



SGLang 0.5.20 Upgrade Guide: CUDA, APIs and Rollback

Plan and build private AI compute with GPU Smith. We connect workload requirements with GPU hardware, networking, deployment and operating decisions so your infrastructure fits the work it must perform.

Published by GPU Smith · 2026-10-17 · sglang 0.5.20 · sglang upgrade · cuda 13 · cuda 12 migration · responses api · rocm · inference infrastructure · rollback validation · canary deployment

Original article: <https://gpusmith.com/articles/en/sglang-0-5-20-upgrade-guide>



In this article

1. Executive Summary
 2. Introduction and Background
 3. Key Changes
 4. Implementation Considerations and Process Changes
 5. Compatibility Tests and Canary Design
 6. Rollback Procedure and State Continuity
 7. Data Analysis and Evidence
 8. Implications and Future Directions
 9. Conclusion
-

Executive Summary

SGLang 0.5.20 requires a staged migration for fleets that depend on CUDA 12 artifacts, retained Responses API state, legacy configuration, or removed kernel paths. The release shipped on **September 18, 2026**; its published wheels cover multiple interpreter and processor combinations. It is a specific migration target, rather than the newest available package in the release history checked for this report. (pypi.org) The practical decision is **go to canary**, **pin the existing deployment**, or **no-go for the proposed configuration**, with artifact, application, and rollback evidence required separately. (docs.docker.com)

The decisive infrastructure change is the end of published CUDA 12 wheels and images: **0.5.19 is the final CUDA 12 lane**, while already published artifacts remain available. (github.com) NVIDIA's generic compatibility table gives CUDA 13.x a **580-or-newer driver branch** floor, but that floor does not certify every architecture or just-in-time kernel. (docs.nvidia.com) The installation guide is now rolling documentation for a newer stack; use the target release's dependency file rather than assuming its current examples describe this version. (docs.sglang.io) For AMD and Intel fleets, select platform-specific artifacts and validate their exact framework and kernel combination. (rocm.docs.amd.com) (docs.pytorch.org)

The application change is equally material. Responses storage is disabled unless explicitly enabled, and the enabled store remains process-local and in memory. A successful foreground response, even one echoing `store=true`, is insufficient evidence of retention. (github.com) Under prefill/decode disaggregation, stateful Responses operations are unsupported; clients should supply explicit conversation history. (docs.sglang.io) Test the application's actual chaining, retrieval, streaming, cancellation, and background behavior before moving traffic. OpenAI's current background contract also differs from SGLang's requirement concerning `store=false`, so compatibility must be checked feature by feature. (developers.openai.com)

Approve expansion only after the candidate passes a workload-specific canary and a rehearsal that returns traffic to the saved baseline. Pin container digests and model revisions, preserve the old package inventory, and keep separate compiler/cache directories as an operator precaution. Docker digests identify immutable image content. (docs.docker.com) Kubernetes rollback restores the Pod template rather than application state. (kubernetes.io) Compare time to first token, inter-token latency, goodput, memory, and application errors using paired workloads and a predeclared practical-change threshold. (www.itl.nist.gov) A faster benchmark cannot compensate for lost conversation continuity or an unavailable recovery artifact. The paired comparison must preserve corresponding workload conditions. (www.itl.nist.gov)

Introduction and Background

This **SGLang 0.5.20 upgrade guide** addresses operators moving from an earlier release, particularly fleets whose images, launch manifests, adapters, and application assumptions have accumulated over time. The release combines **713 pull requests** from **237 contributors**. (github.com) That breadth makes a feature-by-feature migration inventory more useful than treating a package version change as a complete deployment plan.

The report takes the infrastructure engineering perspective appropriate to GPU Smith's independent advisory role: select a bounded target, expose compatibility assumptions, and require reproducible evidence before expanding traffic. GPU Smith is the publishing perspective here, rather than an inference runtime competing

with SGLang. The recommendations below are proposed operating procedures, not claims about customer deployments or measurements performed for this article.

