Behavioral Consistency and Transparency Analysis on Large Language Model API Gateways

Guanjie Lin
guanjie.lin001@umb.edu
University of Massachusetts Boston
Boston, USA

Yinxin Wan
yinxin.wan@umb.edu
University of Massachusetts Boston
Boston, USA

Shichao Pei
shichao.pei@umb.edu
University of Massachusetts Boston
Boston, USA

Ting Xu
ting.xu001@umb.edu
University of Massachusetts Boston
Boston, USA

Kuai Xu
kuai.xu@asu.edu
Arizona State University
Tempe, USA

Guoliang Xue
xue@asu.edu
Arizona State University
Tempe, USA

ABSTRACT

Third-party Large Language Model (LLM) API gateways are rapidly emerging as unified access points to models offered by multiple vendors. However, the internal routing, caching, and billing policies of these gateways are largely undisclosed, leaving users with limited visibility into whether requests are served by the advertised models, whether responses remain faithful to upstream APIs, or whether invoices accurately reflect public pricing policies. To address this gap, we introduce VeriFlow, a lightweight black-box measurement framework for evaluating behavioral consistency and operational transparency in commercial LLM gateways. VeriFlow is designed to detect key misbehaviors, including model downgrading or switching, truncation, billing inaccuracies, and instability in latency by auditing gateways along four critical dimensions: response content analysis, multi-turn conversation performance, billing accuracy, and latency characteristics. Our measurements across 10 real-world commercial LLM API gateways reveal frequent gaps between expected and actual behaviors, including silent model substitutions, degraded memory retention, deviations from announced pricing, and substantial variation in latency stability across platforms.

1 INTRODUCTION

Large Language Models (LLMs) such as OpenAI's GPT, Anthropic's Claude, and Google's Gemini are increasingly integrated into a broad range of applications [7, 12]. Although these models are widely accessible through cloud-based inference APIs, each vendor maintains its own independent API platform. For developers and enterprise users who require access to multiple models from different providers, this fragmented ecosystem introduces substantial operational overhead, including complex backend integration, API key management, and billing processes [17].

LLM gateways are emerging as a promising solution, offering a unified API interface with centralized billing that aggregates access to models from multiple vendors. In such gateways, a single API key can be used to invoke different models, significantly simplifying deployment workflows and reducing integration and key management complexity. In addition, LLM API gateways often offer additional benefits such as load balancing across models, usage-based optimization, and in some cases discounted token pricing. However, from the client's perspective, these gateways operate as *black-box* intermediaries with little visibility into their internal behaviors. Thus, users cannot reliably verify whether requests use the intended model, whether context is truncated, or whether billing is accurate [2, 6, 21, 23, 24].

Auditing these black-box gateways is challenging because their routing and model selection behaviors are not exposed to clients. Existing efforts such as LLMmap [20] use active probing to differentiate between LLMs for mostly open-source models, but does not target the behavioral transparency of commercial LLM gateways. Other work on instructional fingerprints [22, 26, 27], UTF [5], AuditLLM [3], and FDLLM [8] focuses on analyzing model traits or downstream artifacts rather than evaluating whether third-party API gateways preserve upstream model behavior or comply with pricing rules. These gaps motivate an efficient framework that measures the transparency and behavioral consistency of LLM gateways from the client's perspective.

Building on these needs, we present VeriFlow, a systematic framework for audits gateway behavior transparency

This manuscript was submitted to the ACM Internet Measurement Conference (IMC) 2026 Cycle 1 on November 20, 2025, and accepted by IMC 2026 on March 13, 2026.

and consistency across four key dimensions. VeriFlow performs response content analysis using a structured set of probing queries to generate behavioral profiles for different models, enabling the detection of potential model downgrading or substitution. It evaluates multi-turn conversational consistency to identify model switching during extended conversations and to determine whether silent truncation occurs before reaching documented context or output limits. VeriFlow further examines billing accuracy by comparing actual token consumption with reported usage and assessing differences between expected and reported costs, accounting for cached tokens. Finally, it measures latency characteristics, quantifying variability in request and response delays.

We first demonstrate VeriFlow's effectiveness and efficiency through controlled validation using official vendor model endpoints. We then apply VeriFlow to 10 commercial LLM API gateways, where we uncover significant gaps between expected and observed behaviors, including model substitutions, degraded memory retention, inaccurate billing, and substantial variability in request latency.

Our key research contributions are summarized as follows:

- We identify the lack of transparency and consistency guarantees in LLM API gateways and motivate the need for systematic client-side auditing.
- We design VeriFlow, a lightweight framework that measures gateway behavior across four dimensions: response content, multi-turn conversation consistency, billing accuracy, and latency characteristics.
- We validate VeriFlow in controlled settings and apply it to 10 commercial gateways, uncovering substantial discrepancies between expected and observed behavior.

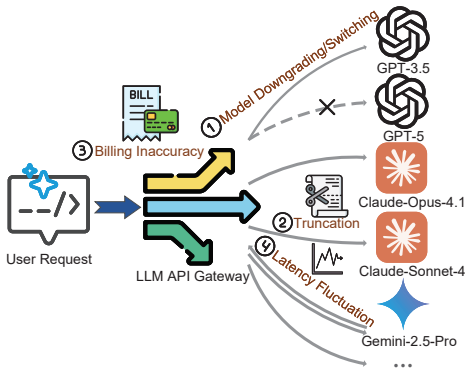

Figure 1: Overview of LLM API gateway architecture and the main transparency and consistency challenges.

2 BACKGROUND AND MOTIVATION

LLM API gateways are increasingly adopted to abstract vendor differences and provide unified interfaces, load balancing, and centralized billing [1]. As illustrated in Figure 1, a gateway acts as a proxy between client applications and upstream model providers. However, the internal operation of these gateways remains largely non-transparent. Users cannot observe how requests are routed, which upstream models are selected, what inference parameters are applied, or how billing is computed. As a result, clients cannot determine whether a gateway follows documented policies or behaves consistently. This lack of visibility raises concerns around several critical issues as shown in Figure 1:

- Model downgrading/switching: gateways may route requests to cheaper or less capable models despite advertising a premium model, often using internal routing policies driven by cost or latency considerations. Such routing decisions are typically undisclosed, leaving users unaware of which model actually served a request.
- Prompt/response truncation: gateways may enforce their own context or output limits that are smaller than the model's native capacity, silently truncating multi-turn prompts or responses without signaling the change and degrading service quality.
- Billing inaccuracy: reported usage and actual token consumption may diverge, including cases where gateways charge for cached tokens [4, 19] or add extra tokens that were not processed.
- Latency fluctuation: limited computational resources, especially during peak hours, along with extra processing and forwarding steps inside the gateway, as well as behaviors such as switching models, modifying user input, or changing API endpoints, can introduce significant variability in response latency at LLM gateways, negatively impacting service quality and reliability.

Prior work has examined hidden behaviors in LLM services, such as prompt caching, hidden token usage, and API-level security risks, but these studies primarily focus on first-party vendor's LLM API platforms rather than third-party LLM API gateways [9, 11, 16, 21, 23–25]. These challenges collectively motivate VeriFlow, a framework for systematically auditing behavioral transparency and consistency in black-box LLM API gateways.

3 DESIGN OF VERIFLOW

This section describes the design of VeriFlow in detail.

3.1 Overview of VeriFlow

VeriFlow is a lightweight, systematic auditing framework that operates entirely through the public APIs exposed by LLM gateways. All measurements are performed using the OpenAI-compatible `/v1/chat/completions` endpoint, ensuring a consistent request–response format across platforms. The framework relies solely on information accessible to ordinary clients through standard interactions, without requiring privileged access or internal instrumentation.

