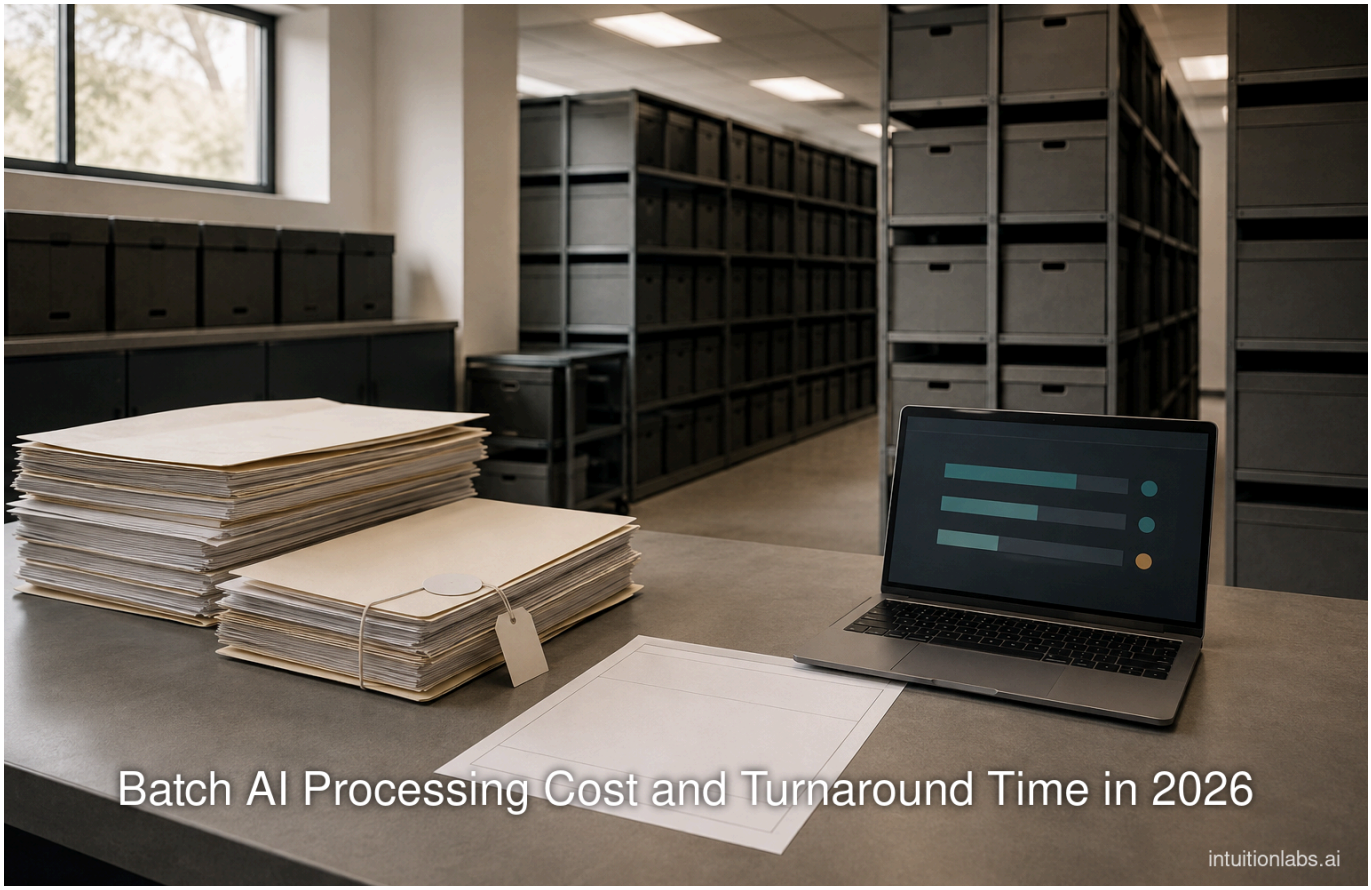
Batch AI Processing Cost and Turnaround Time in 2026

IntuitionLabs.ai · Adrien Laurent · 2026-09-05 · batch ai processing · llm batch api pricing · batch inference turnaround time · openai batch api · anthropic batch api · aws bedrock batch inference · async inference · idempotency · prompt caching · llm cost optimization



Executive Summary

Batch (asynchronous) processing has become the default cost lever for large-scale large language model (LLM) workloads. As of September 2026, the cited OpenAI, Anthropic, Gemini API, Azure OpenAI Global Batch, and qualifying AWS Bedrock batch offerings advertise a 50% reduction relative to their cited synchronous prices; buyers should verify the applicable model, modality, service, and region. In particular, Google lists Gemini Embedding batch input at $0.00012 per 1,000 input tokens versus $0.00015 for online input tokens, a 20% reduction. OpenAI's Batch API documents a "50% cost discount compared to synchronous APIs" (platform.openai.com), Anthropic's Message Batches API offers "a 50% discount on all usage compared to standard API prices" (docs.claude.com), Google's Gemini Batch API is "designed to process large volumes of requests asynchronously at 50% of the standard cost" (ai.google.dev), and Azure OpenAI's Batch offering "returns completions within 24 hours for a 50% discount on Global Standard Pricing" (azure.microsoft.com). AWS Bedrock's batch inference is priced "at a 50% lower price compared to on-demand inference pricing" (aws.amazon.com).

