the same bootstrap draws. This difference is a weighting effect, not an arithmetic contradiction. The original manuscript described this secondary estimand while reporting the pooled result.

In the Anthropic artifact, the baseline has no cache reads or writes. The optimized arm records 72,447 read tokens and 1,767 written tokens across 42 provider executions for 62 logical requests. The pricing contrast

$$D_{\text{cache}} = \frac{(2.00 - 0.20) \cdot 72{,}447 \; - \; (2.50 - 2.00) \cdot 1{,}767}{10^6} = 0.1295211 \text{ USD} \tag{8}$$

is 41.08 % of the baseline cost. This values those observed tokens against ordinary input pricing. It does not isolate a causal effect while holding response reuse, model outputs and cache history fixed. In particular, absent calls cannot also read the provider cache.

6.2 Counterfactual gateway replay

The original report gives 2.49 % [1.50; 3.57] for 19,779 requests, 51 sessions and 13 days. This is retained as an author-reported historical counterfactual estimate, not as a live paired measurement. Reused measured counters and reconstructed components must retain separate provenance; a mixed estimate cannot be upgraded wholesale to measured evidence. Raw gateway logs remain private. Before using this as a primary independently reproducible result, publish the exact session-level baseline/optimized totals, their inclusion rules and a runnable analysis tied to this interval.

6.3 Batch deduplication checks

A counting adapter verifies that byte-identical bodies can be executed once and the resulting response replayed to duplicates. For equal-weight synthetic requests, duplicate fractions of 10, 30

and 50 % produce the corresponding execution-count reductions. This is a deterministic adapter check. It does not prove that two independent provider executions would return identical answers, even at temperature zero. Monetary and token reductions equal the duplicate fraction only under the relevant equal-cost or equal-token assumptions. Replay must be an explicit part of the application's output contract.

6.4 Prefix reordering: observed cost and structural limitation

A mechanical 1,000-request check reports one stable prefix after reordering and 69.6 % of system bytes in that span. Byte stability is an eligibility condition, not a provider hit-rate measurement; eligibility also depends on length, expiration, scope and provider policy. A changed prefix may still leave an earlier eligible prefix reusable.

The public 20-request-per-arm `claude-haiku-4-5` summary reports no reads and 126,260 written tokens without reordering. With reordering, it reports 119,567 read tokens, 6,293 written tokens and a 95 % request hit rate. Recorded costs are \$0.166225 and \$0.02865795, a reduction of **82.76 %** in this constructed sequence. These totals include the write premium. No quality non-inferiority or interval across independent runs is established.

The line-based transformation is **not structure-preserving in general**. For example, moving an ISO-timestamp property line out of a multiline JSON system block can make valid JSON invalid. An offline regression case demonstrates this failure without invoking a model. Therefore the transformation must not be described as universally lossless. Explicit caller-supplied stable and dynamic segments are a proposed replacement; they are not claimed to be implemented by the reviewed release. Prompt-order sensitivity still requires task evaluation.

6.5 Repeat-rate pilot: corrected denominator

The generator uses 20 unique questions and appends $\lfloor 20q \rfloor$ repetitions. Consequently q is repeats per unique question; the realized fraction among all requests is $\rho = \lfloor 20q \rfloor / (20 + \lfloor 20q \rfloor)$.

Generator q	Requests	Actual repeat fraction	Avoided repeats	Reported saving
0.3	26	$6/26 = 23.08\,\%$	5/6	14.67 %
0.6	32	$12/32 = 37.50\,\%$	10/12	23.67 %

Table 2: Exploratory `gpt-4o-mini` runs; savings taken from stored ratio fields.

The stored dollar totals are rounded and cannot reproduce the ratio fields exactly. Publish unrounded totals or integer accounting units in the next artifact. The summary does not explain why every repeat was not served from cache; request-level admission and skip reasons are needed. These two runs are not a general response curve or a calibrated quality evaluation.

6.6 Exploratory factorial: response reuse and prefix caching

The four cells contain the same 12 questions and 12 repeats on `claude-haiku-4-5`, with a per-cell salt intended to isolate provider caches. Here "dedup" refers to the exact-response-cache policy in the live harness, not the batch API operation.