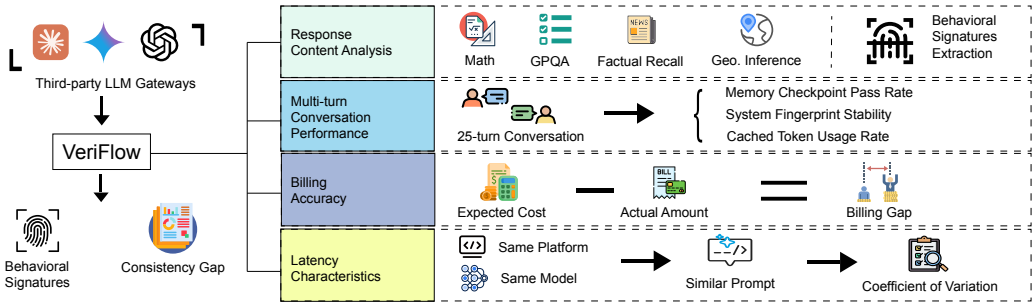

Figure 2: Overall architecture of the VeriFlow framework.

Figure 2 illustrates the overall structure of VeriFlow, which implements a multi-dimensional auditing pipeline to evaluate the response content, multi-turn conversation, billing accuracy, and latency characteristics.

3.2 Response Content Analysis

The goal of this dimension is to determine whether a gateway actually serves the requested model or silently replaces it with a different and often cheaper model. To detect such substitution, we analyze the response behavior of different LLMs and use it to differentiate between models. This is challenging because even the same model may generate varied responses to the same prompt, especially for open-ended questions, making naive text matching unreliable. Prior work on LLM fingerprinting and identification primarily distinguishes models using lexical, semantic, or logit-space features extracted from free-form text [20, 28]. However, these approaches are often sensitive to generation randomness and frequently fail to capture meaningful behavioral differences in critical fields (e.g., the final answer in a math query is far more important than the surrounding derivation text) when relying solely on text-level analysis.

Instead, our prompt design follows three requirements: (1) responses to the same probe should remain stable across repeated queries to the same model, (2) responses should differ across models because they reflect model-specific capabilities, and these differences should be identifiable through key feature dimensions, and (3) the prompt suite should be lightweight so that auditing remains efficient.

To formalize this analysis, we adopt a formulation following prior work [20, 28], treating each model version v as a stochastic mapping $u \sim E_v(p, \theta)$, where a probe $p \in \mathcal{P}$ and generation parameters θ produce an output u with metadata m.

3.2.1 Prompt Design. Following the three requirements mentioned above, we design prompts that (i) structure model responses for analysis and (ii) expose key response features that can effectively differentiate models. Each prompt instructs the LLM to generate explicit step-by-step reasoning leading to a final answer, and the response is required to follow a shared JSON schema that enforces a consistent output format across models and runs.

When designing the prompt content, We deliberately avoid two unsuitable choices. First, we avoid banner-grabbing questions such as "What is your model version?", since they are easily spoofed through system-prompt manipulation and yield inconsistent identifiers across gateways [20]. Second, we avoid using large public benchmarks at API scale, as they are expensive to run and risk data contamination [13, 14].

Motivated by these requirements and constraints, we design a probe suite $\mathcal{S}$ that captures distinctive behavioral signatures across models, including differences in reasoning ability, knowledge cutoff, mathematical competence, and output style (e.g., preferred symbols or LaTeX formatting). To cover these dimensions, we design prompts across four domains: (1) *AIME (American Invitational Mathematics Examination)*: numeric-answer problems that test precise computation and mathematical reasoning [18]; (2) *GPQA (Graduate-Level Google-Proof Q&A)*: multiple-choice questions designed to expose deep reasoning and deliberation capabilities across models [10]; (3) *Factual Recall*: questions about major events between 2021–2025 that assess knowledge cutoff and factual retrieval ability; (4) *Geographic Information Inference*: implicit reasoning tasks that require step-by-step linking of intermediate facts to identify a specific geographic feature.

Building on discrete concept composition (DCC) [15], each prompt instructs LLM to compose disconnected facts across multiple reasoning hops, making intermediate choices such as ordering, granularity, and error types directly observable. Concretely, each response is required to follow a structured format with two fields: `knowledge_path` (an array of intermediate reasoning steps) and `final_answer` (the final result). A complete specification of the request template can be found in Appendix B. This structure helps to remove most free-form variations in responses so that observable differences mainly reflect model capabilities rather than surface text patterns. It also exposes reasoning strategies that remain stable across repetitions. For example, when asked *"What is the highest geographic feature associated with the origin area*

of the Starbucks corporation?", GPT-5 consistently avoids detailed step-by-step reasoning and instead produces a short summary, whereas other models produce explicit multi-hop chains of inference.

3.2.2 Model Signature Extraction. For each structured response obtained from a prompt, we extract a d-dimensional signature vector $\Phi(u, m)$ rather than relying on text-level embeddings. This enables robust differentiation between models by capturing behavioral and structural characteristics. The signature vector is composed of five feature families: (i) answer quality: correctness of the final answer and whether it appears in the correct location, (ii) reasoning structure: reasoning depth, mean step length, and variance, (iii) scale: response length and density, (iv) style: presence of numeric expressions or LaTeX formatting, and (v) parsing quality: whether parsing occurs and at what degree. Full feature definitions can be found in Appendix E. Collecting signatures across probes and repetitions yields, for each model version, a behavioral signature matrix.Collecting signatures across probes and repetitions yields, for each model version, a behavioral signature matrix. Using these signatures, we train lightweight per-model classifiers. For each model v, a one-vs-rest XGBoost classifier is trained to distinguish signatures produced by v from those of all other models.

3.3 Multi-turn Conversation Performance

LLM applications typically involve multi-turn conversations, where users expect the model to retain context, chat history, and intermediate results across turns. Since most LLMs are inherently stateless, the full conversation history is included in every request to preserve context in multi-turn interactions. Under normal conditions, this enables consistent behavior and correct recall of information introduced earlier in the dialog. However, a black-box gateway can silently break these expectations by switching back-end models to cheaper alternatives or truncating conversation history to reduce token consumption.

In this dimension, our goal is to evaluate gateway performance in multi-turn interactions and detect failures arising from silent model switching and hidden prompt or response truncation. To accomplish this, we construct a multi-turn testing procedure that embeds repeated questions and explicit memory checkpoints to assess whether early information is retained and updated correctly, and whether response consistency is preserved throughout the conversation.

3.3.1 Multi-Turn Conversation Design. We design a 25-turn conversation template specifically for testing third-party LLM API gateways. A 25-turn length offers a practical balance, providing enough depth to expose multi-turn consistency issues and cover diverse test cases while remaining lightweight and efficient in token usage. A detailed template with example content is provided in Appendix D.

The conversation begins by assigning the model a stable identity and a user-specific preference. Two segments of distractor discussion on unrelated topics are then inserted to reduce the immediate salience of this information. Midway through the dialog, the preference is first checked and then explicitly updated by the user, after which the conversation again shifts to unrelated content. In the final two turns, the model is asked to restate both the identity and the current preference. During the process, we also collect *metadata from all responses*, including the `system_fingerprint` and cached token counts. `system_fingerprint` serves as a good sign for potential model switching, as it typically changes when the underlying model is replaced. The number of cached tokens is also informative, since switching models resets the cache-hit mechanism, causing a drop in cached token usage. This structure allows us to test whether the gateway-served model (i) retains information introduced early in the conversation, (ii) correctly overwrites outdated preferences, (iii) avoids hallucinating attributes that were never provided, and (iv) maintains consistent behaviors during the entire interaction.

3.3.2 Indicators and Detection Goals. For each model offered through each gateway, we run multi-turn conversations using the template and evaluate performance based on three indicators that collectively quantify conversational consistency and context retention: (i) Memory Checkpoint Pass Rate: the fraction of correctly recalled identity and preference values at predefined checkpoints (e.g., at turns 10, 24, and 25), (ii) System Fingerprint Stability: the number of distinct `system_fingerprint` values observed across turns, which can indicate silent model switching;, and (iii) Cached Token Usage Rate, the ratio of cached prompt tokens to total input tokens as reported by API metadata.

