

JAX

Release Notes

RN-11517-001_v26.11 | November 2026

Table of Contents

Chapter 1. JAX Overview 1
Chapter 2. Pulling A Container 2
Chapter 3. Running JAX 3
Chapter 4. JAX Release 26.11 5
Chapter 5. JAX Release 26.10 9
Chapter 6. JAX Release 26.09 12
Chapter 7. JAX Release 26.08 16
Chapter 8. JAX Release 26.07 19
Chapter 9. JAX Release 26.06 22
Chapter 10. JAX Release 26.05 24
Chapter 11. JAX Release 26.04 27
Chapter 12. JAX Release 26.03 30
Chapter 13. JAX Release 26.02 32
Chapter 14. JAX Release 26.01 34
Chapter 15. JAX Release 25.12 37
Chapter 16. JAX Release 25.11 38
Chapter 17. JAX Release 25.10 39
Chapter 18. JAX Release 25.09 42
Chapter 19. JAX Release 25.08 43
Chapter 20. JAX Release 25.06 46
Chapter 21. JAX Release 25.05 47
Chapter 22. JAX Release 25.04 48
Chapter 23. JAX Release 25.03 51
Chapter 24. JAX Release 25.02 52
Chapter 25. JAX Release 25.01 53
Chapter 26. JAX Release 24.12 56
Chapter 27. JAX Release 24.11 57
Chapter 28. JAX Release 24.10 58
Chapter 29. JAX Release 24.09 62

Chapter 30. JAX Release 24.08 63
Chapter 31. JAX Release 24.07 64
Chapter 32. JAX Release 24.06 65
Chapter 33. JAX Release 24.05 66
Chapter 34. JAX Release 24.04 67
Chapter 35. JAX Release 24.03 71
Chapter 36. JAX Release 24.02 72
Chapter 37. JAX Release 23.12 73
Chapter 38. JAX Release 23.10 74
Chapter 39. JAX Release 23.08 79

Chapter 1. JAX Overview

JAX is a framework for high-performance numerical computing and machine learning research. It combines numpy-like APIs, automatic differentiation, XLA acceleration and simple primitives for scaling across GPUs.

The JAX NGC Container comes with all dependencies included, providing an easy place to start developing applications in areas such as natural language processing, computer vision, multimodality, physics-based simulations, reinforcement learning, drug discovery, material discovery and physical AI's foundational models.

The JAX NGC Container is optimized for GPU acceleration, and contains a validated set of libraries that enable and optimize GPU performance. This container may also include modifications to the JAX source code in order to maximize performance and compatibility. This container also includes software for accelerated Training (cuDNN, NCCL, TransformerEngine).

To work with deep learning networks, the JAX NGC Container includes Flax, a neural network library with support for common deep learning models, layers and optimizers. We also include an additional container for training LLMs with MaxText. You can use the JAX or MaxText containers for your deep learning workloads or install your own favorite libraries on top of them.

Starting January 2026, the JAX containers are released on a monthly basis.

Separately, the GitHub code contributions are being sent upstream regularly.

All libraries and contributions have been tested, tuned, and optimized for use on NVIDIA hardware.

Chapter 2. Pulling A Container

About this task

Before you can pull a container from the NGC container registry:

- Install Docker.
  - For NVIDIA DGX™ users, see Preparing to use NVIDIA Containers Getting Started Guide.
  - For non-DGX users, see NVIDIA® GPU Cloud™ (NGC) container registry installation documentation based on your platform.
- (Optional) Ensure that you have access and can log in to the NGC container registry.

  Refer to NGC Getting Started Guide for more information.

The deep learning frameworks, the NGC Docker containers, and the deep learning framework containers are stored in the `nvcr.io/nvidia` repository.

Chapter 3. Running JAX

Before you begin

Before you can run an NGC deep learning framework container, your Docker® environment must support NVIDIA GPUs. To run a container, issue the appropriate command as explained in Running A Container and specify the registry, repository, and tags.

About this task

On a system with GPU support for NGC containers, when you run a container, the following occurs:

- The container runtime loads the image into a container which runs the software.
- You define the runtime resources of the container by including additional flags and settings that are used with the command.

  These flags and settings are described in Running A Container.
- The GPUs are explicitly defined for the Docker container (defaults to all GPUs, but can be specified by using the `NVIDIA_VISIBLE_DEVICES` environment variable).

Procedure

1. Issue the command for the applicable release of the container that you want.

   The following command assumes you want to pull the latest JAX core container, where yy.mm is the container version (mm denotes the month, and yy the year of container release).

   ```
   docker pull nvcr.io/nvidia/jax:yy.mm-py3
   ```

   To pull the latest MaxText container:

   ```
   docker pull nvcr.io/nvidia/jax:yy.mm-maxtext-py3
   ```
2. In ther terminal, paste the above command.

   Ensure that the pull successfully completes before you proceed to step 3.
3. Run the container image:
   - Use the following commands to run the container:

- Optimized communication collective operation on sub-byte types such as int4 and fp4 eliminating redundant cast operations and reducing the volume of data moved.

```
docker run --gpus all -it --rm -v local_dir:container_dir nvcr.io/nvidia/
jax:yy.mm-py3
```

If you use multiprocessing for multi-threaded data loaders, the default shared memory segment size with which the container runs might not be enough. Therefore, you should increase the shared memory size add one of these extra parameters to the docker command line:

- **`--ipc=host`**
- **`--shm-size=<requested memory size>`**

To pull data and model descriptions from locations outside the container for use by JAX or save results to locations outside the container, mount one or more host directories as Docker® data volumes. This is done via the `-v` parameter in the example above.

Chapter 4. JAX Release 26.11

The NVIDIA JAX Release 26.11 is made up of two container images available on NGC: JAX and MaxText.

Contents of the JAX container

This container image contains the complete source for the following software:

- JAX: `/opt/jax`
- XLA: `/opt/xla`
- Flax: `/opt/flax`
- TransformerEngine: `/opt/transformer-engine`

The MaxText container image is based on the JAX container. Additionally, it includes:

- MaxText: `/opt/maxtext`

The JAX runtime package jaxlib is prebuilt and installed in the default Python environment (/usr/local/lib/python3.10/dist-packages/jaxlib) in the container image.

Versions of packages included in both of these containers:

- CUDA 13.4.1
  - Please refer to the CUDA DL 26.11 release notes section for the list of libraries inherited from the CUDA container.
- JAX v0.12.0
- MaxText 40b1fd3
- Flax 41deea3
- XLA d867969
- Transformer Engine 33f3b35

Driver Requirements

Release 26.11 is based on CUDA 13.5.0. For comprehensive and up-to-date driver compatibility information, please refer to the following documentation:

- NVIDIA CUDA Compatibility Guide - Compatibility information between CUDA versions and driver releases.
- CUDA Toolkit Release Notes - Driver version requirements and compatibility matrices.
- NVIDIA Drivers Download - Latest NVIDIA drivers.

Key Features and Enhancements

- Starting in 26.11, the DLFW containers - JAX & MaxText, introduce general availability support for the NVIDIA Rubin GPU architecture.

XLA compiler support for Vera Rubin

- XLA compute cost model supports Rubin GPU specs, such as its number of compute units, memory size and speed, and cache sizes. XLA can also now target newer GPU instruction versions (PTX 9.4) even if it was built with an older CUDA toolkit; the installed NVIDIA assembler (ptxas) still sets the upper limit. Two related fixes make cuDNN operations compile correctly on CUDA 13.4 for multiple GPU models, including B200.

Horizontal Scaling

- **Faster GPU-to-GPU communication at a larger scale.** The kernels can also run inside CUDA graphs, which cuts per-iteration graph update time from about 0.25 ms to about 0.05 ms. This is on by default and can be turned off with --xla_gpu_enable_command_buffer=-collectives_kernel. A separate fix stops the multi-GPU barrier used by ragged-all-to-all from crashing above 32 GPUs; it now supports up to 128.
- **More collective operations and smarter scheduling.** XLA adds support for collective-reduce, which combines results from all GPUs onto one chosen GPU. This is motivated by the layerwise FSDP used to optimize the Muon optimizer in Deepseek-v4. It is off by default; turn it on with --xla_gpu_emit_collective_reduce. JAX's broadcast can now choose its source GPU at run time. A scheduler bug is also fixed. Collective calls assigned to a stream were counted twice against a resource limit, which could make compilation fail when overlap limits were tight.
- **Faster transfers between CPU and GPU memory.** Copies between CPU memory and GPU memory now run on two dedicated streams instead of sharing streams with compute work. This lets transfers, such as host-memory offloading, overlap with computation and communication. The setting is on by default and affects every GPU program. The changes report no speed measurements.
- **More reliable runs with many GPUs in one process.** Several fixes make it safer to drive many GPUs from a single process. A timing problem in loop unrolling could make compilation inconsistent and hang multi-GPU runs; the decision is now made once, up front. Execution timing data was shared by all GPUs, which caused a data race; each GPU now keeps its own record. Each cuDNN graph is now built once per GPU, which prevents CUDNN_STATUS_EXECUTION_FAILED errors, and a new per-GPU state

container supports this. Operations that cannot be captured in a CUDA graph now keep their order relative to captured ones, which prevents races. TransformerEngine also sets up its softmax custom calls before first use to avoid certain deadlock scenarios.

XLA Compiler

- **Experimental new GPU code generator (TensorIR).** XLA adds an experimental code generator, called TensorIR, that writes GPU kernels using CUDA Tile IR. After a follow-up change it handles more kinds of fused operations, including FP8 and operations with several outputs. It is off by default; turn it on with --xla_gpu_experimental_enable_tensor_ir. It needs an sm_90 or newer GPU and CUDA 13.3 or newer. Because XLA's autotuner can choose it, turning it on can change which kernels run.
- **Using PGLE with NCCL symmetric memory requires NCCL 2.32.2 or newer.** Training jobs that enable both profile-guided latency estimation (PGLE) and NCCL symmetric buffers (--xla_gpu_experimental_enable_nccl_symmetric_buffers=true) could fail right after PGLE recompiles the program, with an NCCL “Allocation failed” error. The cause is that XLA registers symmetric memory windows each time a GPU program is loaded, and PGLE loads a second program in the same process. Older NCCL versions did not reuse those windows, so the repeated registrations exhausted NCCL’s symmetric allocation space. NCCL 2.32 adds window reuse, which fixes the failure. Use NCCL 2.32.2 or newer whenever PGLE and symmetric buffers are both enabled. Running either feature alone is not affected.

MoE

- **DeepSeek V3 on Blackwell and Vera Rubin GPUs.** DeepSeek V3 performance on MXFP8 has been optimized for both Blackwell and Vera Rubin GPUs in this release. To reproduce these results, please refer to the configuration settings that will be published in JAX-Toolbox.
- **Mixture-of-experts training with TransformerEngine on GPU.** These changes help train mixture-of-experts (MoE) models, such as DeepSeek-style models, on GPU with TransformerEngine. TransformerEngine can now spread experts across several GPUs within one process, reusing XLA's NCCL communicator. It also names MoE checkpoints so they can be saved and recomputed selectively. MaxText builds on this to cut communication during training and to fix a sizing bug that could set expert capacity too low when gradients are accumulated.
- **Faster mixture-of-experts communication and math.** MoE models send tokens between GPUs, and these changes make that exchange faster. The ragged all-to-all kernel now spreads network transfers over all available connections, and it sends data between machines before copying within a machine, so the two overlap. cuDNN can also compute the weight-gradient step of MoE training (a grouped matrix multiply). That needs cuDNN 9.24 or newer and an experimental flag.

Attention

- **Attention enchancements.** JAX SDPA now supports flex attention, which lets users customize how attention scores are computed. TransformerEngine's fused attention now handles gradient sharding consistently in the backward pass, with more tests. In MaxText, cuDNN flash attention works during autoregressive inference when the key-value cache and the query have different batch sizes.