Three evidence categories remain separate throughout. **Release-specific facts** come from tagged metadata, official change descriptions, and package records. **Rolling documentation** explains current behavior but can move beyond the target. **Acceptance criteria** are operator decisions that must be filled with local measurements. The distinction matters because the current installation guide specifies a newer Python baseline, while the target's published distributions include older interpreter builds. (docs.sglang.io) (pypi.org)

An inference upgrade also includes more than the server package. Model and tokenizer revisions, platform-specific binaries, drivers, runtime libraries, launch arguments, routing, and application state together define the service. Wheel compatibility tags encode interpreter, application binary interface (ABI), and platform requirements; they do not certify the entire GPU execution path. (packaging.python.org) A model download can likewise follow a moving branch unless its revision is pinned. (huggingface.co)

The bounded question is therefore whether an identified candidate can preserve the required application contract on a particular fleet and return to an identified baseline. A reproducible image reference and a captured environment make that question testable. (docs.docker.com) (pip.pypa.io) (docs.docker.com) The following matrices translate the upstream changes into those production decisions.

Key Changes

Artifact retirement and platform selection

The release removes the CUDA 12 publishing lane, rather than deleting artifacts already published for older releases. A fleet requiring those binaries should preserve its known-good baseline while preparing a separate migration environment. (github.com) *Table 1* gives a go/pin/no-go screen. Its decisions are engineering recommendations; upstream compatibility statements are cited in the evidence column.

Fleet or dependency	Evidence and boundary	Recommended decision
CUDA 12 artifact dependency	Target release stops this publishing lane; the prior release remains the last lane. (github.com)	Pin until a replacement runtime and host combination passes testing.
CUDA 13 candidate	Generic driver floor is branch 580 ; toolkit compatibility still has feature limitations. (docs.nvidia.com)	Go to canary after driver, architecture, import, and execution checks. Test just-in-time (JIT) compilation paths separately. (docs.nvidia.com)
Older NVIDIA architectures	NVIDIA lists CUDA 12.x as the final toolkit support for Maxwell, Pascal, and Volta. (docs.nvidia.com)	No-go for assuming a CUDA 13 image is interchangeable; evaluate another supported deployment path.
AMD / ROCm	The current AMD matrix lists an ecosystem combination with an earlier SGLang version, not certification of this target. (rocm.docs.amd.com)	Staged migration with an exact platform image, driver, and quantization inventory.
Intel XPU	SGLang documents a dedicated XPU dependency file and package channel. (docs.sglang.io)	Go to platform canary after selecting the matching release image and driver.

Fleet or dependency	Evidence and boundary	Recommended decision
Stateful disaggregated Responses	Official documentation excludes store-backed stateful features under prefill/decode disaggregation. (docs.sglang.io)	No-go for that design; use an explicitly validated application-state alternative.

The table deliberately separates an incompatible proposed configuration from a blanket judgment about a hardware vendor. A runtime label is only the beginning of the inventory. For example, AMD quantization paths differ from NVIDIA paths; the documented AMD AWQ implementation does not have the faster Marlin path. (docs.sglang.io) Intel also requires its GPU driver even when framework support is native. (www.intel.com)

The release makes **ROCm 10** the default AMD lane, adds MI30x/MI35x images with matching kernel wheels, retires ROCm 7.0 CI and new image/wheel publishing, and retains ROCm 7.2.x images. Tagged Intel images use `lmsysorg/sglang:vX.Y.Z-xpu`. (github.com)

Responses storage and application state

/v1/responses storage is off by default. (www.sglang.io) Standalone deployments requiring retrieval, `previous_response_id`, or background requests must opt in with `--enable-response-store`. Foreground generation and streaming can still accept the API's default `store=true` without retaining data. The returned field echoes the request rather than proving storage. With storage disabled, retrieval and cancellation return **HTTP 400** before lookup or abort, while unsupported chaining and background requests are rejected at admission. (github.com)

Enabling the store does not create a durable conversation database. Upstream describes it as **process-local, in-memory, unbounded, and without a time-to-live (TTL)**. (github.com) Treat restart, replica routing, traffic migration, and recovery as separate state-continuity questions. An application needing persistent conversation history should record its own requirements and prove its chosen state design rather than infer persistence from an identifier. OpenAI's documented conversation chaining describes the client pattern, rather than SGLang's persistence guarantees. (developers.openai.com)

Prefill/decode (PD) disaggregation separates prompt processing from token generation. Its documented Responses path supports foreground generation, but the process-local stores cannot provide coherent shared state. Store-backed features are rejected at admission with **HTTP 400**, and explicit conversation history is the documented alternative. (docs.sglang.io) Do not add the storage flag to both roles as a proposed workaround.

Removed controls, kernel paths, and configuration objects

Table 2 is an affected-feature inventory. Detection examples are suggested searches over the operator's manifests and adapters, not commands executed against a fleet for this report.