Turnaround, not just price, is the second axis this report measures. AWS Bedrock does not share that fixed 24-hour completion window: its `timeoutDurationInHours` setting has a valid range of 24 to 168 hours (AWS API reference). OpenAI states batch jobs complete "within 24 hours (and often more quickly)" (platform.openai.com); Anthropic documents "most batches completing within 1 hour" (docs.claude.com); Google states its "target turnaround time is 24 hours, but in majority of cases, it is much quicker" (ai.google.dev); and Azure's documentation says its service "aims to process batch requests within 24 hours, but it doesn't expire jobs that take longer" (learn.microsoft.com), a materially different failure mode than OpenAI's or Anthropic's hard 24-hour expiration.

Using a reproducible monthly workload of 6,000,000 input tokens and 600,000 output tokens (100 documents at 60,000 input and 6,000 output tokens each, a volume IntuitionLabs previously documented for enterprise summarization (intuitionlabs.ai)), this report calculates batch costs ranging from **$0.63** (OpenAI GPT-4o mini) to **$22.50** (Anthropic Claude Opus 5) per month, against roughly double those figures at each provider's documented synchronous rate.

Failure recovery and idempotency separate the providers further. OpenAI writes failed individual requests to a dedicated error file rather than failing the whole batch (platform.openai.com); Anthropic explicitly states that "errored" and "expired" requests are never billed (docs.claude.com); and AWS Bedrock's job-creation API exposes a `clientRequestToken` explicitly "to ensure the API request completes only once" (docs.aws.amazon.com), the clearest documented idempotency mechanism among the five providers. Result retention also varies materially: Anthropic keeps results downloadable for 29 days (docs.claude.com), OpenAI automatically deletes its output file 30 days after the batch is complete (OpenAI Batch API), and Azure lets customers configure retention from 14 to 30 days (learn.microsoft.com). This report documents the pricing, turnaround commitments, expiration behavior, retry semantics, and idempotency mechanics of each provider's batch offering, sets out a reproducible method for costing a batch workload, and distinguishes documented vendor commitments from operational targets throughout.

Introduction and Background

Batch AI processing (also called asynchronous inference) refers to submitting a large collection of model requests as a single job that a provider processes outside the interactive request/response cycle, typically returning results within a defined completion window rather than within seconds. It is a distinct question from headline per-token pricing: two providers can charge similar synchronous rates while offering very different asynchronous discounts, turnaround guarantees, and failure semantics, and those differences compound at scale. Organizations that batch-process legal contracts, clinical trial documents, financial filings, or customer support transcripts by the tens of thousands make purchasing and architecture decisions based on this specific set of tradeoffs, which is why this report treats cost and turnaround as a paired question rather than pricing alone.

As of September 2026, five providers publish a documented batch or asynchronous tier for LLM inference: OpenAI's Batch API, Anthropic's Message Batches API, Google's Gemini Batch API (and the equivalent Vertex AI batch prediction tier), Microsoft's Azure OpenAI Service Batch (Global Batch) deployment, and AWS Bedrock's batch inference feature. All five apply a discount relative to synchronous pricing. AWS Bedrock's `timeoutDurationInHours` setting, however, has a valid range of 24 to 168 hours rather than a fixed 24-hour completion window (AWS API reference).

This report builds on IntuitionLabs' existing analysis of LLM API pricing (intuitionlabs.ai), which compared synchronous per-token rates across providers; that article did not address batch-specific economics, and this report is intended as its complement. IntuitionLabs describes itself as a life-sciences and artificial intelligence (AI) consultancy founded in 2023 (intuitionlabs.ai), and its existing pricing work is used here as one documented reference point for a realistic document-processing workload, not as a benchmark of batch performance itself.

The remainder of this report proceeds provider by provider (OpenAI, Anthropic, Google and Azure, then AWS Bedrock), synthesizes failure-recovery and idempotency practices across all of them, builds a comparative matrix and a reproducible worked cost example, and closes with implications for organizations planning large-scale document-processing pipelines. Every price, turnaround figure, and technical limit below is dated to its September 2026 observation and sourced to the vendor's own documentation, since these figures are among the most volatile in the industry and are expected to change.