Profiling and Observability

- **More accurate profiling.** Profiling results are more trustworthy. For operations made of several GPU kernels, cost is now measured from the first kernel's start to the last kernel's end, instead of an average that under-reported it. XLA's automatic profile-guided scheduling (AutoPGLE) now works while running under Nsight Systems, and fixed offsets mean XLA metadata such as module name and source locations shows correctly there. Mosaic GPU supports CUPTI V2, so several profiling sessions can run at once. A tool that builds matmul performance tables now handles matmuls that launch several kernels; it does not change compiled programs.

JAX Toolbox

The JAX Toolbox projects focuses on achieving the best performance and convergence on NVIDIA Ampere, Hopper, and Blackwell, and Rubin architectures families and provides the latest deep learning models and scripts and container images for training and fine-tuning the latest open source models. These examples are tested against a nightly CI as well as each NGC container release to ensure consistent accuracy and performance over time.

Nightly Containers

In addition to projects, JAX Toolbox includes nightly containers for libraries across the JAX ecosystem.

Container	Type	Image URI
jax	-	ghcr.io/nvidia/jax:jax-YYYY-MM-DD
maxtext	LLM framework	ghcr.io/nvidia/jax:maxtext-YYYY-MM-DD
axlearn	LLM framework	ghcr.io/nvidia/jax:axlearn-YYYY-MM-DD

Known Issues

- None.

Chapter 5. JAX Release 26.10

The NVIDIA JAX Release 26.10 is made up of two container images available on NGC: JAX and MaxText.

Contents of the JAX container

This container image contains the complete source for the following software:

- JAX: `/opt/jax`
- XLA: `/opt/xla`
- Flax: `/opt/flax`
- TransformerEngine: `/opt/transformer-engine`

The MaxText container image is based on the JAX container. Additionally, it includes:

- MaxText: `/opt/maxtext`

The JAX runtime package jaxlib is prebuilt and installed in the default Python environment (/usr/local/lib/python3.10/dist-packages/jaxlib) in the container image.

Versions of packages included in both of these containers:

- CUDA 13.4.1
  - Please refer to the CUDA DL 26.10 release notes section for the list of libraries inherited from the CUDA container.
- JAX jax-v0.11.2
- MaxText f178b07
- Flax 4f42ed0
- XLA 91888df
- Transformer Engine v2.20.2 (6ea2a74a)

Driver Requirements

Release 26.10 is based on CUDA 13.4.3. For comprehensive and up-to-date driver compatibility information, please refer to the following documentation:

- NVIDIA CUDA Compatibility Guide - Compatibility information between CUDA versions and driver releases.
- CUDA Toolkit Release Notes - Driver version requirements and compatibility matrices.
- NVIDIA Drivers Download - Latest NVIDIA drivers.

Key Features and Enhancements

- Updated cuDNN FrontEnd library to version 1.26.
- Shipped JAX FFI extension headers (collectives, record) in the JAX wheel.

MoE

- Added sqrtsoftplus scoring to fused MoE routing in JAX. (TransformerEngine).
- Enabled Expert Parallelism to borrow XLA's NCCL communicator when supported (TransformerEngine).
- Supported single-process multi-device execution in MoE block (TransformerEngine).

Attention

- Fixed JAX fused-attention backward output-gradient sharding, rotated THD metadata handling, and compound DP/FSDP test-data placement (TransformerEngine).

Precision

- Added Hopper BF16 grouped GEMM support in JAX (TransformerEngine).
- Fixed FP8 custom-gradient state overwrite in MaxText NNX training.

Collective and Communication

- Added a new CollectiveReduce HLO operation for GPU collectives.

Compiler and Runtime

- JAX upgraded to v0.11.1 (from v0.10.2) with CUDA 13.4 support.
- Fixed quadratic tracing behaviour in JAX for improved compilation performance

JAX Toolbox

JAX Toolbox focuses on achieving the best performance and convergence on NVIDIA Ampere, Hopper,Blackwell, and Rubin architectures and provides scripts and container images for training and fine-tuning the latest open source models. These examples are tested against a nightly CI as well as each NGC container release to ensure consistent accuracy and performance.

Nightly Containers

In addition to projects, JAX Toolbox includes nightly containers for libraries across the JAX ecosystem.

Container	Type	Image URI
jax	-	ghcr.io/nvidia/jax:jax-YYYY-MM-DD
maxtext	LLM framework	ghcr.io/nvidia/jax:maxtext-YYYY-MM-DD
axlearn	LLM framework	ghcr.io/nvidia/jax:axlearn-YYYY-MM-DD

Known Issues

- None.

Chapter 6. JAX Release 26.09

The NVIDIA JAX Release 26.09 is made up of two container images available on NGC: JAX and MaxText.

Contents of the JAX container

This container image contains the complete source for the following software:

- JAX: `/opt/jax`
- XLA: `/opt/xla`
- Flax: `/opt/flax`
- TransformerEngine: `/opt/transformer-engine`

The MaxText container image is based on the JAX container. Additionally, it includes:

- MaxText: `/opt/maxtext`

The JAX runtime package jaxlib is prebuilt and installed in the default Python environment (/usr/local/lib/python3.10/dist-packages/jaxlib) in the container image.

Versions of packages included in both of these containers:

- CUDA 13.4.1
  - Please refer to the CUDA DL 26.09 release notes section for the list of libraries inherited from the CUDA container.
- JAX jax-v0.11.1
- MaxText 0c794a5
- Flax c5f2e5b
- XLA dcf304b
- Transformer Engine v2.18 (b8b2ea2)

Driver Requirements

Release 26.09 is based on CUDA 13.4.1. For comprehensive and up-to-date driver compatibility information, please refer to the following documentation:

- NVIDIA CUDA Compatibility Guide - Compatibility information between CUDA versions and driver releases.
- CUDA Toolkit Release Notes - Driver version requirements and compatibility matrices.
- NVIDIA Drivers Download - Latest NVIDIA drivers.

Key Features and Enhancements

- The 26.09 DLFW containers introduce initial support for the NVIDIA Rubin GPU architecture.
- Enabled CUDA 13 support and PTX9 instruction set in XLA, allowing JAX programs to take advantage of the latest CUDA toolchain capabilities.
- Upgraded cuRAND to version 11, maintaining the latest CUDA toolchain capabilities.
- Backported Blackwell FP4 Triton fixes to ensure numerically correct and stable FP4 operations on Blackwell GPUs.
- Added auto-detection of cluster configuration for Open MPI 5.0, improving multi-node JAX workload portability.

MOE

- Integrated TransformerEngine Collective GEMM into MaxText for MoE layers, enabling GPU communication and computation overlap during training. (MaxText).
- Enabled overlapping MoE expert communications with collective matrix multiplication to improve GPU utilization during MoE forward and backward passes. (MaxText)
- Added Expert Parallelism via a NCCL-EP backend in TransformerEngine for JAX, with bitmap top-k for faster MoE routing. (TransformerEngine)
- Extended communication+GEMM overlap API in JAX to support the cuBLASMp backend. (TransformerEngine)
- Made the linear offset of ClampedSwiGLU configurable in JAX for improved MoE activation control. (TransformerEngine).

Attention

- Added cuDNN-backed flex attention support in JAX. (TransformerEngine)
- Enabled cuDNN fused attention backward pass for attention head dimension 256 on Blackwell GPUs (compute capability 10.x), closing a performance gap for large-head-dim attention workloads. (TransformerEngine)
- Fixed cuDNN packed-attention key-value offsets for GQA and MQA configurations, resolving incorrect attention outputs in grouped-query and multi-query scenarios. (JAX)
- Fixed cuDNN attention batching rules for operands without a vmap axis and for FP8 attention, improving correctness of batched multi-head attention under JAX vmap transforms. (JAX)

Precision

- Introduced packed storage shapes for FP4 XTile operands in XLA, reducing memory footprint for low-precision computations on Blackwell (XLA).
- Added lowering of scaled-dot operand transposes in XTile-to-Triton, enabling efficient FP4 matrix multiplication with transposed inputs. (XLA).
- Restored the Triton TMEM scale padding workaround for Blackwell, maintaining numerical correctness in FP4 tensor memory operations. (XLA).
- Added a BF16 fallback path for FP8 cuDNN convolutions when no valid cuDNN plan is available, preventing silent failures on unsupported configurations.

Collective and Communication

- Enabled symmetric buffers for reduce-scatter, reducing memory allocation overhead and improving bandwidth utilization in distributed reductions.
- Added a device-initiated ragged all-to-all kernel using NCCL LSA/GIN primitives, enabling efficient irregular collective communication patterns on NVIDIA GPUs and optimizing to saturate all SMs.
- Introduced a backend-agnostic collectives FFI extension in XLA, providing a stable C API for integrating custom collective operations across compute backends.
- Added support for true stable memory pointers for collective and user-specified buffer allocations, improving memory safety in long-running distributed programs.

Compiler and Runtime

- JAX upgraded to v0.11.1 (from v0.10.2) with CUDA 13.4 support.
- Added VA remapping of parameter and output buffer allocations, enabling the GPU allocator to remap virtual memory for more efficient buffer reuse across computation steps. (XLA)
- Introduced SKIP_PROFILED command buffer update mode, allowing command buffers to skip re-profiling steps that have already been measured. (XLA)
- Added CommandBuffer fallback mode during profile steps of SKIP_PROFILED update mode, ensuring correctness when a profiled kernel is unavailable. (XLA)
- Implemented batched VMM allocator deferred teardown, reducing allocation overhead for workloads that repeatedly create and destroy large GPU buffers. (XLA)
- Avoided persistent allocation scans for unchanged command buffer commands, reducing CPU overhead during repeated execution of static computation graphs. (XLA)
- Added the SchedulerMemoryFencing pass to bound peak memory usage for the left-hand side of scheduled operations, improving memory predictability for large models. (XLA)
- Enabled selective overlap of D2D copy operations with compute-bound kernels, improving GPU utilization by hiding data movement latency.

- Added XLA flag-driven CUPTI 13.3 multi-subscriber support with CUPTI 13.2 fallback, enabling coexistence of multiple profiling tools alongside JAX on the same GPU process. (XLA)
- Added RAFT select_k for unstable TopK on GPU and lowered approx_top_k to stable sort+slice. (XLA)

JAX Toolbox

The JAX Toolbox projects focus on achieving the best performance and convergence on NVIDIA Ampere, Hopper, and Blackwell architecture families and provide the latest deep learning models and scripts for training and fine-tuning. These examples are tested against a nightly CI as well as each NGC container release to ensure consistent accuracy and performance over time.

Nightly Containers

In addition to projects, JAX Toolbox includes nightly containers for libraries across the JAX ecosystem.

Container	Type	Image URI
jax	-	ghcr.io/nvidia/jax:jax-YYYY-MM-DD
maxtext	LLM framework	ghcr.io/nvidia/jax:maxtext-YYYY-MM-DD
axlearn	LLM framework	ghcr.io/nvidia/jax:axlearn-YYYY-MM-DD

Known Issues

- None.

Chapter 7. JAX Release 26.08

The NVIDIA JAX Release 26.08 is made up of two container images available on NGC: JAX and MaxText.

Contents of the JAX container

This container image contains the complete source for the following software:

- JAX: `/opt/jax`
- XLA: `/opt/xla`
- Flax: `/opt/flax`
- TransformerEngine: `/opt/transformer-engine`

The MaxText container image is based on the JAX container. Additionally, it includes:

- MaxText: `/opt/maxtext`

The JAX runtime package jaxlib is prebuilt and installed in the default Python environment (/usr/local/lib/python3.10/dist-packages/jaxlib) in the container image.