Surface and detection	Migration action	Rollback or stop trigger
CP v1: search for SGLANG_ENABLE_CP_V2, legacy CP arguments, and versioned imports.	Use the strategy-based context parallelism (CP) path and versionless APIs. Generic v1 runtime branches are removed. (www.sglang.io)	Required model or platform cannot launch through the supported path.
Sampling masks: search for SGLANG_DISAGGREGATION_SAMPLING_MASK_MAX_TOKENS.	Replace the retired setting with <code>--sampling-mask-max-tokens</code> ; upstream defaults capacity to 4096 . PD mask requests also require <code>SGLANG_ENABLE_DISAGG_SAMPLING_MASK=1</code> on both servers. (github.com)	Rollout requests exceed the intended support capacity or fail contract checks.
Attention / quantization: inspect explicit <code>cutlass_mla</code> , <code>gptq</code> , <code>AWQ</code> , <code>auto-round</code> , and <code>mixture-of-experts (MoE)</code> selections.	Removed paths include non-Marlin GPTQ, CUTLASS MLA, AWQ ahead-of-time dequantization, and Dual Chunk Flash Attention. Confirm the applicable replacement, including compatible GPTQ-Marlin auto-upgrades. (github.com)	No supported replacement preserves required precision, model support, or output quality.
Deprecated command-line interface (CLI) aliases: compare saved arguments with candidate <code>--help</code> .	Translate all removed spellings; selected mappings appear below. (github.com)	Startup rejects an argument, or resolved behavior differs from the approved configuration.
ServerArgs consumers: search for <code>dataclasses.asdict</code> and <code>dataclass</code> reflection.	Use <code>msgspec.structs.asdict</code> for raw input fields; <code>resolved_dict()</code> serves resolved values. (github.com)	Inventory, policy, or configuration exports silently lose required fields.
Global argument readers: search for <code>get_global_server_args()</code> .	It raises <code>RuntimeError</code> ; use the relevant namespace bag for effective values. <code>get_server_args()</code> represents raw input. (github.com)	Admission or scheduling policy reads the wrong effective setting.

The inventory is a starting point for a manifest diff, rather than proof that every spelling appears in a particular fleet. The removed aliases include these mappings: `--stream-output` to `--incremental-streaming-output`; the `--nsa-prefill-backend` and `--nsa-decode-backend` names to their `--dsa-` equivalents; and `--cuda-graph-bs` and `--cuda-graph-max-bs` to their `-decode` forms. (github.com)

The other translations are `--piecewise-cuda-graph-tokens` to `--cuda-graph-bs-prefill`, `--piecewise-cuda-graph-max-tokens` to `--cuda-graph-max-bs-prefill`, and `--piecewise-cuda-graph-compiler` to `--cuda-graph-tc-compiler`. The former `--enable-breakable-cuda-graph`, `--disable-piecewise-cuda-graph`, and `--enforce-piecewise-cuda-graph` switches become `--cuda-graph-backend-prefill` values `breakable`, `disabled`, and `tc_piecewise`. Rename `--mamba-scheduler-strategy` to `--mamba-radix-cache-strategy`, `--speculative-dflash-draft-window-size` to `--speculative-draft-window-size`, and `--enable-gdn-replayssm-spec` to `--enable-linear-replayssm-spec`. The removed `--collect-tokens-histogram`, `--enable-expert-distribution-metrics`, and `--prefill-round-robin-balance` have no replacements. (github.com)

Implementation Considerations and Process Changes

Additional release surfaces need explicit inventory: the CP gate disappears and CP APIs become versionless; HIP, NPU, and MUSA reject `--enable-prefill-cp` while their ports are pending. (github.com) (www.sglang.io) Decode context parallelism (DCP) communication now selects `fi_a2a` on eligible Blackwell topology, otherwise `a2a` on CUDA/ROCm; explicit `ag_rs` restores the old default. SM120 per-channel FP8 moves to per-tensor `cudnn/nvjet`. For **32 GB** GPUs, decode graph caps change from **24 to 48**, or **160** at tensor parallelism **4 or greater**, and chunked prefill from **2K to 4K**. (github.com) Undersized sliding-window-attention (SWA) pools now fail startup. `GetIsReady` becomes `WatchEngineState`. Diffusion changes GPU residency, measured scaled dot-product attention (SDPA) selection and resolver defaults, retiring `allow_cudnn_sdp`; CPU images move to **Ubuntu 26.04/GCC 15.2**, and contributor formatting moves from black to ruff-format. (github.com)

Capture the target before changing production

Build a candidate from an exact version and retain the complete resulting environment. The tagged CUDA dependency file pins **PyTorch 2.13.0**, **Transformers 5.12.1**, **FlashInfer 0.6.18** with its CUDA 13 extra, **sglang-kernel 0.4.7**, and **sgl-deep-gemm 0.2.0**. (github.com) These facts should take precedence over installing whatever an unpinned command resolves on the day of deployment.

