NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet

Release Notes
# Table of Contents

Chapter 1. NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet
Overview............................................................................................................................1

Chapter 2. Pulling A Container............................................................................................ 2

Chapter 3. Running NVIDIA Optimized Deep Learning Framework, powered by Apache
MXNet ................................................................................................................................3

Chapter 4. Release 20.07..................................................................................................... 5

Chapter 5. Release 20.06...................................................................................................10

Chapter 6. Release 20.03...................................................................................................15

Chapter 7. Release 20.02...................................................................................................19

Chapter 8. Release 20.01...................................................................................................23

Chapter 9. Release 19.12...................................................................................................27

Chapter 10. Release 19.11.................................................................................................31

Chapter 11. Release 19.10.................................................................................................35

Chapter 12. Release 19.09.................................................................................................39

Chapter 13. Release 19.08.................................................................................................43

Chapter 14. Release 19.07.................................................................................................47

Chapter 15. Release 19.06.................................................................................................50

Chapter 16. Release 19.05.................................................................................................53

Chapter 17. Release 19.04.................................................................................................57

Chapter 18. Release 19.03.................................................................................................60

Chapter 19. Release 19.02.................................................................................................63

Chapter 20. Release 19.01.................................................................................................66

Chapter 21. Release 18.12.................................................................................................69

Chapter 22. Release 18.11.................................................................................................71

Chapter 23. Release 18.10.................................................................................................74

Chapter 24. Release 18.09.................................................................................................76

Chapter 25. Release 18.08.................................................................................................78

Chapter 26. Release 18.07.................................................................................................80

Chapter 27. Release 18.06.................................................................................................82
Chapter 1. NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet

Overview

The NVIDIA Deep Learning SDK accelerates widely-used deep learning frameworks such as Apache MXNet.

Apache MXNet is designed for both efficiency and flexibility. It allows you to mix the flavors of symbolic programming and imperative programming to maximize efficiency and productivity.

In its core is a dynamic dependency scheduler that automatically parallelizes both symbolic and imperative operations on the fly. A graph optimization layer on top of that makes symbolic execution fast and memory efficient. The library is portable and lightweight, and it scales to multiple GPUs and multiple machines.

Apache MXNet is also more than a deep learning project. It is also a collection of blueprints and guidelines for building deep learning systems and interesting insights of deep learning systems for hackers.

See `/workspace/README.md` inside the container for information on customizing your Optimized Deep Learning Framework image. For more information about Apache MXNet, including tutorials, documentation, and examples, see:

- Apache MXNet website
- Apache MXNet project

This document describes the key features, software enhancements and improvements, any known issues, and how to run this container.
Chapter 2. Pulling A Container

About this task

Before you can pull a container from the NGC container registry, you must have Docker installed. For DGX users, this is explained in Preparing to use NVIDIA Containers Getting Started Guide.

For users other than DGX, follow the NVIDIA® GPU Cloud™ (NGC) container registry installation documentation based on your platform.

You must also have access and be logged into the NGC container registry as explained in the NGC Getting Started Guide.

The deep learning frameworks are stored in the following repository where you can find the NGC Docker containers.

nvcr.io/nvidia

The deep learning framework containers are stored in the nvcr.io/nvidia repository.
Chapter 3. Running NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet

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 the Running A Container chapter in the NVIDIA Containers And Frameworks User Guide and specify the registry, repository, and tags.

About this task

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

- 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 using NVIDIA_VISIBLE_DEVICES environment variable). Starting in Docker 19.03, follow the steps as outlined below. For more information, refer to the nvidia-docker documentation here.

The method implemented in your system depends on the DGX OS version installed (for DGX systems), the specific NGC Cloud Image provided by a Cloud Service Provider, or the software that you have installed in preparation for running NGC containers on TITAN PCs, Quadro PCs, or 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 container.

   ```
   docker pull nvcr.io/nvidia/mxnet:20.07-py3
   ```
2. Open a command prompt and paste the pull command. The pulling of the container image begins. Ensure the pull completes successfully before proceeding to the next step.

3. Run the container image. A typical command to launch the container is:

   If you have Docker 19.03 or later, a typical command to launch the container is:
   ```bash
docker run --gpus all -it --rm -v local_dir:container_dir nvcr.io/nvidia/mxnet:<xx.xx>-py3
   ```

   If you have Docker 19.02 or earlier, a typical command to launch the container is:
   ```bash
   nvidia-docker run -it --rm -v local_dir:container_dir nvcr.io/nvidia/mxnet:<xx.xx>-py3
   ```

   The NVIDIA Optimized Deep Learning Framework is run simply by importing it as a Python module:
   ```bash
   $ python
   Python 3.5.2 (default, Nov 23 2017, 16:37:01)
   [GCC 5.4.0 20160609] on linux
   >>> import mxnet as mx
   >>> a = mx.nd.ones((2,3), mx.gpu())
   >>> print((a*2).asnumpy())
   [[ 2.  2.  2.]
    [ 2.  2.  2.]]
   ```

   You might want to pull in data and model descriptions from locations outside the container for use by the Optimized Deep Learning Framework or save results to locations outside the container. To accomplish this, the easiest method is to mount one or more host directories as **Docker data volumes**.

   **Note:** In order to share data between ranks, NVIDIA® Collective Communications Library (NCCL) may require shared system memory for IPC and pinned (page-locked) system memory resources. The operating system’s limits on these resources may need to be increased accordingly. Refer to your system’s documentation for details.

   In particular, Docker® containers default to limited shared and pinned memory resources. When using NCCL inside a container, it is recommended that you increase these resources by issuing:
   ```bash
   --shm-size=1g --ulimit memlock=-1
   ```
   in the command line to:
   ```bash
docker run --gpus all
   ```
Chapter 4. Release 20.07

The container image for NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 20.07, is available on NGC.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the NVIDIA Optimized Deep Learning Framework, which is based upon Apache MXNet version 1.6.0. It is pre-built and installed to the Python path.

The container also includes the following:

- Ubuntu 18.04 including Python 3.6
- NVIDIA CUDA 11.0.194 including cuBLAS 11.1.0
- NVIDIA cuDNN 8.0.1
- NVIDIA NCCL 2.7.6 (optimized for NVLink™)
- ONNX exporter 0.1 for CNN classification models
  
  **Note:** The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- Amazon Labs Sockeye sequence-to-sequence framework 1.18.99 (for machine translation)
- MLNX OFED
- OpenMPI 3.1.6
- Horovod 0.19.1 with enhancements
- Nsight Compute 2020.1.1.8
- Nsight Systems 2020.3.2.6
- GluonCV Toolkit 0.5
- GluonNLP Toolkit 0.8.1
- TensorRT 7.1.3
- DALI 0.23
- Tensor Core optimized example:
  - ResNet-50 v1.5
- Jupyter and JupyterLab:
  - Jupyter Client 6.0.0
  - Jupyter Core 4.6.1
  - Jupyter Notebook 6.0.3
  - JupyterLab 1.2.6
  - JupyterLab Server 1.0.6
  - Jupyter-TensorBoard

**Driver Requirements**

Release 20.07 is based on NVIDIA CUDA 11.0.194, which requires NVIDIA Driver release 450 or later. However, if you are running on Tesla (for example, T4 or any other Tesla board), you may use NVIDIA driver release 418.xx or 440.30. The CUDA driver’s compatibility package only supports particular drivers. For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

**GPU Requirements**

Release 20.07 supports CUDA compute capability 6.0 and higher. This corresponds to GPUs in the NVIDIA Pascal, Volta, Turing, and Ampere Architecture GPU families. Specifically, for a list of GPUs that this compute capability corresponds to, see CUDA GPUs. For additional support details, see Deep Learning Frameworks Support Matrix.

**Key Features and Enhancements**

This Optimized Deep Learning Framework release includes the following key features and enhancements.

- NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 20.07 is based on Apache MXNet 1.6.0
- Latest version of NVIDIA CUDA 11.0.194 including cuBLAS 11.1.0
- Latest version of DALI 0.23
- Latest version of NVIDIA NCCL 2.7.6
- Latest version of Nsight Compute 2020.1.1.8
- Latest version of Nsight Systems 2020.3.2.6
- Latest version of TensorRT 7.1.3
- Ubuntu 18.04 with June 2020 updates

**Announcements**

- Deep learning framework containers 19.11 and later include experimental support for Singularity v3.0.
NVIDIA Optimized Deep Learning Framework Container Versions

The following table shows what versions of Ubuntu, CUDA, Apache MXNet, and TensorRT are supported in each of the NVIDIA containers for the Optimized Deep Learning Framework. For older container versions, refer to the Frameworks Support Matrix.

<table>
<thead>
<tr>
<th>Container Version</th>
<th>Ubuntu</th>
<th>CUDA Toolkit</th>
<th>Apache MXNet</th>
<th>TensorRT</th>
</tr>
</thead>
<tbody>
<tr>
<td>20.07</td>
<td>18.04</td>
<td>NVIDIA CUDA 11.0.194</td>
<td>1.6.0</td>
<td>TensorRT 7.1.3</td>
</tr>
<tr>
<td>20.06</td>
<td></td>
<td>NVIDIA CUDA 11.0.167</td>
<td>1.6.0</td>
<td>TensorRT 7.1.2</td>
</tr>
<tr>
<td>20.03</td>
<td></td>
<td>NVIDIA CUDA 10.2.89</td>
<td>1.6.0.rc2</td>
<td>TensorRT 7.0.0</td>
</tr>
<tr>
<td>20.02</td>
<td></td>
<td></td>
<td>1.5.0.rc2</td>
<td>TensorRT 6.0.1</td>
</tr>
<tr>
<td>20.01</td>
<td></td>
<td></td>
<td>1.5.1 commit c98184806 from September 4, 2019</td>
<td>TensorRT 6.0.1</td>
</tr>
<tr>
<td>19.12</td>
<td></td>
<td></td>
<td>1.5.0 commit 006486af3 from August 28, 2019</td>
<td>TensorRT 5.1.5</td>
</tr>
<tr>
<td>19.11</td>
<td></td>
<td></td>
<td>1.5.0 commit 75a9e187d from June 27, 2019</td>
<td>TensorRT 5.1.5</td>
</tr>
<tr>
<td>19.10</td>
<td></td>
<td></td>
<td>1.5.0 commit 75a9e187d from June 27, 2019</td>
<td>TensorRT 5.1.5</td>
</tr>
<tr>
<td>19.09</td>
<td></td>
<td></td>
<td>1.5.0 commit 75a9e187d from June 27, 2019</td>
<td>TensorRT 5.1.5</td>
</tr>
<tr>
<td>19.08</td>
<td></td>
<td></td>
<td>1.5.0 commit 75a9e187d from June 27, 2019</td>
<td>TensorRT 5.1.5</td>
</tr>
</tbody>
</table>

Tensor Core Examples

The tensor core examples provided in GitHub and NVIDIA GPU Cloud [NGC] focus on achieving the best performance and convergence from NVIDIA Volta tensor cores by using the latest deep learning example networks and model scripts for training.

Each example model trains with mixed precision Tensor Cores on Volta and Turing, therefore you can get results much faster than training without tensor cores. This model is tested against each NGC monthly container release to ensure consistent accuracy and performance over time. This container includes the following tensor core examples.

- The ResNet50 v1.5 model is a slightly modified version of the original ResNet50 v1 model that trains to a greater accuracy. This model script is available on GitHub as well as NVIDIA GPU Cloud [NGC].
**Automatic Mixed Precision (AMP)**

Training deep learning networks is a very computationally intensive task. Novel model architectures tend to have an increasing number of layers and parameters, which slows down training. Fortunately, new generations of training hardware as well as software optimizations make training these new models a feasible task.