OpenAI Batch API: Pricing, Turnaround, and Operational Limits

OpenAI's Batch API is designed, in the company's own words, to "send asynchronous groups of requests" at a "50% cost discount compared to synchronous APIs" (platform.openai.com). The discount applies uniformly across supported models and endpoints, and OpenAI states each batch "completes within 24 hours (and often more quickly)" (platform.openai.com); as of this writing, the completion window is fixed rather than configurable to a shorter or longer period (platform.openai.com).

On per-model pricing, OpenAI's official pricing reference lists batch rates for its current model lineup, including **GPT-4o** at $1.25 input and $5.00 output per 1 million (1M) tokens, **GPT-4o mini** at $0.075 input and $0.30 output per 1M tokens, **GPT-4.1** at $1.00 input and $4.00 output per 1M tokens, and **GPT-5** at $0.625 input and $5.00 output per 1M tokens (platform.openai.com), each representing half of the corresponding synchronous rate under OpenAI's documented 50% batch discount policy.

Several operational specifics distinguish OpenAI's implementation:

- **Separate rate-limit pool:** "Batch API rate limits are separate from existing per-model rate limits" (platform.openai.com), and OpenAI confirms that "using the Batch API will not consume tokens from your standard per-model rate limits" (platform.openai.com), meaning a large batch job does not compete with production, real-time traffic for capacity.
- **Size limits:** "A single batch may include up to 50,000 requests, and a batch input file can be up to 200 MB in size" (platform.openai.com), and organizations "can create up to 2,000 batches per hour" (platform.openai.com).
- **Expiration behavior:** if a batch does not finish within its window, "unfinished requests within that batch are cancelled" while completed responses remain available, and the customer is billed "for tokens consumed from any completed requests" (platform.openai.com); individual expired requests surface a `batch_expired` error stating "this request could not be executed before the completion window expired" (platform.openai.com).
- **Partial failure isolation:** "any failed requests in the batch will have their error information written to an error file" (platform.openai.com) rather than failing the entire job; a batch only reaches an outright "failed" state when "the input file has failed the validation process" before any requests run (platform.openai.com).

- **Cancellation lag:** cancelling a batch is not instantaneous; the job's status moves to "cancelling" until "in-flight requests are complete (up to 10 minutes)" (platform.openai.com).
- **Result correlation, not guaranteed order:** OpenAI explicitly warns that its output line order may not match the input line order (platform.openai.com), so every request requires a caller-supplied `custom_id` for matching results back to inputs rather than relying on file position.

OpenAI's Batch API guide states that "The output file will automatically be deleted 30 days after the batch is complete," and the API supports multiple endpoint types beyond chat completions, including the Responses API, embeddings, and image and video generation, all through the same asynchronous submission model (OpenAI Batch API).

Anthropic Message Batches API: Pricing, Turnaround, and Operational Limits

Anthropic's Message Batches API is built around the same discount principle, processing "large volumes of Messages requests asynchronously" while "cutting costs by 50% and increasing throughput," and its documentation states plainly that "the Message Batches API offers a 50% discount on all usage compared to standard API prices" (docs.claude.com), applying to input tokens, output tokens, and cache-related token categories alike.

On per-model batch pricing, Anthropic's official pricing table lists **Claude Sonnet 5** at $1 per million-token (MTok) input and $5 per MTok output (docs.claude.com), **Claude Opus 5** at $2.50 per MTok input and $12.50 per MTok output (docs.claude.com), and **Claude Haiku 4.5** at $0.50 per MTok input and $2.50 per MTok output (docs.claude.com), each already representing the batch (discounted) rate.

On turnaround, Anthropic's documentation states that the system processes "each batch as fast as possible, with most batches completing within 1 hour" (docs.claude.com); results become accessible "when all messages have completed or after 24 hours, whichever comes first" (docs.claude.com), and "batches expire if processing does not complete within 24 hours" (docs.claude.com). Status is retrieved by polling rather than a push notification, and results become retrievable once processing ends.

Anthropic bounds each batch job at "either 100,000 Message requests or 256 MB in size, whichever is reached first" (docs.claude.com), roughly double OpenAI's 50,000-request cap. Results are retrievable as a downloadable or streamable `.jsonl` file in which "each line is a valid JSON object representing the result of a single request" (docs.claude.com), and remain downloadable for 29 days: "batch results are available for 29 days after creation" (docs.claude.com), after which the batch object itself is still viewable but its outputs are not.

On failure semantics, Anthropic defines four terminal per-request outcomes, succeeded, errored, canceled, and expired, and is explicit that the latter three carry no charge. An errored request is one where a request "encountered an error and a message was not created," with the documentation noting "possible errors include invalid requests and internal server errors" (docs.claude.com). A canceled request is one where the "user canceled the batch before this request could be sent to the model" (docs.claude.com), and an expired request is one where the "batch reached its 24-hour expiration before this request could be sent to the model" (docs.claude.com); Anthropic states plainly that users are not billed for any of these three outcomes. Cancelling a running batch produces partial results

for whatever completed before cancellation took effect, and once submitted a batch cannot be edited: "once a batch has been submitted, it cannot be modified" (docs.claude.com), so corrections require cancelling and resubmitting a new batch.