Capture the old and candidate environments with the same tooling. `pip inspect` emits JSON describing installed distributions and environment context, while `pip check` verifies declared dependency compatibility. (pip.pypa.io) Neither replaces a real model launch and execution check. Retain a wheelhouse that matches the recovery environment. (pip.pypa.io) A clean dependency graph can still select a binary combination that does not execute the required workload.

Suggested evidence capture, run separately inside each deployment image:

```
python -m pip inspect > environment.json
python -m pip freeze > requirements.snapshot.txt
python -m pip check
python -m slang.launch_server --help > launch-help.txt
```

Preserve the environment report with each deployment record. (pip.pypa.io)

The package snapshot should include transitive dependencies. For disconnected installations, retain the relevant wheels rather than expecting a version pin to recreate an unavailable download. A hash-checked requirements set should cover every dependency. (pip.pypa.io) `pip` documents both transitive pinning and wheelhouse-based installation, while warning that compiled archives may not be portable across machines. (pip.pypa.io)

Apply the same discipline to models. Pin the complete model repository commit, tokenizer and configuration files, adapters, and any required auxiliary assets. Hugging Face requires the full commit hash when using a commit revision, and `snapshot_download` can cache an entire repository at a selected revision. (huggingface.co) Rehearse startup with outbound access unavailable; `HF_HUB_OFFLINE` uses cached assets and fails when a required file is missing. (huggingface.co) (pip.pypa.io)

Inventory hardware and effective configuration

Record the host GPU architecture, driver, operating system, image platform, selected framework build, and actual loaded libraries. In a multi-platform container publication, an image index can reference distinct platform digests. Save the selected platform alongside the image reference. (docs.docker.com) For wheels, inspect interpreter, ABI, architecture, and glibc requirements rather than assuming that a package name implies compatibility. (packaging.python.org)

On AMD, do not identify the platform only through `torch.cuda.is_available()`: PyTorch intentionally reuses CUDA interfaces for HIP, AMD's runtime interface. Record `torch.version.hip` and `torch.version.cuda` as well. (docs.pytorch.org) For Intel, use the documented XPU build channel and inspect XPU device availability before the inference test. (docs.pytorch.org) (www.intel.com) Native framework support does not remove the GPU-driver prerequisite. (www.intel.com)

The configuration review should compare **raw operator input** with **resolved behavior**. Export effective settings after initialization, then review graph capture, prefill chunking, memory allocation, backend choice, parallelism, speculative decoding, and cache configuration. This is especially important when a wrapper previously serialized `ServerArgs` as a dataclass or read a global argument object. (github.com)

Use a candidate-specific cache directory and preserve the baseline's directory untouched as a conservative deployment policy. This report does not establish cross-version compiler-cache, key-value (KV) cache, or PD-transfer compatibility. A rollback plan should be able to rebuild ephemeral state rather than depend on an unverified shared cache.

Compatibility Tests and Canary Design

Test application contracts before load

Treat OpenAI-compatible as a collection of supported behaviors. OpenAI documents `previous_response_id` as a mechanism for threaded conversation context, but that does not establish equivalent state placement or persistence in another server. (developers.openai.com) Current OpenAI background documentation allows `store=false`; SGLang's change description requires `store=true` for background requests. (developers.openai.com) Test the actual client library and request combinations used by the application.

Use separate standalone-store-disabled, standalone-store-enabled, and PD test environments. The following are proposed acceptance cases; the stated capability failures come from the upstream change and PD documentation. (docs.sglang.io)

- **Foreground generation:** exercise both non-streaming and streaming requests with explicit `store=false` and `store=true`; verify a valid result and completed stream, then test retention separately.
- **Retrieval:** after generating a response, call the retrieval operation through the real gateway. A disabled store must produce the documented capability error rather than being treated as an eventually available object.