Most of the hardware and software training optimization opportunities involve exploiting lower precision like FP16 in order to utilize the Tensor Cores available on new Volta and Turing GPUs. While training in FP16 showed great success in image classification tasks, other more complicated neural networks typically stayed in FP32 due to difficulties in applying the FP16 training guidelines that are needed to ensure proper model training.

That is where AMP (Automatic Mixed Precision) comes into play—it automatically applies the guidelines of FP16 training, using FP16 precision where it provides the most benefit, while conservatively keeping in full FP32 precision operations unsafe to do in FP16.

The NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet AMP tutorial, located in `/opt/mxnet/nvidia-examples/AMP/AMP_tutorial.md` inside this container, shows how to get started with mixed precision training using AMP for Apache MXNet, using by example the SSD network from GluonCV.

For more information about AMP, see the Training With Mixed Precision Guide.

**Known Issues**

- The Apache MXNet KVStore GPU peer-to-peer communication tree discovery, as of release 18.09, is not compatible with DGX-1V. Only users that set the environment variable `MXNET_KVSTORE_USETREE=1` will experience issues, which will be resolved in a subsequent release. Issue tracked under 13341.
- The default setting of the environment variable `MXNET_GPU_COPY_NTHREADS=1` in the container may not be optimal for all networks. Networks with a high ratio of parameters and computation, like AlexNet, may achieve greater multi-GPU training speeds with the setting `MXNET_GPU_COPY_NTHREADS=2`. Users are encouraged to try this setting for their own use case.
- There is a known issue in the BERT QA demo where for some values of sequence length the inference may fail with the error `CUDA Driver: operation failed due to a previous error during capture`. To test those values of sequence length, change the line `export MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH=1` to `export MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH=0` in the `test_bert_inference` script inside the demo directory.
- Performance regression (up to 40%) was noticed for ResNet50 mixed precision training with small batch sizes, such as 16 or 32 on V100 compared to 20.03. This will be addressed in a future release.
Performance regression (up to 6%) was noticed for ResNet50 mixed precision training with batch size 128 and 256 on T4 compared to the 20.03. This will be addressed in a future release.
Chapter 5.  Release 20.06

The container image for NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 20.06, is available on NGC.

Contents of the MXNet container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, which is based upon Apache MXNet version 1.6.0. It is pre-built and installed to the Python path.

The container also includes the following:

- Ubuntu 18.04 including Python 3.6
- NVIDIA CUDA 11.0.167 including cuBLAS 11.1.0
- NVIDIA cuDNN 8.0.1
- NVIDIA NCCL 2.7.5 (optimized for NVLink™)
- ONNX exporter 0.1 for CNN classification models

**Note:** The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- Amazon Labs Sockeye sequence-to-sequence framework 1.18.99 (for machine translation)
- MLNX_OFED
- OpenMPI 3.1.6
- Horovod 0.19.1 with enhancements
- Nsight Compute 2020.1.0.33
- Nsight Systems 2020.2.5.8
- GluonCV Toolkit 0.5
- GluonNLP Toolkit 0.8.1
- TensorRT 7.1.2
- DALI 0.22
- Tensor Core optimized example:
  - ResNet-50 v1.5
Jupyter and JupyterLab:

- Jupyter Client 6.0.0
- Jupyter Core 4.6.1
- Jupyter Notebook 6.0.3
- JupyterLab 1.2.6
- JupyterLab Server 1.0.6
- Jupyter-TensorBoard

Driver Requirements

Release 20.06 is based on NVIDIA CUDA 11.0., which requires NVIDIA Driver release 450 or later. However, if you are running on Tesla (for example, T4 or any other Tesla board), you may use NVIDIA driver release 418.xx or 440.30. The CUDA driver’s compatibility package only supports particular drivers. For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

GPU Requirements

Release 20.06 supports CUDA compute capability 6.0 and higher. This corresponds to GPUs in the NVIDIA Pascal, Volta, Turing, and Ampere Architecture GPU families. Specifically, for a list of GPUs that this compute capability corresponds to, see CUDA GPUs. For additional support details, see Deep Learning Frameworks Support Matrix.

Key Features and Enhancements

This NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet release includes the following key features and enhancements.

- NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 20.06 is based on MXNet 1.6.0
- Latest version of NVIDIA CUDA 11.0.167 including cuBLAS 11.1.0
- Latest version of NVIDIA cuDNN 8.0.1
- Latest version of DALI 0.22
- Latest version of NVIDIA NCCL 2.7.5
- Latest version of Nsight Compute 2020.1.0.33
- Latest version of Nsight Systems 2020.2.5.8
- Latest version of TensorRT 7.1.2
- Integrated latest NVIDIA Deep Learning SDK to support NVIDIA A100 using CUDA 11 and cuDNN 8
- Performance improvement of the NHWC FP16 BatchNorm operation
- Performance improvement of broadcast operations, such as broadcast_add
- Enabled support of the MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH in Gluon models hybridized using hybridize(static_alloc=True, static_shape=True) option
- Enabled pointwise fusion to be applied to more Gluon models
- Ubuntu 18.04 with May 2020 updates

Announcements

- Deep learning framework containers 19.11 and later include experimental support for Singularity v3.0.

NVIDIA MXNet Container Versions

The following table shows what versions of Ubuntu, CUDA, Apache MXNet, and TensorRT are supported in each of the NVIDIA containers for the Optimized Deep Learning Framework. For older container versions, refer to the Frameworks Support Matrix.

<table>
<thead>
<tr>
<th>Container Version</th>
<th>Ubuntu</th>
<th>CUDA Toolkit</th>
<th>Apache MXNet</th>
<th>TensorRT</th>
</tr>
</thead>
<tbody>
<tr>
<td>20.06</td>
<td>18.04</td>
<td>NVIDIA CUDA 11.0.167</td>
<td>1.6.0</td>
<td>TensorRT 7.1.2</td>
</tr>
<tr>
<td>20.03</td>
<td></td>
<td>NVIDIA CUDA 10.2.89</td>
<td>1.6.0.rc2</td>
<td>TensorRT 7.0.0</td>
</tr>
<tr>
<td>20.02</td>
<td></td>
<td></td>
<td>1.5.1 commit</td>
<td>TensorRT 6.0.1</td>
</tr>
<tr>
<td>20.01</td>
<td></td>
<td></td>
<td>c98184806 from</td>
<td></td>
</tr>
<tr>
<td>19.12</td>
<td></td>
<td>NVIDIA CUDA 10.1.243</td>
<td>1.5.0 commit</td>
<td>TensorRT 5.1.5</td>
</tr>
<tr>
<td>19.11</td>
<td></td>
<td></td>
<td>006486af3 from</td>
<td></td>
</tr>
<tr>
<td>19.10</td>
<td></td>
<td></td>
<td>August 28, 2019</td>
<td></td>
</tr>
<tr>
<td>19.09</td>
<td></td>
<td></td>
<td>1.5.0 commit</td>
<td></td>
</tr>
<tr>
<td>19.08</td>
<td></td>
<td></td>
<td>75a9e187d from June 27, 2019</td>
<td></td>
</tr>
</tbody>
</table>

Tensor Core Examples

The tensor core examples provided in GitHub and NVIDIA GPU Cloud [NGC] focus on achieving the best performance and convergence from NVIDIA Volta tensor cores by using the latest deep learning example networks and model scripts for training.

Each example model trains with mixed precision Tensor Cores on Volta and Turing, therefore you can get results much faster than training without tensor cores. This model is tested
against each NGC monthly container release to ensure consistent accuracy and performance over time. This container includes the following tensor core examples.

- The **ResNet50 v1.5 model** is a slightly modified version of the original ResNet50 v1 model that trains to a greater accuracy. This model script is available on GitHub as well as NVIDIA GPU Cloud (NGC).

### Automatic Mixed Precision (AMP)

Training deep learning networks is a very computationally intensive task. Novel model architectures tend to have an increasing number of layers and parameters, which slows down training. Fortunately, new generations of training hardware as well as software optimizations make training these new models a feasible task.

Most of the hardware and software training optimization opportunities involve exploiting lower precision like FP16 in order to utilize the Tensor Cores available on new Volta and Turing GPUs. While training in FP16 showed great success in image classification tasks, other more complicated neural networks typically stayed in FP32 due to difficulties in applying the FP16 training guidelines that are needed to ensure proper model training.

That is where AMP (Automatic Mixed Precision) comes into play—it automatically applies the guidelines of FP16 training, using FP16 precision where it provides the most benefit, while conservatively keeping in full FP32 precision operations unsafe to do in FP16.

The NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet AMP tutorial, located in `/opt/mxnet/nvidia-examples/AMP/AMP_tutorial.md` inside this container, shows how to get started with mixed precision training using AMP for Apache MXNet, using by example the SSD network from GluonCV.

For more information about AMP, see the [Training With Mixed Precision Guide](#).

### Known Issues

- The MXNet KVStore GPU peer-to-peer communication tree discovery, as of release 18.09, is not compatible with DGX-1V. Only users that set the environment variable `MXNET_KVSTORE_USETREE=1` will experience issues, which will be resolved in a subsequent release. Issue tracked under [13341](#).

- The default setting of the environment variable `MXNET_GPU_COPY_NTHREADS=1` in the container may not be optimal for all networks. Networks with a high ratio of parameters and computation, like AlexNet, may achieve greater multi-GPU training speeds with the setting `MXNET_GPU_COPY_NTHREADS=2`. Users are encouraged to try this setting for their own use case.

- There is a known issue in the BERT QA demo where for some values of sequence length the inference may fail with the error `CUDA Driver: operation failed due to a previous error during capture`. To test those values of sequence length, change the line `export MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH=1` to `export`
MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH=0 in the test_bert_inference script inside the demo directory.

- Performance regression (up to 40%) was noticed for ResNet50 mixed precision training with small batch sizes, such as 16 or 32 on V100 compared to 20.03. This will be addressed in a future release.

- Performance regression (up to 6%) was noticed for ResNet50 mixed precision training with batch size 128 and 256 on T4 compared to the 20.03. This will be addressed in a future release.

- There is a known performance regression of up to 15% compared to the 20.03 release in the BERT QA demo with batch size of 1 on V100. This will be addressed in a future release.
Chapter 6. Release 20.03

The container image for NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 20.03, is available on NGC.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in `/opt/mxnet`. It is pre-built and installed to the Python path.

The container also includes the following:

- Ubuntu 18.04 including Python 3.6
- NVIDIA CUDA 10.2.89 including cuBLAS 10.2.2.89
- NVIDIA cuDNN 7.6.5
- NVIDIA NCCL 2.6.3 (optimized for NVLink™)
- ONNX exporter 0.1 for CNN classification models

**Note:** The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- Amazon Labs Sockeye sequence-to-sequence framework 1.18.99 (for machine translation)
- MLNX_OFED
- OpenMPI 3.1.4
- Horovod 0.19.0
- Nsight Compute 2019.5.0
- Nsight Systems 2020.1.1
- GluonCV Toolkit 0.5
- GluonNLP Toolkit 0.8.1
- TensorRT 7.0.0
- DALI 0.19.0
- Tensor Core optimized example:
  - ResNet-50 v1.5
Jupyter and JupyterLab:

- Jupyter Client 6.0.0
- Jupyter Core 4.6.1
- Jupyter Notebook 6.0.3
- JupyterLab 1.2.6
- JupyterLab Server 1.0.6
- Jupyter-TensorBoard

Driver Requirements

Release 20.03 is based on NVIDIA CUDA 10.2.89, which requires NVIDIA Driver release 440.33.01. However, if you are running on Tesla (for example, T4 or any other Tesla board), you may use NVIDIA driver release 396, 384.111+, 410, 418.xx or 440.30. The CUDA driver’s compatibility package only supports particular drivers. For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

GPU Requirements

Release 20.03 supports CUDA compute capability 6.0 and higher. This corresponds to GPUs in the Pascal, Volta, and Turing families. Specifically, for a list of GPUs that this compute capability corresponds to, see CUDA GPUs. For additional support details, see Deep Learning Frameworks Support Matrix.