With $C_{00}, C_{01}, C_{10}, C_{11}$ in table order, the cost interaction is $IC = C_{11} - C_{10} - C_{01} + C_{00} = 0.052605$ USD. The combined reduction is $1 - C_{11}/C_{00} = 85.94\,\%$. The two baseline-relative savings cannot be added without subtracting IC: avoided calls remove opportunities for cache reads. This is an algebraic decomposition of recorded cell totals. The difference in admitted repeats (8 versus 7), fixed execution sequence and lack of independent replicated cells prevent a precise causal attribution claim with a sampling interval. The cached block recorded in this artifact is 7,229 provider tokens; approximate token lengths in earlier prose should not substitute for that counter.

Response reuse	Prefix caching	Cost USD	Avoided calls
Off	Off	0.178171	0
Off	On	0.031128	0
On	Off	0.119482	8
On	On	0.025044	7

Table 3: Public aggregate cells from 15 November 2026. No replicated-cell uncertainty is reported.

6.7 Cascade cost pilot; quality result not validated

On 24 trivia questions, the published summary compares `claude-opus-5-5` with a cascade serving all 24 requests through `claude-haiku-4-5`, with no escalations. Costs of \$0.064568 and \$0.020718 imply a **67.91 %** reduction. The artifact records 24/24 accepted answers in each arm and 23/24 agreement after text normalization, rather than verbatim agreement.

The evaluation function accepts a response containing the expected answer as a substring. For example, it accepts "not Paris" for "paris" and "12" for "2". This proves that the grader is permissive; it does not prove that those errors occurred in the recorded responses. Because individual responses are absent from the public summary, its 24/24 scores cannot be independently regraded. These scores must not support a non-inferiority claim.

A future paired non-inferiority evaluation must specify a quality metric, margin δ, one-sided significance level, sampling population, sample-size justification and a validated grader before collection. For accuracy difference $\Delta = p_B - p_A$, the criterion is a suitable one-sided lower confidence bound exceeding $-\delta$. Even 24 independently sampled and perfectly graded cases with zero additional failures give a one-sided 95 % upper bound $1 - 0.05^{1/24} = 11.73\,\%$ on the probability of an additional failure (baseline correct and cascade incorrect). This illustration is not a retrospective non-inferiority test of the present selected trivia set.

Following this review, a corrected exploratory re-run (15 November 2026) replaced the grader with an exact normalized match against per-question declared acceptance sets and published the individual answers of both arms. It records 24/24 accepted answers in each arm, 23/24 normalized agreement, costs \$0.064568 and \$0.020718 (−67.91 %), and zero escalations (artifact: `benchmarks/evidencia/cascada-piloto-corregido-2026-11-15.json`). This is a regradeable exploratory pilot, not a non-inferiority test; the caveats above remain in force.

7 Economic scope and deployment implications

Model price comparisons are conditional arithmetic. Cross-model savings require complete usage, escalation costs and a validated quality gate; an output schema verifies form rather than truth. Historical price rows should be frozen with effective date, model identifier, region, tier, cache-write lifetime and currency assumptions, rather than reconstructed from a moving latest catalog.

For a deployment with baseline list-price spend B, applicable pooled reduction R, incremental operating cost H and service fee F, net benefit is $RB - H - F$. A historical example $F = 15$ EUR and $R = 0.315$ gives $B^* = 47.62$ EUR only if $H = 0$ and the workload and currency basis are appropriate. It is not a current price offer or a forecast. Gross provider savings and all-in deployment economics should be reported separately.

8 Threats to validity

The experiments use small constructed workloads with deliberate repetition. Families may share cache history, and the original family bootstrap does not remove dependence between families. A stronger design randomizes policy order across independent workloads or time blocks, preserves

chronological execution within each block, and isolates or explicitly measures warm-up. Provider model changes, output stochasticity, time-varying tariffs and cache policies limit transportability.

Public artifacts support offline recomputation but are not signed provider receipts or reconciled invoices. Some November artifacts contain only rounded aggregates and selected traces. Counterfactual replay introduces additional modeling uncertainty not represented by resampling observed records alone. Results from SDK 0.4.31 do not validate all behavior of SDK 1.2.0. No independent live replication or population-level quality, freshness or latency guarantee has been established.

9 Limitations

The layer integrates known optimization mechanisms; this report does not establish exclusive scientific priority or patentability. The diagnostic ledgers prevent particular reporting confusions only if event identity and cost provenance are correct. They do not by themselves identify causal savings. Community mode's absence of a revocation channel is a distribution property, not an experimental result. Generic line rearrangement, compression and semantic cache hits require application-specific validity checks. The reviewed subset of implementation is not a comprehensive security or billing audit.