Each request requires a caller-assigned `custom_id`, which "must be 1 to 64 characters and contain only alphanumeric characters, hyphens, and underscores" (docs.claude.com), serving the same result-matching function as OpenAI's identically named field. Anthropic also notes that prompt caching can stack with the batch discount, though because batch requests are processed asynchronously and concurrently, cache hits are best-effort: "cache hit rates ranging from 30% to 98%, depending on their traffic patterns" (docs.claude.com) are typical.

Google Gemini and Azure OpenAI Batch Offerings

Google Gemini Batch API and Vertex AI

Google's Gemini Batch API follows the same 50% discount structure: it is "designed to process large volumes of requests asynchronously at 50% of the standard cost" (ai.google.dev), a discount Google's own developer pricing page repeats as a feature line: "Batch API (50% cost reduction)" (ai.google.dev). On turnaround, Google states its "target turnaround time is 24 hours, but in majority of cases, it is much quicker" (ai.google.dev), and formalizes this as a documented service level objective: "batch jobs are designed to complete within a 24-hour turnaround time" (ai.google.dev). The same mechanism extends to image generation, where developers can "get higher rate limits in exchange for a turnaround of up to 24 hours" (ai.google.dev).

Google's expiration window is notably longer than OpenAI's or Anthropic's: a Gemini batch job is only marked expired "because it was running or pending for more than 48 hours," at which point "the job will not have any results to retrieve" (ai.google.dev), a 48-hour hard ceiling against a 24-hour soft target, effectively giving Gemini batch jobs a full extra day of grace before results are lost entirely. On Vertex AI, Google's enterprise cloud platform, the generative AI pricing page presents a distinct "Priority" versus "Flex/Batch" pricing toggle for each model, confirming batch as a formally separate pricing tier from standard interactive pricing (cloud.google.com). Notably, the discount is not uniform across every line item: for Gemini Embedding, Vertex AI's own pricing page lists online input requests at $0.00015 per 1,000 tokens against $0.00012 per 1,000 tokens for batch requests (cloud.google.com), a 20% reduction rather than the 50% figure quoted in Google's general batch messaging. This is a documented discrepancy worth flagging for buyers: Google's headline "50% of standard cost" language describes its Gemini API batch tier broadly, while at least one Vertex AI line item shows a smaller discount, so the blanket percentage should not be assumed to apply identically to every model and modality without checking the specific line item.

Azure OpenAI Service Batch

Microsoft's Azure OpenAI Service offers the same underlying OpenAI models through a Global Batch deployment type that "returns completions within 24 hours for a 50% discount on Global Standard Pricing" (azure.microsoft.com), and Microsoft states that "language models are also now available in the Batch API for global deployments and three regions" (azure.microsoft.com) as of this observation. Azure's operational model diverges from OpenAI's own API in one important respect: rather than expiring unfinished jobs at the 24-hour mark,

Microsoft Learn's documentation states "the service aims to process batch requests within 24 hours, but it doesn't expire jobs that take longer. You can cancel the job anytime" (learn.microsoft.com). If a customer does cancel, billing is limited to completed work: "you pay for any completed work" (learn.microsoft.com).

The `completion_window` parameter itself is rigid: "if you set any other value than 24h your job will fail," and "jobs taking longer than 24 hours will continue to execute until canceled" (learn.microsoft.com), meaning the 24-hour figure functions as a target rather than a hard cutoff on Azure, the opposite failure mode from OpenAI's own API. Output-file retention is configurable rather than fixed: batch results otherwise "expire 14 days after they're created," but customers can "set seconds to a value from 1209600 through 2592000 to select an expiration period from 14 through 30 days" (learn.microsoft.com). Azure isolates batch capacity from production traffic the same way OpenAI and Anthropic do: "global batch requests have a separate enqueued token quota" (learn.microsoft.com), and Microsoft's documented retry pattern for quota exhaustion is to queue "multiple batch jobs with exponential backoff" (learn.microsoft.com) and let the next job start automatically once quota frees up.

AWS Bedrock Batch Inference and Idempotent Job Design

Amazon Bedrock offers batch inference for select foundation models from providers including Anthropic, Meta, and Mistral AI, and its own Amazon models, priced, per AWS's own pricing page, at "a 50% lower price compared to on-demand inference pricing" (aws.amazon.com), consistent with the discount level documented by every other provider in this report. AWS documents `timeoutDurationInHours` as a batch-job timeout setting with a valid range of 24 to 168 hours (AWS API reference). Bedrock's implementation is also notable for two specific design choices around reliability and idempotency that are less explicit in OpenAI's, Anthropic's, or Google's own documentation.