Key Features and Enhancements

This NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet release includes the following key features and enhancements.

- NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 20.03 is based on Apache MXNet 1.6.0
- Increased performance of BatchNorm operator when the bn_group parameter is set to more than 1 and enabled grouping for 8 and 16 GPUs in a single node
- Increased performance of the ArgSort operator
- Exposed cudnn control parameters such as cudnn_algo_fwd, previously only available in mx.sym.Convolution to Gluon’s gluon.nn.Conv[1D,2D,3D]
- Increased performance of the elementwise operations such as elementwise_add, most notably for FP16 precision
- The latest version of DALI 0.19.0
- Ubuntu 18.04 with February 2020 updates
Announcements

‣ Deep learning framework containers 19.11 and later include experimental support for Singularity v3.0.

NVIDIA MXNet Container Versions

The following table shows what versions of Ubuntu, CUDA, Apache MXNet, and TensorRT are supported in each of the NVIDIA containers for the Optimized Deep Learning Framework. For older container versions, refer to the Frameworks Support Matrix.

<table>
<thead>
<tr>
<th>Container Version</th>
<th>Ubuntu</th>
<th>CUDA Toolkit</th>
<th>Apache MXNet</th>
<th>TensorRT</th>
</tr>
</thead>
<tbody>
<tr>
<td>20.03</td>
<td>18.04</td>
<td>NVIDIA CUDA 10.2.89</td>
<td>1.6.0</td>
<td>TensorRT 7.0.0</td>
</tr>
<tr>
<td>20.02</td>
<td>16.04</td>
<td>NVIDIA CUDA 10.2.89</td>
<td>1.6.0.rc2</td>
<td></td>
</tr>
<tr>
<td>20.01</td>
<td></td>
<td>NVIDIA CUDA 10.1.243</td>
<td>1.5.1 commit c98184806 from September 4, 2019</td>
<td>TensorRT 6.0.1</td>
</tr>
<tr>
<td>19.12</td>
<td></td>
<td>NVIDIA CUDA 10.1.243</td>
<td>1.5.0 commit 006486af3 from August 28, 2019</td>
<td></td>
</tr>
<tr>
<td>19.11</td>
<td></td>
<td>NVIDIA CUDA 10.1.243</td>
<td>1.5.0 commit 75a9e187d from June 27, 2019</td>
<td></td>
</tr>
<tr>
<td>19.10</td>
<td></td>
<td>NVIDIA CUDA 10.1.243</td>
<td>TensorRT 5.1.5</td>
<td></td>
</tr>
<tr>
<td>19.09</td>
<td></td>
<td>NVIDIA CUDA 10.1.243</td>
<td>TensorRT 5.1.5</td>
<td></td>
</tr>
<tr>
<td>19.08</td>
<td></td>
<td>NVIDIA CUDA 10.1.243</td>
<td>TensorRT 5.1.5</td>
<td></td>
</tr>
</tbody>
</table>

Tensor Core Examples

The tensor core examples provided in GitHub and NVIDIA GPU Cloud [NGC] focus on achieving the best performance and convergence from NVIDIA Volta tensor cores by using the latest deep learning example networks and model scripts for training.

Each example model trains with mixed precision Tensor Cores on Volta and Turing, therefore you can get results much faster than training without tensor cores. This model is tested against each NGC monthly container release to ensure consistent accuracy and performance over time. This container includes the following tensor core examples.

‣ The ResNet50 v1.5 model is a slightly modified version of the original ResNet50 v1 model that trains to a greater accuracy. This model script is available on GitHub as well as NVIDIA GPU Cloud [NGC].
Automatic Mixed Precision (AMP)

Training deep learning networks is a very computationally intensive task. Novel model architectures tend to have an increasing number of layers and parameters, which slows down training. Fortunately, new generations of training hardware as well as software optimizations make training these new models a feasible task.

Most of the hardware and software training optimization opportunities involve exploiting lower precision like FP16 in order to utilize the Tensor Cores available on new Volta and Turing GPUs. While training in FP16 showed great success in image classification tasks, other more complicated neural networks typically stayed in FP32 due to difficulties in applying the FP16 training guidelines that are needed to ensure proper model training.

That is where AMP (Automatic Mixed Precision) comes into play—it automatically applies the guidelines of FP16 training, using FP16 precision where it provides the most benefit, while conservatively keeping in full FP32 precision operations unsafe to do in FP16.

The NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet AMP tutorial, located in /opt/mxnet/nvidia-examples/AMP/AMP_tutorial.md inside this container, shows how to get started with mixed precision training using AMP for Apache MXNet, using by example the SSD network from GluonCV.

For more information about AMP, see the Training With Mixed Precision Guide.

Known Issues

▶ The Apache MXNet KVStore GPU peer-to-peer communication tree discovery, as of release 18.09, is not compatible with DGX-1V. Only users that set the environment variable MXNET_KVSTORE_USETREE=1 will experience issues, which will be resolved in a subsequent release. Issue tracked under 13341.

▶ The default setting of the environment variable MXNET_GPU_COPY_NTHREADS=1 in the container may not be optimal for all networks. Networks with a high ratio of parameters and computation, like AlexNet, may achieve greater multi-GPU training speeds with the setting MXNET_GPU_COPY_NTHREADS=2. Users are encouraged to try this setting for their own use case.

▶ There is a known issue in the BERT QA demo, where for some values of sequence length the inference may fail with CUDA Driver: operation failed due to a previous error during captureerror. To test those values of sequence length, change the line export MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH=1 to export MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH=0 in the test_bert_inference script inside the demo directory.
Chapter 7. Release 20.02

The container image for NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 20.02, is available on NGC.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of the NVIDIA Optimized Deep Learning Framework in `/opt/mxnet`. It is pre-built and installed to the Python path.

The container also includes the following:

- Ubuntu 18.04 including Python 3.6
- NVIDIA CUDA 10.2.89 including cuBLAS 10.2.89
- NVIDIA cuDNN 7.6.5
- NVIDIA NCCL 2.5.6 (optimized for NVLink™)
- ONNX exporter 0.1 for CNN classification models

Note: The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- Amazon Labs Sockeye sequence-to-sequence framework 1.18.99 [for machine translation]
- MLNX_OFED
- OpenMPI 3.1.4
- Horovod 0.19.0
- Nsight Compute 2019.5.0
- Nsight Systems 2020.1.1
- GluonCV Toolkit 0.5
- GluonNLP Toolkit 0.8.1
- TensorRT 7.0.0
- DALI 0.18.0 Beta
- Tensor Core optimized example:
  - ResNet-50 v1.5
- Jupyter and JupyterLab:
Jupyter Client 5.3.4
Jupyter Core 4.6.1
Jupyter Notebook 6.0.3
JupyterLab 1.2.6
JupyterLab Server 1.0.6
Jupyter-TensorBoard

Driver Requirements

Release 20.02 is based on NVIDIA CUDA 10.2.89, which requires NVIDIA Driver release 440.33.01. However, if you are running on Tesla (for example, T4 or any other Tesla board), you may use NVIDIA driver release 396, 384.111+, 410, 418.xx or 440.30. The CUDA driver’s compatibility package only supports particular drivers. For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

GPU Requirements

Release 20.02 supports CUDA compute capability 6.0 and higher. This corresponds to GPUs in the Pascal, Volta, and Turing families. Specifically, for a list of GPUs that this compute capability corresponds to, see CUDA GPUs. For additional support details, see Deep Learning Frameworks Support Matrix.

Key Features and Enhancements

This Optimized Deep Learning Framework release includes the following key features and enhancements.

- NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 20.02 is based on Apache MXNet 1.6.0.rc2 and includes upstream commits up through commit c9814806 from September 4, 2019.
- Latest version of Nsight Systems 2020.1.1
- Latest version of Horovod 0.19.0
- Latest version of DALI 0.18.0 Beta
- Latest version of Jupyter Notebook 6.0.3, JupyterLab 1.2.6
- Ubuntu 18.04 with January 2020 updates
- A new option was added for Automatic Mixed Precision: layout optimization. It can be triggered by specifying layout_optimization=True option in the call to amp.init(layout_optimization=True). Setting this option might significantly increase throughput of training and inference of convolutional neural networks.
- Significantly improved performance of the BatchNorm operator when use_global_stats=True is set.
Announcements

- Deep learning framework containers 19.11 and later include experimental support for Singularity v3.0.

NVIDIA MXNet Container Versions

The following table shows what versions of Ubuntu, CUDA, Apache MXNet, and TensorRT are supported in each of the NVIDIA containers for the Optimized Deep Learning Framework. For older container versions, refer to the Frameworks Support Matrix.

<table>
<thead>
<tr>
<th>Container Version</th>
<th>Ubuntu</th>
<th>CUDA Toolkit</th>
<th>Apache MXNet</th>
<th>TensorRT</th>
</tr>
</thead>
<tbody>
<tr>
<td>20.02</td>
<td>18.04</td>
<td>NVIDIA CUDA 10.2.89</td>
<td>1.6.0.rc2</td>
<td>TensorRT 7.0.0</td>
</tr>
<tr>
<td>20.01</td>
<td>16.04</td>
<td></td>
<td>1.5.1 commit c98184806 from September 4, 2019</td>
<td>TensorRT 6.0.1</td>
</tr>
<tr>
<td>19.12</td>
<td></td>
<td>NVIDIA CUDA 10.1.243</td>
<td>1.5.0 commit 006486af3 from August 28, 2019</td>
<td>TensorRT 5.1.5</td>
</tr>
<tr>
<td>19.11</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19.10</td>
<td></td>
<td></td>
<td>1.5.0 commit 75a9e187d from June 27, 2019</td>
<td></td>
</tr>
<tr>
<td>19.09</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19.08</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Tensor Core Examples

The tensor core examples provided in GitHub and NVIDIA GPU Cloud [NGC] focus on achieving the best performance and convergence from NVIDIA Volta tensor cores by using the latest deep learning example networks and model scripts for training.

Each example model trains with mixed precision Tensor Cores on Volta and Turing, therefore you can get results much faster than training without tensor cores. This model is tested against each NGC monthly container release to ensure consistent accuracy and performance over time. This container includes the following tensor core examples.

- The ResNet50 v1.5 model is a slightly modified version of the original ResNet50 v1 model that trains to a greater accuracy. This model script is available on GitHub as well as NVIDIA GPU Cloud [NGC].
Automatic Mixed Precision (AMP)

Training deep learning networks is a very computationally intensive task. Novel model architectures tend to have an increasing number of layers and parameters, which slows down training. Fortunately, new generations of training hardware as well as software optimizations make training these new models a feasible task.

Most of the hardware and software training optimization opportunities involve exploiting lower precision like FP16 in order to utilize the Tensor Cores available on new Volta and Turing GPUs. While training in FP16 showed great success in image classification tasks, other more complicated neural networks typically stayed in FP32 due to difficulties in applying the FP16 training guidelines that are needed to ensure proper model training.

That is where AMP (Automatic Mixed Precision) comes into play- it automatically applies the guidelines of FP16 training, using FP16 precision where it provides the most benefit, while conservatively keeping in full FP32 precision operations unsafe to do in FP16.

The NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet AMP tutorial, located in /opt/mxnet/nvidia-examples/AMP/AMP_tutorial.md inside this container, shows how to get started with mixed precision training using AMP for Apache MXNet, using by example the SSD network from GluonCV.

For more information about AMP, see the Training With Mixed Precision Guide.