Versions of packages included in both of these containers:

- CUDA 13.4.1
  - Please refer to the CUDA DL 26.08 release notes section for the list of libraries inherited from the CUDA container.
- JAX jax-v0.11.0
- MaxText be69fa6
- Flax a31129e
- XLA 131bf41
- Transformer Engine 27486e03c

Driver Requirements

Release 26.08 is based on CUDA 13.4.1. For comprehensive and up-to-date driver compatibility information, please refer to the following documentation:

- NVIDIA CUDA Compatibility Guide - Compatibility information between CUDA versions and driver releases.
- CUDA Toolkit Release Notes - Driver version requirements and compatibility matrices.
- NVIDIA Drivers Download - Latest NVIDIA drivers.

Key Features and Enhancements

MOE

- Integrated TransformerEngine Collective GEMM into MaxText for MoE layers, enabling GPU communication and computation overlap during training. (MaxText).
- Added fused MoE FFN1 combining wi_0 and wi_1 into a single grouped GEMM, reducing GPU kernel launch overhead for MoE MLP layers. (MaxText)
- Enabled overlapping MoE expert communications with collective matrix multiplication to improve GPU utilization during MoE forward and backward passes. (MaxText)
- Fixed ragged all-to-all buffer factor handling in DeepSeek-V3 MoE routing collectives. (MaxText)
- Added Expert Parallelism via a NCCL-EP backend in TransformerEngine for JAX, with bitmap top-k for faster MoE routing. (TransformerEngine)
- Extended communication+GEMM overlap API in JAX to support the cuBLASMp backend. (TransformerEngine)
- Made the linear offset of ClampedSwiGLU configurable in JAX for improved MoE activation control. (TransformerEngine)

Attention

- Added packed all-gather for context parallelism, improving GPU memory efficiency in long-context training. (MaxText)
- Added cuDNN-backed flex attention support in JAX. (TransformerEngine)

Compiler and Runtime

- JAX upgraded to v0.11.0 (from v0.10.2) with CUDA 13.4 support.
- Enabled experimental ragged all-to-all barrier with NCCL by default for improved GPU collective performance. (XLA)
- Added RAFT select_k for unstable TopK on GPU and lowered approx_top_k to stable sort+slice. (XLA)
- Integrated latest Triton updates and fixed thread pool starvation deadlock in GPU autotuning. (XLA)

JAX Toolbox

The JAX Toolbox projects focus on achieving the best performance and convergence on NVIDIA Ampere, Hopper, and Blackwell architecture families and provide the latest deep learning models and scripts for training and fine-tuning. These examples are tested

against a nightly CI as well as each NGC container release to ensure consistent accuracy and performance over time.

Nightly Containers

In addition to projects, JAX Toolbox includes nightly containers for libraries across the JAX ecosystem.

Container	Type	Image URI
jax	-	ghcr.io/nvidia/jax:jax-YYYY-MM-DD
maxtext	LLM framework	ghcr.io/nvidia/jax:maxtext-YYYY-MM-DD
axlearn	LLM framework	ghcr.io/nvidia/jax:axlearn-YYYY-MM-DD

Known Issues

- None.

Chapter 8. JAX Release 26.07

The NVIDIA JAX Release 26.07 is made up of two container images available on NGC: JAX and MaxText.

Contents of the JAX container

This container image contains the complete source for the following software:

- JAX: `/opt/jax`
- XLA: `/opt/xla`
- Flax: `/opt/flax`
- TransformerEngine: `/opt/transformer-engine`

The MaxText container image is based on the JAX container. Additionally, it includes:

- MaxText: `/opt/maxtext`

The JAX runtime package jaxlib is prebuilt and installed in the default Python environment (/usr/local/lib/python3.10/dist-packages/jaxlib) in the container image.

Versions of packages included in both of these containers:

- CUDA 13.3 U1
  - Please refer to the CUDA DL 26.07 release notes section for the list of libraries inherited from the CUDA container.
- JAX jax-v0.10.2
- MaxText 454e55f
- Flax 5768bb5
- XLA 5a9e73c (internal)
- Transformer Engine 4220403
- AXLearn c8624c8

Driver Requirements

Release 26.07 is based on CUDA 13.3.1. For comprehensive and up-to-date driver compatibility information, please refer to the following documentation:

- NVIDIA CUDA Compatibility Guide - Compatibility information between CUDA versions and driver releases.
- CUDA Toolkit Release Notes - Driver version requirements and compatibility matrices.
- NVIDIA Drivers Download - Latest NVIDIA drivers.

Key Features and Enhancements

- The GCP NCCL plugin package is now installed in the container. On Google Cloud Platform hosts (A3 Ultra, A4, A4X and A4X Max machine types), the container automatically detects and activates the GCP NCCL network, tuner, profiler, and environment plugins for optimized multi-node communication.
  - On GCP hosts with the GCP NCCL plugins installed, NCCL telemetry is enabled by default starting in this release.
  - A runtime advisory message is displayed when running on GCP hosts with the GCP NCCL plugins active, directing users to the GCP NCCL telemetry documentation.
- PDL launch inserted after Triton GEMM MMA loops before epilogues by default, increasing producer-consumer overlap with e2e speed up on smaller models.
- cuDNN-backed flex attention in jax.dot_product_attention reenabled.
- Support for CuDNN's AoT/deviceless compilation in XLA. cuDNN backend invoked in deviceless mode in XLA, improving compilation time for graphs with cuDNN calls.
- Fixed incorrect gradients in zero-padded positions in jax.dot_product_attention.
- Redundant zero-initialization removal in DS/DUS scan-loops, giving small end to end speedups in LLM model training.

Deprecations

- Removal of NVSHMEM calls/library dependency from XLA. Developers using NVSHMEM in custom kernels are recommended to leverage NCCL device-side APIs.

JAX Toolbox

The JAX Toolbox projects focus on achieving the best performance and convergence on NVIDIA Ampere, Hopper, and Blackwell architecture families and provide the latest deep learning models and scripts for training and fine-tuning. These examples are tested against a nightly CI as well as each NGC container release to ensure consistent accuracy and performance over time.

Nightly Containers

In addition to projects, JAX Toolbox includes nightly containers for libraries across the JAX ecosystem.

Container	Type	Image URI
jax	-	ghcr.io/nvidia/jax:jax-YYYY-MM-DD
maxtext	LLM framework	ghcr.io/nvidia/jax:maxtext-YYYY-MM-DD
axlearn	LLM framework	ghcr.io/nvidia/jax:axlearn-YYYY-MM-DD

Known Issues

- None.

Chapter 9. JAX Release 26.06

The NVIDIA JAX Release 26.06 is made up of two container images available on NGC: JAX and MaxText.

Contents of the JAX container

This container image contains the complete source for the following software:

- JAX: `/opt/jax`
- XLA: `/opt/xla`
- Flax: `/opt/flax`
- TransformerEngine: `/opt/transformer-engine`

The MaxText container image is based on the JAX container. Additionally, it includes:

- MaxText: `/opt/maxtext`

The JAX runtime package jaxlib is prebuilt and installed in the default Python environment (/usr/local/lib/python3.10/dist-packages/jaxlib) in the container image.

Versions of packages included in both of these containers:

- CUDA 13.3.0
  - Please refer to the CUDA DL 26.06 release notes section for the list of libraries inherited from the CUDA container.
- JAX 0.10.1
- Maxtext (23b4787)
- Flax (d044de8)
- XLA (9b63591)
- TransformerEngine 2.16

Driver Requirements

Release 26.06 is based on CUDA 13.3.0. For comprehensive and up-to-date driver compatibility information, please refer to the following documentation:

- NVIDIA CUDA Compatibility Guide - Compatibility information between CUDA versions and driver releases.
- CUDA Toolkit Release Notes - Driver version requirements and compatibility matrices.
- NVIDIA Drivers Download - Latest NVIDIA drivers.

Key Features and Enhancements

- Upgraded JAX to 0.10.0.
- Pallas/Triton kernels now compiled to PTX by default, improving portability and load times.
- Added CUDA 13 build target.
- MLIR-to-LLVM IR compilation is now asynchronous for MlirKernelFusion.
- Improved GPU dot fusion cost model and introduced tileable GemmFusion.

JAX Toolbox

The JAX Toolbox projects focus on achieving the best performance and convergence on NVIDIA Ampere, Hopper, and Blackwell architecture families and provide the latest deep learning models and scripts for training and fine-tuning. These examples are tested against a nightly CI as well as each NGC container release to ensure consistent accuracy and performance over time.

Nightly Containers

In addition to projects, JAX Toolbox includes nightly containers for libraries across the JAX ecosystem.

Container	Type	Image URI
jax	-	ghcr.io/nvidia/jax:jax-YYYY-MM-DD
maxtext	LLM framework	ghcr.io/nvidia/jax:maxtext-YYYY-MM-DD
axlearn	LLM framework	ghcr.io/nvidia/jax:axlearn-YYYY-MM-DD

Known Issues

- None.

Chapter 10. JAX Release 26.05

The NVIDIA JAX Release 26.05 is made up of two container images available on NGC: JAX and MaxText.

Contents of the JAX container

This container image contains the complete source for the following software:

- JAX: `/opt/jax`
- XLA: `/opt/xla`
- Flax: `/opt/flax`
- TransformerEngine: `/opt/transformer-engine`

The MaxText container image is based on the JAX container. Additionally, it includes:

- MaxText: `/opt/maxtext`

The JAX runtime package jaxlib is prebuilt and installed in the default Python environment (/usr/local/lib/python3.10/dist-packages/jaxlib) in the container image.

Versions of packages included in both of these containers:

- CUDA 13.2.1
  - Please refer to the CUDA DL 26.05 release notes section for the list of libraries inherited from the CUDA container.
- JAX 0.10.0
- Maxtext (083293f)
- Flax (901efc3)
- XLA (b6f37ab)
- TransformerEngine 2.15

Driver Requirements

Release 26.05 is based on CUDA 13.2.1. For comprehensive and up-to-date driver compatibility information, please refer to the following documentation:

- NVIDIA CUDA Compatibility Guide - Compatibility information between CUDA versions and driver releases.
- CUDA Toolkit Release Notes - Driver version requirements and compatibility matrices.
- NVIDIA Drivers Download - Latest NVIDIA drivers.

Key Features and Enhancements

- jax.ragged_dot has better performance. It's powered by XLA-cuDNN. Use sparse_matmul=true in MaxText to invoke jax.ragged_dot.
- cuBLASLt is an alternative gemm backend, replacing cuBLAS. The autotuning of gemms now spans across Triton, cuDNN and cuBLASLt.
- Enabled deviceless XLA-cuDNN fusion for AoT workflow to reduce compilation time.
- Expanded fusion support of convolution with XLA-cuDNN graph API integration with NHWC layout.
- Added CUPTI 13.2 multi-subscriber tracing, allowing XLA profiling to run in the same process as other CUPTI-based tools such as NSight.

Performance Optimizations

- Accelerated jax.ragged_dot with XLA-cuDNN integration. fprop/dgrad powered by ragged gemm while wgrad is by batched dense gemm. Use xla_gpu_experimental_use_ragged_dot_fusion = ON.
- Added basic PDL (Programmatic Dependent Launch) in XLA, uplifting E2E performance of inference models.For example, Gemma3's perf uplifted by single digit percentage.
- Introduced native kernel codegen improvements for FP4 training, including all-gather-transpose optimization, higher unroll/vectorization factors for narrow data types, and lowering slice/slice-update from FP4 activation offloading to D2H memcpy.
- Enabled async multi-stream collectives in XLA, permitting inter-node communication overlap with that of intra-node communication. For example, Llama3-405B perf uplifted by single digit percentage.
- Enhanced XLA to leverage NCCL 2.28's copy engine to reduce SM usage of XLA's all-gather & all-2-all communication collectives. For example, MaxText's Llama3 perf uplifted by single digit percentage.
- Improved D2H & H2D copy overlap with compute, to help accelerate MoE models on GB200/GB300 systems.