First, on partial failure, AWS documents that a completed batch job produces a manifest file recording total, processed, success, and error record counts, and that individual failures are represented inline rather than by a separate error file: "an error object replaces the modelOutput field in any line where there was an error in inference" (docs.aws.amazon.com). While a job is still running, Bedrock exposes live progress counters, including a `processedRecordCount` field described as "the number of records processed so far, which includes both successes and errors" (docs.aws.amazon.com), giving operators mid-job visibility that none of the other four providers document as explicitly. If a job is stopped before completion, AWS states plainly that the customer is "charged for tokens that have already been processed" (docs.aws.amazon.com), the same completed-work billing principle documented by Azure and OpenAI.

Second, and most directly relevant to the "idempotency" question this report was asked to address, Bedrock's `CreateModelInvocationJob` API accepts a `clientRequestToken` parameter whose stated purpose is "to ensure the API request completes only once" (docs.aws.amazon.com), a first-class idempotency key at the job-submission level. This is a materially different mechanism from OpenAI's and Anthropic's `custom_id` fields, which correlate individual request-level results back to inputs inside an already-created batch, but do not prevent the batch-creation call itself from being duplicated if retried. For comparison, AWS's own broader compute service, **AWS Batch**, documents an automatic retry strategy of "1 to 10 attempts" triggered by failure conditions including "any non-zero exit code from a container job" (docs.aws.amazon.com), while explicitly excluding cancelled jobs from automatic retry: "jobs that are cancelled or terminated aren't retried" (docs.aws.amazon.com). AWS Step Functions, often used to orchestrate multi-stage batch pipelines around these APIs, documents a comparable

idempotency pattern at the workflow level: "StartExecution is idempotent for STANDARD workflows" (docs.aws.amazon.com), deduplicating identical calls by execution name, and that name becomes reusable only "90 days after it closes" (docs.aws.amazon.com).

Failure Recovery, Retries, and Idempotency Across Providers

Read together, the five providers' documentation converges on a small set of shared design patterns for handling failure in an asynchronous, high-volume system, even where the specific mechanics differ:

- **Isolate individual failures from the batch as a whole.** OpenAI writes failed requests to a separate error file rather than failing the job; Anthropic marks individual requests as errored while the rest of the batch proceeds; AWS Bedrock embeds an error object in place of a successful result for the affected line only, as documented above.
- **Do not bill for work the provider did not deliver.** OpenAI, Anthropic, and Azure each document that expired or cancelled requests are excluded from billing, or that cancellation billing is limited to completed work; Anthropic states this most explicitly of the three, repeating variations of "you will not be billed for these requests" across its errored, canceled, and expired request definitions.
- **Use a caller-supplied identifier to correlate results, never row order.** OpenAI's `custom_id` and Anthropic's `custom_id` serve the same function, matching an asynchronous result back to the request that produced it, and OpenAI explicitly warns that output ordering is not guaranteed (platform.openai.com).
- **Treat job cancellation as eventually consistent, not instantaneous.** OpenAI's cancellation can take "up to 10 minutes" to finish in-flight work (platform.openai.com), and Azure similarly allows in-flight work to complete after a cancel request is issued.
- **Deduplicate at the submission layer, not just the result layer, where the provider supports it.** AWS Bedrock's `clientRequestToken` and AWS Step Functions' name-based `StartExecution` idempotency are the clearest documented examples of this pattern among the sources reviewed; OpenAI's and Anthropic's public batch documentation does not describe an equivalent job-creation idempotency key, only per-request correlation identifiers, which is a genuine gap in what those two providers currently document rather than evidence that no such mechanism exists internally.
- **Use the monitoring mechanism each provider documents.** Anthropic's documented workflow is to "poll for the status of the batch" (docs.claude.com) and retrieve results once processing ends. For Bedrock, Amazon EventBridge can send automatic notifications when a batch inference job completes or changes state instead of polling.

For teams building a retry layer on top of any of these APIs, the practical implication is that idempotent request design has to be assembled from what each vendor actually exposes rather than assumed: a `clientRequestToken` on Bedrock genuinely prevents duplicate job creation, while a `custom_id` on OpenAI or Anthropic only prevents duplicate result attribution within a job that has already been created, so a retried batch-submission call on those two platforms can still create a second, billable batch unless the calling application enforces its own submission-level deduplication.

Comparative Context and Market Positioning

Table 1 below summarizes the documented discount, turnaround commitment, result-retention window, and per-job size limit for each provider's batch offering as of September 2026.