Known Issues

- The Apache MXNet KVStore GPU peer-to-peer communication tree discovery, as of release 18.09, is not compatible with DGX-1V. Only users that set the environment variable MXNET_KVSTORE_USETREE=1 will experience issues, which will be resolved in a subsequent release. Issue tracked under 13341.

- The default setting of the environment variable MXNET_GPU_COPY_NTHREADS=1 in the container may not be optimal for all networks. Networks with a high ratio of parameters and computation, like AlexNet, may achieve greater multi-GPU training speeds with the setting MXNET_GPU_COPY_NTHREADS=2. Users are encouraged to try this setting for their own use case.

- There is a known issue in the BERT QA demo, where for some values of sequence length the inference may fail with CUDA Driver: operation failed due to a previous error during captureerror. To test those values of sequence length, change the line export MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH=1 to export MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH=0 in the test_bert_inference script inside the demo directory.
Chapter 8. Release 20.01

The NVIDIA container image for NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 20.01, is available on NGC.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu 18.04** including **Python 3.6**
- **NVIDIA CUDA 10.2.89** including **cuBLAS 10.2.2.89**
- **NVIDIA cuDNN 7.6.5**
- **NVIDIA NCCL 2.5.6** (optimized for NVLink™)
- **ONNX exporter 0.1** for CNN classification models

  **Note:** The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- **Amazon Labs Sockeye sequence-to-sequence framework 1.18.99** (for machine translation)
- **MLNX_OFED**
- **OpenMPI 3.1.4**
- **Horovod 0.18.1**
- **Nsight Compute 2019.5.0**
- **Nsight Systems 2019.6.1**
- **GluonCV Toolkit 0.5**
- **GluonNLP Toolkit 0.8.1**
- **TensorRT 7.0.0**
- **DALI 0.17.0 Beta**
- **Tensor Core optimized example:**
  - **ResNet-50 v1.5**
Jupyter and JupyterLab:
- Jupyter Client 5.3.4
- Jupyter Core 4.6.1
- Jupyter Notebook 6.0.2
- JupyterLab 1.2.4
- JupyterLab Server 1.0.6
- Jupyter-TensorBoard

Driver Requirements
Release 20.01 is based on NVIDIA CUDA 10.2.89, which requires NVIDIA Driver release 440.33.01. However, if you are running on Tesla (for example, T4 or any other Tesla board), you may use NVIDIA driver release 396, 384.111+, 410, 418.xx or 440.30. The CUDA driver’s compatibility package only supports particular drivers. For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

GPU Requirements
Release 20.01 supports CUDA compute capability 6.0 and higher. This corresponds to GPUs in the Pascal, Volta, and Turing families. Specifically, for a list of GPUs that this compute capability corresponds to, see CUDA GPUs. For additional support details, see Deep Learning Frameworks Support Matrix.

Key Features and Enhancements
This Optimized Deep Learning Framework release release includes the following key features and enhancements.
- NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 20.01 is based on MXNet 1.5.1 and includes upstream commits up through commit c98184806 from September 4, 2019.
- Latest version of TensorRT 7.0.0
- Latest version of DALI 0.17.0 Beta
- Latest version of JupyterLab 1.2.4
- Ubuntu 18.04 with December 2019 updates

Announcements
- Deep learning framework containers 19.11 and later include experimental support for Singularity v3.0.
NVIDIA MXNet Container Versions

The following table shows what versions of Ubuntu, CUDA, Apache MXNet, and TensorRT are supported in each of the NVIDIA containers for the Optimized Deep Learning Framework. For older container versions, refer to the Frameworks Support Matrix.

<table>
<thead>
<tr>
<th>Container Version</th>
<th>Ubuntu</th>
<th>CUDA Toolkit</th>
<th>Apache MXNet</th>
<th>TensorRT</th>
</tr>
</thead>
<tbody>
<tr>
<td>20.01</td>
<td>18.04</td>
<td>NVIDIA CUDA 10.2.89</td>
<td>1.5.1 commit c98194806 from September 4, 2019</td>
<td>TensorRT 7.0.0</td>
</tr>
<tr>
<td>19.12</td>
<td>16.04</td>
<td>NVIDIA CUDA 10.1.243</td>
<td>1.5.0 commit 006486af3 from August 28, 2019</td>
<td>TensorRT 6.0.1</td>
</tr>
<tr>
<td>19.11</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19.10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19.09</td>
<td></td>
<td></td>
<td></td>
<td>TensorRT 5.1.5</td>
</tr>
<tr>
<td>19.08</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Tensor Core Examples

The tensor core examples provided in GitHub and NVIDIA GPU Cloud (NGC) focus on achieving the best performance and convergence from NVIDIA Volta tensor cores by using the latest deep learning example networks and model scripts for training.

Each example model trains with mixed precision Tensor Cores on Volta and Turing, therefore you can get results much faster than training without tensor cores. This model is tested against each NGC monthly container release to ensure consistent accuracy and performance over time. This container includes the following tensor core examples.

- The ResNet50 v1.5 model is a slightly modified version of the original ResNet50 v1 model that trains to a greater accuracy. This model script is available on GitHub as well as NVIDIA GPU Cloud (NGC).

Automatic Mixed Precision (AMP)

Training deep learning networks is a very computationally intensive task. Novel model architectures tend to have an increasing number of layers and parameters, which slows down training. Fortunately, new generations of training hardware as well as software optimizations make training these new models a feasible task.

Most of the hardware and software training optimization opportunities involve exploiting lower precision like FP16 in order to utilize the Tensor Cores available on new Volta and Turing GPUs. While training in FP16 showed great success in image classification tasks, other more
complicated neural networks typically stayed in FP32 due to difficulties in applying the FP16 training guidelines that are needed to ensure proper model training.

That is where AMP (Automatic Mixed Precision) comes into play - it automatically applies the guidelines of FP16 training, using FP16 precision where it provides the most benefit, while conservatively keeping in full FP32 precision operations unsafe to do in FP16.

The NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet AMP tutorial, located in `/opt/mxnet/nvidia-examples/AMP/AMP_tutorial.md` inside this container, shows how to get started with mixed precision training using AMP for Apache MXNet, using by example the SSD network from GluonCV.

For more information about AMP, see the Training With Mixed Precision Guide.

**Known Issues**

- The Apache MXNet KVStore GPU peer-to-peer communication tree discovery, as of release 18.09, is not compatible with DGX-1V. Only users that set the environment variable `MXNET_KVSTORE_USETREE=1` will experience issues, which will be resolved in a subsequent release. Issue tracked under 13341.

- The default setting of the environment variable `MXNET_GPU_COPY_NTHREADS=1` in the container may not be optimal for all networks. Networks with a high ratio of parameters and computation, like AlexNet, may achieve greater multi-GPU training speeds with the setting `MXNET_GPU_COPY_NTHREADS=2`. Users are encouraged to try this setting for their own use case.

- There is a known issue in the BERT QA demo where for some values of sequence length the inference may fail with the error CUDA Driver: operation failed due to a previous error during capture. To test those values of sequence length, change the line `export MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH=1` to `export MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH=0` in the test_bert_inference script inside the demo directory.

The container image for NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 19.12, is available on NGC.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- Ubuntu 18.04 including Python 3.6
- NVIDIA CUDA 10.2.89 including cuBLAS 10.2.2.89
- NVIDIA cuDNN 7.6.5
- NVIDIA NCCL 2.5.6 (optimized for NVLink™)
- ONNX exporter 0.1 for CNN classification models

Note: The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- Amazon Labs Sockeye sequence-to-sequence framework 1.18.99 [for machine translation]
- MLNX OFED
- OpenMPI 3.1.4
- Horovod 0.18.1
- Nsight Compute 2019.5.0
- Nsight Systems 2019.6.1
- GluonCV Toolkit 0.5
- GluonNLP Toolkit 0.8.1
- TensorRT 6.0.1
- DALI 0.16.0 Beta
- Tensor Core optimized example:
  - ResNet-50 v1.5
Driver Requirements

Release 19.12 is based on NVIDIA CUDA 10.2.89, which requires NVIDIA Driver release 440.30. However, if you are running on Tesla (for example, T4 or any other Tesla board), you may use NVIDIA driver release 396, 384.111+, 410, 418.xx or 440.30. The CUDA driver’s compatibility package only supports particular drivers. For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

GPU Requirements

Release 19.12 supports CUDA compute capability 6.0 and higher. This corresponds to GPUs in the Pascal, Volta, and Turing families. Specifically, for a list of GPUs that this compute capability corresponds to, see CUDA GPUs. For additional support details, see Deep Learning Frameworks Support Matrix.

Key Features and Enhancements

This Optimized Deep Learning Framework release includes the following key features and enhancements.

- NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 19.12 is based on Apache MXNet 1.5.1 and includes upstream commits up through commit c98184806 from September 4, 2019.
- The compilation speed of fused pointwise kernels in many models was improved.
- Latest version of DALI 0.16.0 Beta
- Latest version of Nsight Systems 2019.6.1
- Latest version of JupyterLab 1.2.3
- Ubuntu 18.04 with November 2019 updates

Announcements

- Deep learning framework containers 19.11 and later include experimental support for Singularity v3.0.
Tensor Core Examples

The tensor core examples provided in GitHub and NVIDIA GPU Cloud [NGC] focus on achieving the best performance and convergence from NVIDIA Volta tensor cores by using the latest deep learning example networks and model scripts for training.

Each example model trains with mixed precision Tensor Cores on Volta and Turing, therefore you can get results much faster than training without tensor cores. This model is tested against each NGC monthly container release to ensure consistent accuracy and performance over time. This container includes the following tensor core examples.

- The ResNet50 v1.5 model is a slightly modified version of the original ResNet50 v1 model that trains to a greater accuracy. This model script is available on GitHub as well as NVIDIA GPU Cloud [NGC].

Automatic Mixed Precision (AMP)

Training deep learning networks is a very computationally intensive task. Novel model architectures tend to have an increasing number of layers and parameters, which slows down training. Fortunately, new generations of training hardware as well as software optimizations make training these new models a feasible task.

Most of the hardware and software training optimization opportunities involve exploiting lower precision like FP16 in order to utilize the Tensor Cores available on new Volta and Turing GPUs. While training in FP16 showed great success in image classification tasks, other more complicated neural networks typically stayed in FP32 due to difficulties in applying the FP16 training guidelines that are needed to ensure proper model training.

That is where AMP (Automatic Mixed Precision) comes into play- it automatically applies the guidelines of FP16 training, using FP16 precision where it provides the most benefit, while conservatively keeping in full FP32 precision operations unsafe to do in FP16.

The NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet AMP tutorial, located in `/opt/mxnet/nvidia-examples/AMP/AMP_tutorial.md` inside this container, shows how to get started with mixed precision training using AMP for Apache MXNet, using by example the SSD network from GluonCV.

For more information about AMP, see the Training With Mixed Precision Guide.

Known Issues

- The Apache MXNet KVStore GPU peer-to-peer communication tree discovery, as of release 18.09, is not compatible with DGX-1V. Only users that set the environment variable `MXNET_KVSTORE_USETREE=1` will experience issues, which will be resolved in a subsequent release. Issue tracked under 13341.

- The default setting of the environment variable `MXNET_GPU_COPY_NTHREADS=1` in the container may not be optimal for all networks. Networks with a high ratio of parameters...
and computation, like AlexNet, may achieve greater multi-GPU training speeds with the setting `MXNET_GPU_COPY_NTHREADS=2`. Users are encouraged to try this setting for their own use case.