JAX Toolbox

The JAX Toolbox projects focus on achieving the best performance and convergence on NVIDIA Ampere, Hopper, and Blackwell architecture families and provide the latest deep learning models and scripts for training and fine-tuning. These examples are tested

against a nightly CI as well as each NGC container release to ensure consistent accuracy and performance over time.

Nightly Containers

In addition to projects, JAX Toolbox includes nightly containers for libraries across the JAX ecosystem.

Container	Type	Image URI
jax	-	ghcr.io/nvidia/jax:jax-YYYY-MM-DD
maxtext	LLM framework	ghcr.io/nvidia/jax:maxtext-YYYY-MM-DD
axlearn	LLM framework	ghcr.io/nvidia/jax:axlearn-YYYY-MM-DD

Known Issues

- Transformer Engine version 2.14 shipped with 26.04 has an issue resulting in a nondeterministic wrong answer when using MXFP8 training and bias is present. Please upgrade to Transformer Engine version 2.14.1, where the issue was fixed.

Chapter 11. JAX Release 26.04

The NVIDIA JAX Release 26.04 is made up of two container images available on NGC: JAX and MaxText.

Contents of the JAX container

This container image contains the complete source for the following software:

- JAX: `/opt/jax`
- XLA: `/opt/xla`
- Flax: `/opt/flax`
- TransformerEngine: `/opt/transformer-engine`

The MaxText container image is based on the JAX container. Additionally, it includes:

- MaxText: `/opt/maxtext`

The JAX runtime package jaxlib is prebuilt and installed in the default Python environment (/usr/local/lib/python3.10/dist-packages/jaxlib) in the container image.

Versions of packages included in both of these containers:

- CUDA 13.2.1
  - Please refer to the CUDA DL 26.04 release notes section for the list of libraries inherited from the CUDA container.
- JAX 0.9.2
- Maxtext (093ab89)
- Flax (1fcc2a5)
- XLA (187a5eb)
- TransformerEngine 2.14
- AXLearn (5602a78)

Driver Requirements

Release 26.04 is based on CUDA 13.2.1. For comprehensive and up-to-date driver compatibility information, please refer to the following documentation:

- NVIDIA CUDA Compatibility Guide - Compatibility information between CUDA versions and driver releases.
- CUDA Toolkit Release Notes - Driver version requirements and compatibility matrices.
- NVIDIA Drivers Download - Latest NVIDIA drivers.

Key Features and Enhancements

- Integrated NCCL 2.29+ one-side APIs: ncclPutSignal, ncclSignal, and ncclWaitSignal in XLA to enable custom kernels and ragged all-2-all communication to use NCCL's symmetric memory.
- Completed Thor support in JAX/XLA with full Triton support - Enabled TCGen5 MMA in triton with CUDA 13.1.

Performance optimizations

- Enabled FP8 reduction ops in XLA-NCCL, helping improve communication throughput by ~1.6x-2x over BF16.
- Improved collective cost model of GB200/GB300 systems to improve the accuracy of latency estimators.
- In PGLE-disabled scenario, improved heuristic to expand the overlap coverage, delivering a single digit perf for dense models.

JAX Toolbox

The JAX Toolbox projects focus on achieving the best performance and convergence on NVIDIA Ampere, Hopper, and Blackwell architecture families and provide the latest deep learning models and scripts for training and fine-tuning. These examples are tested against a nightly CI as well as each NGC container release to ensure consistent accuracy and performance over time.

Nightly Containers

In addition to projects, JAX Toolbox includes nightly containers for libraries across the JAX ecosystem.

Container	Type	Image URI
jax	-	ghcr.io/nvidia/jax:jax-YYYY-MM-DD
maxtext	LLM framework	ghcr.io/nvidia/jax:maxtext-YYYY-MM-DD
axlearn	LLM framework	ghcr.io/nvidia/jax:axlearn-YYYY-MM-DD

Known Issues

- Transformer Engine version 2.14 shipped with 26.04 has an issue resulting in a nondeterministic wrong answer when using MXFP8 training and bias is present. Please upgrade to Transformer Engine version 2.14.1, where the issue was fixed.

Chapter 12. JAX Release 26.03

The NVIDIA JAX Release 26.03 is made up of two container images available on NGC: JAX and MaxText.

Contents of the JAX container

This container image contains the complete source for the following software:

- JAX: `/opt/jax`
- XLA: `/opt/xla`
- Flax: `/opt/flax`
- TransformerEngine: `/opt/transformer-engine`

The MaxText container image is based on the JAX container. Additionally, it includes:

- MaxText: `/opt/maxtext`

The JAX runtime package jaxlib is prebuilt and installed in the default Python environment (/usr/local/lib/python3.10/dist-packages/jaxlib) in the container image.

Versions of packages included in both of these containers:

- CUDA 13.2
  - Please refer to the CUDA DL 26.03 release notes section for the list of libraries inherited from the CUDA container.
- JAX 0.9.0.1
- Maxtext (d4ed226)
- Flax (7e5c70a)
- XLA (bb760b0)
- TransformerEngine 2.13
- AXLearn (fd34795)

Driver Requirements

Release 26.03 is based on CUDA 13.2.0. For comprehensive and up-to-date driver compatibility information, please refer to the following documentation:

- NVIDIA CUDA Compatibility Guide - Compatibility information between CUDA versions and driver releases.
- CUDA Toolkit Release Notes - Driver version requirements and compatibility matrices.
- NVIDIA Drivers Download - Latest NVIDIA drivers.

Key Features and Enhancements

This JAX release includes the following key features and enhancements.

- The current release is based on JAX 0.9.0.1 and CUDA 13.2
- The new cuBLAS version included in the container contains fix for issue present in former JAX container release (wherein some cublasLtMatmul kernels can lead to incorrect results when executed concurrently with another kernel that uses Tensor Memory on GPUs with Compute Capability 10.x and 11.x).

JAX Toolbox

The JAX Toolbox projects focus on achieving the best performance and convergence on NVIDIA Ampere, Hopper, and Blackwell architecture families and provide the latest deep learning models and scripts for training and fine-tuning. These examples are tested against a nightly CI as well as each NGC container release to ensure consistent accuracy and performance over time.

Nightly Containers

In addition to projects, JAX Toolbox includes nightly containers for libraries across the JAX ecosystem.

Container	Type	Image URI
jax	-	ghcr.io/nvidia/jax:jax-YYYY-MM-DD
maxtext	LLM framework	ghcr.io/nvidia/jax:maxtext-YYYY-MM-DD
equinox	layer library	ghcr.io/nvidia/jax:equinox-YYYY-MM-DD
axlearn	LLM framework	ghcr.io/nvidia/jax:axlearn-YYYY-MM-DD

Known Issues

- None.

Chapter 13. JAX Release 26.02

The NVIDIA JAX Release 26.02 is made up of two container images available on NGC: JAX and MaxText.

Contents of the JAX container

This container image contains the complete source for the following software:

- JAX: `/opt/jax`
- XLA: `/opt/xla`
- Flax: `/opt/flax`
- TransformerEngine: `/opt/transformer-engine`

The MaxText container image is based on the JAX container. Additionally, it includes:

- MaxText: `/opt/maxtext`

The JAX runtime package jaxlib is prebuilt and installed in the default Python environment (/usr/local/lib/python3.10/dist-packages/jaxlib) in the container image.

Versions of packages included in both of these containers:

- CUDA 13.1.U1
  - Please refer to the CUDA DL 26.02 release notes section for the list of libraries inherited from the CUDA container.
- JAX 0.9.0.1
- XLA (bb760b0)
- TransformerEngine 2.12
- Flax 0.12.1
- Maxtext (aca5b24)

Driver Requirements

Release 26.02 is based on CUDA 13.1 U1 (Toolkit) which requires CUDA Driver release 590.48. Please refer to the latest Drivers and CTKsupport table for additional information.

For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

Key Features and Enhancements

This JAX release includes the following key features and enhancements.

- The current release is based on JAX 0.9.0.1 and CUDA 13.1 U1

JAX Toolbox

The JAX Toolbox projects focus on achieving the best performance and convergence on NVIDIA Ampere, Hopper, and Blackwell architecture families and provide the latest deep learning models and scripts for training and fine-tuning. These examples are tested against a nightly CI as well as each NGC container release to ensure consistent accuracy and performance over time.

Nightly Containers

In addition to projects, JAX Toolbox includes nightly containers for libraries across the JAX ecosystem.

Container	Type	Image URI
jax	-	ghcr.io/nvidia/jax:jax-YYYY-MM-DD
maxtext	LLM framework	ghcr.io/nvidia/jax:maxtext-YYYY-MM-DD
equinox	layer library	ghcr.io/nvidia/jax:equinox-YYYY-MM-DD
axlearn	LLM framework	ghcr.io/nvidia/jax:axlearn-YYYY-MM-DD

Known Issues

- The version of cuBLAS included in the container has a known issue where some cublasLtMatmul kernels can lead to incorrect results when executed concurrently with another kernel that uses Tensor Memory on GPUs with Compute Capability 10.x and 11.x.

Chapter 14. JAX Release 26.01

The NVIDIA JAX Release 26.01 is made up of two container images available on NGC: JAX and MaxText.

Contents of the JAX container

This container image contains the complete source for the following software:

- JAX: `/opt/jax`
- XLA: `/opt/xla`
- Flax: `/opt/flax`
- TransformerEngine: `/opt/transformer-engine`

The MaxText container image is based on the JAX container. Additionally, it includes:

- MaxText: `/opt/maxtext`

The JAX runtime package jaxlib is prebuilt and installed in the default Python environment (/usr/local/lib/python3.10/dist-packages/jaxlib) in the container image.

Versions of packages included in both of these containers:

- CUDA 13.1.1
  - Please refer to the CUDA DL 26.01 release notes section for the list of libraries inherited from the CUDA container.
- JAX 0.8.1
- XLA (a7e5887)
- TransformerEngine 2.11
- Flax 0.12.1
- Maxtext (aca5b24)

Driver Requirements

Release 26.01 is based on CUDA 13.1 U1 (Toolkit) which requires CUDA Driver release 590.48. Please refer to the latest Drivers and CTKsupport table for additional information.

For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

Key Features and Enhancements

This JAX release includes the following key features and enhancements.

- The current release is based on JAX 0.8.1 and CUDA 13.1 U1
- Added support for DGX Spark and Jetson Thor
- Added NVLink domain capability for NVSHMEM to accelerate long-context training in multi-node multi-gpu environments. Use `--xla_gpu_experimental_enable_nvshmem=true` flag.
- Fixed problem with running in a configuration with multiple GPUs per process where JAX would hang when loading a kernel to GPUs on the first run of the kernel.
- Enabled boolean, int8 and uint8 datatypes in NVSHMEM reduction collectives.
- Improved collective optimization pipeline to enable all-reduces to lower into efficient reduce-scatters.
- Optimizations for sub-byte data types:
  - Generate hardware intrinsics for efficient conversion of fp4 types.
  - Optimized communication collective operation on sub-byte types such as int4 and fp4 eliminating redundant cast operations and reducing the volume of data moved.
- Enable int4 in cuDNN Gemm fusions. Set flag `xla_gpu_cudnn_gemm_fusion_level > 0` to use.
- Support forward convolution with dilation and added heuristic to differentiate between forward and backward convolution and dispatch to optimized cuDNN kernels.

JAX Toolbox

The JAX Toolbox projects focus on achieving the best performance and convergence on NVIDIA Ampere, Hopper, and Blackwell architecture families and provide the latest deep learning models and scripts for training and fine-tuning. These examples are tested against a nightly CI as well as each NGC container release to ensure consistent accuracy and performance over time.