3.4 Billing Accuracy

Billing is a critical factor for third-party LLM gateways, as users ultimately care about whether they are charged fairly. In black-box API gateways, cost discrepancies can arise from (i) incorrect token consumption reporting, (ii) improper handling of cached tokens, and (iii) incorrect cost calculations based on published pricing.

In the billing accuracy analysis, we first compute the actual token consumption locally and compare it against the values reported in the gateway console. We also evaluate whether each gateway charges fees consistent with the reported token usage and the unit prices it publishes. For each request, we record the number of input tokens n_{in}, cached tokens n_{cached}, and output tokens n_{out}, along with

the published per-token rates p_{in}, p_{cached}, and p_{out}. The expected cost for a single call is then computed as: $C_{\text{expected}} = (n_{\text{in}} - n_{\text{cached}})\, p_{\text{in}} + n_{\text{cached}}\, p_{\text{cached}} + n_{\text{out}}\, p_{\text{out}}$, gateways that do not support separate cache pricing or fail to apply cache discounts are evaluated with $n_{\text{cached}} = 0$.

3.5 Latency Characteristics

Latency is an important performance metric in LLM gateways and can also serve as an indicator of potential misbehaviors such as model switching or downgrading, since different models often exhibit distinct processing times. As a result, switching to a different model may introduce noticeable variation in latency. In this evaluation, we conduct latency measurements by issuing repeated requests for a given probe across gateways within a tightly bounded time window. This controlled setup reduces external influences such as network dynamics, allowing latency variation to more reliably reflect gateway-side behavior.

When a gateway consistently serves a fixed claimed model on probes with comparable prompt and output lengths under fixed parameters, the resulting latency distribution should remain stable. We quantify this stability using the coefficient of variation (CV), $\text{CV} = \sigma_T/\mu_T$, where σ_T and μ_T are the standard deviation and mean over K repeated invocations of the same probe. In contrast, model switching across heterogeneous execution paths typically increases CV and can lead to bimodal or multi-modal latency distributions. Elevated or highly inconsistent CV therefore serves as an indicator of unstable back-end behaviors for LLM API gateways.

4 EVALUATION

This section evaluates VeriFlow's effectiveness in measuring behavioral transparency and consistency. We first validate the accuracy and reliability of VeriFlow under controlled conditions using official model endpoints, and then apply it to audit commercial gateway services.

4.1 Data Collection

For official vendor platforms, we collect behavioral signatures from 24 LLMs exposed through their native APIs. Using the standardized probe suite $\mathcal{S}$ under fixed parameters, each model is queried with $K = 12$ repetitions per probe, producing 15,840 records in total (660 per model) covering response-content features and latency measurements. The full sampling protocol can be found in Appendix C.

For third-party platforms, we audit 10 commercial LLM gateways using the same probe suite and parameters. Gateways are anonymized using first–last-letter encodings: `oer`, `aix`, `oub`, `zng`, `oey`, `uipi`, `bie`, `unpi`, `ayi`, `lub`. Results represent a point-in-time snapshot rather than a long-term ranking of individual services. For single-turn evaluation, each model on each gateway is queried with the prompt suite using five repetitions. For multi-turn evaluation, we run the standardized 25-turn conversation template, also repeated five times per model per gateway. Overall, approximately **1M tokens** are consumed per model during testing, which remains reasonably sustainable in practice. This cost can be further reduced by lowering the repetition count if needed.

4.2 Controlled Validation

Before applying VeriFlow to third-party gateways, we first validate on official vendor APIs that the response-content signatures introduced in Section 3.2 are sufficiently discriminative under controlled settings where the ground-truth model identity is known.

For each model version v available through official vendor APIs, we train a classifier that distinguishes signatures generated by v from those generated by all other models, using the probe suite $\mathcal{S}$ and the feature configuration described in Section 3.2. Data splits, threshold selection, and hyperparameter settings are summarized in Appendix E, with detailed per-model metrics reported in Appendix F. Evaluating these classifiers on labeled test data collected by querying vendor endpoints with $\mathcal{S}$, we observe an average $F1$ score of 0.968 ± 0.085 across all 24 models, with most models exceeding 0.95. Considering that our evaluation includes closely related variants within the same model family (e.g., `gpt-4o-mini` and `gpt-4.1-nano`), these results demonstrate that our response-content analysis exhibits strong discriminative power and can reliably distinguish LLM models.

We also evaluate the robustness of our method on **unseen models** by applying the trained ensemble to 5 additional LLMs (Claude sonnet 4.5, Claude haiku 4.5, Qwen-plus, Qwen-turbo and DeepSeek-V3.2-Exp) not included in the original set of 24. In the evaluation, none of these unseen models is mistakenly classified as any known model. This indicates that our method behaves conservatively in controlled settings and avoids misclassification when encountering new model variants. Extending the ensemble to identify new models is also lightweight as training an additional classifier requires *less than* 1 *second* per model.

4.3 Third-party API Gateway Evaluation

4.3.1 Results of Response Content Evaluation. reports the proportion of responses classified as the *claimed* upstream model for four representative models across different gateways (extended results for all models can be found in Appendix G). Each percentage is computed over 275 responses per model.

For gpt-4o, most gateways generate responses that are consistently identified as the claimed model, while platforms such as `bie` and `ayi` exhibit noticeably lower consistency.

For gpt-5, identification rates vary significantly across gateways, with several platforms frequently classified as models outside the `gpt-5` family—indicating possible model substitution. For `gemini-2.5-pro` and `claude-sonnet-4.0`, platform support is limited and behavior is unstable. The official APIs and a few gateways maintain high identification rates above 95%, whereas several others fall below 60%, showing that they do not reliably preserve the identity of the claimed model.

Takeaway: For certain models, responses from several gateways show substantial variation. This indicates that these gateways may route user's prompt to a different model than the requested one.

Table 1: Response content evaluation using per-model probe subsets. Each cell shows the percentage of responses classified as the claimed model. Extended results are presented in Appendix G.

Gateway	gpt-4o (%)	gpt-5 (%)	Gemini-2.5-pro (%)	Claude Sonnet 4.0 (%)
Baseline	98.18	97.09	96.00	97.09
a*yi	62.18	48.00	54.91	70.91
a*ix	88.00	70.91	N/A	83.27
b*ie	46.91	13.09	N/A	62.91
l*ub	90.91	94.18	84.00	58.18
o*ub	86.18	34.91	90.18	N/A
o*er	93.09	97.09	89.09	86.91
o*ey	90.18	92.00	96.00	N/A
un*pi	82.18	89.09	73.09	N/A
ui*pi	76.00	N/A	N/A	N/A
z*ng	95.27	94.91	N/A	97.09

4.3.2 Results of Multi-turn Conversation Evaluation. Table 2 summarizes the multi-turn conversation performacnes for gpt-4o and gpt-4o-mini across the vendor's baseline API platforms and 10 third-party gateways, where T10/24/25 indicates checkpoint pass count at turn 10, 24, and 25, FC indicates system_fingerprint count, and CR indicates cache rate(%). Extended results are presented in Appendix H.

For gpt-4o, the official API passes all checkpoints with only 1 unique fingerprint and moderate cache reuse. Gateways such as l*ub, o*er, and un*pi largely match this behavior, while others do not pass checkpoints at turns 24 and 25 or introduce additional fingerprints in most cases. For gpt-4o-mini, behavior is less stable even on the baseline, and most gateways forgets identity and preference by the final checkpoints. Two platforms show notable anomalies. Platform a*yi has very similar checkpoint passing and cache rate performances for gpt-4o and gpt-4o-mini, indicating both of them may be switched to the same model. Platform o*ey achieves the highest checkpoint passing counts for gpt-4o-mini among all gateways while reporting zero cache usage, suggesting either a non-canonical gpt-4o-mini implementation or aggressive, opaque caching that alters long-context behavior relative to the upstream API.