- There is a known issue in the BERT QA demo, where for some values of sequence length the inference may fail with CUDA Driver: operation failed due to a previous error during capture error. To test those values of sequence length, change the line `export MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH=1` to `export MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH=0` in the `test_bert_inference` script inside the demo directory.
Chapter 10. Release 19.11

The container image for NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 19.11, is available on NGC.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- Ubuntu 18.04 including Python 3.6
- NVIDIA CUDA 10.2.89 including cuBLAS 10.2.2.89
- NVIDIA cuDNN 7.6.5
- NVIDIA NCCL 2.5.6 (optimized for NVLink™)
- ONNX exporter 0.1 for CNN classification models

**Note:** The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- Amazon Labs Sockeye sequence-to-sequence framework 1.18.99 (for machine translation)
- MLNX_OFED
- OpenMPI 3.1.4
- Horovod 0.18.1
- Nsight Compute 2019.5.0
- Nsight Systems 2019.5.2
- GluonCV Toolkit 0.5
- GluonNLP Toolkit 0.8.1
- TensorRT 6.0.1
- DALI 0.15.0 Beta
- Tensor Core optimized example:
  - ResNet-50 v1.5
Jupyter and JupyterLab:

- Jupyter Client 5.3.4
- Jupyter Core 4.6.1
- Jupyter Notebook 6.0.2
- JupyterLab 1.2.2
- JupyterLab Server 1.0.6
- Jupyter-TensorBoard

Driver Requirements

Release 19.11 is based on NVIDIA CUDA 10.2.89, which requires NVIDIA Driver release 440.30. However, if you are running on Tesla (for example, T4 or any other Tesla board), you may use NVIDIA driver release 396, 384.111+, 410 or 418.xx. The CUDA driver’s compatibility package only supports particular drivers. For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

GPU Requirements

Release 19.11 supports CUDA compute capability 6.0 and higher. This corresponds to GPUs in the Pascal, Volta, and Turing families. Specifically, for a list of GPUs that this compute capability corresponds to, see CUDA GPUs. For additional support details, see Deep Learning Frameworks Support Matrix.

Key Features and Enhancements

This Optimized Deep Learning Framework release includes the following key features and enhancements.

- NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 19.11 is based on MXNet 1.5.1 and includes upstream commits up through commit c98184806 from September 4, 2019.
- Improved performance of the BERT QA inference demo, especially for T4 GPU on large batch sizes.
- Improved performance of `box_nms` operator.
- Improved performance of backward pass when using `Embedding` operator.
- Improved performance of `argmax` operator.
- Improved performance of PointWise fusion when interacting with scalars.
- Latest version of Python 3.6
- Latest version of NVIDIA CUDA 10.2.89 including cuBLAS 10.2.2.89
- Latest version of NVIDIA cuDNN 7.6.5
- Latest version of NVIDIA NCCL 2.5.6
- Latest version of Nsight Compute 2019.5.0
Latest version of Nsight Systems 2019.5.2
Latest version of DALI 0.15.0 Beta
Latest versions of Jupyter Client 5.3.4, Jupyter Core 4.6.1, JupyterLab 1.2.2, and Jupyter Notebook 6.0.2
Ubuntu 18.04 with October 2019 updates

Announcements

- Deep learning framework containers 19.11 and later include experimental support for Singularity v3.0.

Tensor Core Examples

The tensor core examples provided in GitHub and NVIDIA GPU Cloud (NGC) focus on achieving the best performance and convergence from NVIDIA Volta tensor cores by using the latest deep learning example networks and model scripts for training.

Each example model trains with mixed precision Tensor Cores on Volta and Turing, therefore you can get results much faster than training without tensor cores. This model is tested against each NGC monthly container release to ensure consistent accuracy and performance over time. This container includes the following tensor core examples.

- The ResNet50 v1.5 model is a slightly modified version of the original ResNet50 v1 model that trains to a greater accuracy. This model script is available on GitHub as well as NVIDIA GPU Cloud (NGC).

Automatic Mixed Precision (AMP)

Training deep learning networks is a very computationally intensive task. Novel model architectures tend to have an increasing number of layers and parameters, which slows down training. Fortunately, new generations of training hardware as well as software optimizations make training these new models a feasible task.

Most of the hardware and software training optimization opportunities involve exploiting lower precision like FP16 in order to utilize the Tensor Cores available on new Volta and Turing GPUs. While training in FP16 showed great success in image classification tasks, other more complicated neural networks typically stayed in FP32 due to difficulties in applying the FP16 training guidelines that are needed to ensure proper model training.

That is where AMP (Automatic Mixed Precision) comes into play- it automatically applies the guidelines of FP16 training, using FP16 precision where it provides the most benefit, while conservatively keeping in full FP32 precision operations unsafe to do in FP16.

The NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet AMP tutorial, located in /opt/mxnet/nvidia-examples/AMP/AMP_tutorial.md inside this container, shows how to get started with mixed precision training using AMP for Apache MXNet, using by example the SSD network from GluonCV.
For more information about AMP, see the Training With Mixed Precision Guide.

**Known Issues**

- There is a known performance regression for Apache MXNet Inception V3 and Alexnet training with multi-GPU configurations. This will be fixed in 19.12.

- The Apache MXNet KVStore GPU peer-to-peer communication tree discovery, as of release 18.09, is not compatible with DGX-1V. Only users that set the environment variable `MXNET_KVSTORE_USETREE=1` will experience issues, which will be resolved in a subsequent release. Issue tracked under 13341.

- The default setting of the environment variable `MXNET_GPU_COPY_NTHREADS=1` in the container may not be optimal for all networks. Networks with a high ratio of parameters and computation, like AlexNet, may achieve greater multi-GPU training speeds with the setting `MXNET_GPU_COPY_NTHREADS=2`. Users are encouraged to try this setting for their own use case.

- There is a known issue in the BERT QA demo, where for some values of sequence length the inference may fail with CUDA Driver: operation failed due to a previous error during captureerror. To test those values of sequence length, change the line `export MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH=1` to `export MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH=0` in the `test_bert_inference` script inside the demo directory.
Chapter 11. Release 19.10

The container image for NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 19.10, is available on NGC.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- Ubuntu 18.04 including Python 3.5
- NVIDIA CUDA 10.1.243 including cuBLAS 10.2.1.243
- NVIDIA cuDNN 7.6.4
- NVIDIA NCCL 2.4.8 (optimized for NVLink™)
- ONNX exporter 0.1 for CNN classification models

Note: The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- Amazon Labs Sockeye sequence-to-sequence framework 1.18.99 (for machine translation)
- MLNX_OFED
- OpenMPI 3.1.4
- Horovod 0.18.1
- Nsight Compute 2019.4.0
- Nsight Systems 2019.5.1
- GluonCV Toolkit 0.5
- GluonNLP Toolkit 0.8.1
- TensorRT 6.0.1
- DALI 0.14.0 Beta
- Tensor Core optimized example:
  - ResNet-50 v1.5
Jupyter and JupyterLab:

- Jupyter Client 5.3.3
- Jupyter Core 4.5.0
- Jupyter Notebook 6.0.1
- JupyterLab 1.1.4
- JupyterLab Server 1.0.6
- Jupyter-TensorBoard

Driver Requirements

Release 19.10 is based on NVIDIA CUDA 10.1.243, which requires NVIDIA Driver release 418.xx. However, if you are running on Tesla (for example, T4 or any other Tesla board), you may use NVIDIA driver release 396, 384.111+ or 410. However, if you are running on Tesla (Tesla V100, Tesla P4, Tesla P40, or Tesla P100), you may use NVIDIA driver release 384.111+ or 410. The CUDA driver’s compatibility package only supports particular drivers. For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

GPU Requirements

Release 19.10 supports CUDA compute capability 6.0 and higher. This corresponds to GPUs in the Pascal, Volta, and Turing families. Specifically, for a list of GPUs that this compute capability corresponds to, see CUDA GPUs. For additional support details, see Deep Learning Frameworks Support Matrix.

Key Features and Enhancements

This Optimized Deep Learning Framework release includes the following key features and enhancements.

- NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 19.10 is based on Apache MXNet 1.5.1 and includes upstream commits up through commit c98184806 from September 4, 2019.
- Latest version of NVIDIA cuDNN 7.6.4
- Latest version of Nsight Systems 2019.5.1
- Latest version of DALI 0.14.0 Beta
- Latest version of GluonCV Toolkit 0.5
- Latest versions of Jupyter Client 5.3.3 and JupyterLab 1.1.4
- Ubuntu 18.04 with September 2019 updates
Tensor Core Examples

These examples focus on achieving the best performance and convergence from NVIDIA Volta Tensor Cores by using the latest deep learning example networks for training.

Each example model trains with mixed precision Tensor Cores on Volta, therefore you can get results much faster than training without tensor cores. This model is tested against each NGC monthly container release to ensure consistent accuracy and performance over time. This container includes the following tensor core examples.

- The ResNet50 v1.5 model is a slightly modified version of the original ResNet50 v1 model that trains to a greater accuracy. This model script is available on GitHub as well as NVIDIA GPU Cloud [NGC].

Automatic Mixed Precision (AMP)

Training deep learning networks is a very computationally intensive task. Novel model architectures tend to have an increasing number of layers and parameters, which slows down training. Fortunately, new generations of training hardware as well as software optimizations make training these new models a feasible task.

Most of the hardware and software training optimization opportunities involve exploiting lower precision like FP16 in order to utilize the Tensor Cores available on new Volta and Turing GPUs. While training in FP16 showed great success in image classification tasks, other more complicated neural networks typically stayed in FP32 due to difficulties in applying the FP16 training guidelines that are needed to ensure proper model training.

That is where AMP (Automatic Mixed Precision) comes into play- it automatically applies the guidelines of FP16 training, using FP16 precision where it provides the most benefit, while conservatively keeping in full FP32 precision operations unsafe to do in FP16.

The NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet AMP tutorial, located in /opt/mxnet/nvidia-examples/AMP/AMP_tutorial.md inside this container, shows how to get started with mixed precision training using AMP for Apache MXNet, using by example the SSD network from GluonCV.

For more information about AMP, see the Training With Mixed Precision Guide.

Known Issues

- The Apache MXNet KVStore GPU peer-to-peer communication tree discovery, as of release 18.09, is not compatible with DGX-1V. Only users that set the environment variable MXNET_KVSTORE_USETREE=1 will experience issues, which will be resolved in a subsequent release. Issue tracked under 13341.

- The default setting of the environment variable MXNET_GPU_COPY_NTHREADS=1 in the container may not be optimal for all networks. Networks with a high ratio of parameters and computation, like AlexNet, may achieve greater multi-GPU training speeds with the
setting MXNET_GPU_COPY_NTHREADS=2. Users are encouraged to try this setting for their own use case.

- There is a known issue in the BERT QA demo, where for some values of sequence length the inference may fail with CUDA Driver: operation failed due to a previous error during captureerror. To test those values of sequence length, change the line `export MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH=1` to `export MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH=0` in the `test_bert_inference` script inside the demo directory.
Chapter 12. Release 19.09

The container image for NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 19.09, is available on NGC.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu 18.04** including **Python 3.5**
- **NVIDIA CUDA 10.1.243** including **cuBLAS 10.2.1.243**
- **NVIDIA cuDNN 7.6.3**
- **NVIDIA NCCL 2.4.8** (optimized for NVLink™)
- **ONNX exporter 0.1** for CNN classification models

**Note:** The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- **Amazon Labs Sockeye sequence-to-sequence framework 1.18.99** (for machine translation)
- **MLNX OFED**
- **OpenMPI 3.1.4**
- **Horovod 0.18.1**
- **Nsight Compute 2019.4.0**
- **Nsight Systems 2019.4.2**
- **GluonCV Toolkit 0.4**
- **GluonNLP Toolkit 0.8.1**
- **TensorRT 6.0.1**
- **DALI 0.13.0 Beta**
- Tensor Core optimized example:
  - **ResNet-50 v1.5**
Jupyter and JupyterLab:

- Jupyter Client 5.3.1
- Jupyter Core 4.5.0
- Jupyter Notebook 6.0.1
- JupyterLab 1.1.1
- JupyterLab Server 1.0.6
- Jupyter-TensorBoard

Driver Requirements

Release 19.09 is based on NVIDIA CUDA 10.1.243, which requires NVIDIA Driver release 418.xx. However, if you are running on Tesla (for example, T4 or any other Tesla board), you may use NVIDIA driver release 396, 384.111+ or 410. However, if you are running on Tesla (Tesla V100, Tesla P4, Tesla P40, or Tesla P100), you may use NVIDIA driver release 384.111+ or 410. The CUDA driver’s compatibility package only supports particular drivers. For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

GPU Requirements

Release 19.09 supports CUDA compute capability 6.0 and higher. This corresponds to GPUs in the Pascal, Volta, and Turing families. Specifically, for a list of GPUs that this compute capability corresponds to, see CUDA GPUs. For additional support details, see Deep Learning Frameworks Support Matrix.