- **Chaining:** send a follow-up with `previous_response_id`; verify the required contextual answer on the enabled standalone lane and an admission rejection on the unsupported lane.
- **Background work:** test submission, polling, cancellation, and completion. Reject `background=true` with `store=false` under the target's documented SGLang contract.
- **Replica and restart boundaries:** attempt retrieval after a restart and after routing to a different replica. Record the application's intended behavior rather than assuming process-local memory follows the request.
- **Disaggregated admission:** verify retrieval, chaining, background work, and cancellation fail before generation on unsupported paths; test foreground requests with explicit history as the supported alternative. (docs.sglang.io)

Classify capability errors separately from service-availability errors. The HTTP specification describes **400** as a perceived client error and **503** as temporary overload or scheduled maintenance. (www.rfc-editor.org) Repeated retries cannot supply a capability the deployment deliberately does not support. Preserve the returned error body, endpoint, request identifier, and execution-side observations so that a gateway cannot hide a contract change behind a generic availability response.

Cover the workload dimensions that change the decision

Create independent baseline and candidate deployments and route controlled traffic between them.

Kubernetes explicitly describes multiple Deployments, one per release, as a canary pattern. (kubernetes.io) Choose expansion steps and stop conditions before viewing candidate results. Retain the online benchmark's per-run record and detailed request evidence. (docs.sglang.io)

- **Models and precision:** include each deployed model family, tokenizer revision, adapter combination, weight quantization, and KV precision; preserve a deterministic semantic test set.
- **Context distribution:** cover ordinary prompts, long-context tails, intended maximum context, and output-length limits; preserve the same prompt mix across paired runs.
- **Concurrency and arrival pattern:** test interactive traffic, sustained load, bursts, and the expected queueing limit rather than comparing unmatched throughput maxima.
- **Speculation:** test speculative decoding disabled and in its intended configuration; verify completion quality alongside accepted-token and latency measurements.
- **Prefix reuse:** distinguish cold prompts, repeated system prompts, branching conversations, and reused prefixes after cache pressure.
- **Hardware topology:** include each GPU family, parallelism layout, interconnect, and disaggregated routing arrangement that will receive traffic.

Use the online serving benchmark for scheduler-inclusive measurements. Upstream distinguishes it from `bench_one_batch`, which bypasses the scheduler, and recommends a prompt count at least **five times maximum concurrency** for steady-state testing. (docs.sglang.io) Keep load-generator settings, request

corpus, model revision, warmup, and cache state with each result. The benchmark's detailed output can retain per-request texts and errors alongside timing arrays. (docs.sglang.io)

Rollback Procedure and State Continuity

Prepare a return path before the canary

A rollback artifact is a saved executable deployment, not an intention to reinstall an old version. Pin and retain the baseline container digest, platform, package inventory, manifests, model revision, router configuration, and required assets. Docker documents pulling by digest to select an exact image version. (docs.docker.com) Preserve the same evidence for the candidate so the eventual comparison can be reproduced.

For package-based deployments, prepare a complete wheelhouse for each environment and validate installation without depending on a changing index. Hash-checking mode requires hashes for all dependencies, making it useful for an approved requirements set. (pip.pypa.io) Wheel compatibility remains environment-specific, so build and test the recovery bundle on the actual target platform. (packaging.python.org) (pip.pypa.io)

The proposed recovery sequence is:

1. **Stop expansion:** freeze the rollout and stop sending new traffic to the candidate when a predeclared contract, quality, capacity, or reliability gate fails.
2. **Drain traffic:** remove the candidate from normal routing and observe in-flight stream completion or explicit termination. Kubernetes terminating endpoints are not ready for regular traffic, but connection-draining behavior still needs validation. (kubernetes.io)
3. **Preserve evidence:** capture the failing request class, logs, resolved settings, package inventory, and measurements before replacing the candidate. Keep user data handling consistent with the deployment's own retention rules.
4. **Restore the approved deployment:** select the saved baseline manifest or an explicitly chosen deployment revision. `kubectl rollout undo` supports `--to-revision`; verify that the intended historical revision still exists. (kubernetes.io)
5. **Rebuild ephemeral state:** warm the baseline with its own caches and assets. Restore application-managed state through the application's tested mechanism, rather than expecting the deployment command to recover it.
6. **Validate re-entry:** repeat health, representative generation, streaming, state behavior, and load checks before expanding baseline traffic. Retain paired measurements for the incident-free recovery rehearsal.