Provider	Documented Batch Discount	Turnaround Commitment	Result Retention	Per-Job Capacity Limit
OpenAI Batch API	50% vs. synchronous (documented above)	24-hour window; unfinished requests cancelled at expiry	Output file automatically deleted 30 days after batch completion (OpenAI Batch API)	50,000 requests or 200 MB
Anthropic Message Batches API	50% on all usage (documented above)	Most complete under 1 hour; 24-hour hard expiration	Downloadable 29 days after creation	100,000 requests or 256 MB
Google Gemini Batch API	50% of standard cost, though at least one Vertex AI line item shows 20% (ai.google.dev) (cloud.google.com)	24-hour target SLO; 48-hour hard expiration (ai.google.dev)	Not specified in reviewed documentation	Inline requests total under 20 MB; input files up to 2 GB (Google Batch API documentation)
Azure OpenAI Batch (Global Batch)	50% vs. Global Standard pricing (azure.microsoft.com)	24-hour target; jobs are **not** expired, only cancellable (learn.microsoft.com)	Configurable 14 to 30 days (learn.microsoft.com)	100,000 requests per input file (Microsoft batch limits)
AWS Bedrock Batch Inference	50% vs. on-demand (aws.amazon.com)	Configurable 24–168-hour timeout (AWS API reference)	Not specified in reviewed documentation	Not specified in reviewed documentation

The cited 50% discount is an important benchmark, but it should not be read as a guarantee that every model, modality, or line item within a given provider carries exactly that discount, as the Gemini Embedding counter-example shows. Turnaround commitments are not uniform: OpenAI and Anthropic hard-expire unfinished work at 24 hours, Google extends the hard cutoff to 48 hours, Azure does not expire jobs at all, and AWS Bedrock permits a 24–168-hour timeout setting (AWS API reference). For a workload where completeness matters more than a fixed delivery time (for example, a one-time document backlog that must fully complete no matter how long it takes), Azure's model removes the expiration risk that OpenAI's and Anthropic's models carry; for a workload with a hard downstream deadline, Anthropic may be attractive when its typical sub-hour completion is suitable, but this is an operational tendency rather than a guaranteed or empirically comparative turnaround time.

IntuitionLabs, a life-sciences and AI consultancy, is not itself a batch AI infrastructure provider and does not appear as an option in the table above; it is an implementation and advisory partner that helps pharmaceutical and life-sciences organizations, among others, design document-processing pipelines that sit on top of providers like the ones compared here. Founded in 2023 by Adrien Laurent and based in San Jose, California (intuitionlabs.ai), the firm's relevant perspective is that regulated life-sciences workloads, such as batch summarization of regulatory

submissions or clinical documentation, often value the predictability of a hard, well-documented expiration window over raw completion speed, because a job that silently drops incomplete work at hour 24 has different validation and audit implications than one that simply keeps running until cancelled.

Data Analysis and Evidence

Method. To make the cost comparison reproducible, this report defines a fixed monthly document-processing workload: 100 documents per month, each requiring 60,000 input tokens and producing a 6,000-token output, a volume IntuitionLabs previously used to illustrate enterprise document summarization costs (intuitionlabs.ai). That yields a total monthly volume of 6,000,000 input tokens and 600,000 output tokens. For reference, OpenAI's own token-counting guidance estimates that "1 token is approximately 4 characters" and "100 tokens are approximately 75 words" (help.openai.com), and Anthropic's document-processing guidance states that a scanned or text page "typically uses 1,500 to 3,000 tokens per page depending on content density" (docs.anthropic.com), placing a 60,000-input-token document in the range of a 20 to 40 page filing. This report applies each provider's own published per-token batch rate to that fixed workload and separately derives the equivalent synchronous cost by doubling the batch rate, consistent with each provider's own documented "50% discount" policy language rather than an independently estimated multiplier.

Table 2 below shows the resulting monthly cost for six representative models.

Model	Batch Input Rate (per 1M tokens)	Batch Output Rate (per 1M tokens)	Monthly Batch Cost	Derived Synchronous Cost
OpenAI GPT-4o mini	$0.075	$0.30	$0.63	$1.26
Anthropic Claude Haiku 4.5	$0.50	$2.50	$4.50	$9.00
OpenAI GPT-5	$0.625	$5.00	$6.75	$13.50
Anthropic Claude Sonnet 5	$1.00	$5.00	$9.00	$18.00
OpenAI GPT-4o	$1.25	$5.00	$10.50	$21.00
Anthropic Claude Opus 5	$2.50	$12.50	$22.50	$45.00

Rates for OpenAI models are drawn from OpenAI's published pricing table cited above, and rates for Anthropic models from Anthropic's published pricing table cited above. At this workload size, the absolute dollar gap between the cheapest and highest-priced model in this sample is roughly $22 per month in batch mode, a difference that scales linearly with volume; an organization processing 100,000 documents a month rather than 100 would see the same ratios apply to figures roughly three orders of magnitude larger. No vendor in this report publishes an empirical, size-indexed completion-time distribution for a workload of this specific shape, so the provider-specific timeouts, targets, and qualifiers in Table 1 are the only documented commitments available, including AWS Bedrock's configurable 24–168-hour timeout (AWS API reference); this report does not claim a measured completion time for the specific 6-million-token workload above, only the vendors' general documented turnaround behavior.