Key Features and Enhancements

This Optimized Deep Learning Framework release includes the following key features and enhancements.

- NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 19.09 is based on Apache MXNet 1.5.0 and includes upstream commits up through commit 006486af3 from August 28, 2019.
- Significantly improved BERT pretraining performance when using GluonNLP.
- Improved performance of backward computation of softmax operator.
- Improved performance of backward computation of FullyConnected operator when using bias.
- Added FP16 support for TopK operator.
- Added automatic fusion of operators for BatchNorm + activation, BatchnNormAddRelu and NormConvolution.
- Latest version of NVIDIA cuDNN 7.6.3
- Latest version of Horovod 0.18.1
- Latest version of GluonNLP Toolkit 0.8.1
Tensor Core Examples

These examples focus on achieving the best performance and convergence from NVIDIA Volta Tensor Cores by using the latest deep learning example networks for training. Each example model trains with mixed precision Tensor Cores on Volta, therefore you can get results much faster than training without tensor cores. This model is tested against each NGC monthly container release to ensure consistent accuracy and performance over time. This container includes the following tensor core examples.

- The ResNet50 v1.5 model is a slightly modified version of the original ResNet50 v1 model that trains to a greater accuracy. This model script is available on GitHub as well as NVIDIA GPU Cloud [NGC].

Automatic Mixed Precision (AMP)

Training deep learning networks is a very computationally intensive task. Novel model architectures tend to have an increasing number of layers and parameters, which slows down training. Fortunately, new generations of training hardware as well as software optimizations make training these new models a feasible task.

Most of the hardware and software training optimization opportunities involve exploiting lower precision like FP16 in order to utilize the Tensor Cores available on new Volta and Turing GPUs. While training in FP16 showed great success in image classification tasks, other more complicated neural networks typically stayed in FP32 due to difficulties in applying the FP16 training guidelines that are needed to ensure proper model training.

That is where AMP (Automatic Mixed Precision) comes into play- it automatically applies the guidelines of FP16 training, using FP16 precision where it provides the most benefit, while conservatively keeping in full FP32 precision operations unsafe to do in FP16.

The NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet AMP tutorial, located in /opt/mxnet/nvidia-examples/AMP/AMP_tutorial.md inside this container, shows how to get started with mixed precision training using AMP for Apache MXNet, using by example the SSD network from GluonCV.

For more information about AMP, see the Training With Mixed Precision Guide.

Known Issues

- The Apache MXNet KVStore GPU peer-to-peer communication tree discovery, as of release 18.09, is not compatible with DGX-1V. Only users that set the environment variable
MXNET_KVSTORE_USETREE=1 will experience issues, which will be resolved in a subsequent release. Issue tracked under 13341.

- There is a known performance issue with the resnet50v1.5 MXNet 19.09 script on T4 and Titan RTX GPUs. The NHWC_BATCNORM_LAUNCH_MARGIN environment variable contains a sub-optimal value. As a workaround, remove line 78 (`os.environ['NHWC_BATCNORM_LAUNCH_MARGIN'] = "64"`) from the `/opt/mxnet/nvidia-examples/resnet50v1.5/runner` file.

- The default setting of the environment variable MXNET_GPU_COPY_NTHREADS=1 in the container may not be optimal for all networks. Networks with a high ratio of parameters and computation, like AlexNet, may achieve greater multi-GPU training speeds with the setting MXNET_GPU_COPY_NTHREADS=2. Users are encouraged to try this setting for their own use case.

- There is a known issue in the BERT QA demo, where for some values of sequence length the inference may fail with CUDA Driver: operation failed due to a previous error during captureerror. To test those values of sequence length, change the line `export MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH=1` to `export MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH=0` in the `test_bert_inference` script inside the demo directory.
Chapter 13. Release 19.08

The container image for NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 19.08, is available on NGC.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- Ubuntu 18.04 including Python 3.5
- NVIDIA CUDA 10.1.243 including cuBLAS 10.2.1.243
- NVIDIA cuDNN 7.6.2
- NVIDIA NCCL 2.4.8 (optimized for NVLink™)
- ONNX exporter 0.1 for CNN classification models

Note: The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- Amazon Labs Sockeye sequence-to-sequence framework 1.18.99 (for machine translation)
- MLNX_OFED 4.0
- OpenMPI 3.1.4
- Horovod 0.16.4
- Nsight Compute 10.1.168
- Nsight Systems 2019.3.7.9
- TensorRT 5.1.5
- GluonCV Toolkit 0.4
- GluonNLP Toolkit 0.7.1
- DALI 0.12.0 Beta
- Tensor Core optimized example:
  - ResNet-50 v1.5
» Jupyter and JupyterLab:
  » Jupyter Client 5.3.1
  » Jupyter Core 4.5.0
  » Jupyter Notebook 6.0.0
  » JupyterLab 1.0.5
  » JupyterLab Server 1.0.0
  » Jupyter-TensorBoard

**Driver Requirements**

Release 19.08 is based on NVIDIA CUDA 10.1.243, which requires NVIDIA Driver release 418.87. However, if you are running on Tesla (Tesla V100, Tesla P4, Tesla P40, or Tesla P100), you may use NVIDIA driver release 384.111+ or 410. The CUDA driver’s compatibility package only supports particular drivers. For a complete list of supported drivers, see the [CUDA Application Compatibility](#) topic. For more information, see [CUDA Compatibility and Upgrades](#).

**GPU Requirements**

Release 19.08 supports CUDA compute capability 6.0 and higher. This corresponds to GPUs in the Pascal, Volta, and Turing families. Specifically, for a list of GPUs that this compute capability corresponds to, see [CUDA GPUs](#). For additional support details, see [Deep Learning Frameworks Support Matrix](#).

**Key Features and Enhancements**

This Optimized Deep Learning Framework release includes the following key features and enhancements.

» NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 19.08 is based on Apache MXNet 1.5.0 and includes upstream commits up through commit 75a9e187d from June 27, 2019.
» Latest version of NVIDIA CUDA 10.1.243 including cuBLAS 10.2.1.243
» Latest version of NVIDIA cuDNN 7.6.2
» Latest version of NVIDIA NCCL 2.4.8
» Latest version of Horovod 0.16.4
» Added GluonCV Toolkit 0.4 and GluonNLP Toolkit 0.7.1 to the container.
» Latest version of DALI 0.12.0 Beta
» Latest version of OpenMPI 3.1.4
» Latest version of Nsight Systems 2019.3.7.9
» Latest version of MLNX OFED +4.0
» Latest versions of Jupyter Notebook 6.0.0 and JupyterLab 1.0.5
» Included support for length parameter in softmax [PR 15159]
Improved performance of forward computation of the `softmax` operator

Improved performance of forward computation of the `FullyConnected` operator when using bias.

Added a demo for fast BERT QA inference using Apache MXNet in `/workspace/examples/gluon/bert_inference`.

Improved latency of inference done with networks using the `Dropout` operator.

Added experimental support for CUDA Graphs inside Apache MXNet guarded by the `MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH` environment variable.

Ubuntu 18.04 with July 2019 updates

Tensor Core Examples

These examples focus on achieving the best performance and convergence from NVIDIA Volta Tensor Cores by using the latest deep learning example networks for training.

Each example model trains with mixed precision Tensor Cores on Volta, therefore you can get results much faster than training without tensor cores. This model is tested against each NGC monthly container release to ensure consistent accuracy and performance over time. This container includes the following tensor core examples.

- The `ResNet50 v1.5 model` is a slightly modified version of the original `ResNet50 v1 model` that trains to a greater accuracy. This model script is available on [GitHub](https://github.com) as well as [NVIDIA GPU Cloud (NGC)](https://nvidia.com/).

Automatic Mixed Precision (AMP)

Training deep learning networks is a very computationally intensive task. Novel model architectures tend to have an increasing number of layers and parameters, which slows down training. Fortunately, new generations of training hardware as well as software optimizations make training these new models a feasible task.

Most of the hardware and software training optimization opportunities involve exploiting lower precision like FP16 in order to utilize the Tensor Cores available on new Volta and Turing GPUs. While training in FP16 showed great success in image classification tasks, other more complicated neural networks typically stayed in FP32 due to difficulties in applying the FP16 training guidelines that are needed to ensure proper model training.

That is where AMP (Automatic Mixed Precision) comes into play- it automatically applies the guidelines of FP16 training, using FP16 precision where it provides the most benefit, while conservatively keeping in full FP32 precision operations unsafe to do in FP16.

The NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet AMP tutorial, located in `/opt/mxnet/nvidia-examples/AMP/AMP_tutorial.md` inside this container, shows how to get started with mixed precision training using AMP for Apache MXNet, using by example the SSD network from GluonCV.

For more information about AMP, see the [Training With Mixed Precision Guide](https://docs.nvidia.com/deeplearning/mxnet/user-guide/index.html#training-with-mixed-precision).
Known Issues

- The Apache MXNet KVStore GPU peer-to-peer communication tree discovery, as of release 18.09, is not compatible with DGX-1V. Only users that set the environment variable `MXNET_KVSTORE_USETREE=1` will experience issues, which will be resolved in a subsequent release. Issue tracked under 13341.

- The default setting of the environment variable `MXNET_GPU_COPY_NTHREADS=1` in the container may not be optimal for all networks. Networks with a high ratio of parameters and computation, like AlexNet, may achieve greater multi-GPU training speeds with the setting `MXNET_GPU_COPY_NTHREADS=2`. Users are encouraged to try this setting for their own use case.

- There is a known issue in the BERT QA demo, where for some values of sequence length the inference may fail with CUDA Driver: operation failed due to a previous error during captureerror. To test those values of sequence length, change the line `export MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH=1` to `export MXNET_EXPERIMENTAL_ENABLE_CUDA_GRAPH=0` in the `test_bert_inference` script inside the demo directory.
Chapter 14. Release 19.07

The container image for NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 19.07, is available on NGC.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in `/opt/mxnet`. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu 18.04** including **Python 3.5**
- **NVIDIA CUDA 10.1.168** including **cuBLAS 10.2.0.168**
- **NVIDIA cuDNN 7.6.1**
- **NVIDIA NCCL 2.4.7** (optimized for NVLink™)
- **ONNX exporter 0.1** for CNN classification models

  **Note:** The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the **master branch**.

- **Amazon Labs Sockeye sequence-to-sequence framework 1.18.99** (for machine translation)
- **MLNX_OFED +3.4**
- **OpenMPI 3.1.3**
- **Horovod 0.16.2**
- **TensorRT 5.1.5**
- **DALI 0.11.0 Beta**
- Tensor Core optimized example:
  - **ResNet-50 v1.5**
- Jupyter and JupyterLab:
  - **Jupyter Client 5.3.1**
  - **Jupyter Core 4.5.0**
  - **Jupyter Notebook 5.7.8**
Driver Requirements

Release 19.07 is based on NVIDIA CUDA 10.1.168, which requires NVIDIA Driver release 418.67. However, if you are running on Tesla (Tesla V100, Tesla P4, Tesla P40, or Tesla P100), you may use NVIDIA driver release 384.111+ or 410. The CUDA driver’s compatibility package only supports particular drivers. For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

GPU Requirements

Release 19.07 supports CUDA compute capability 6.0 and higher. This corresponds to GPUs in the Pascal, Volta, and Turing families. Specifically, for a list of GPUs that this compute capability corresponds to, see CUDA GPUs. For additional support details, see Deep Learning Frameworks Support Matrix.

