Starting April 29, 2025, Gemini 1.5 Pro and Gemini 1.5 Flash models are not available in projects that have no prior usage of these models, including new projects. For details, see Model versions and lifecycle (/vertex-ai/generative-ai/docs/learn/model-versions#legacy-stable).

Context caching overview

Release Notes

To see an example of context caching, run the "Intro to context caching" notebook in one of the following environments:

Open in Colab (https://colab.research.google.com/github/GoogleCloudPlatform/generative-ai/blob/main/gemini/context-caching/intro_context_caching.ipynb)

|

Open in Colab Enterprise (https://console.cloud.google.com/vertex-ai/colab/import/https%3A%2F%2Fraw.githubusercontent.com%2FGoogleCloudPlatform%2Fgenerative-ai%2Fmain%2Fgemini%2Fcontext-caching%2Fintro_context_caching.ipynb)

|

Open in Vertex AI Workbench (https://console.cloud.google.com/vertex-ai/workbench/deploy-notebook?download_url=https%3A%2F%2Fraw.githubusercontent.com%2FGoogleCloudPlatform%2Fgenerative-ai%2Fmain%2Fgemini%2Fcontext-caching%2Fintro_context_caching.ipynb)

|

View on GitHub (https://github.com/GoogleCloudPlatform/generative-ai/blob/main/gemini/context-caching/intro_context_caching.ipynb)

Context caching aims to reduce the cost and the latency of requests to Gemini that contain repeated content.

By default, Google automatically caches inputs for all Gemini models to reduce latency and accelerate responses for subsequent prompts.

For Gemini 2.5 Flash (minimum input token count of 1,024) and Gemini 2.5 Pro (minimum input token count of 2,048) models, the cached input tokens are charged at a 75% discount (https://cloud.google.com/vertex-ai/generative-ai/pricing) relative to standard input tokens when a cache hit occurs.

View cache hit token information in the responses metadata field. To disable this, refer to Generative AI and data governance (/vertex-ai/generative-ai/docs/data-governance#customer_data_retention_and_achieving_zero_data_retention).

Through the Vertex AI API, you can create context caches (/vertex-ai/generative-ai/docs/context-cache/context-cache-create) and exercise more control over them by:

- Using a context cache (/vertex-ai/generative-ai/docs/context-cache/context-cache-use) by referencing its contents in a prompt request with its resource name.
- Updating the expiration time (TTL) of a context cache (/vertex-ai/generative-ai/docs/context-cache/context-cache-update) beyond the default 60 minutes.
- Deleting a context cache (/vertex-ai/generative-ai/docs/context-cache/context-cache-delete) when you no longer need it.

You can also use the Vertex AI API to get information about a context cache (/vertex-ai/generative-ai/docs/context-cache/context-cache-getinfo).

Important: Context caching using the Vertex AI API is only supported when you use regional endpoints (https://cloud.google.com/about/locations).

Note that caching requests using the Vertex AI API charges input tokens at the same 75% discount relative to standard input tokens and provides assured cost savings. There is also a storage charge based on the amount of time data is stored.

When to use context caching

Context caching is particularly well suited to scenarios where a substantial initial context is referenced repeatedly by subsequent requests.

Cached context items, such as a large amount of text, an audio file, or a video file, can be used in prompt requests to the Gemini API to generate output. Requests that use the same cache in the prompt also include text unique to each prompt. For example, each prompt request that composes a

chat conversation might include the same context cache that references a video along with unique text that comprises each turn in the chat.

Consider using context caching for use cases such as:

- Chatbots with extensive system instructions
- Repetitive analysis of lengthy video files
- Recurring queries against large document sets
- Frequent code repository analysis or bug fixing

Important: Using the same context cache and prompt doesn't guarantee consistent model responses because the responses from LLMs are nondeterministic. A context cache doesn't cache any output.

Cost-efficiency through caching

Context caching is a paid feature designed to reduce overall operational costs. Billing is based on the following factors:

- **Cache token count:** The number of input tokens cached, billed at a reduced rate when included in subsequent prompts.
- **Storage duration:** The amount of time cached tokens are stored, billed hourly. The cached tokens are deleted when a context cache expires.
- **Other factors:** Other charges apply, such as for non-cached input tokens and output tokens.

Note: When you create the cached contents, the first call allocates the cache storage. For this initial call, you are charged at the normal rate based on the number of input tokens. Subsequent calls that refer to the cached contents are charged at the reduced rate. For pricing details, see Gemini and context caching on the Gemini pricing page (/vertex-ai/generative-ai/pricing).

The number of tokens in the cached part of your input can be found in the metadata field of your response, under the `cachedContentTokenCount` (/vertex-ai/docs/reference/rest/v1/GenerateContentResponse#UsageMetadata) field.

Context caching support for Provisioned Throughput is in Preview (/products#product-launch-stages) for default caching. Context caching using the Vertex AI API is not supported for Provisioned Throughput. Refer to the Provisioned Throughput guide

(/vertex-ai/generative-ai/docs/provisioned-throughput/measure-provisioned-throughput#context_caching) for more details.

Note: Context caching is now supported by both base and fine-tuned Gemini models(see Context cache for fine-tuned Gemini models (/vertex-ai/generative-ai/docs/context-cache/context-cache-for-tuned-gemini)).

Supported models

The following Gemini models support context caching:

- Gemini 2.5 Flash-Lite (/vertex-ai/generative-ai/docs/models/gemini/2-5-flash-lite)
- Gemini 2.5 Pro (/vertex-ai/generative-ai/docs/models/gemini/2-5-pro)
- Gemini 2.5 Flash (/vertex-ai/generative-ai/docs/models/gemini/2-5-flash)
- Gemini 2.0 Flash (/vertex-ai/generative-ai/docs/models/gemini/2-0-flash)
- Gemini 2.0 Flash-Lite (/vertex-ai/generative-ai/docs/models/gemini/2-0-flash-lite)

For more information, see Available Gemini *stable* model versions (/vertex-ai/generative-ai/docs/learn/model-versioning#stable-versions-available). Note that context caching supports all MIME types for supported models.

Availability

Context caching is available in regions where Generative AI on Vertex AI is available. For more information, see Generative AI on Vertex AI locations (/vertex-ai/generative-ai/docs/learn/locations).

VPC Service Controls support

Context caching supports VPC Service Controls, meaning your cache cannot be exfiltrated beyond your service perimeter. If you use Cloud Storage to build your cache, include your bucket in your service perimeter as well to protect your cache content.

For more information, see VPC Service Controls with Vertex AI (/vertex-ai/docs/general/vpc-service-controls) in the Vertex AI documentation.

What's next

- Learn about the Gemini API (/vertex-ai/generative-ai/docs/overview).
- Learn how to use multimodal prompts (/vertex-ai/generative-ai/docs/multimodal/send-multimodal-prompts).

Last updated 2025-07-26 UTC.