10 Prospective research plan

1. Freeze a protocol before the next collection: hypotheses, estimands, randomization unit, cache initialization, stopping rules, quality margin and analysis code.
2. Replicate the response-reuse/prefix-cache factorial across independent blocks; publish all four cells, execution order, uncertainty and cache-state handling.
3. Sweep the realized repeat fraction, recording token and cost weights, cache admission, skip reasons and final usage for every request.
4. Replace permissive substring grading with task-specific validated grading and publish paired outputs or suitable auditable access. Separate gate selection data from retained evaluation data.
5. Evaluate explicit stable/dynamic prompt segmentation, structured data, negations, tool calls, freshness and tenant boundaries.
6. Reconcile one deployment with provider invoices and measure incremental infrastructure and verification costs.
7. Archive immutable data/code/catalog versions and obtain an independent live replication.

These items are a prospective plan. The presence of similar topics in an earlier roadmap is not itself evidence of a fully specified preregistration.

11 Data and code availability

Public repository: https://github.com/BiVelio/bivelio-savings-layer. Review snapshot: `239c84b99fe8320f7efaa9da9e63def08d62aed6`.

Historical primary artifacts are `docs/benchmarks/2026-09-28/gsm8k-0431.json` and `docs/benchmarks/20` with the corresponding CSVs. The estimator is implemented in `benchmarks/stats.ts` and called by `benchmarks/publicados/gsm8k/medir.ts`. November summaries are in `benchmarks/evidencia/`;

harnesses are in `benchmarks/experimentos/`. The public benchmark index is https://savings.bivelio.com/benchmarks.

The companion review package contains an offline recomputation script, selected public fixtures, hashes and targeted counterexamples. It supports recalculation of the claims explicitly identified as recomputed. New API experiments require credentials and execution costs and were not run for this revision. The draft needs author review and the identified quality-evidence work before submission.

12 Conflict of interest

The original author is the founder of BiVelio Inc, which develops the evaluated product. This proposed editorial and technical revision does not constitute independent live replication or peer-review acceptance.

Appendix A. Exact recomputation procedure

For the two historical GSM8K artifacts:

1. Pair baseline and optimized observations by the recorded logical request, using the published ordering and family assignment.
2. Sum costs within each family to obtain (b_f, o_f).
3. Compute $1 - \sum_f o_f / \sum_f b_f$.
4. For each of 10,000 replicates, draw F family indices with replacement using the published mulberry32 generator and seed `0xa11`; sum the sampled paired costs and recompute the ratio.
5. Sort the replicate ratios. Select the zero-based order statistic at index $\lfloor (B-1)p \rfloor$ for $p = 0.025$ and 0.975, with no interpolation, matching the published implementation.
6. As a separate secondary analysis, average the per-family ratios and bootstrap that mean using the same sampled indices.

All observed and resampled primary denominators in these artifacts are positive. A general implementation must report undefined ratios instead of silently replacing them by zero. The companion script checks input hashes and reproduces the published pooled points and endpoints. It is a computational audit of stored observations, not a new sample of model behavior.

Appendix B. Publication evidence still required

The primary estimator correction can be made without changing the historical headline percentages. Claims of general losslessness, independently identified mechanism effects and quality non-inferiority cannot be restored by editorial changes alone. They require implementation changes or additional experiments. Specifically, the quality pilot needs saved outputs and validated grading; generalized prefix rewriting needs a sound applicability contract; causal factorial claims need independent replicated, controlled assignments; and the gateway interval needs sufficient session-level data. Each released table should identify its exact artifact, model, run date, SDK version, catalog and scope.

Appendix C. Resumen en español

Esta revisión conserva los resultados que se han podido recalcular y corrige su interpretación. El ahorro agregado es el cociente entre sumas de costes: 31,50 % en OpenAI y 75,08 % en Anthropic. La media no ponderada entre familias da 22,88 % y 70,71 %, respectivamente, y constituye otra métrica. Se amplía la fórmula de coste para separar entrada normal, lecturas y escrituras de caché por duración y salida facturable. Los cuatro libros describen actividad; no demuestran una partición causal del ahorro.

Los ensayos de noviembre se presentan como exploratorios. Se corrigen las tasas reales de repetición y la interpretación de las interacciones. El ensayo de cascada no demuestra no inferioridad: el evaluador admite coincidencias parciales y no publica respuestas individuales. La reordenación por líneas puede romper estructuras JSON. El borrador propone contratos explícitos y pruebas adicionales antes de sostener garantías generales. No se han ejecutado nuevas llamadas a proveedores ni conciliado facturas.