Key Features and Enhancements

This Optimized Deep Learning Framework release includes the following key features and enhancements.

- NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 19.07 is based on Apache MXNet 1.5.0.rc2 and includes upstream commits up through commit 75a9e187d from June 27, 2019.
- Pointwise operators fusion was introduced to improve performance of training and inference and is controlled by the MXNET_USE_FUSION environment variable (on by default).
- Latest version of NVIDIA cuDNN 7.6.1
- Latest version of MLNX_OFED+3.4
- Latest versions of Jupyter Client 5.3.1, Jupyter Core 4.5.0, JupyterLab 1.0.0 and JupyterLab Server 1.0.0, including Jupyter-TensorBoard integration.
- Latest version of DALI 0.11.0 Beta
- Latest version of Amazon Labs Sockeye sequence-to-sequence framework 1.18.99
- Latest version of Ubuntu 18.04

Tensor Core Examples

These examples focus on achieving the best performance and convergence from NVIDIA Volta Tensor Cores by using the latest deep learning example networks for training.
Each example model trains with mixed precision Tensor Cores on Volta, therefore you can get results much faster than training without tensor cores. This model is tested against each NGC monthly container release to ensure consistent accuracy and performance over time. This container includes the following tensor core examples.

- **ResNet50 v1.5 model.** The ResNet50 v1.5 model is a slightly modified version of the original ResNet50 v1 model that trains to a greater accuracy.

**Automatic Mixed Precision (AMP)**

Training deep learning networks is a very computationally intensive task. Novel model architectures tend to have an increasing number of layers and parameters, which slows down training. Fortunately, new generations of training hardware as well as software optimizations make training these new models a feasible task.

Most of the hardware and software training optimization opportunities involve exploiting lower precision like FP16 in order to utilize the Tensor Cores available on new Volta and Turing GPUs. While training in FP16 showed great success in image classification tasks, other more complicated neural networks typically stayed in FP32 due to difficulties in applying the FP16 training guidelines that are needed to ensure proper model training.

That is where AMP (Automatic Mixed Precision) comes into play- it automatically applies the guidelines of FP16 training, using FP16 precision where it provides the most benefit, while conservatively keeping in full FP32 precision operations unsafe to do in FP16.

The NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet AMP tutorial, located in `/opt/mxnet/nvidia-examples/AMP/AMP_tutorial.md` inside this container, shows how to get started with mixed precision training using AMP for Apache MXNet, using by example the SSD network from GluonCV.

For more information about AMP, see the [Training With Mixed Precision Guide](#).

**Known Issues**

- The Apache MXNet KVStore GPU peer-to-peer communication tree discovery, as of release 18.09, is not compatible with DGX-1V. Only users that set the environment variable `MXNET_KVSTORE_USETREE=1` will experience issues, which will be resolved in a subsequent release. Issue tracked under [13341](#).

- The default setting of the environment variable `MXNET_GPU_COPY_NTHREADS=1` in the container may not be optimal for all networks. Networks with a high ratio of parameters and computation, like AlexNet, may achieve greater multi-GPU training speeds with the setting `MXNET_GPU_COPY_NTHREADS=2`. Users are encouraged to try this setting for their own use case.

- There is a known issue with the pointwise fusion when calculating the gradient of the erf function. When training a network containing the erf operator, set the `MXNET_USE_FUSION=0` environment variable.
Chapter 15. Release 19.06

The container image for NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 19.06, is available on NGC.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu 16.04** including **Python 3.5**
- **NVIDIA CUDA 10.1.168** including **cuBLAS 10.2.0.168**
- **NVIDIA cuDNN 7.6.0**
- **NVIDIA NCCL 2.4.7** (optimized for NVLink™)
- **ONNX exporter 0.1** for CNN classification models

Note: The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- **Amazon Labs Sockeye sequence-to-sequence framework 1.18.61** (for machine translation)
- **OpenMPI 3.1.3**
- **Horovod 0.16.2**
- **TensorRT 5.1.5**
- **DALI 0.10.0 Beta**
- Tensor Core optimized example:
  - **ResNet-50 v1.5**
- Jupyter and JupyterLab:
  - **Jupyter Client 5.2.4**
  - **Jupyter Core 4.4.0**
  - **Jupyter Notebook 5.7.8**
  - **JupyterLab 0.35.6**
JupyterLab Server 0.2.0

**Driver Requirements**

Release 19.06 is based on **NVIDIA CUDA 10.1.168**, which requires **NVIDIA Driver** release 418.xx. However, if you are running on Tesla (Tesla V100, Tesla P4, Tesla P40, or Tesla P100), you may use NVIDIA driver release 384.111+ or 410. The CUDA driver’s compatibility package only supports particular drivers. For a complete list of supported drivers, see the **CUDA Application Compatibility** topic. For more information, see **CUDA Compatibility and Upgrades**.

**GPU Requirements**

Release 19.06 supports CUDA compute capability 6.0 and higher. This corresponds to GPUs in the Pascal, Volta, and Turing families. Specifically, for a list of GPUs that this compute capability corresponds to, see **CUDA GPUs**. For additional support details, see **Deep Learning Frameworks Support Matrix**.

**Key Features and Enhancements**

This Optimized Deep Learning Framework release includes the following key features and enhancements.

- NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 19.06 is based on **Apache MXNet 1.4.1**. The image also contains upstream commits up through commit 01cf29d0a from May 26, 2019 and has much of the functionality from **Apache MXNet 1.5.0.rc0**.
- Latest version of **NVIDIA CUDA 10.1.168** including **cuBLAS 10.2.0.168**
- Latest version of **NVIDIA NCCL 2.4.7**
- Latest version of **DALI 0.10.0 Beta**
- Latest version of **JupyterLab 0.35.6**
- Latest version of **Horovod 0.16.2**
- Ubuntu 16.04 with May 2019 updates [see Announcements]

**Tensor Core Examples**

These examples focus on achieving the best performance and convergence from NVIDIA Volta Tensor Cores by using the latest deep learning example networks for training.

Each example model trains with mixed precision Tensor Cores on Volta, therefore you can get results much faster than training without tensor cores. This model is tested against each NGC monthly container release to ensure consistent accuracy and performance over time. This container includes the following tensor core examples.

- **ResNet50 v1.5 model**. The ResNet50 v1.5 model is a slightly modified version of the original **ResNet50 v1 model** that trains to a greater accuracy.
Automatic Mixed Precision (AMP)

Training deep learning networks is a very computationally intensive task. Novel model architectures tend to have an increasing number of layers and parameters, which slows down training. Fortunately, new generations of training hardware as well as software optimizations make training these new models a feasible task.

Most of the hardware and software training optimization opportunities involve exploiting lower precision like FP16 in order to utilize the Tensor Cores available on new Volta and Turing GPUs. While training in FP16 showed great success in image classification tasks, other more complicated neural networks typically stayed in FP32 due to difficulties in applying the FP16 training guidelines that are needed to ensure proper model training.

That is where AMP (Automatic Mixed Precision) comes into play— it automatically applies the guidelines of FP16 training, using FP16 precision where it provides the most benefit, while conservatively keeping in full FP32 precision operations unsafe to do in FP16.

The NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet AMP tutorial, located in /opt/mxnet/nvidia-examples/AMP/AMP_tutorial.md inside this container, shows how to get started with mixed precision training using AMP for Apache MXNet, using by example the SSD network from GluonCV.

For more information about AMP, see the Training With Mixed Precision Guide.

Announcements

In the next release, we will no longer support Ubuntu 16.04. Release 19.07 will instead support Ubuntu 18.04.

Known Issues

- The Apache MXNet KVStore GPU peer-to-peer communication tree discovery, as of release 18.09, is not compatible with DGX-1V. Only users that set the environment variable MXNET_KVSTORE_USETREE=1 will experience issues, which will be resolved in a subsequent release. Issue tracked under 13341.
- The default setting of the environment variable MXNET_GPU_COPY_NTHREADS=1 in the container may not be optimal for all networks. Networks with a high ratio of parameters and computation, like AlexNet, may achieve greater multi-GPU training speeds with the setting MXNET_GPU_COPY_NTHREADS=2. Users are encouraged to try this setting for their own use case.
Chapter 16. Release 19.05

The container image for NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 19.05, is available on NGC.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- Ubuntu 16.04 including Python 3.5
- NVIDIA CUDA 10.1 Update 1 including cuBLAS 10.1 Update 1
- NVIDIA cuDNN 7.6.0
- NVIDIA NCCL 2.4.6 (optimized for NVLink™)
- ONNX exporter 0.1 for CNN classification models

Note: The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- Amazon Labs Sockeye sequence-to-sequence framework 1.18.61 (for machine translation)
- OpenMPI 3.1.3
- Horovod 0.16.1
- TensorRT 5.1.5
- DALI 0.9.1 Beta
- Tensor Core optimized example:
  - ResNet-50 v1.5
- Jupyter and JupyterLab:
  - Jupyter Client 5.2.4
  - Jupyter Core 4.4.0
  - JupyterLab 0.35.4
  - JupyterLab Server 0.2.0
Driver Requirements

Release 19.05 is based on CUDA 10.1 Update 1, which requires NVIDIA Driver release 418.xx. However, if you are running on Tesla (Tesla V100, Tesla P4, Tesla P40, or Tesla P100), you may use NVIDIA driver release 384.111+ or 410. The CUDA driver’s compatibility package only supports particular drivers. For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

GPU Requirements

Release 19.05 supports CUDA compute capability 6.0 and higher. This corresponds to GPUs in the Pascal, Volta, and Turing families. Specifically, for a list of GPUs that this compute capability corresponds to, see CUDA GPUs. For additional support details, see Deep Learning Frameworks Support Matrix.

Key Features and Enhancements

This Optimized Deep Learning Framework release includes the following key features and enhancements.

- NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 19.05 is based on MXNet 1.4.0commit 87c7addcd from February 12, 2019.
- Latest version of NVIDIA CUDA 10.1 Update 1 including cuBLAS 10.1 Update 1
- Latest version of NVIDIA cuDNN 7.6.0
- Latest version of TensorRT 5.1.5
- Latest version of DALI 0.9.1 Beta
- Updated to Horovod 0.16.1
- New experimental NormalizedConvolution operator (see below).
- Ubuntu 16.04 with April 2019 updates

NormalizedConvolution Operator

NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet 19.05 includes a new operator [NormalizedConvolution] to improve training speeds of CNN’s like ResNet-50. The NormalizedConvolution operator combines the functions of BatchNorm and Convolution into one operator to reduce data transfers to and from GPU global memory. For more information regarding its use through the ResNet-50 sample model script, see ./example/image-classification/symbols/resnet-v1b-normconv-f1.py.

NormalizedConvolution is supported by new APIs of cuDNN v7.6, however, its Python API is experimental until it becomes incorporated into upstream Apache MXNet.
**Tensor Core Examples**

These examples focus on achieving the best performance and convergence from NVIDIA Volta Tensor Cores by using the latest deep learning example networks for training.