Kubernetes rollback restores only the Deployment's Pod template. It does not restore an external database, a deleted model artifact, or an in-memory response store. (kubernetes.io) Keep schema and configuration snapshots for application-managed state in the recovery bundle, and test restoration only through a mechanism the application actually provides. This report establishes no SGLang response-store export/import guarantee.

Define baseline re-entry criteria

Re-entry requires more than a healthy process. The old image must start on the current host, load the pinned model and tokenizer, produce acceptable outputs, complete the required endpoints, and sustain the recovery load. NVIDIA documents backward compatibility from newer drivers to applications built with older CUDA toolkits, but this does not certify the complete old application stack. (docs.nvidia.com) Rehearse that combination before a host-driver change removes the original environment from service.

Run dependency validation again after restoring the environment and compare the captured inventory. (pip.pypa.io) Repeat the disconnected-asset test if recovery depends on an internal mirror or local cache. (huggingface.co) A successful rollback record should state which conversations completed, restarted, or required application reconstruction. That record turns state continuity into an observable acceptance criterion.

Data Analysis and Evidence

Separate upstream opportunities from fleet measurements

The project describes branching-point caching as retaining sliding-window state where requests fork from a shared prefix. (www.sglang.io) Upstream's branching-point cache work includes synthetic shared-prefix and conversational scenarios with different effects. Its reported results concern a particular patch and workload, rather than an independently reproduced comparison of the operator's old and candidate releases. The fetched methodology did not identify the GPU hardware clearly enough for this report to reproduce its performance numbers, so those numbers are deliberately omitted. (github.com) Treat radix/cache changes, graph-memory behavior, and model-specific kernel paths as separate test hypotheses.

For each metric M , calculate the analyst-defined relative change as $100 \times (M_{\text{candidate}} / M_{\text{baseline}} - 1)$. State direction: negative values favor lower-is-better latency and memory; positive values favor higher-is-better goodput. A positive throughput change is useful only if the candidate still meets its application contract and service-level objective (SLO). Research on LLM-serving evaluation likewise argues for metrics that account for request-level service objectives and token-delivery behavior. (arxiv.org)

Use paired workload blocks with matched inputs and load. NIST defines paired samples through one-to-one correspondence. (www.itl.nist.gov) For uncertainty on a relative-change statistic, a paired bootstrap can resample matched blocks together; SciPy's paired option applies the same sampled indices to input arrays. (docs.scipy.org) A **95% confidence interval** is an explicit analysis setting, consistent with SciPy's documented default, rather than an upstream deployment guarantee. (docs.scipy.org)

Declare a minimum practical improvement δ and a maximum tolerated regression r before the run. Advance for performance only when the favorable confidence bound clears δ , while every protected metric remains within r . (www.itl.nist.gov) Choose both from operating requirements, not from the observed result. NIST discusses adopting a new process only when improvement exceeds a selected threshold. (www.itl.nist.gov) Temporal correlation and workload shifts still require care; resampling cannot correct a mismatched experiment.

Table 3 is a results template, not fabricated canary output. Every blank must be filled from the same experiment definition and accompanied by the retained raw observations.

Metric / evidence	Baseline	Candidate	Delta and uncertainty	Acceptance gate
TTFT , time to first token	To measure	To measure	Relative change and interval, using matched observations. (docs.scipy.org)	Interactive and long-prompt SLOs pass.
ITL , inter-token latency	To measure	To measure	Tail distribution and interval	Streaming delivery meets the agreed limit.
Goodput	To measure	To measure	SLO-compliant completed work per unit time. (arxiv.org)	Improvement clears δ without protected regressions.
GPU memory	To measure	To measure	Peak, steady-state, allocated, and reserved. (docs.pytorch.org)	Capacity and restart headroom remain acceptable.
Application errors	To measure	To measure	Classified counts and rates	Expected capability errors separated from unexpected failures.
Contract / rollback	To verify	To verify	Recorded test outcomes	Required state behavior and baseline re-entry pass.

Interpret the table at fleet scope. Prometheus warns that precomputed summary quantiles cannot be aggregated, so averaging replica percentiles does not produce a fleet percentile. (prometheus.io) Retain compatible distributions or raw observations and account for histogram bucket error. (prometheus.io) On ROCm, distinguish allocated tensor memory from allocator-reserved memory; unused cached memory can still appear used in device tooling. (docs.pytorch.org)

Finally, confirm measurement semantics. Non-streaming chat completion makes the benchmark's TTFT equal total latency, so that output cannot substitute for a streaming first-token measurement. (docs.sglang.io) Record generation lengths and errors alongside latency; otherwise shorter, incomplete, or rejected requests could make an apparently favorable result misleading.