Takeaway. Long-context retention and infrastructure behavior vary substantially across gateways: some preserve the upstream LLM's behavior, while others lose state or change back-end configurations during the conversation.

Table 2: Multi-turn conversation performance comparison for gpt-4o and gpt-4o-mini. Extended results are presented in Appendix H.

Gateway	gpt-4o					gpt-4o-mini				
	T10	T24	T25	FC	CR(%)	T10	T24	T25	FC	CR(%)
Baseline	5	5	5	1	48.7	5	1	2	1	40.3
a*yi	3	1	2	4	4.2	3	1	1	2	5.3
a*ix	5	4	5	1	34.7	5	0	0	2	21.5
b*ie	5	4	4	2	15.7	4	0	0	3	14.2
l*ub	5	5	5	2	18.2	3	0	1	2	16.3
o*ub	5	5	3	2	32.5	4	0	1	1	29.7
o*er	5	5	5	1	32.8	5	1	0	1	52.3
o*ey	4	3	4	2	0.0	5	4	4	3	0.0
un*pi	5	5	4	2	19.3	4	0	0	2	18.7
ui*pi	5	5	4	2	12.8	5	1	2	2	11.5
z*ng	5	5	4	2	23.5	5	0	0	2	32.6

4.3.3 Results of Billing Accuracy Evaluation. Table 3 demonstartes the billing accuracy results for gpt-4o by comparing the expected cost from public prices with the actual amount charged by each gateway. Extended results are presented in Appendix I.

Most gateways match the expected cost and do not have any billing gap. A small number of platforms charge more than expected. a*ix has a billing gap of 7.6% while o*ey has the billing gap as 62.8% even though the token usage is similar to other platforms.

Takeaway. Billing outcomes vary across gateways. Most platforms follow vendor pricing while a few charge significantly more than the expected cost.

Table 3: Billing accuracy for `gpt-4o`: baseline token counts and expected vs. actual costs (each gateway has different rate). Extended results are presented in Appendix I.

Gateway	Input	Cached	Output	C_{exp}	C_{act}	Gap%
Baseline	139,025	67,705	201,060	2.27	2.27	0.0
a*yi	137,475	0	203,346	2.30	2.30	0.0
a*ix	140,436	0	196,651	6.07	6.53	+7.6
b*ie	139,085	0	208,578	2.35	2.35	0.0
l*ub	141,712	3,685	201,749	2.32	2.32	0.0
o*ub	138,762	0	365,097	8.50	8.50	0.0
o*er	139,649	31,256	199,124	2.30	2.30	0.0
o*ey	140,987	0	213,446	6.81	11.09	+62.8
un*pi	142,318	28,501	204,763	2.27	2.27	0.0
ui*pi	139,421	0	217,790	2.36	2.36	0.0
z*ng	137,884	0	226,935	10.14	10.14	0.0

4.3.4 Results of Latency Evaluation. Table 4 summarizes the latecny measurement results for gpt-4o across the across the vendor's baseline API platforms and 10 third-party gateways. Extended results are presented in Appendix J. Gateway b*ie exhibits marked instability, characterized by anomalously

large CV vlaues and wide disparities between minimum and maximum response times. For instance, in the Math task, latencies span from ≈ 6 seconds to over 273 seconds, resulting in a CV (1.10) almost double the baseline. This indicates that b*ie may switch between different models.

Takeaway. Latency stability is not uniform across gateways. High variation may indicate suspicious back-end behavior such as model downgrading or switching.

Table 4: Latency ranges (min, max, in seconds) and coefficients of variation (CV) for `gpt-4o` across gateways. Extended results are presented in Appendix J.

Gateway	Math			GPQA			Factual			Geo		
	Min	Max	CV	Min	Max	CV	Min	Max	CV	Min	Max	CV
Baseline	2.67	121.84	0.63	1.72	15.56	0.67	1.38	8.29	0.35	0.77	6.33	0.42
a*yi	2.98	136.11	0.68	1.91	17.27	0.72	1.55	9.34	0.38	0.77	6.33	0.42
a*ix	1.33	60.92	0.25	0.87	7.87	0.27	0.84	5.03	0.18	0.44	3.63	0.20
b*ie	5.99	273.38	1.10	3.30	29.85	1.05	4.74	28.49	0.82	2.25	18.50	0.88
l*ub	2.51	114.51	0.58	1.62	14.68	0.62	1.29	7.75	0.32	0.77	6.33	0.42
o*ub	2.48	113.03	0.57	1.60	14.50	0.61	1.26	7.57	0.31	0.76	6.22	0.41
o*er	4.62	210.98	0.92	3.07	27.81	1.00	3.02	18.11	0.60	1.45	11.92	0.65
o*ey	2.25	102.45	0.50	1.44	13.05	0.53	1.23	7.38	0.30	0.71	5.87	0.38
un*pi	2.57	117.46	0.60	1.66	15.03	0.64	1.35	8.11	0.34	0.77	6.33	0.42
ui*pi	2.86	130.34	0.66	1.83	16.58	0.70	1.50	8.99	0.37	0.82	6.77	0.44
z*ng	1.97	89.89	0.42	1.28	11.54	0.45	1.14	6.82	0.27	0.64	5.28	0.33

5 CONCLUSION

We introduced VeriFlow, a framework for auditing third-party LLM gateways along content, multi-turn, billing, and latency dimensions. Our measurements on official APIs and ten commercial gateways reveal significant transparency gaps, including downgraded models and pricing deviations. We hope VeriFlow will serve as a starting point for long-term, multi-region monitoring and stronger accountability mechanisms in the gateway ecosystem.

REFERENCES

[1] An overview of OpenRouter's API. https://openrouter.ai/docs/api-reference/overview.

[2] Apparently all third party providers downgrade, none of them provide a max quality model. https://www.reddit.com/r/LocalLLaMA/comments/1nqkx7o/apparently_all_third_party_providers_downgrade/.

[3] Maryam Amirizaniani, Elias Martin, Tanya Roosta, Aman Chadha, and Chirag Shah. 2024. AuditLLM: A Tool for Auditing Large Language Models Using Multiprobe Approach. *arXiv preprint arXiv:2402.09334* (2024).

[4] Gemini API. Context caching. https://ai.google.dev/gemini-api/docs/caching.

[5] Jiacheng Cai, Jiahao Yu, Yangguang Shao, Yuhang Wu, and Xinyu Xing. 2025. UTF: Under-trained Tokens as Fingerprints —— a Novel Approach to LLM Identification. In *Proc. of ACL*.

[6] Ruizhi Cheng, Surendra Pathak, Guowu Xie, Matteo Varvello, Songqing Chen, and Bo Han. 2025. Hello, GenAI? Dissecting Human to Generative-AI Calling. In *Proc. of IMC*.

[7] Stanford Institute for Human-Centered Artificial Intelligence. 2025. Artificial Intelligence Index Report 2025. Stanford HAI. https://hai.stanford.edu/assets/files/hai_ai_index_report_2025.pdf.

[8] Junbang Fu, Wenlong Dong, Chong Wang, Xutong Zhao, Jingwei Gao, Zhirui Hu, Shangbo Wu, Dong Wang, Yinzhen Wang, Xiaojuan Qi, Shuai He, Yujun Chen, and Tianyu Du. 2025. FDLLM: A Dedicated Detector for Black-Box LLMs Fingerprinting. *arXiv preprint arXiv:2501.16029* (2025).

[9] Yao Fu, Leyang Xue, Yeqi Huang, Andrei-Octavian Brabete, Dmitrii Ustiugov, Yuvraj Patel, and Luo Mai. 2024. ServerlessLLM: Low-Latency Serverless Inference for Large Language Models. In *Proc. of USENIX OSDI*.

[10] GPQA-Diamond. https://huggingface.co/datasets/fingertap/GPQA-Diamond.