Nightly Containers

In addition to projects, JAX Toolbox includes nightly containers for libraries across the JAX ecosystem.

Container	Type	Image URI
jax	-	ghcr.io/nvidia/jax:jax-YYYY-MM-DD

Container	Type	Image URI
maxtext	LLM framework	ghcr.io/nvidia/jax:maxtext-YYYY-MM-DD
equinox	layer library	ghcr.io/nvidia/jax:equinox-YYYY-MM-DD
axlearn	LLM framework	ghcr.io/nvidia/jax:axlearn-YYYY-MM-DD

Known Issues

There are no known issues.

Chapter 15. JAX Release 25.12

There is no JAX container in DLFW release 25.12.

Chapter 16. JAX Release 25.11

There is no JAX container in DLFW release 25.11.

Chapter 17. JAX Release 25.10

The NVIDIA JAX Release 25.10 is made up of two container images available on NGC: JAX and MaxText.

Contents of the JAX container

This container image contains the complete source for the following software:

- JAX: `/opt/jax`
- XLA: `/opt/xla`
- Flax: `/opt/flax`
- TransformerEngine: `/opt/transformer-engine`

The MaxText container image is based on the JAX container. Additionally, it includes:

- MaxText: `/opt/maxtext`

The JAX runtime package jaxlib is prebuilt and installed in the default Python environment (/usr/local/lib/python3.10/dist-packages/jaxlib) in the container image.

Versions of packages included in both of these containers:

- JAX 0.7.2
- XLA (@0fccb8a)
- TransformerEngine 2.8
- Flax 0.11.2
- Maxtext (@8fdac10)
- Ubuntu 24.04 including Python 3.12
- NVIDIA CUDA® 13.0.2
- NVSHMEM 3.3.20
- NVIDIA cuBLAS 13.1.0.3
- NVIDIA cuDNN 9.14.0.64
- NVIDIA NCCL 2.27.7
- NVIDIA DALI 1.51.2
- rdma-core 50.0

- NVIDIA HPC-X 2.24
- GDRCopy 2.4.1
- Nsight Compute 2025.3.1.4
- Nsight Systems 2025.5.1.121

Driver Requirements

Release 25.10 is based on CUDA 13.0.2 (Toolkit) which requires CUDA Driver release 580.65. Please refer to the latest Drivers and CTKsupport table for additional information.

For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

Key Features and Enhancements

This JAX release includes the following key features and enhancements.

- The current release is based on JAX 0.7.2 and CUDA 13.0.2.
- G/B300 performant support.
- JAX comes with the Shardy partitioner on by default. See Shardy Migration guide for more details or instructions on how to disable the new partitioner.
- Fixed global scale application in NVFP4 quantization.
- Added and improved Shardy sharding rules for JAX SDPA and JAX scaled matmul APIs.
- Enabled XLA support for symmetric memory kernels in NCCL 2.27 collective operations. Use the flag `–xla_gpu_experimental_enable_nccl_symmetric_buffers` to enable it. Symmetric memory kernels enable an improvement in NCCL communication latencies while reducing the number of SMs used for collectives, thus allowing for better overlap with compute operations.
- Transformer Engine added fused swizzling operation for the scaling factor inverse and transpose calculation of the data.

JAX Toolbox

The JAX Toolbox projects focus on achieving the best performance and convergence on NVIDIA Ampere, Hopper, and Blackwell architecture families and provide the latest deep learning models and scripts for training and fine-tuning. These examples are tested against a nightly CI as well as each NGC container release to ensure consistent accuracy and performance over time.

Nightly Containers

In addition to projects, JAX Toolbox includes nightly containers for libraries across the JAX ecosystem.

Container	Type	Image URI
jax	-	ghcr.io/nvidia/jax:jax
t5x	LLM framework	ghcr.io/nvidia/jax:t5x
maxtext	LLM framework	ghcr.io/nvidia/jax:maxtext
equinox	layer library	ghcr.io/nvidia/jax:equinox
axlearn	LLM framework	ghcr.io/nvidia/jax:axlearn

Known Issues

When running in a configuration with multiple GPUs per process, JAX can hang when loading a kernel to GPUs on the first run of the kernel. This has been mitigated by preloading Transformer Engine kernels. NVIDIA recommends running in a process-per-GPU configuration (this is the default when using SLURM automatic configuration).

Chapter 18. JAX Release 25.09

There is no JAX container in DLFW release 25.09.

Chapter 19. JAX Release 25.08

The NVIDIA JAX Release 25.08 is made up of two container images available on NGC: JAX and MaxText.

Contents of the JAX container

This container image contains the complete source for the following software:

- JAX: `/opt/jax`
- XLA: `/opt/xla`
- Flax: `/opt/flax`
- TransformerEngine: `/opt/transformer-engine`

The MaxText container image is based on the JAX container. Additionally, it includes:

- MaxText: `/opt/maxtext`

The JAX runtime package jaxlib is prebuilt and installed in the default Python environment (/usr/local/lib/python3.10/dist-packages/jaxlib) in the container image.

Versions of packages included in both of these containers:

- JAX 0.6.2
- Flax 0.10.5
- maxtext.git@0b1fde4513d9 (only in the maxtext container
- Ubuntu 24.04 including Python 3.12
- NVIDIA CUDA® 13.0.0
- NVSHMEM 3.3.20
- NVIDIA cuBLAS 13.0.0.19
- NVIDIA cuDNN 9.12.0.46
- NVIDIA NCCL 2.27.7
- NVIDIA DALI 1.51.2
- rdma-core 50.0
- NVIDIA HPC-X 2.24
- GDRCopy 2.4.1

- Nsight Compute 2025.3.0.19
- Nsight Systems 2025.4.1.136
- TransformerEngine 2.5

Driver Requirements

Release 25.08 is based on CUDA 13.0.0 (Toolkit) which requires CUDA Driver release 580.65. Please refer to the latest Drivers and CTKsupport table for additional information.

For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

Key Features and Enhancements

This JAX release includes the following key features and enhancements.

- The current release is based on JAX 0.6.2.
- Compatibility with CUDA 13.0
- New hardware: RTX PRO™ 6000 Blackwell Server Edition functional support.
- Max Text
  - JAX SDPA API enabled for training and inference models.
  - Added Context Parallelism using TE, cuDNN's Flash attention & all-gather mechanism to support long-context model training.
- Added CuDNN's paged attention in jax._src_.cudnn package.
- Enabled multi-stream collective overlap in XLA.
- XLA overlaps host offloading of array slices with computation for efficient activation offloading.
- New tutorial on Ray-based resilient training in JAX Toolbox.
- The compiler now automatically overlaps host offloading of array slices with computation. The overlap enables efficient implementations of activation offloading.
- Added hermetic support for CUDA, CuDNN, NCCL and NVSHMEM in XLA.

JAX Toolbox

The JAX Toolbox projects focus on achieving the best performance and convergence on NVIDIA Ampere, Hopper, and Blackwell architecture families and provide the latest deep learning models and scripts for training and fine-tuning. These examples are tested against a nightly CI as well as each NGC container release to ensure consistent accuracy and performance over time.

Nightly Containers

In addition to projects, JAX Toolbox includes nightly containers for libraries across the JAX ecosystem.

Container	Type	Image URI
jax	-	ghcr.io/nvidia/jax:jax
t5x	LLM framework	ghcr.io/nvidia/jax:t5x
maxtext	LLM framework	ghcr.io/nvidia/jax:maxtext
equinox	layer library	ghcr.io/nvidia/jax:equinox
axlearn	LLM framework	ghcr.io/nvidia/jax:axlearn

Known Issues

- When running in a configuration with multiple GPUs per process, JAX can hang when loading a kernel to GPUs on the first run of the kernel. NVIDIA recommends running in a process-per-GPU configuration (this is the default when using SLURM automatic configuration).

Known Issues Fixed

- Numerical correctness problems in jax.nn.scaled_matmul and jax.nn.scaled_dot_general for MXFP8 and NVFP4
- Tensor memory deallocation in Mosaic GPU.

Chapter 20. JAX Release 25.06

There is no JAX container in DLFW release 25.06.

Chapter 21. JAX Release 25.05

There is no JAX container in DLFW release 25.05.

Chapter 22. JAX Release 25.04

The NVIDIA JAX Release 25.04 is made up of two container images available on NGC: JAX and MaxText.

Contents of the JAX container

This container image contains the complete source for the following software:

- JAX: `/opt/jax`
- XLA: `/opt/xla`
- Flax: `/opt/flax`
- TransformerEngine: `/opt/transformer-engine`

The MaxText container image is based on the JAX container. Additionally, it includes:

- MaxText: `/opt/maxtext`

The JAX runtime package jaxlib is prebuilt and installed in the default Python environment (/usr/local/lib/python3.10/dist-packages/jaxlib) in the container image.

Versions of packages included in both of these containers:

- JAX 0.5.3
- Flax 0.10.5
- maxtext.git@0b1fde4513d9 (only in the maxtext container
- Ubuntu 24.04 including Python 3.12
- NVIDIA CUDA® 12.9.0
- NVIDIA cuBLAS 12.9.0
- NVIDIA cuDNN 9.9.0.48
- NVIDIA NCCL 2.26.3
- rdma-core 50.0
- NVIDIA HPC-X 2.22.1
- GDRCopy 2.4.1
- Nsight Compute 2025.2.0.11
- Nsight Systems 2025.2.1.130

- TransformerEngine 2.2

Driver Requirements

Release 25.04 is based on CUDA 12.9.0 which requires NVIDIA Driver release 575 or later. However, if you are running on a data center GPU (for example, B100, L40, or any other data center GPU), you can use NVIDIA driver release 470.57 (or later R470), 525.85 (or later R525), 535.86 (or later R535), or 550.54 (or later R550) in forward-compatibility mode.

The CUDA driver's compatibility package only supports particular drivers. Thus, users should upgrade from all R418, R440, R450, R460, R510, R520, R530, R545 and R555 and R560 drivers, which are not forward-compatible with CUDA 12.9. For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

Key Features and Enhancements

This JAX release includes the following key features and enhancements.

- JAX container images in version 25.04 are based on `jaxlib==0.5.3`.
- Experimental support for MXFP8 & FP4 at `jax.nn.scaled_matmul`.
- Enhanced jax SDPA API to support sequence packing.
- Added explicit collective grouping capability. It can be enabled with `set_xla_metadata(_collectives_group="", inlineable="false")` context manager.
- Enabled weight and optimizer state offloading. Reasonable memory saving observed with scan operation.
- Added experimental `compute_on` API to assign compute/communication kernels on a specific stream along with `shard_map`. Placed it behind the experimental flag `xla_gpu_experimental_stream_annotation`.
- Replaced stream capture-based graph creation with explicit graph construction for invoking cuDNN's graph APIs.
- NUMA-aware XLA delivers up-to 2x speedup on D2H/H2D transfers.
- Improved sequence parallelism perf with CSE optimization on collective ops.
- Enabled collective permute optimization to combine small NCCL messages for cpu overhead reduction.
- Observed 10% (geomean) perf gain on MaxText's LLM models over version 25.01 on B200.
- New JAX_COMPILATION_CACHE_EXPECT_PGLE flag to leverage PGLE-optimized compilation caches when using AutoPGLE. Turning on this flag allows AutoPGLE to work in tandem with the NSight profiler.
- `nsys-jax` is now pip installable

JAX Toolbox

The JAX Toolbox projects focus on achieving the best performance and convergence on NVIDIA Ampere, Hopper, and Blackwell architecture families and provide the latest deep learning models and scripts for training and fine-tuning. These examples are tested against a nightly CI as well as each NGC container release to ensure consistent accuracy and performance over time.

Nightly Containers

In addition to projects, JAX Toolbox includes nightly containers for libraries across the JAX ecosystem.