Implications and Future Directions

A production acceptance checklist

The decision should be documented as **go**, **pin**, or **no-go for the proposed configuration**, with an owner and retained evidence. The following is a proposed checklist for the release under consideration.

- **Target identified:** the exact image digest, selected platform, source revision, and distribution hashes are recorded.
- **Host qualified:** the actual architecture, driver, framework, runtime, and kernel combination launches and executes the required model.

- **Configuration migrated:** deprecated spellings, environment variables, backend selections, and configuration consumers have been reviewed.
- **Application verified:** streaming, retrieval, chaining, background work, cancellation, error handling, and replica routing match the intended contract.
- **State bounded:** required conversation continuity has an explicit location and tested restart behavior.
- **Quality protected:** representative outputs, tools, structured output, and precision-sensitive tasks meet the application's acceptance criteria.
- **Capacity measured:** prompt lengths, concurrency, prefix reuse, and speculation cover the traffic intended for expansion.
- **Evidence reproducible:** raw measurements, environment reports, model revisions, and experimental settings are retained.
- **Recovery rehearsed:** traffic draining and restoration of the saved baseline have passed application and load checks.
- **Expansion bounded:** traffic expansion stays within the hardware and workload combinations covered by the accepted test record.
- **Recovery ownership:** the operating team records who can stop expansion, initiate restoration, and confirm re-entry.
- **Decision bounded:** practical improvement, tolerated regression, and stop criteria were declared before reading the result.

Artifact immutability and model revision pinning make the checklist auditable rather than descriptive.

(docs.docker.com) (huggingface.co) Use it again when selecting a later release; the availability of a newer package does not establish that its migration requirements are identical.

Security status should also remain an evidence boundary. The official repository advisory view fetched for this report displayed no published advisories, but that observation is not a comprehensive applicability assessment for the target package or its dependencies. (github.com) Record the exact dependency inventory and assess it against the organization's approved advisory sources before expansion. Do not label the deployment secure on the strength of an empty repository view.

Future optimization work should be separated from the first compatibility migration. Establish the target's accepted behavior, then change graph settings, cache policy, parallelism, or kernel selection in controlled experiments. Retaining detailed benchmark output makes later experiments comparable. (docs.sglang.io) This sequence helps explain whether an improvement came from the release, a configuration change, or a different workload.

Conclusion

The production decision is a bounded compatibility decision. Pin the existing environment when a required artifact lane or application contract has no tested replacement. Advance a candidate only when its selected hardware, dependencies, effective configuration, and endpoints pass the requirements of the service that will receive traffic. Match the target wheels to the interpreter, ABI, and platform before execution testing.

(packaging.python.org)

The migration record should distinguish what upstream documents, what rolling documentation currently describes, and what the operator has actually measured. A successful foreground request does not establish retained conversation state; a package resolver does not establish kernel execution; and a deployment reversal does not reconstruct application data. Keep each proof obligation visible.

Use the artifact matrix to identify blocking dependencies, the affected-feature inventory to translate configuration, and the results template to record canary evidence. Require output quality, state behavior, capacity, and recovery alongside performance. If any required gate fails, retain the baseline and narrow the missing migration work before another attempt.

The useful outcome is a reproducible go/pin/no-go record with an executable return path. Preserve both deployments, their model assets and configuration evidence, and the observations supporting the decision. Digest references make the preserved image selectable. (docs.docker.com) That makes the next release evaluation a controlled comparison rather than another reconstruction of how the fleet was assembled.

Source links

Links cited in this article, in order of first appearance. Full addresses are provided for readers using extracted text.

1. <https://pypi.org/project/sglang/0.5.20>
2. <https://docs.docker.com/reference/cli/docker/image/pull>
3. <https://github.com/sgl-project/sglang/pull/38404>
4. <https://docs.nvidia.com/deploy/cuda-compatibility/minor-version-compatibility.html>
5. <https://docs.sglang.io/docs/get-started/install>
6. <https://rocm.docs.amd.com/en/latest/compatibility/compatibility-matrix.html>
7. https://docs.pytorch.org/docs/2.14/notes/get_start_xpu.html
8. <https://github.com/sgl-project/sglang/pull/39122>
9. https://docs.sglang.io/docs/advanced_features/pd_disaggregation
10. <https://developers.openai.com/api/docs/guides/background>
11. <https://docs.docker.com/dhi/explore/security-concepts/digests>
12. <https://kubernetes.io/docs/concepts/workloads/controllers/deployment>
13. <https://www.itl.nist.gov/div898/handbook/eda/section3/eda353.htm>
14. <https://github.com/sgl-project/sglang/releases/tag/v0.5.20>
15. <https://packaging.python.org/en/latest/specifications/platform-compatibility-tags>
16. https://huggingface.co/docs/huggingface_hub/en/guides/download