Stacking further discounts on top of the batch rate is possible through prompt caching, a separate mechanism that discounts repeated context rather than job type. Anthropic states that "using cached content is significantly cheaper, costing only 10% of the base input token price," while "writing to the cache costs 25% more than our base input token price" (claude.com), with overall savings of "up to 90%" on cost and "up to 85%" on latency for long prompts (claude.com). OpenAI documents a comparable effect, letting a caller "pay the model's reduced cached-input rate for reused tokens, discounted up to 90%" (developers.openai.com). As noted above, Anthropic's own batch documentation flags that cache hit rates inside asynchronous batch jobs are best-effort, ranging from 30% to 98% depending on traffic patterns, rather than guaranteed, which means the two discounts (batch and cache) are additive in principle but not fully predictable in combination.

The underlying reason providers can sustain a 50% batch discount is rooted in inference-serving techniques that trade latency for throughput, a body of work with an independent academic record. The PagedAttention paper underlying the widely used vLLM serving engine reports a 2 to 4 times throughput improvement "with the same level of latency compared to the state-of-the-art systems, such as FasterTransformer and Orca" (arxiv.org), and the vLLM project's own published benchmark states "vLLM achieves 8.5x - 15x higher throughput than HF" (Hugging Face Transformers) "and 3.3x - 3.5x higher throughput than TGI" (Text Generation Inference) (vllm.ai) (vllm.ai). Quantization delivers a separate, compounding gain: the GPTQ paper reports end-to-end generation speedups of "3.25x when using high-end GPUs (NVIDIA A100) and 4.5x when using more cost-effective ones (NVIDIA A6000)" (arxiv.org), and the AWQ paper's TinyChat inference framework "offers more than 3x speedup over the Huggingface FP16 implementation on both desktop and mobile GPUs" (arxiv.org). NVIDIA's own technical documentation on serving optimization states that "in-flight batching can therefore greatly increase the overall GPU utilization in real-world use cases" (developer.nvidia.com), the same continuous-batching principle that lets providers pack many customers' asynchronous jobs onto shared graphics processing unit (GPU) capacity at lower marginal cost than reserving capacity for instant, unpredictable synchronous traffic. Independent research firm Epoch AI has also tracked the broader inference-price trend this technology stack enables, finding that "the rate of decline varies dramatically depending on the performance milestone, ranging from 9x to 900x per year" in the price needed to reach a fixed capability level (epoch.ai), context for why both the synchronous baseline and the batch discount on top of it have been falling in absolute terms even as the relative 50% batch discount itself has stayed roughly constant across providers.

Implications and Future Directions

The clearest signal from this research is that turnaround is not a single, comparable commitment across providers: OpenAI and Anthropic use a hard 24-hour expiration, Azure has a soft target with no expiration, Google has a soft target with a longer 48-hour hard expiration, and AWS Bedrock permits a configurable 24–168-hour timeout (AWS API reference). Organizations selecting a batch provider for a deadline-sensitive pipeline, such as a nightly document-ingestion job that must be complete before a downstream process starts, should treat this as a primary selection criterion rather than an implementation detail, since the cost of a missed deadline (having to detect an expired job, identify the unprocessed subset, and resubmit it) is an operational cost the pricing comparison alone does not capture.

Formal service-level commitments are also unevenly documented. Google Cloud publishes an explicit Vertex AI service level agreement (SLA) covering "Training, Deployment, and Batch Prediction" workloads with a stated monthly uptime commitment (cloud.google.com), and AWS publishes a Bedrock SLA with tiered service credits for

uptime shortfalls, stating that "AWS will use commercially reasonable efforts to make Amazon Bedrock available with the Monthly Uptime Percentages set forth in the table below" (aws.amazon.com). By contrast, this research did not locate an equivalent, batch-specific, contractual SLA page for OpenAI's or Anthropic's self-serve API tiers; both providers describe their 24-hour and sub-hour figures in operational, documentation-level language ("completes within," "most batches completing within") rather than in a dedicated SLA document, which is a meaningful distinction for procurement teams that require a contractual uptime commitment rather than a stated target. This is a documented gap in what was publicly locatable during this research, not a claim that no such commitment exists in enterprise agreements outside the public documentation reviewed here.