Container	Type	Image URI
jax	-	ghcr.io/nvidia/jax:jax
t5x	LLM framework	ghcr.io/nvidia/jax:t5x
levanter	LLM framework	ghcr.io/nvidia/jax:levanter
maxtext	LLM framework	ghcr.io/nvidia/jax:maxtext
equinox	layer library	ghcr.io/nvidia/jax:equinox
gemma	model	ghcr.io/nvidia/jax:gemma
axlearn	LLM framework	ghcr.io/nvidia/jax:axlearn

Known Issues

- In a single process/per gpu setup, `jax.nn.scaled_matmul` with MXFP8 may cause incorrect values. The suggested workaround is to use `cublasLt` from CUDA 12.8 or use 1 process per GPU.
- Mosaic GPU kernels in this release do not deallocate tensor memory before kernel exit, which causes a runtime check failure that is enabled starting with CUDA 12.8.1. The issue was fixed in JAX upstream PR. We recommend Mosaic GPU users to use the nightly releases of JAX to get the latest fixes.
- To set visible GPU devices, we recommend to set the "CUDA_VISIBLE_DEVICES" environment variable directly. The JAX "jax_cuda_visible_devices" config in Python is not a reliable way to set visible devices.'

Bug Fixes

- Out-of-memory caused by incorrect handling of memory limit #23271.
- Resolved crash with both inter-node FSDP and inter-node DDP enabled on Blackwell.
- Resolved hangs related to running sharded autotuning with cached auto-tuning.

Chapter 23. JAX Release 25.03

There is no JAX container in DLFW release 25.03.

Chapter 24. JAX Release 25.02

There is no JAX container in DLFW release 25.02.

Chapter 25. JAX Release 25.01

The NVIDIA JAX Release 25.01 is made up of two container images available on NGC: JAX and MaxText.

Contents of the JAX container

This container image contains the complete source for the following software:

- JAX: `/opt/jax`
- XLA: `/opt/xla`
- Flax: `/opt/flax`
- TransformerEngine: `/opt/transformer-engine`

The MaxText container image is based on the JAX container. Additionally, it includes:

- MaxText: `/opt/maxtext`

The JAX runtime package jaxlib is prebuilt and installed in the default Python environment (/usr/local/lib/python3.10/dist-packages/jaxlib) in the container image.

Versions of packages included in both of these containers:

- JAX 0.4.38 (Includes release-specific patches and mxfp8.)
- Flax 0.10.2
- maxtext.git@4651cb3c73de (only in the maxtext container
- Ubuntu 24.04 including Python 3.12
- NVIDIA CUDA® 12.8.0.038
- NVIDIA cuBLAS 12.8.3.14
- NVIDIA cuDNN 9.7.0.66
- NVIDIA NCCL 2.25.1
- rdma-core 50.0
- NVIDIA HPC-X 2.21
- OpenMPI 4.1.7
- GDRCopy 2.3.1-1
- Nsight Compute 2025.1.0.14

- Nsight Systems 2024.2.6.2.225
- TransformerEngine 1.14

Driver Requirements

Release 25.01 is based on CUDA 12.8.0 which requires NVIDIA Driver release 570 or later. However, if you are running on a data center GPU (for example, B100, L40, or any other data center GPU), you can use NVIDIA driver release 470.57 (or later R470), 525.85 (or later R525), 535.86 (or later R535), or 550.54 (or later R550) in forward-compatibility mode.

The CUDA driver's compatibility package only supports particular drivers. Thus, users should upgrade from all R418, R440, R450, R460, R510, R520, R530, R545 and R555 and R560 drivers, which are not forward-compatible with CUDA 12.8. For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

Key Features and Enhancements

This JAX release includes the following key features and enhancements.

- JAX container images in version 25.01 are based on `jaxlib==0.4.38`.
- Added Blackwell GPU Architecture support.
- Experimental support for the MXFP8 dtype.
- Experimental support and testing for AWS networking. H100 instances on AWS (P5) have been evaluated. For optimal performance in LLM training and other distributed workloads with high communication costs, NVIDIA recommends the following:
  - AWS: the NCCL plugin supporting AWS EFA is included in the container and will be enabled automatically.

JAX Toolbox

The JAX Toolbox projects focus on achieving the best performance and convergence on NVIDIA Ampere, Hopper, and Blackwell architecture families and provide the latest deep learning models and scripts for training and fine-tuning. These examples are tested against a nightly CI as well as each NGC container release to ensure consistent accuracy and performance over time.

Nightly Containers

In addition to projects, JAX Toolbox includes nightly containers for libraries across the JAX ecosystem.

Container	Type	Image URI
jax	-	ghcr.io/nvidia/jax:jax

Container	Type	Image URI
t5x	LLM framework	ghcr.io/nvidia/jax:t5x
levanter	LLM framework	ghcr.io/nvidia/jax:levanter
maxtext	LLM framework	ghcr.io/nvidia/jax:maxtext
triton	JAX extension	ghcr.io/nvidia/jax:triton
equinox	layer library	ghcr.io/nvidia/jax:equinox
grok	model	ghcr.io/nvidia/jax:grok

Known Issues

- AWS EFA plugin crashes in 1 case with 256 GPUs. You can use this environment variable to WAR the bug: `NCCL_RUNTIME_CONNECT=0` .
- This version of XLA can hang during compilation in rare cases. You can use this environment variable to work around the issue for now: `XLA_FLAGS=--xla_gpu_shard_autotuning=false`.
- On gamer Blackwell(sm_120) GPUs, JAX random number generator is non-deterministic..

Chapter 26. JAX Release 24.12

There is no JAX container in DLFW release 24.12.

Chapter 27. JAX Release 24.11

There is no JAX container in DLFW release 24.11.

Chapter 28. JAX Release 24.10

The NVIDIA container image for JAX, release 24.10 is available on NGC.

Contents of the JAX container

This container image contains the complete source for the following software:

- JAX: `/opt/jax`
- XLA: `/opt/xla`
- Flax: `/opt/flax`
- TransformerEngine: `/opt/transformer-engine`

Additionally, the max container image tag includes:

- MaxText: `/opt/maxtext`

The JAX runtime package jaxlib is prebuilt and installed in the default Python environment (/usr/local/lib/python3.10/dist-packages/jaxlib) in the container image.

The container also includes the following:

- JAX 0.4.33.dev20241018+44e618246
- Flax 0.9.0
- maxtext.git@20f57fb7
- Ubuntu 22.04 including Python 3.10.12
- NVIDIA CUDA® 12.4.2.004
- NVIDIA cuBLAS 12.6.3.3
- NVIDIA cuDNN 9.5.0.50
- NVIDIA NCCL 2.22.3
- rdma-core 39.0
- NVIDIA HPC-X 2.20
- OpenMPI 4.1.7
- GDRCopy 2.3.1-1
- Nsight Compute 2024.3.2.3
- Nsight Systems 2024.2.6.1.90

- TransformerEngine 1.11

Driver Requirements

Release 24.10 is based on CUDA 12.6.2 which requires NVIDIA Driver release 560 or later. However, if you are running on a data center GPU (for example, T4 or any other data center GPU), you can use NVIDIA driver release 470.57 (or later R470), 525.85 (or later R525), 535.86 (or later R535), or 545.23 (or later R545).

The CUDA driver's compatibility package only supports particular drivers. Thus, users should upgrade from all R418, R440, R450, R460, R510, R520, R530, R545 and R555 drivers, which are not forward-compatible with CUDA 12.6. For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

Key Features and Enhancements

This JAX release includes the following key features and enhancements.

- JAX container images in version 24.10 are based on `jaxlib==0.4.33`.
- Experimental support and testing for AWS and GCP networking. H100 instances on both AWS (P5) and GCP (A3-Mega) have been evaluated. For optimal performance in LLM training and other distributed workloads with high communication costs, NVIDIA recommends the following:
  - AWS: Run the script at `/usr/local/bin/install-efa.sh` or include it in a new Dockerfile to leverage AWS EFA.
  - GCP: Follow the guide to set up a cluster and enable GCP's GPUDirect-TCPXO NCCL plugin.

JAX Toolbox

The JAX Toolbox projects focus on achieving the best performance and convergence on NVIDIA Hopper and Ampere tensor cores and provide the latest deep learning models and scripts for training and fine-tuning. These examples are tested against a nightly CI as well as each NGC container release to ensure consistent accuracy and performance over time.

Projects

- GPT: Decoder-only LLM used in many modern text applications. You can also find the latest performance and convergence on the model card.
- LLaMA-2: Decoder-only LLM ranging in scale from 7B to 70B parameters. You can find the latest performance and convergence on our model card.
  - This release also introduces LoRA support for fine-tuning of LLaMA models. NVIDIA includes an example script that finetunes LLama 2 7B on BoolQ and achieves significant metric improvements. You can find the convergence and performance results on the above model card.

  - LLaMA-2 recipes are also available in maxtext. You can find instructions and performance results on our model card.
- MoE: GLaM style MoE with configs up to 1.14T parameter scale. You can find the latest information on our model card.
- T5: This model was introduced in the Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer paper. This is a popular model for sequence-to-sequence modeling tasks. You can find the latest performance and convergence on our model card.
- ViT: The Vision Transformer was introduced in the An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale paper. This is a popular computer vision model used for image classification, but is also a foundational model for GenAI applications such as Multimodal Language Models. The ViT model also showcases NVIDIA DALI data pipelines. This is particularly helpful for smaller models where data pipelines tend to be the bottleneck.
- Imagen: Text-to-image generative diffusion model based on Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding paper. You can find the latest performance and convergence on our model card.
- PaliGemma: PaliGemma is a vision language model (VLM). These models are well-suited for a variety of tasks that require visually-situated text understanding and object localization.

Nightly Containers

In addition to projects, JAX Toolbox includes nightly containers for libraries across the JAX ecosystem.

Container	Type	Image URI
jax	-	ghcr.io/nvidia/jax:jax
paxml	LLM framework	ghcr.io/nvidia/jax:pax
t5x	LLM framework	ghcr.io/nvidia/jax:t5x
levanter	LLM framework	ghcr.io/nvidia/jax:levanter
maxtext	LLM framework	ghcr.io/nvidia/jax:maxtext
triton	JAX extension	ghcr.io/nvidia/jax:triton
equinox	layer library	ghcr.io/nvidia/jax:equinox
grok	model	ghcr.io/nvidia/jax:grok

Known Issues

- A JAX eigensolver unit test is failing due to a flaw in the test itself using a hard-coded workspace size. This should not impact real applications, where the workspace size is computed by the cuSolver API.

- There is a known convergence issue with GPT 126M when Flash Attention is enabled.
- There is a known divergence issue with Flash Attention in PAXML when enabled natively via XLA without Transformer Engine. If training models with TransformerEngine disabled/absent, NVIDIA recommends running without Flash Attention by passing `--xla_gpu_enable_cudnn_fmha=false`.
- LLaMA fine-tuning via Maxtext does not currently support FP8.
- Transformer Engine is currently not supported with GLaM models. Future releases will include TE support with GLaM.
- There is a known issue in the Maxtext container where failures may be observed if `--xla_gpu_enable_custom_fusions` and `--xla_gpu_enable_address_computation_fusion` flags are enabled. It is advised to disable these two flags by adding `--xla_gpu_enable_custom_fusions=false` and `--xla_gpu_enable_address_computation_fusion=false` to the XLA_FLAGS environment variable.

Chapter 29. JAX Release 24.09

There is no JAX container in DLFW release 24.09.

Chapter 30. JAX Release 24.08

There is no JAX container in DLFW release 24.08.

Chapter 31. JAX Release 24.07

There is no JAX container in DLFW release 24.07.

Chapter 32. JAX Release 24.06

There is no JAX container in DLFW release 24.06.

