JAX

Release Notes

RN-11517-001_v24.03 | April 2024

Table of Contents

Chapter 1. JAX Overview....1
Chapter 2. Pulling A Container....2
Chapter 3. Running JAX....3
Chapter 4. JAX Release 24.03....5
Chapter 5. JAX Release 24.02....6
Chapter 6. JAX Release 23.12....7
Chapter 7. JAX Release 23.10....8
Chapter 8. JAX Release 23.08....13

Chapter 1. JAX Overview

JAX is a framework for high-performance numerical computing and machine learning research. It includes 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 NLP, Computer Vision, Multimodality, physics-based simulations, reinforcement learning, drug discovery, and neural rendering.

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 accelerating ETL (DALI) and Training (cuDNN, NCCL).

For working with neural networks, the JAX NGC Container includes Flax, a neural network library with support for common deep learning models, layers and optimizers. We also include containers for training GPT models with a Paxml container, and training T5 and ViT models with our T5x container. You can use the JAX, Paxml, or T5x containers for your deep learning workloads or install your own favorite libraries on top of them.

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 Docker engine 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).

Note: Starting in Docker 19.03, complete the steps below.

The method implemented in your system depends on the DGX OS version that you installed (for DGX systems), the NGC Cloud Image that was provided by a Cloud Service Provider, or the software that you installed to prepare to run NGC containers on TITAN PCs, Quadro PCs, or NVIDIA Virtual GPUs (vGPUs).

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 container, where 23.10 is the container version. For example, 23.10 for October 2023 release:

   ```
   docker pull nvcr.io/nvidia/jax:23.10-py3
   ```

   To pull the latest Paxml container:

```
docker pull nvcr.io/nvidia/jax:23.10-paxml-py3
```

To pull the latest T5x container:

```
docker pull nvcr.io/nvidia/jax:23.10-t5x-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, where 23.10 is the container version:

     ```
     docker run --gpus all -it --rm -v local_dir:container_dir nvcr.io/nvidia/jax:23.10-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 by issuing one of the following commands:

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

     in the command line to

     ```
     docker run --gpus all
     ```

     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 24.03

There is no JAX container in DLFW release 24.03.

Chapter 5. JAX Release 24.02

There is no JAX container in DLFW release 24.02.

Chapter 6. JAX Release 23.12

There is no JAX container in DLFW release 23.12.

Chapter 7. 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 8. 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

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