[11] Chenchen Gu, Xiang Lisa Li, Rohith Kuditipudi, Percy Liang, and Tatsunori Hashimoto. 2025. Auditing Prompt Caching in Language Model APIs. *arXiv preprint arXiv:2502.07776* (2025).

[12] Wei Hao, Van Tran, Vincent Rideout, Zixi Wang, AnMei Dasbach-Prisk, M. H. Afifi, Junfeng Yang, Ethan Katz-Bassett, Grant Ho, and Asaf Cidon. 2025. Do Spammers Dream of Electric Sheep? Characterizing the Prevalence of LLM-Generated Malicious Emails. *Proc. of IMC* (2025).

[13] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring Mathematical Problem Solving With the MATH Dataset. *Proc. of NeurIPS* (2021).

[14] Dan Hendrycks, Collin Burns, Andy Zou, Steven Basart, Andy Lee, Dawn Kohlmeier, Wenliang Ju, Dawn Xiaodong Song, and Jacob Steinhardt. 2020. Measuring Massive Multitask Language Understanding. *arXiv preprint arXiv:2006.04019* (2020).

[15] Guan Zhe Hong, Bhavya Vasudeva, Vatsal Sharan, Cyrus Rashtchian, Prabhakar Raghavan, and Rina Panigrahy. 2025. Latent Concept Disentanglement in Transformer-based Language Models. *arXiv preprint arXiv:2506.16975* (2025).

[16] Sangwon Hyun, Mingyu Guo, and M. Ali Babar. 2024. METAL: Metamorphic Testing Framework for Analyzing Large-Language Model Qualities. In *Proc. of IEEE ICST*.

[17] In-Depth Analysis of AI Gateway: The New Generation of Intelligent Traffic Control Hub. https://jimmysong.io/en/blog/ai-gateway-in-depth/.

[18] Zihan Liu, Yang Chen, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2025. AceMath: Advancing Frontier Math Reasoning with Post-Training and Reward Modeling.

[19] OpenAI. Prompt caching. https://platform.openai.com/docs/guides/prompt-caching.

[20] Dario Pasquini, Evgenios M. Kornaropoulos, and Giuseppe Ateniese. 2025. LLMmap: Fingerprinting for Large Language Models. In *Proc. of USENIX Security*.

[21] Yashothara Shanmugarasa, Ming Ding, M. A. P Chamikara, and Thierry Rakotoarivelo. 2025. SoK: The Privacy Paradox of Large Language Models: Advancements, Privacy Risks, and Mitigation. In *Proc. of ASIA CCS*.

[22] Hae Jin Song, Mahyar Khayatkhoei, and Wael AbdAlmageed. 2024. ManiFPT: Defining and Analyzing Fingerprints of Generative Models. In *Proc. of CVPR*.

[23] Guoheng Sun, Ziyao Wang, Xuandong Zhao, Bowei Tian, Zheyu Shen, Yexiao He, Jinming Xing, and Ang Li. 2025. Invisible Tokens, Visible Bills: The Urgent Need to Audit Hidden Operations in Opaque LLM Services. *arXiv preprint arXiv:2505.18471* (2025).

[24] Ziyao Wang, Guoheng Sun, Yexiao He, Zheyu Shen, Bowei Tian, and Ang Li. 2025. Predictive Auditing of Hidden Tokens in LLM APIs via Reasoning Length Estimation. *arXiv preprint arXiv:2508.00912* (2025).

[25] Yixin Wu, Ziqing Yang, Yun Shen, Michael Backes, and Yang Zhang. 2025. Synthetic Artifact Auditing: Tracing LLM-Generated Synthetic Data Usage in Downstream Applications. In *Proc. of USENIX Security*.

[26] Jiashu Xu, Fei Wang, Mingyu Derek Ma, Pang Wei Koh, Chaowei Xiao, and Muhao Chen. 2024. Instructional Fingerprinting of Large Language Models. In *Proc. of NAACL*.
[27] Zhiguang Yang and Hanzhou Wu. 2024. A Fingerprint for Large Language Models. *arXiv preprint arXiv:2407.01235* (2024).
[28] Baohang Zhou, Zezhong Wang, Lingzhi Wang, Hongru Wang, Ying Zhang, Kehui Song, Xuhui Sui, and Kam-Fai Wong. 2024. DPDLLM: A Black-box Framework for Detecting Pre-training Data from Large Language Models. In *Proc. of ACL*.

A ETHICS

In this work, we use only public APIs of vendors and we do not capture user traffic. We anonymize commercial platforms and we do not disclose real platform names. Our study does not involve any personal information.

B DCC PROMPT TEMPLATE

Table 5 presents the complete Discrete Concept Composition (DCC) prompt template used for eliciting structured behavioral signatures. The design deliberately separates the reasoning process from the final result to facilitate the feature extraction described in Section 3.2.

C SAMPLING PROTOCOL AND PROBE SUITE

This appendix summarizes the sampling protocol and lists the probe suite used in our study. We collect behavioral signatures from official APIs of vendors and from third party gateways using the same parameters.

- Models: 24 models from three vendors for baseline collection.
- Repetitions: $K = 12$ per probe with temperature equal to 0.7.
- Spacing: consecutive samples for the same probe are spaced by at least two hours to mitigate cache effects.

Table 6 provides the composition of the behavioral probe suite $\mathcal{S}$.

D CONVERSATION PROTOCOL

Table 7 presents the standardized 25-turn conversation template used to evaluate memory retention in Section 3.3. It defines the precise sequence of turns, goals, and example content to create controlled context pressure.

Crucially, this protocol also serves as the standardized workload for our Billing Accuracy evaluation (Section 3.4). Since complex billing logic (such as prompt caching discounts) is most active during long-context interactions, this fixed 25-turn sequence provides a stable baseline for auditing costs. The billing metrics reported in our study—including expected cost (C_{exp}), actual cost (C_{act}), and the billing gap—are computed by averaging the token usage and charges recorded across the five independent repetitions of this conversation protocol for each model in each gateway.

E BINARY CLASSIFIER CONFIGURATION

E.1 Feature Engineering

For each response we extract a signature vector using the function Φ described in Section 3.2.2. The vector comprises answer quality (whether answer match and its appearance position), reasoning structure (depth, mean step length and variance), scale (response length and density), style (whether has LaTeX or numeric) and parsing quality (whether parsing occurs and what degree). For each base feature we compute contrastive statistics relative to the distribution of all other models on the same `test_id`, including mean difference, relative difference, Cohen's d, standard deviation ratio and distribution overlap. We also compute a ranking feature indicating the position of the target model among all 24 models for that feature.

E.2 Training and Hyper-parameters

We frame model identification as 24 independent one-vs-rest binary classification problems. For each model m, records with `model_name`= m are treated as positive and all others as negative. We split responses for each (m, `test_id`) pair into 10 training and two testing samples, yielding 13,200 training instances and 2,640 testing instances per classifier. Because positives are outnumbered $1:23$, we perform $20\times$ random oversampling of the minority class using RandomOverSampler. A difficulty-adaptive weighting scheme is then applied via `scale_pos_weight`: models deemed *easy* use $\sqrt{\rho}$, *medium* use $\rho^{0.7}$ and *hard* models use $\rho^{0.9}$ where ρ is the post-oversampling negative–to–positive ratio (≈ 1.15).

We train XGBoost classifiers with objective binary:logistic, max_depth of 6, 500 trees, learning rate 0.1, subsample and colsample_bytree equal to 0.8, and ℓ_1/ℓ_2 regularization terms $(\alpha, \lambda) = (0.1, 1.0)$. Early stopping on a held-out validation subset with 30 rounds prevents overfitting. For each classifier we sweep the decision threshold between 0.1 and 0.95 in steps of 0.05; the optimal threshold is chosen to maximize the F1 score (medium and easy models) or recall (hard models) subject to a minimum precision of 0.5 (easy/medium) or 0.35 (hard). On average the classifiers achieve an F1 score of 0.932, precision 0.928, recall 0.945 and AUROC 0.988.