Chapter 33. JAX Release 24.05

There is no JAX container in DLFW release 24.05.

Chapter 34. JAX Release 24.04

The NVIDIA container image for JAX, release 24.04 is available on NGC.

Contents of the JAX container

This container image contains the complete source for the following software:

- JAX: /opt/jax
- XLA: /opt/xla
- Flax: /opt/flax
- TransformerEngine: /opt/transformer-engine

The JAX runtime package jaxlib is prebuilt and installed in the default Python environment (/usr/local/lib/python3.10/dist-packages/jaxlib) in the container image.

The container also includes the following:

- JAX v0.4.26
- Ubuntu 22.04.2 LTS with Python 3.10
- NVIDIA CUDA® 12.4.1
- NVIDIA cuBLAS 12.4.5.8
- NVIDIA cuDNN 9.1.0.70
- NVIDIA NCCL 2.21.5
- rdma-core 39.0
- NVIDIA HPC-X 2.18
- Nsight Compute 2024.1.1.0
- Nsight Systems 2024.2.1.106
- TransformerEngine 1.5

Driver Requirements

Release 23.10 is based on CUDA 12.4.1, which requires NVIDIA Driver release 545 or later. However, if you are running on a data center GPU (for example, T4 or any other data center GPU), you can use NVIDIA driver release 470.57 (or later R470), 525.85 (or later R525), 535.86 (or later R535), or 545.23 (or later R545).

The CUDA driver's compatibility package only supports particular drivers. Thus, users should upgrade from all R418, R440, R450, R460, R510, and R520 drivers, which are not forward-compatible with CUDA 12.3. For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

Key Features and Enhancements

This JAX release includes the following key features and enhancements.

- JAX container images in version 24.04 are based on jaxlib==0.4.26.
- PAXML and Maxtext containers are available in version 24.04.

JAX Toolbox

The JAX Toolbox projects focus on achieving the best performance and convergence on NVIDIA Hopper and Ampere tensor cores and provide the latest deep learning models and scripts for training and fine-tuning. These examples are tested against a nightly CI as well as each NGC container release to ensure consistent accuracy and performance over time.

Projects

- GPT: Decoder-only LLM used in many modern text applications. You can also find the latest performance and convergence on the model card.
- LLaMA-2: Decoder-only LLM ranging in scale from 7B to 70B parameters. You can find the latest performance and convergence on our model card.
  - This release also introduces LoRA support for fine-tuning of LLaMA models. We include an example script that finetunes LLama 2 7B on BoolQ and achieves significant metric improvements. You can find the convergence and performance results on the above model card.
  - LLaMA-2 recipes are also available in maxtext. You can find instructions and performance results on our model card.
- MoE: GLaM style MoE with configs up to 1.14T parameter scale. You can find the latest information on our model card.
- T5: This model was introduced in the Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer paper. This is a popular model for sequence-to-sequence modeling tasks. You can find the latest performance and convergence on our model card.
- ViT: The Vision Transformer was introduced in the An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale paper. This is a popular computer vision model used for image classification, but is also a foundational model for GenAI applications such as Multimodal Language Models. The ViT model also showcases NVIDIA DALI data pipelines. This is particularly helpful for smaller models where data pipelines tend to be the bottleneck.

- Imagen: Text-to-image generative diffusion model based on Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding paper. You can find the latest performance and convergence on our model card.

Nightly Containers

In addition to projects, JAX Toolbox includes nightly containers for libraries across the JAX ecosystem.

Container	Type	Image URI
jax	-	ghcr.io/nvidia/jax:jax
paxml	LLM framework	ghcr.io/nvidia/jax:pax
t5x	LLM framework	ghcr.io/nvidia/jax:t5x
levanter	LLM framework	ghcr.io/nvidia/jax:levanter
maxtext	LLM framework	ghcr.io/nvidia/jax:maxtext
triton	JAX extension	ghcr.io/nvidia/jax:triton
equinox	layer library	ghcr.io/nvidia/jax:equinox
grok	model	ghcr.io/nvidia/jax:grok

NVIDIA Tx5 and Paxml Container Versions

The following table shows what versions of CUDA and JAX are supported in each of the NVIDIA containers for JAX.

The JAX version in the Paxml and T5x containers are development versions. See `/opt/jax-source` and `/opt/xla-source` for the version used in each container.

Known Issues

- A JAX eigensolver unit test is failing due to a flaw in the test itself using a hard-coded workspace size. This should not impact real applications, where the workspace size is computed by the cuSolver API.
- There is a known convergence issue with GPT 126M when Flash Attention is enabled.
- There is a known divergence issue with Flash Attention in PAXML when enabled natively via XLA without Transformer Engine. If training models with TransformerEngine disabled/absent, it is recommended to run without Flash Attention by passing `--xla_gpu_enable_cudnn_fmha=false`.
- LLaMA fine-tuning via PAXML does not currently support FP8.
- MoE does not currently support TransformerEngine.
- There is a known CVE that affects the amd64 Paxml container related to TensorFlow 2.9.x due to pinning TensorFlow to 2.9.x in Paxml. We will fix these in an upcoming release.

- CVE-2023-25668

- There is a known issue in the Maxtext container where failures may be observed if `--xla_gpu_enable_custom_fusions` and `--xla_gpu_enable_address_computation_fusion` flags are enabled. It is advised to disable these two flags by adding `--xla_gpu_enable_custom_fusions=false` and `--xla_gpu_enable_address_computation_fusion=false` to the XLA_FLAGS environment variable.

Chapter 35. JAX Release 24.03

There is no JAX container in DLFW release 24.03.

Chapter 36. JAX Release 24.02

There is no JAX container in DLFW release 24.02.

Chapter 37. JAX Release 23.12

There is no JAX container in DLFW release 23.12.

Chapter 38. JAX Release 23.10

The NVIDIA container image for JAX, release 23.10 is available on NGC.

Contents of the JAX container

This container image contains the complete source of the version of JAX in `/opt/jax-source`. It is prebuilt and installed within the container image.

The container also includes the following:

- Ubuntu 22.04.2 LTS with Python 3.10.12
- NVIDIA CUDA® 12.2.0
- NVIDIA cuBLAS 12.2.5.6
- NVIDIA cuDNN 8.9.5
- NVIDIA NCCL 2.19.3
- rdma-core 39.0
- Mellanox OFED 5.9-0.5.6.0
- Nsight Compute 2023.2.2.3
- Nsight Systems 2023.3.1.92
- NVIDIA DALI® 1.30.0 in T5x containers
- TransformerEngine
  - 1.0.0.dev0+1f4c397 in Paxml container
  - 0.13.0+511241c in T5x container

Driver Requirements

Release 23.10 is based on CUDA 12.2.0, which requires NVIDIA Driver release 535 or later. However, if you are running on a data center GPU (for example, T4 or any other data center GPU), you can use NVIDIA driver release 450.51 (or later R450), 470.57 (or later R470), 510.47 (or later R510), or 525.85 (or later R525), or 535.86 (or later R535).

The CUDA driver's compatibility package only supports particular drivers. Thus, users should upgrade from all R418, R440, R460, and R520 drivers, which are not forward-compatible with CUDA 12.2. For a complete list of supported drivers, see the CUDA

Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

GPU Requirements

Release 23.10 supports CUDA compute capability 6.0 and later. This corresponds to GPUs in the NVIDIA Pascal, NVIDIA Volta™, NVIDIA Turing™, NVIDIA Ampere architecture, and NVIDIA Hopper™ architecture families. For a list of GPUs to which this compute capability corresponds, see CUDA GPUs. For additional support details, see Deep Learning Frameworks Support Matrix.

Key Features and Enhancements

This JAX release includes the following key features and enhancements.

- JAX container images in version 23.10 are based on jaxlib==0.4.17.
- T5X and PAXML containers are available in version 23.10.
- PAXML arm64 containers are available: nvcr.io/nvidia/jax:23.10-paxml-py3-arm64.
- PAXML ships with Hopper support for FP8 training.

Announcements

- Transformer Engine is a library for accelerating Transformer models on NVIDIA GPUs. It includes support for 8-bit floating point (FP8) precision on Hopper GPUs which provides better training and inference performance with lower memory utilization. Transformer Engine also includes a collection of highly optimized modules for popular Transformer architectures and an automatic mixed precision-like API that can be used seamlessly with your JAX code.

NVIDIA JAX Container Versions

The following table shows what versions of CUDA and JAX are supported in each of the NVIDIA containers for JAX.

Container Version	CUDA Toolkit	JAX
23.10	NVIDIA CUDA 12.2.0	0.4.17
23.08	NVIDIA CUDA 12.1.1	0.4.14

NVIDIA Tx5 and Paxml Container Versions

The following table shows what versions of CUDA and JAX are supported in each of the NVIDIA containers for JAX.

	Container Version	CUDA Toolkit	JAX
Paxml	23.10	NVIDIA CUDA 12.2.2	0.4.17.dev20231010

	Container Version	CUDA Toolkit	JAX
	23.08	NVIDIA CUDA 12.1.1	0.4.14 (579808d98)
T5x	23.10	NVIDIA CUDA 12.2.2	0.4.17.dev20231010
	23.08	NVIDIA CUDA 12.1.1	0.4.14 (603eeb190)

The JAX version in the Paxml and T5x containers are development versions. See `/opt/jax-source` and `/opt/xla-source` for the version used in each container.

Inspecting Source code in NVIDIA T5x and Paxml Containers

If you would like to inspect the pax's source code (paxml and praxis) to learn more about what is being run, you can do so by inspecting the source within the `nvcr.io/nvidia/jax:23.10-paxml-py3` container. Their locations within the container are:

- Paxml: `/opt/paxml`
- Praxis: `/opt/praxis`

Similarly, for t5x's source code in `nvcr.io/nvidia/jax:23.10-t5x-py3`:

- t5x: `/opt/t5x`

JAX Toolbox Examples

The JAX Toolbox examples focus on achieving the best performance and convergence from NVIDIA Hopper and NVIDIA Ampere architecture tensor cores by using the latest deep learning example networks and model scripts for training.

These examples are tested against a nightly CI as well as each NGC container release to ensure consistent accuracy and performance over time.

- GPT model: This is a decoder-only LLM architecture used in many modern text applications.This model script is available on Github. You can also find the latest performance and secured convergence on the model card on Github.
- T5 model: This model was introduced in the Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer paper. This is a popular model for sequence-to-sequence modeling tasks involving text.This model script is available on Github. You can also find the latest performance and secured convergence on the model card on Github.
- ViT model: The Vision Transformer was introduced in the An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale paper. This is a popular computer vision model used for image classification, but is also a foundational model for GenAI applications such as Multimodal Language Models. The ViT model also showcases NVIDIA DALI data pipelines. This is particularly helpful for smaller models where data pipelines tend to be the bottleneck. The model script is available on Github. You can also find the latest performance and secured convergence on the model card on Github.