17. <https://pip.pypa.io/en/stable/reference/inspect-report>
18. <https://docs.nvidia.com/datacenter/tesla/drivers/latest/cuda-toolkit-driver-and-architecture-matrix.html>
19. <https://docs.sglang.io/docs/hardware-platforms/xpu>
20. https://docs.sglang.io/docs/hardware-platforms/amd_gpu
21. <https://www.intel.com/content/www/us/en/developer/tools/oneapi/optimization-for-pytorch.html>
22. <https://www.sglang.io>
23. <https://developers.openai.com/api/docs/guides/conversation-state>
24. <https://github.com/sgl-project/sglang/pull/36631>
25. <https://github.com/sgl-project/sglang/pull/32114>
26. <https://github.com/sgl-project/sglang/pull/38375>
27. <https://github.com/sgl-project/sglang/pull/38958>
28. <https://github.com/sgl-project/sglang/pull/36229>
29. <https://github.com/sgl-project/sglang/blob/v0.5.20/python/pyproject.toml>
30. <https://pip.pypa.io/en/stable/topics/repeatable-installs>
31. <https://pip.pypa.io/en/stable/topics/secure-installs>
32. https://huggingface.co/docs/huggingface_hub/en/package_reference/environment_variables
33. <https://docs.pytorch.org/docs/2.14/notes/hip.html>
34. <https://www.rfc-editor.org/rfc/rfc9110.html>
35. https://docs.sglang.io/docs/developer_guide/bench_serving
36. https://docs.sglang.io/docs/developer_guide/benchmark_and_profiling
37. https://pip.pypa.io/en/stable/cli/pip_check
38. <https://kubernetes.io/docs/tutorials/services/pods-and-endpoint-termination-flow>
39. https://kubernetes.io/docs/reference/kubectl/generated/kubectl_rollout/kubectl_rollout_undo
40. <https://github.com/sgl-project/sglang/pull/34565>
41. <https://arxiv.org/abs/2410.14257>
42. <https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.bootstrap.html>
43. <https://www.itl.nist.gov/div898/handbook/prc/section3/prc312.htm>
44. <https://prometheus.io/docs/practices/histograms>
45. <https://github.com/sgl-project/sglang/security/advisories>

THE PUBLISHER

About GPU Smith

GPU Smith provides independent engineering and research for private AI infrastructure. We help technical buyers and operators reason about GPU workload sizing, hardware selection, procurement, deployment and inference operations, from the compute system to the networking, power and cooling requirements around it.

Start with the workload

Useful infrastructure decisions begin with the models, data, throughput, latency and operating constraints a team actually has. GPU Smith helps connect those requirements with system design and deployment choices. The goal is a privately controlled AI environment whose capacity and operational demands are understood before a purchasing decision.

Hardware and supplier research

Our public reference library covers GPUs, complete systems and networking components. The server-vendor directory and manufacturing research help teams investigate suppliers, compare options and follow sources behind technical claims. We distinguish manufacturer specifications from measured benchmarks and advertised capabilities from tested configurations. Reference pages are research resources, not a live inventory listing or a binding equipment quote.

Deployment and operations

GPU Smith's engineering scope includes infrastructure integration, networking, power, cooling and the practical operation of inference workloads. Our published articles explain the tradeoffs behind deployment, procurement and ongoing operations so teams can ask better questions and document decisions.

Work with GPU Smith

Explore the [hardware library](#), [GPU references](#), [networking references](#) and [vendor research](#). [Contact GPU Smith](#) with your workloads and deployment constraints to discuss a project and confirm current scope, pricing and availability.

Inclusion of a manufacturer or product in our research does not imply a partnership, certification or endorsement.

Website: <https://gpusmith.com>

This article is educational. Verify consequential medical, legal, financial, regulatory and technical decisions with appropriate professionals and the cited primary sources. Company services are described separately from the article's research findings.