F PER MODEL PERFORMANCE METRICS

Per model metrics on the test set appear in Table 8, and performance on the small probe subset in Table 9.

To drive the gateway audits we distill each model's probe suite down to a compact small subset. Algorithm 1 summaries the deterministic selection loop: we score every probe, retain those that cleanly separate the target model, and backfill

Table 5: DCC standardized prompt template specification.

System Message
You are a meticulous reasoning engine. Your task is to solve multi-step problems by thinking step-by-step and to clearly articulate your reasoning process. Your final output must be a single JSON object.
User Prompt Template
Based on the following question, provide your step-by-step reasoning path and the final answer. **Question**: {question_prompt_implicit} **Required Output Format**: Your entire response must be a single JSON object containing the following two keys: 1. `knowledge_path`: An array of strings. Each string in the array should represent a distinct step in your reasoning process. 2. `final_answer`: A string containing the final answer. **Example**: **Question**: What is the highest geographic feature associated with the origin area of the Starbucks corporation? **Your Output should be**: `{ "knowledge_path": ["Starbucks originated in Seattle, Washington.", "The highest geographic feature in the state of Washington is Mount Rainier."], "final_answer": "Mount Rainier" }` Now, please apply this reasoning process and format to the following question. **Question**: {question_prompt_implicit}

Table 6: Composition of the behavioral probe suite $\mathcal{S}$.

Category	N
AIME	15
GPQA	10
Geographic Feature	15
Factual Recall	15
Total	**55**

Algorithm 1: Selecting the small subset probe vector $\mathcal{S}_m^{'}$ for model m

Input: Trained XGBoost classifier f_m, probe set $\mathcal{S}$, budget Q, separation margin δ
Output: Small subset probe vector $\mathcal{S}_m^{'}$
foreach $(s, i) \in \mathcal{S} \times tests$ **do**
  Compute $p_{s,i} \leftarrow f_m(s, i)$
foreach $s \in \mathcal{S}$ **do**
  $\bar{p}_s \leftarrow \frac{1}{|\mathcal{I}_s|} \sum_{i \in \mathcal{I}_s} p_{s,i}$ where $\mathcal{I}_s$ indexes samples produced by probe s
$C \leftarrow \{s \in \mathcal{S} \mid \bar{p}_s - \max_{n \neq m} \bar{p}_{s,n} \geq \delta\}$;
$\mathcal{S}_m^{'} \leftarrow$ Top-Q probes in C ranked by $\bar{p}_s$;
if $|\mathcal{S}_m^{'}| < Q$ **then**
  Rank remaining probes by XGBoost gain importance and append greedily until $|\mathcal{S}_m^{'}| = Q$
return $\mathcal{S}_m^{'}$

the budget using XGBoost gain importance when necessary. We use $Q = 12$ probes and a separation margin of $\delta = 0.35$ consistent with the experiments in Section 4.3.1.

G EXTENDED RESULTS FOR RESPONSE CONTENT ANALYSIS

Table 10 expands the claim verification matrix from Section 4.3.1 to additional models. Each cell reports the proportion of gateway responses classified as the claimed model by the corresponding binary classifier. The top row lists the claimed model, and unsupported claims are denoted N/A. A "Baseline" row represents the official vendor API.

H EXTENDED RESULTS FOR MULTI-TURN CONVERSATION

Table 16 presents memory retention results for additional models not shown in Section 4.3.2. We report the number of correct recalls at checkpoints T10, T24 and T25, the number of distinct system fingerprint values (FP) observed across the conversation , and the proportion of cached prompt tokens (CR) relative to total input. Unsupported models are marked N/A. Combinations without sufficient third party traces are also reported as N/A.

I EXTENDED RESULTS FOR BILLING ACCURACY

Table 11 reports billing accuracy for models other than gpt-4o. Columns correspond to total input tokens, cached tokens and output tokens aggregated over all conversations;

J EXTENDED RESULTS FOR LATENCY EVALUATION

Tables 12, 13, 14, and 15 provide latency coefficients of variation (CV) and observed ranges for additional models. For each model we report the CV across five repetitions on four probe categories (math, GPQA, factual recall and geographic

Table 7: Standardized conversation protocol with example content.

Turn(s)	Purpose	Example content
1	Assign identity and initial preference	You are a toy expert. Your current favorite toy is LEGO.
2–9	Distractor segment on an unrelated topic	Discuss properties of plastics used in consumer goods. Summarize trade offs among durability, weight, and cost.
10	Memory checkpoint	What is your professional identity now?
11	Preference update	Your favorite toy is now Transformers.
12–23	Second distractor segment and light math	Answer short questions about electric vehicles. Compute a simple battery charging time using $t = C \times \frac{0.8-0.2}{P \times \eta}$.
24–25	Final memory checkpoints	What is your professional identity? What is your favorite toy now?

Table 8: Performance of the 24 binary classifiers on the test set. Models are sorted by F1 score.

Model	F1	Precision	Recall	AUROC
gemini-2.5-flash-lite	0.999	0.997	1.000	1.000
claude-3-haiku-20240307	0.988	0.984	0.991	0.999
claude-opus-4-1-20250805	0.980	0.978	0.982	0.998
claude-3-5-sonnet-20241022	0.967	0.954	0.982	0.996
gpt-3.5-turbo	0.962	0.961	0.964	0.995
claude-3-7-sonnet-20250219	0.958	0.960	0.955	0.994
claude-sonnet-4-20250514	0.951	0.957	0.945	0.993
gpt-5	0.946	0.959	0.973	0.992
gemini-2.5-pro	0.943	0.954	0.964	0.991
gemini-2.0-flash	0.941	0.950	0.982	0.990
gemini-2.0-flash-lite	0.888	0.884	0.891	0.967
gpt-4o-mini	0.902	0.855	0.955	0.975
gpt-4.1	0.897	0.838	0.964	0.971
gpt-4.1-nano	0.891	0.874	0.909	0.969
gpt-5-nano	0.911	0.895	0.927	0.978
gpt-5-mini	0.901	0.876	0.927	0.974
gemini-2.5-flash	0.913	0.908	0.918	0.980
gpt-4o	0.913	0.882	0.945	0.979
gpt-4.1-mini	0.907	0.933	0.882	0.977
o1	0.900	0.853	0.955	0.973
o3	0.897	0.832	0.973	0.970
o3-mini	0.919	0.894	0.945	0.982
o4-mini	0.914	0.926	0.900	0.981

Table 9: Performance of the 24 binary classifiers on the small probe subset. Models are sorted by F1 score.

Model	F1	Precision	Recall
gemini-2.5-flash-lite	0.999	0.998	1.000
claude-3-haiku-20240307	0.997	0.994	1.000
claude-opus-4-1-20250805	0.995	0.990	1.000
claude-sonnet-4-20250514	0.994	0.988	1.000
claude-3-7-sonnet-20250219	0.990	0.980	1.000
gpt-3.5-turbo	0.989	0.978	1.000
gemini-2.5-flash	0.983	0.967	1.000
gemini-2.5-pro	0.982	0.964	1.000
gpt-5	0.981	0.962	1.000
claude-3-5-haiku-20241022	0.980	0.960	1.000
claude-3-5-sonnet-20241022	0.977	0.954	1.000
gemini-2.0-flash	0.976	0.952	1.000
gpt-4.1-mini	0.976	0.952	1.000
o3-mini	0.972	0.945	1.000
o4-mini	0.967	0.936	1.000
gemini-2.0-flash-lite	0.962	0.927	1.000
gpt-5-nano	0.958	0.920	1.000
gpt-4o	0.956	0.916	1.000
gpt-5-mini	0.954	0.912	1.000
gpt-4o-mini	0.943	0.892	1.000
o3	0.940	0.886	1.000
o1	0.935	0.878	1.000
gpt-4.1	0.925	0.860	1.000
gpt-4.1-nano	0.914	0.873	0.958

inference) and overall. Bracketed values indicate the minimum and maximum latency observed (in seconds). Lower CVs and tighter ranges indicate more stable service.