Known Issues

- Pipeline parallelism is not supported with NVIDIA Transformer Engine enabled in the Paxml container.
- There could be random failures when running JAX in single-process with multi-GPU. This is a known issue due to the XLA version in the 23.10 containers. The issue will be resolved in the next release and the recommendation is to run with multi-processing enabled (1 GPU per process).
- There is a known degradation in accuracy in the Paxml container when training with enabling dropout in transformer layers from TransformerEngine. This issue will be resolved in a future container release.
- Performance degradation:
  - There is a 10% performance regression on Paxml's 126M parameter model, trained with BF16 on A100s, when NVIDIA Transformer Engine is disabled. This will be fixed in a future release.
  - There is a 1-2% performance regression in the 23.10 Paxml container compared to the 23.08 Paxml container on A100s.
  - ViT has an 11% pretraining performance regression in the T5X container compared to the initial JAX-Toolbox release container.
  - The T5 model in T5x has a 6-9% performance regression.
- The JAX, T5x, and Paxml containers are an early release and have some differences from the NGC TensorFlow container that will be addressed in future releases. Some differences include CUDA minor version, support for 3rd party network devices via HPC-X, and versions of CTK/cuDNN/NCCL and so on. Some packages that do not currently ship include:
  - JupyterLab including Jupyter-TensorBoard
  - NVIDIA TensorRT
  - TensorBoard
  - NVIDIA HPC-X with UCX and OpenMPI
  - OpenMPI
  - NVIDIA RAPIDS
  - cuTENSOR
  - GDRCopy
- The `jax.experimental.sparse` sparse matrix package currently produces incorrect matrix multiplication results on NVIDIA Ampere architecture/Hopper GPUs. Volta GPUs appear to not be affected. See detailed issue here.
- There are known CVEs that affect the amd64 Paxml container related to TensorFlow 2.9.x due to pinning TensorFlow to 2.9.x in Paxml and Lingvo. We will fix these in an upcoming release. The known CVEs are:

- CVE-2023-25668
- CVE-2023-25658
- CVE-2023-25663
- CVE-2023-25664
- CVE-2023-25664
- CVE-2023-25672
- CVE-2023-25674
- CVE-2023-25660
- CVE-2023-27579
- CVE-2023-25671
- CVE-2023-25659
- CVE-2023-25662
- CVE-2023-25675
- CVE-2023-25801
- CVE-2023-25670
- CVE-2023-25669
- CVE-2023-25665
- CVE-2023-25673
- CVE-2023-25666

Chapter 39. JAX Release 23.08

The NVIDIA container image for JAX, release 23.08 is available on NGC.

Contents of the JAX container

This container image contains the complete source of the version of JAX in `/opt/jax-source`. It is prebuilt and installed within the container image.

The container also includes the following:

- Ubuntu 22.04.2 LTS with Python 3.10.12
- NVIDIA CUDA® 12.1.1
- NVIDIA cuBLAS 12.2.5
- NVIDIA cuDNN 8.9.4
- NVIDIA NCCL 2.18.3
- rdma-core 39.0
- Mellanox OFED 5.9-0.5.6.0
- Nsight Compute 2023.2.0
- Nsight Systems 2023.2.0
- NVIDIA DALI® 1.28.0 in Paxml and T5x containers
- TransformerEngine 0.13.0.dev0+03202c3 in Paxml and T5x containers

Driver Requirements

Release 23.08 is based on CUDA 12.1.1, which requires NVIDIA Driver release 525.60.13 or later. However, if you are running on a data center GPU (for example, T4 or any other data center GPU), you can use NVIDIA driver release 450.51 (or later R450), 470.57 (or later R470), 510.47 (or later R510), or 525.85 (or later R525).

The CUDA driver's compatibility package only supports particular drivers. Thus, users should upgrade from all R418, R440, R460, and R520 drivers, which are not forward-compatible with CUDA 12.1. For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

GPU Requirements

Release 23.08 supports CUDA compute capability 6.0 and later. This corresponds to GPUs in the NVIDIA Pascal, NVIDIA Volta™, NVIDIA Turing™, NVIDIA Ampere architecture, and NVIDIA Hopper™ architecture families. For a list of GPUs to which this compute capability corresponds, see CUDA GPUs. For additional support details, see Deep Learning Frameworks Support Matrix.

Key Features and Enhancements

This JAX release includes the following key features and enhancements.

- JAX container images in version 23.08 are based on jaxlib==0.4.14.
- T5X and PAXML containers are available in version 23.08.

Announcements

- Transformer Engine is a library for accelerating Transformer models on NVIDIA GPUs. It includes support for 8-bit floating point (FP8) precision on Hopper GPUs which provides better training and inference performance with lower memory utilization. Transformer Engine also includes a collection of highly optimized modules for popular Transformer architectures and an automatic mixed precision-like API that can be used seamlessly with your JAX code.

NVIDIA JAX Container Versions

The following table shows what versions of CUDA and JAX are supported in each of the NVIDIA containers for JAX.

Container Version	CUDA Toolkit	JAX
23.08	NVIDIA CUDA 12.1.1	0.4.14

NVIDIA Tx5 and Paxml Container Versions

The following table shows what versions of CUDA and JAX are supported in each of the NVIDIA containers for JAX.

	Container Version	CUDA Toolkit	JAX
Paxml	23.08	NVIDIA CUDA 12.1.1	0.4.14 (579808d98)
T5x	23.08	NVIDIA CUDA 12.1.1	0.4.14 (603eeb190)

The JAX version in the Paxml and T5x containers are development versions. See `/opt/jax-source` and `/opt/xla-source` for the version used in each container.

Inspecting Source code in NVIDIA T5x and Paxml Containers

If you would like to inspect the pax's source code (paxml and praxis) to learn more about what is being run, you can do so by inspecting the source within the `nvcr.io/nvidia/jax:23.08-paxml-py3` container. Their locations within the container are:

- Paxml: `/opt/paxml`
- Praxis: `/opt/praxis`

Similarly, for t5x's source code in `nvcr.io/nvidia/jax:23.08-t5x-py3`:

- t5x: `/opt/t5x`

JAX Toolbox Examples

The JAX Toolbox examples focus on achieving the best performance and convergence from NVIDIA Hopper and NVIDIA Ampere architecture tensor cores by using the latest deep learning example networks and model scripts for training.

These examples are tested against a nightly CI as well as each NGC container release to ensure consistent accuracy and performance over time.

- GPT model: This is a decoder-only LLM architecture used in many modern text applications.This model script is available on Github. You can also find the latest performance and secured convergence on the model card on Github.
- T5 model: This model was introduced in the Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer paper. This is a popular model for sequence-to-sequence modeling tasks involving text.This model script is available on Github. You can also find the latest performance and secured convergence on the model card on Github.

Known Issues

- Pipeline parallelism is not supported with NVIDIA Transformer Engine enabled in the Paxml container.
- There is a 15% performance regression on Paxml's 126M parameter model, trained with BF16 on A100s, when NVIDIA Transformer Engine is disabled. This will be fixed in a future release.
- The Paxml container (`nvcr.io/nvidia/jax:23.08-paxml-py3`) does not fully support Hopper yet. Future releases will add Hopper support.
- There is a known sporadic NCCL crash that happens when using the T5x container at node counts greater than or equal to 32 nodes. We will fix this in a future release. The issue is tracked here.
- The JAX, T5x, and Paxml containers are an early release and have some differences from the NGC TensorFlow container that will be addressed in future releases. Some differences include CUDA minor version, support for 3rd party network devices via

HPC-X, and versions of CTK/cuDNN/NCCL and so on. Some packages that do not currently ship include:

- JupyterLab including Jupyter-TensorBoard
- NVIDIA TensorRT
- TensorBoard
- NVIDIA HPC-X with UCX and OpenMPI
- OpenMPI
- NVIDIA RAPIDS
- cuTENSOR
- GDRCopy

- The `jax.experimental.sparse` sparse matrix package currently produces incorrect matrix multiplication results on NVIDIA Ampere architecture/Hopper GPUs. Volta GPUs appear to not be affected. See detailed issue here.
- There are known CVEs that affect the Paxml container related to TensorFlow 2.9.x due to pinning TensorFlow to 2.9.x in Paxml and Lingvo. We will fix these in an upcoming release. The known CVEs are:
  - CVE-2023-25668
  - CVE-2023-25658
  - CVE-2023-25663
  - CVE-2023-25664
  - CVE-2023-25664
  - CVE-2023-25672
  - CVE-2023-25674
  - CVE-2023-25660
  - CVE-2023-27579
  - CVE-2023-25671
  - CVE-2023-25659
  - CVE-2023-25662
  - CVE-2023-25675
  - CVE-2023-25801
  - CVE-2023-25670
  - CVE-2023-25669
  - CVE-2023-25665
  - CVE-2023-25673
  - CVE-2023-25666

Notice

This document is provided for information purposes only and shall not be regarded as a warranty of a certain functionality, condition, or quality of a product. NVIDIA Corporation ("NVIDIA") makes no representations or warranties, expressed or implied, as to the accuracy or completeness of the information contained in this document and assumes no responsibility for any errors contained herein. NVIDIA shall have no liability for the consequences or use of such information or for any infringement of patents or other rights of third parties that may result from its use. This document is not a commitment to develop, release, or deliver any Material (defined below), code, or functionality.

NVIDIA reserves the right to make corrections, modifications, enhancements, improvements, and any other changes to this document, at any time without notice.

Customer should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

NVIDIA products are sold subject to the NVIDIA standard terms and conditions of sale supplied at the time of order acknowledgement, unless otherwise agreed in an individual sales agreement signed by authorized representatives of NVIDIA and customer ("Terms of Sale"). NVIDIA hereby expressly objects to applying any customer general terms and conditions with regards to the purchase of the NVIDIA product referenced in this document. No contractual obligations are formed either directly or indirectly by this document.

NVIDIA products are not designed, authorized, or warranted to be suitable for use in medical, military, aircraft, space, or life support equipment, nor in applications where failure or malfunction of the NVIDIA product can reasonably be expected to result in personal injury, death, or property or environmental damage. NVIDIA accepts no liability for inclusion and/or use of NVIDIA products in such equipment or applications and therefore such inclusion and/or use is at customer's own risk.

NVIDIA makes no representation or warranty that products based on this document will be suitable for any specified use. Testing of all parameters of each product is not necessarily performed by NVIDIA. It is customer's sole responsibility to evaluate and determine the applicability of any information contained in this document, ensure the product is suitable and fit for the application planned by customer, and perform the necessary testing for the application in order to avoid a default of the application or the product. Weaknesses in customer's product designs may affect the quality and reliability of the NVIDIA product and may result in additional or different conditions and/or requirements beyond those contained in this document. NVIDIA accepts no liability related to any default, damage, costs, or problem which may be based on or attributable to: (i) the use of the NVIDIA product in any manner that is contrary to this document or (ii) customer product designs.

No license, either expressed or implied, is granted under any NVIDIA patent right, copyright, or other NVIDIA intellectual property right under this document. Information published by NVIDIA regarding third-party products or services does not constitute a license from NVIDIA to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property rights of the third party, or a license from NVIDIA under the patents or other intellectual property rights of NVIDIA.

Reproduction of information in this document is permissible only if approved in advance by NVIDIA in writing, reproduced without alteration and in full compliance with all applicable export laws and regulations, and accompanied by all associated conditions, limitations, and notices.

THIS DOCUMENT AND ALL NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, "MATERIALS") ARE BEING PROVIDED "AS IS." NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL NVIDIA BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF NVIDIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Notwithstanding any damages that customer might incur for any reason whatsoever, NVIDIA's aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms of Sale for the product.

HDMI

HDMI, the HDMI logo, and High-Definition Multimedia Interface are trademarks or registered trademarks of HDMI Licensing LLC.

OpenCL

OpenCL is a trademark of Apple Inc. used under license to the Khronos Group Inc.

NVIDIA Corporation | 2788 San Tomas Expressway, Santa Clara, CA 95051
https://www.nvidia.com

Trademarks

NVIDIA, the NVIDIA logo, and cuBLAS, CUDA, DALI, DGX, DGX-1, DGX-2, DGX Station, DLProf, Jetson, Kepler, Maxwell, NCCL, Nsight Compute, Nsight Systems, NvCaffe, PerfWorks, Pascal, SDK Manager, Tegra, TensorRT, Triton Inference Server, Tesla, TF-TRT, and Volta are trademarks and/or registered trademarks of NVIDIA Corporation in the U.S. and other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

Copyright

© 2024-2026 NVIDIA Corporation & Affiliates. All rights reserved.

NVIDIA Corporation | 2788 San Tomas Expressway, Santa Clara, CA 95051
https://www.nvidia.com