**60-second takeaway**
SIE successfully switched between a small embedding model and a reranker on one RTX 3090 Ti. A 15-second idle policy unloaded each model, and stopping the server returned GPU memory to the 47 MiB host baseline.
This proves the tested on-demand lifecycle. It does not prove SIE's pressure-driven LRU behavior, production throughput, or large-model safety.

# The practical problem

A remote embedding service can save memory on a laptop, but moving the model to a GPU workstation creates another problem: does the model occupy GPU memory all day, even when nobody is searching?

An on-demand server should do four things well:

1. load the requested model when work arrives;
2. tell the client when that model is still loading;
3. switch between model types without restarting the service; and
4. release most model memory after the model becomes idle.

We tested those behaviors in SIE with one embedding model and one reranking model. The goal was not to reproduce a broad benchmark. It was to find out whether SIE could support a practical remote retrieval service on a shared 24 GB GPU.

# Test setup

The server ran on an RTX 3090 Ti and listened only on `127.0.0.1`. Telemetry was disabled, and the idle eviction threshold was set to 15 seconds.

The two models were:

- `sentence-transformers/all-MiniLM-L6-v2` for embeddings;
- `cross-encoder/ms-marco-MiniLM-L-6-v2` for reranking.

The repository, Python environment, package cache, and model cache lived on an external hard drive. That protected the workstation's nearly full NVMe, but it

also made cold installation and model loading slower.

## What happened on CUDA

| Stage | Observed GPU memory | What it establishes |
|---|---|---|
| Host before server | 47 MiB | Idle host baseline |
| Server before a model | 390 MiB | CUDA runtime and server overhead |
| Embedding model active | 510 MiB | Embedding request completed on GPU |
| After embedding eviction | 426 MiB | Most model-attributable memory released |
| Reranker active | 466 MiB | Model switch and reranking completed |
| After reranker eviction | 432 MiB | Most reranker memory released |
| After server shutdown | 47 MiB | Full return to the observed host baseline |

The remaining 36 to 42 MiB above the server's pre-model level after eviction is consistent with CUDA runtime or allocator residue. It is not evidence that the whole model remained resident. The stronger check was server shutdown, which returned the card to the same 47 MiB baseline measured before the service.

## Cold loading changes the client contract

The first embedding request took 36.77 seconds from the external cached model. The reranker initially returned `503 MODEL_LOADING`, became ready after about 9.03 seconds, and then served the request.

That `503` is not merely an inconvenience. It defines part of the API contract. A real client needs:

- a bounded retry policy;
- backoff or a readiness endpoint;
- a timeout longer than the expected cold load;

- a clear distinction between loading, failed loading, and unavailable; and
- protection against many clients triggering the same load simultaneously.

A server that eventually loads a model is not enough. The caller must know what to do while the load is in progress.

## CPU behavior provided a useful control

The CPU run showed the same lifecycle without GPU-specific memory behavior. The embedding model took 107.12 seconds for its initial download and request, a warm request took 21 milliseconds, and reload from the external cache after eviction took 4.80 seconds. The reranker loaded in the background in about 21 seconds and then answered in 11 milliseconds.

These cold numbers combine download, external-drive reads, deserialization, and warmup. They should not be presented as steady-state model speed. They do show why storage placement and warm-request latency need separate measurements.

## What we did not prove

SIE also supports eviction when GPU memory pressure crosses a threshold. That path was not triggered in this experiment.

SIE enforces a minimum pressure threshold of 50 percent. The two small models could not reach that threshold safely, and we did not create artificial pressure with unrelated GPU allocations. Therefore, this experiment does not establish:

- which model pressure-driven LRU evicts first;
- whether an active request is protected during eviction;
- how the server recovers from an out-of-memory load;
- behavior with concurrent requests;
- sustained throughput;
- repeated load and eviction stability; or
- lifecycle behavior for models that occupy several gigabytes each.

Idle eviction and pressure-driven LRU solve related but different problems. Passing one does not prove the other.

## Is SIE useful for a remote embedding service?

For a low-duty service where searches arrive in bursts, the tested result is promising. A model can load when needed, serve a request, and release most of its GPU memory after a short idle period. The workstation can then return to other GPU work without keeping the embedding weights resident permanently.

The tradeoff is cold-start latency. This design fits workflows that tolerate a wait on the first request better than interactive search where every query must return immediately. A practical deployment can choose among three policies:

| Policy | Memory use | First-request latency | Best fit |
|---|---|---|---|
| Always resident | Highest | Lowest | Frequent interactive search |
| Idle eviction | Low between bursts | Cold start after idle | Occasional research and batch work |
| Scheduled warm window | Moderate | Low during known periods | Predictable work sessions |

The right policy depends on workload frequency, not model size alone.

## A production-minded architecture

For a private workstation, keep the model server bound to loopback and expose it through an authenticated private path such as an SSH port forward or a supervised Tailscale service. Do not expose an unauthenticated embedding endpoint directly to the public internet.

The client should keep documents local and send only the text chunks needed for embedding. It should also preserve the model identifier and embedding dimension with every vector index. Switching models without rebuilding or separating the index can silently mix incompatible vector spaces.

This lifecycle layer complements model selection. In our separate Nanbeige4.2-3B versus Qwen3-VL-4B benchmark, both models fit on the same card, but their task results differed. SIE answers a different question: how can several useful models share the workstation without remaining loaded forever?

## The next decisive experiment

The next run should use two approved models large enough to cross the 50 percent pressure threshold naturally. It should preregister which model is expected to be evicted and test:

1. one active and one idle model under pressure;
2. simultaneous requests during a cold load;
3. repeated load, serve, evict cycles;
4. a failed or out-of-memory load followed by recovery;
5. readiness and retry behavior from the real client; and
6. complete VRAM attribution before and after shutdown.

Until that run exists, the bounded conclusion is simple: SIE's model switching and idle eviction worked for two small models on this RTX 3090 Ti. Pressure-driven LRU and production reliability remain open questions.