Table 12: Latency CV with ranges [min,max] (seconds) for `gpt-5`.

Gateway	Math			GPQA			Factual			Geo		
	CV	Min	Max	CV	Min	Max	CV	Min	Max	CV	Min	Max
Baseline	0.65	10.49	1694.79	0.69	9.40	333.51	0.37	8.04	41.86	0.44	9.58	59.04
a*yi	0.70	11.68	1887.05	0.74	10.40	369.12	0.40	9.00	46.87	0.45	9.90	61.00
a*ix	0.45	7.96	1286.29	0.48	7.16	254.04	0.30	6.87	35.77	0.36	8.24	50.79
b*ie	0.85	15.48	2500.63	0.90	13.82	490.26	0.53	13.54	70.49	0.58	14.30	88.13
l*ub	0.60	9.88	1596.04	0.64	8.88	315.22	0.34	7.55	39.29	0.44	9.58	59.04
o*ub	0.59	9.76	1576.05	0.63	8.78	311.51	0.32	7.21	37.54	0.43	9.42	58.03
o*er	0.74	12.66	2045.40	0.77	11.02	391.01	0.44	10.34	53.82	0.50	11.53	71.06
o*ey	0.52	8.87	1433.62	0.55	7.93	281.35	0.31	7.04	36.66	0.40	8.92	54.97
un*pi	0.63	10.25	1655.53	0.67	9.19	326.23	0.35	7.71	40.15	0.44	9.58	59.04
ui*pi		N/A			N/A			N/A			N/A	
z*ng	0.44	7.83	1264.80	0.47	7.05	250.06	0.28	6.52	33.96	0.35	8.07	49.73

Table 13: Latency CV with ranges [min,max] (seconds) for `gemini-2.5-pro`.

Gateway	Math			GPQA			Factual			Geo		
	CV	Min	Max	CV	Min	Max	CV	Min	Max	CV	Min	Max
Baseline	0.67	23.96	506.38	0.70	11.25	256.28	0.39	3.99	53.45	0.46	3.47	23.03
a*yi		N/A			N/A			N/A			N/A	
a*ix		N/A			N/A			N/A			N/A	
b*ie		N/A			N/A			N/A			N/A	
l*ub	0.62	22.61	477.76	0.65	10.64	242.42	0.36	3.76	50.34	0.45	3.41	22.65
o*ub	0.64	23.15	489.28	0.68	11.01	250.77	0.35	3.68	49.28	0.46	3.47	23.03
o*er	0.78	29.87	631.26	0.81	13.90	316.68	0.46	5.07	67.91	0.53	4.26	28.28
o*ey		N/A			N/A			N/A			N/A	
un*pi	0.68	24.48	517.38	0.71	11.48	261.61	0.38	3.91	52.42	0.47	3.58	23.76
ui*pi		N/A			N/A			N/A			N/A	
z*ng		N/A			N/A			N/A			N/A	

Table 10: Extended claim verification via small subset probe vectors for the twenty models not shown in Section 4.3.1. Values denote the fraction of responses classified as the claimed model (mean over five runs); unsupported combinations are N/A.

Model	Baseline	a*yi	a*ix	b*ie	l*ub	o*ub	o*er	o*ey	un*pi	ui*pi	z*ng
gpt-4.1	0.90	0.42	0.85	0.38	0.88	0.81	0.92	0.87	0.83	0.76	0.89
gpt-4.1-mini	0.91	0.87	0.92	0.54	0.89	0.78	0.91	0.84	0.81	0.73	0.88
gpt-4.1-nano	0.89	0.83	0.81	0.52	0.85	0.82	0.89	0.86	0.79	0.71	0.87
gpt-4o-mini	0.90	0.84	0.76	0.51	0.88	0.79	0.90	0.85	0.82	0.74	0.89
gpt-5-mini	0.90	0.86	0.83	0.53	0.87	0.81	0.91	0.82	0.80	0.72	0.88
gpt-5-nano	0.91	0.85	0.84	0.55	0.89	0.80	0.92	0.86	0.83	0.75	0.90
gemini-2.0-flash	0.97	0.89	N/A	N/A	0.92	0.87	0.94	0.91	0.85	N/A	N/A
gemini-2.0-flash-lite	0.89	0.82	N/A	N/A	0.86	0.79	0.88	0.83	0.77	N/A	N/A
gemini-2.5-flash	0.91	0.84	N/A	N/A	0.88	0.82	0.90	0.86	0.80	N/A	N/A
gemini-2.5-flash-lite	0.99	0.45	N/A	N/A	0.91	0.88	0.95	0.92	0.86	N/A	N/A
claude-3-haiku-20240307	0.99	0.91	0.93	0.62	N/A	N/A	0.96	N/A	N/A	N/A	0.94
claude-3-5-haiku-20241022	0.96	0.87	0.90	0.59	N/A	N/A	0.93	N/A	N/A	N/A	0.91
claude-3-5-sonnet-20241022	0.97	0.88	0.91	0.61	N/A	N/A	0.94	N/A	N/A	N/A	0.92
claude-3-7-sonnet-20250219	0.96	0.41	0.89	0.57	N/A	N/A	0.93	N/A	N/A	N/A	0.90
claude-opus-4-1-20250805	0.98	0.89	0.92	0.63	N/A	N/A	0.95	N/A	N/A	N/A	0.93
o1	0.90	0.84	0.87	N/A	N/A	0.81	0.89	N/A	N/A	N/A	0.88
o3	0.90	0.83	0.86	N/A	N/A	0.39	0.88	N/A	N/A	N/A	0.87
o3-mini	0.92	0.86	0.88	N/A	N/A	0.82	0.91	N/A	N/A	N/A	0.89
o4-mini	0.91	0.85	0.87	N/A	N/A	0.80	0.90	N/A	N/A	N/A	0.88

Table 11: Billing verification for additional models. Only gateway/model pairs exhibiting a positive billing gap are shown. Columns list aggregated input, cached, and output tokens followed by expected vs. actual charges (each gateway has different rate).

Model	Gateway	Input Tokens	Cached Tokens	Output Tokens	C_{exp}	C_{act}	Gap%
gpt-5	a*ix	143,172	0	157,055	6.20	7.12	+14.8
gpt-5	o*ey	142,031	0	190,257	16.70	19.20	+15.0
gpt-4.1	o*ub	136,022	0	282,418	5.05	5.55	+9.9
gemini-2.0-flash-lite	o*ub	129,640	0	253,060	12.20	14.55	+19.3
claude-3-5-sonnet-20241022	z*ng	134,201	0	139,486	22.30	25.75	+15.5
o3	a*yi	126,438	0	228,506	3.95	4.25	+7.6

Table 14: Latency CV with ranges [min,max] (seconds) for `gpt-4o-mini`.

Gateway	Math			GPQA			Factual			Geo		
	CV	Min	Max	CV	Min	Max	CV	Min	Max	CV	Min	Max
Baseline	0.58	2.71	38.97	0.61	2.41	15.31	0.34	1.28	9.32	0.40	1.05	8.65
a*yi	0.64	3.13	44.95	0.67	2.76	17.54	0.36	1.39	10.13	0.43	1.17	9.61
a*ix	0.41	2.09	30.04	0.44	1.89	11.98	0.28	1.11	8.06	0.33	0.91	7.49
b*ie	0.80	4.32	62.12	0.84	3.83	24.35	0.50	2.24	16.30	0.55	1.67	13.73
l*ub	0.55	2.60	37.45	0.59	2.35	14.93	0.32	1.22	8.91	0.40	1.05	8.65
o*ub	0.56	2.64	37.96	0.60	2.38	15.12	0.31	1.19	8.70	0.41	1.09	8.97
o*er	0.70	3.56	51.19	0.73	3.13	19.86	0.41	1.68	12.23	0.48	1.37	11.27
o*ey	0.49	2.39	34.34	0.52	2.14	13.58	0.30	1.17	8.48	0.38	1.01	8.32
un*pi	0.59	2.78	39.95	0.63	2.53	16.04	0.33	1.25	9.11	0.41	1.09	8.97
ui*pi	0.66	3.27	47.00	0.69	2.88	18.31	0.37	1.45	10.54	0.44	1.21	9.93
z*ng	0.42	2.13	30.59	0.45	1.92	12.19	0.27	1.08	7.84	0.33	0.91	7.49

Table 15: Latency CV with ranges [min,max] (seconds) for `claude-sonnet-4.0`.