References

[1] H. Jiang, Q. Wu, C.-Y. Lin, Y. Yang, L. Qiu. LLMLingua: Compressing prompts for accelerated inference of large language models. EMNLP, 2023.

[2] Z. Pan, Q. Wu, H. Jiang, et al. LLMLingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. Findings of ACL, 2024.

[3] J. Mu, X. L. Li, N. Goodman. Learning to compress prompts with gist tokens. NeurIPS 36, 2023.

[4] I. Gim, G. Chen, S.-s. Lee, et al. Prompt Cache: Modular attention reuse for low-latency inference. arXiv:2311.04934, 2023.

[5] B. Gao, Z. He, P. Sharma, et al. Cost-efficient large language model serving for multi-turn conversations with CachedAttention. USENIX ATC, pp. 111–126, 2024.

[6] L. Zheng, L. Yin, Z. Xie, et al. SGLang: Efficient execution of structured language model programs. arXiv:2312.07104, 2023.

[7] J. Yao, H. Li, Y. Liu, et al. CacheBlend: Fast large language model serving for RAG with cached knowledge fusion. EuroSys, 2025.

[8] Anthropic. Prompt caching — documentation. https://platform.claude.com/docs/en/build-with-claude/prompt-caching

[9] OpenAI. Prompt caching — API guide. https://platform.openai.com/docs/guides/prompt-caching

[10] F. Bang. GPTCache: An open-source semantic cache for LLM applications enabling faster answers and cost savings. NLP-OSS Workshop, 2023.

[11] J. Li, C. Xu, F. Wang, et al. SCALM: Towards semantic caching for automated chat services with large language models. arXiv:2406.00025, 2024.

[12] W. Gill, M. Elidrisi, P. Kalapatapu, et al. MeanCache: User-centric semantic caching for LLM web services. arXiv:2403.02694, 2024.

[13] L. Chen, M. Zaharia, J. Zou. FrugalGPT: How to use large language models while reducing cost and improving performance. arXiv:2305.05176, 2023.

[14] I. Ong, A. Almahairi, V. Wu, et al. RouteLLM: Learning to route LLMs with preference data. ICLR, 2025. arXiv:2406.18665.

[15] W. Kwon, Z. Li, S. Zhuang, et al. Efficient memory management for large language model serving with PagedAttention. ACM SOSP, 2023.

[16] Y. Leviathan, M. Kalman, Y. Matias. Fast inference from transformers via speculative decoding. ICML, 2023.

[17] OpenTelemetry Project. Generative AI semantic conventions. https://opentelemetry.io/docs/specs/semconv/gen-ai/

[18] Joint Committee for Guides in Metrology. Guide to the expression of uncertainty in measurement. JCGM 100:2008, 2008.

[19] Joint Committee for Guides in Metrology. International vocabulary of metrology (VIM), 3rd ed. JCGM 200:2012, 2012.

[20] Anthropic. Claude Code — agentic coding tool documentation. https://www.anthropic.com/claude-code

[21] K. Cobbe, V. Kosaraju, M. Bavarian, et al. Training verifiers to solve math word problems. arXiv:2110.14168, 2021.

[22] B. Efron, R. J. Tibshirani. An Introduction to the Bootstrap. Chapman and Hall/CRC, 1993.

[23] A. C. Davison, D. V. Hinkley. Bootstrap Methods and Their Application. Cambridge University Press, 1997.

[24] L. G. Schroeder, A. Desai, A. Cuadron, et al. vCache: Verified semantic prompt caching. ICLR, 2026. arXiv:2502.03771.

[25] D. Guo, J. Wu, S. M. Yiu. RouteNLP: Closed-loop LLM routing with conformal cascading and distillation co-optimization. arXiv:2604.23577, 2026.

[26] L. Kutschka and B. Geiger. Notation matters: A benchmark study of token-optimized formats in agentic AI systems. arXiv:2605.29676, 2026.

[27] A. Baral, R. Ralev, I. S. Zhechev, S. Rajamohan, J. Agarwal. Closing the operational gap in semantic caching. EMNLP Industry, pp. 988–1011, 2026. https://aclanthology.org/2026.emnlp-industry.60/

[28] BiVelio. Calibrated semantic-cache evaluation and abstention report. Public benchmark documentation, September 2026. https://savings.bivelio.com/benchmarks