For organizations planning to reduce inference cost at scale generally, the evidence in this report suggests the batch discount, prompt caching, and lower-level serving optimizations (continuous batching, quantization) are complementary rather than substitutes: a 50% batch discount and a 90% cache discount apply to different parts of a token bill (job type versus repeated context) and can be combined, while quantization and continuous-batching gains documented in the academic literature accrue to the provider's infrastructure economics and show up to the customer only indirectly, through the stability of the 50% batch price over time rather than as a separately billed line item. For a life-sciences organization such as those IntuitionLabs advises, where document sets (protocols, regulatory submissions, adverse-event reports) are processed in predictable monthly or quarterly batches rather than in real time, the batch tier's discount and its explicit failure semantics (which requests were billed, which were not, and why) are likely to matter more for cost governance and audit purposes than shaving the last few hours off an already-overnight turnaround window.

Frequently Asked Questions (FAQs)

How much does batch AI processing cost compared to real-time API calls? The cited Gemini API, OpenAI, Anthropic, Azure OpenAI Global Batch, and qualifying AWS Bedrock offerings advertise a 50% discount relative to their cited standard or online rates (platform.openai.com). Buyers should verify the model, modality, service, and region: Google lists the separate Gemini Embedding batch input price at $0.00012 per 1,000 input tokens versus $0.00015 online, a 20% reduction (cloud.google.com).

How long does batch inference actually take? The providers document different time commitments: OpenAI and Anthropic use 24-hour completion or expiration windows; Google has a 24-hour target and a 48-hour expiration; Azure has a 24-hour target without expiration; and AWS permits a 24–168-hour timeout. OpenAI states jobs complete "within 24 hours (and often more quickly)" with a hard cutoff (platform.openai.com), Anthropic documents that most batches finish within an hour, Google extends its hard expiration to 48 hours (ai.google.dev), Azure does not expire jobs at all past 24 hours, only allowing cancellation (learn.microsoft.com), and AWS Bedrock exposes a timeout setting with a valid range of 24 to 168 hours (AWS API reference).

What happens to failed requests inside a batch job? In every provider reviewed, an individual request failure does not fail the whole job. OpenAI and Anthropic isolate failures into a per-request error state or error file, while AWS Bedrock replaces the successful output field with an error object for the affected record only (docs.aws.amazon.com), and none of the reviewed documentation describes charging customers for requests that errored, were cancelled, or expired without being sent to the model.

Is there a service level agreement (SLA) for batch AI processing? Google Cloud's Vertex AI publishes an explicit SLA covering batch prediction workloads (cloud.google.com), and AWS publishes a Bedrock SLA with tiered service credits (aws.amazon.com). This research did not locate an equivalent standalone, batch-specific SLA page for OpenAI's or Anthropic's self-serve API tiers; their public documentation frames turnaround in operational rather than contractual terms.

How can LLM inference costs be reduced at scale beyond using batch processing? The two most-documented complementary levers are prompt caching, which Anthropic and OpenAI each document can cut input-token costs by up to 90% for repeated context (claude.com) (developers.openai.com), and choosing a smaller or quantized model where quality permits, since independent benchmarks document quantization speedups of roughly 3x to 4.5x on comparable hardware (arxiv.org).

What is idempotency in async batch inference, and how is it implemented? Idempotency means a retried submission does not create duplicate, separately billed work. AWS Bedrock is the clearest example among the providers reviewed, exposing a `clientRequestToken` explicitly "to ensure the API request completes only once" (docs.aws.amazon.com); OpenAI's and Anthropic's `custom_id` fields serve a related but narrower purpose, correlating results to requests inside a batch that has already been created, rather than preventing the batch-creation call itself from being duplicated.

Conclusion

The cited OpenAI, Anthropic, Gemini API, Azure OpenAI Global Batch, and qualifying AWS Bedrock offerings advertise a 50% reduction relative to their cited synchronous prices, but this is not a universal rate: Google lists Gemini Embedding batch input at $0.00012 per 1,000 input tokens versus $0.00015 online, a 20% reduction. AWS documents a `timeoutDurationInHours` setting with a valid range of 24 to 168 hours for Bedrock batch jobs (AWS API reference). What has not converged is what happens at the edges of that target: whether unfinished work is billed, cancelled, or simply left running; how long results remain retrievable afterward; and whether the platform gives developers a genuine idempotency key or only a result-correlation identifier. Using a reproducible 6.6-million-token monthly document-processing workload, this report calculated batch costs from $0.63 to $22.50 depending on model choice, each roughly half its provider's own synchronous rate, and documented that prompt caching, quantization, and continuous-batching serving techniques are the underlying, independently benchmarked mechanisms that make that discount economically sustainable for providers to offer. For a team choosing among these five options, buyers should verify the applicable model, modality, service, and region rather than assume a uniform discount; the turnaround, expiration, and failure-recovery behavior documented here is where a real, provider-specific decision still has to be made, and it should be made against the provider documentation that distinguishes each commitment, including AWS Bedrock's configurable 24–168-hour timeout (AWS API reference).