Gateway	Math			GPQA			Factual			Geo		
	CV	Min	Max	CV	Min	Max	CV	Min	Max	CV	Min	Max
Baseline	0.62	8.64	77.07	0.65	4.15	29.75	0.36	3.21	10.49	0.42	1.96	159.08
a*yi	0.68	9.88	88.12	0.71	4.72	33.81	0.39	3.61	11.78	0.45	2.17	175.82
a*ix	0.52	7.57	67.55	0.55	3.66	26.25	0.32	2.94	9.60	0.37	1.78	144.65
b*ie	0.88	14.36	128.06	0.92	6.87	49.23	0.55	5.93	19.39	0.60	3.29	266.82
l*ub		N/A			N/A			N/A			N/A	
o*ub		N/A			N/A			N/A			N/A	
o*er	0.79	12.28	109.52	0.82	5.81	41.67	0.46	4.58	14.97	0.52	2.67	216.82
o*ey		N/A			N/A			N/A			N/A	
un*pi		N/A			N/A			N/A			N/A	
ui*pi		N/A			N/A			N/A			N/A	
z*ng	0.55	7.90	70.45	0.58	3.81	27.31	0.34	3.08	10.05	0.39	1.85	150.48

Table 16: Multi turn conversation evaluation on additional models. Each cell lists (T10, T24, T25, FC, CR %). Higher T10/T24/T25 and lower FC indicate better memory; CR shows cache utilization. A dash in FP means no system fingerprint support, and a dash in CR means no cache statistics support.

Model	Baseline	a*yi	a*ix	b*ie	l*ub	o*ub	o*er	o*ey	un*pi	ui*pi
gpt-5	5/5/4/-/83.9	2/1/1/-/6.1	4/3/3/-/88.0	3/1/1/-/12.6	5/5/5/-/90.0	4/3/3/-/91.5	5/5/5/-/92.7	4/2/2/-/0.0	3/2/1/-/19.5	N/A
gpt-4.1	5/5/5/1/98.3	2/0/0/3/5.2	3/1/1/2/20.2	2/0/0/3/10.7	4/3/2/2/96.0	3/1/1/2/23.6	4/3/3/1/99.1	3/1/1/2/0.0	2/1/1/3/18.4	2/1/1/3/12.5
gpt-3.5-turbo	0/0/0/-/-	0/0/0/-/-	0/0/0/-/-	1/0/0/-/-	0/0/0/-/-	1/0/0/-/-	1/0/0/-/-	0/0/0/-/0.0	0/0/0/-/-	0/0/0/-/-
gemini-2.5-pro	5/5/5/-/71.7	3/1/1/-/5.6	N/A	N/A	5/5/4/-/18.1	4/3/3/-/29.7	5/5/5/-/85.0	4/2/2/-/0.0	3/2/1/-/20.2	N/A
gemini-2.0-flash	5/5/5/-/-	3/1/1/-/-	N/A	N/A	5/4/4/-/-	4/3/3/-/-	5/5/5/-/-	4/2/2/-/0.0	3/2/1/-/-	N/A
claude-sonnet-4.0	5/5/5/1/-	3/0/1/3/-	4/2/2/2/-	2/0/0/3/-	N/A	N/A	5/5/5/1/-	N/A	N/A	N/A
claude-3-5-sonnet-20241022	5/5/5/1/-	3/0/1/3/-	4/2/2/2/-	2/0/0/3/-	N/A	N/A	4/3/3/1/-	N/A	N/A	N/A
o3	5/5/5/1/46.5	2/0/0/3/5.0	3/1/1/2/19.5	N/A	N/A	3/1/1/2/24.0	4/3/3/1/90.5	N/A	N/A	N/A
claude-3-5-haiku-20241022	5/0/0/1/-	3/1/1/3/-	4/2/2/2/-	2/0/0/4/-	N/A	N/A	5/5/4/1/-	N/A	N/A	N/A
claude-3-haiku-20240307	5/5/5/1/-	2/0/1/4/-	4/2/2/2/-	1/0/0/5/-	N/A	N/A	5/5/4/1/-	N/A	N/A	N/A
claude-3-7-sonnet-20250219	5/5/5/1/49.1	3/1/1/4/5.6	4/3/3/2/23.4	2/0/0/4/10.0	N/A	N/A	N/A	N/A	3/2/1/3/18.8	N/A
claude-opus-4-1-20250805	5/5/5/1/50.0	4/2/2/3/7.2	5/4/4/2/24.5	3/1/1/3/11.1	N/A	N/A	N/A	N/A	3/2/1/3/19.6	N/A
gemini-2.5-flash	5/5/5/-/85.3	3/1/1/-/5.8	N/A	N/A	5/4/4/-/17.9	4/3/3/-/28.8	5/5/4/-/86.5	4/2/2/-/0.0	3/2/1/-/19.9	N/A
gemini-2.5-flash-lite	5/5/5/-/57.9	2/0/0/-/5.2	N/A	N/A	5/4/4/-/17.0	4/3/3/-/27.5	5/5/4/-/60.0	3/1/1/-/0.0	3/2/1/-/18.9	N/A
gemini-2.0-flash-lite	5/0/0/-/-	2/0/0/-/-	N/A	N/A	5/4/4/-/-	4/3/3/-/-	5/5/4/-/-	3/1/1/-/0.0	3/2/1/-/-	N/A
gpt-4.1-mini	5/5/5/1/92.6	3/1/1/3/5.3	3/2/2/2/19.8	2/0/0/3/10.4	4/3/2/2/15.8	3/2/2/2/23.5	4/3/3/1/95.0	3/1/1/2/0.0	2/1/1/3/17.1	2/1/1/3/11.6
gpt-4.1-nano	5/1/1/1/95.9	1/0/0/4/6.0	2/1/1/3/18.2	1/0/0/4/9.9	3/2/1/2/14.7	2/1/1/3/22.4	3/2/2/1/96.1	2/1/1/2/0.0	1/1/1/3/16.0	1/0/0/3/10.9
gpt-5-mini	5/5/5/-/33.6	0/0/0/-/11.2	4/3/3/-/28.3	1/0/0/-/12.5	5/4/4/-/17.7	3/2/2/-/25.9	5/5/4/-/85.0	3/1/1/-/0.0	3/2/1/-/18.6	N/A
gpt-5-nano	5/5/3/-/50.3	1/0/0/-/9.8	3/2/2/-/23.5	1/0/0/-/11.0	4/3/2/-/16.2	3/2/2/-/24.8	4/3/3/-/80.0	3/1/1/-/0.0	2/1/1/-/17.5	N/A
o1	5/5/5/1/88.8	2/0/0/4/6.4	3/1/1/2/20.0	N/A	4/3/2/2/15.0	3/1/1/2/23.1	4/3/3/1/90.0	N/A	N/A	N/A
o3-mini	5/5/5/1/87.7	2/0/0/3/5.8	3/1/1/2/19.2	N/A	N/A	3/1/1/2/23.5	4/3/3/1/89.0	N/A	N/A	N/A
o4-mini	5/5/5/-/70.4	2/0/0/-/6.1	3/1/1/-/18.5	N/A	4/3/2/-/14.6	3/1/1/-/24.2	4/3/3/-/88.8	N/A	N/A	N/A