Each example model trains with mixed precision Tensor Cores on Volta, therefore you can get results much faster than training without tensor cores. This model is tested against each NGC monthly container release to ensure consistent accuracy and performance over time. This container includes the following tensor core examples.

- An implementation of the ResNet-50 model. The ResNet50 v1.5 model is a slightly modified version of the original ResNet50 v1 model that trains to a greater accuracy.

**Automatic Mixed Precision (AMP)**

Training deep learning networks is a very computationally intensive task. Novel model architectures tend to have increasing number of layers and parameters, which slows down training. Fortunately, new generations of training hardware as well as software optimizations, make it a feasible task.

However, where most of the (both hardware and software) optimization opportunities exists is in exploiting lower precision (like FP16) to, for example, utilize tensor cores available on new Volta and Turing GPUs. While training in FP16 showed great success in image classification tasks, other more complicated neural networks typically stayed in FP32 due to difficulties in applying the FP16 training guidelines.

That is where AMP (Automatic Mixed Precision) comes into play. It automatically applies the guidelines of FP16 training, using FP16 precision where it provides the most benefit, while conservatively keeping in full FP32 precision operations unsafe to do in FP16.

The NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet AMP tutorial, located in `/opt/mxnet/nvidia-examples/AMP/AMP_tutorial.md` inside this container, shows you how to get started with mixed precision training using AMP for Apache MXNet. As an example of a network we will use SSD network from GluonCV.

For more information about AMP, see the Training With Mixed Precision Guide.

**Known Issues**

- The Apache MXNet KVStore GPU peer-to-peer communication tree discovery, as of release 18.09, is not compatible with DGX-1V. Only users that set the environment variable `MXNET_KVSTORE_USETREE=1` will experience issues, which will be resolved in a subsequent release. Issue tracked under [13341](#).

- The default setting of the environment variable `MXNET_GPU_COPY_NTHREADS=1` in the container may not be optimal for all networks. Networks with a high ratio of parameters and computation, like AlexNet, may achieve greater multi-GPU training speeds with the
setting `MXNET_GPU_COPY_NTHREADS=2`. Users are encouraged to try this setting for their own use case.
Chapter 17. Release 19.04

The container image for NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 19.04, is available on NGC.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- Ubuntu 16.04 including Python 3.5
- NVIDIA CUDA 10.1.105 including cuBLAS 10.1.0.105
- NVIDIA cuDNN 7.5.0
- NVIDIA NCCL 2.4.6 (optimized for NVLink™)
- ONNX exporter 0.1 for CNN classification models

Note: The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- Amazon Labs Sockeye sequence-to-sequence framework 1.18.61 (for machine translation)
- OpenMPI 3.1.3
- Horovod 0.13.11
- TensorRT 5.1.2
- DALI 0.8.1 Beta
- Tensor Core optimized example:
  - ResNet-50 v1.5
- Jupyter and JupyterLab:
  - Jupyter Client 5.2.4
  - Jupyter Core 4.4.0
  - JupyterLab 0.35.4
  - JupyterLab Server 0.2.0
Driver Requirements

Release 19.04 is based on CUDA 10.1, which requires NVIDIA Driver release 418.xx.x+. However, if you are running on Tesla (Tesla V100, Tesla P4, Tesla P40, or Tesla P100), you may use NVIDIA driver release 384.111+ or 410. The CUDA driver’s compatibility package only supports particular drivers. For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

GPU Requirements

Release 19.04 supports CUDA compute capability 6.0 and higher. This corresponds to GPUs in the Pascal, Volta, and Turing families. Specifically, for a list of GPUs that this compute capability corresponds to, see CUDA GPUs. For additional support details, see Deep Learning Frameworks Support Matrix.

Key Features and Enhancements

This Optimized Deep Learning Framework release includes the following key features and enhancements.

‣ NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 19.04 is based on Apache MXNet 1.4.0commit 87c7addcd from February 12, 2019.
‣ Latest version of NVIDIA NCCL 2.4.6
‣ Latest version of cuBLAS 10.1.0.105
‣ Latest version of DALI 0.8.1 Beta
‣ Added support for Apache MXNet Automatic Mixed Precision; see below for more information.
‣ Ubuntu 16.04 with March 2019 updates

Tensor Core Examples

These examples focus on achieving the best performance and convergence from NVIDIA Volta Tensor Cores by using the latest deep learning example networks for training.

Each example model trains with mixed precision Tensor Cores on Volta, therefore you can get results much faster than training without tensor cores. This model is tested against each NGC monthly container release to ensure consistent accuracy and performance over time. This container includes the following tensor core examples.

‣ An implementation of the ResNet-50 model. The ResNet50 v1.5 model is a slightly modified version of the original ResNet50 v1 model that trains to a greater accuracy.

Automatic Mixed Precision (AMP)

Training deep learning networks is a very computationally intensive task. Novel model architectures tend to have increasing number of layers and parameters, which slows down
training. Fortunately, new generations of training hardware as well as software optimizations, make it a feasible task.

However, where most of the (both hardware and software) optimization opportunities exists is in exploiting lower precision (like FP16) to, for example, utilize tensor cores available on new Volta and Turing GPUs. While training in FP16 showed great success in image classification tasks, other more complicated neural networks typically stayed in FP32 due to difficulties in applying the FP16 training guidelines.

That is where AMP (Automatic Mixed Precision) comes into play. It automatically applies the guidelines of FP16 training, using FP16 precision where it provides the most benefit, while conservatively keeping in full FP32 precision operations unsafe to do in FP16.

The NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet AMP tutorial, located in the /opt/mxnet/nvidia-examples/AMP/AMP_tutorial.md directory of this container, shows you how to get started with mixed precision training using AMP for Apache MXNet. As an example of a network we will use SSD network from GluonCV.

For more information about AMP, see the Training With Mixed Precision Guide.

Known Issues

‣ The Apache MXNet KVStore GPU peer-to-peer communication tree discovery, as of release 18.09, is not compatible with DGX-1V. Only users that set the environment variable MXNET_KVSTORE_USETREE=1 will experience issues, which will be resolved in a subsequent release. Issue tracked under 13341.

‣ The default setting of the environment variable MXNET_GPU_COPY_NTHREADS=1 in the container may not be optimal for all networks. Networks with a high ratio of parameters and computation, like AlexNet, may achieve greater multi-GPU training speeds with the setting MXNET_GPU_COPY_NTHREADS=2. Users are encouraged to try this setting for their own use case.
Chapter 18. Release 19.03

The container image for NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 19.03, is available on NGC.

### Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in `/opt/mxnet`. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu 16.04 including Python 3.5**
- **NVIDIA CUDA 10.1.105 including cuBLAS 10.1.105**
- **NVIDIA cuDNN 7.5.0**
- **NVIDIA NCCL 2.4.3** (optimized for NVLink™)
- **ONNX exporter 0.1** for CNN classification models

*Note:* The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- Amazon Labs Sockeye sequence-to-sequence framework 1.18.61 (for machine translation)
- **OpenMPI 3.1.3**
- **Horovod 0.13.11**
- **TensorRT 5.1.2**
- **DALI 0.7 Beta**
- Tensor Core optimized example:
  - ResNet-50 v1.5
- Jupyter and JupyterLab:
  - Jupyter Client 5.2.4
  - Jupyter Core 4.4.0
  - JupyterLab 0.35.4
  - JupyterLab Server 0.2.0
Driver Requirements

Release 19.03 is based on CUDA 10.1, which requires NVIDIA Driver release 418.xx+. However, if you are running on Tesla (Tesla V100, Tesla P4, Tesla P40, or Tesla P100), you may use NVIDIA driver release 384.111+ or 410. The CUDA driver’s compatibility package only supports particular drivers. For a complete list of supported drivers, see the CUDA Application Compatibility topic. For more information, see CUDA Compatibility and Upgrades.

GPU Requirements

Release 19.03 supports CUDA compute capability 6.0 and higher. This corresponds to GPUs in the Pascal, Volta, and Turing families. Specifically, for a list of GPUs that this compute capability corresponds to, see CUDA GPUs. For additional support details, see Deep Learning Frameworks Support Matrix.

Key Features and Enhancements

This Optimized Deep Learning Framework release includes the following key features and enhancements.

‣ NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 19.03 is based on Apache MXNet 1.4.0.
‣ Latest version of NVIDIA CUDA 10.1.105 including cuBLAS 10.1.105
‣ Latest version of NVIDIA cuDNN 7.5.0
‣ Latest version of NVIDIA NCCL 2.4.3
‣ Latest version of DALI 0.7 Beta
‣ Latest version of TensorRT 5.1.2
‣ Optimized NMS operator performance
‣ Ubuntu 16.04 with February 2019 updates

Tensor Core Examples

These examples focus on achieving the best performance and convergence from NVIDIA Volta Tensor Cores by using the latest deep learning example networks for training.

Each example model trains with mixed precision Tensor Cores on Volta, therefore you can get results much faster than training without tensor cores. This model is tested against each NGC monthly container release to ensure consistent accuracy and performance over time. This container includes the following tensor core examples.

‣ An implementation of the ResNet-50 model. The ResNet50 v1.5 model is a slightly modified version of the original ResNet50 v1 model that trains to a greater accuracy.
Known Issues

- The Apache MXNet KVStore GPU peer-to-peer communication tree discovery, as of release 18.09, is not compatible with DGX-1V. Only users that set the environment variable `MXNET_KVSTORE_USETREE=1` will experience issues, which will be resolved in a subsequent release. Issue tracked under 13341.

- The default setting of the environment variable `MXNET_GPU_COPY_NTHREADS=1` in the container may not be optimal for all networks. Networks with a high ratio of parameters and computation, like AlexNet, may achieve greater multi-GPU training speeds with the setting `MXNET_GPU_COPY_NTHREADS=2`. Users are encouraged to try this setting for their own use case.

- If using or upgrading to a 3-part-version driver, for example, a driver that takes the format of `xxx.yy.zz`, you will receive a `Failed to detect NVIDIA driver version` message. This is due to a known bug in the entry point script’s parsing of 3-part driver versions. This message is non-fatal and can be ignored. This will be fixed in the 19.04 release.
Chapter 19. Release 19.02

The container image for NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 19.02, is available.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in `/opt/mxnet`. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu 16.04 including Python 3.5**
- **NVIDIA CUDA 10.0.130 including CUDA® Basic Linear Algebra Subroutines library™ (cuBLAS) 10.0.130**
- **NVIDIA CUDA® Deep Neural Network library™ (cuDNN) 7.4.2**
- **NVIDIA Collective Communications Library (NCCL) 2.3.7 (optimized for NVLink™)**
- **ONNX exporter 0.1** for CNN classification models

Note: The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- **Amazon Labs Sockeye sequence-to-sequence framework 1.18.61** [for machine translation]
- **OpenMPI 3.1.3**
- **Horovod 0.13.11**
- **TensorRT 5.0.2**
- **DALI 0.6.1 Beta**
- **Tensor Core optimized example:**
  - **ResNet50 v1.5**
- **Jupyter and JupyterLab:**
  - **Jupyter Client 5.2.4**
  - **Jupyter Core 4.4.0**
  - **JupyterLab 0.35.4**
Driver Requirements

Release 19.02 is based on CUDA 10, which requires NVIDIA Driver release 410.xx. However, if you are running on Tesla (Tesla V100, Tesla P4, Tesla P40, or Tesla P100), you may use NVIDIA driver release 384. For more information, see CUDA Compatibility and Upgrades.

GPU Requirements

Release 19.02 supports CUDA compute capability 6.0 and higher. This corresponds to GPUs in the Pascal, Volta, and Turing families. Specifically, for a list of GPUs that this compute capability corresponds to, see CUDA GPUs. For additional support details, see Deep Learning Frameworks Support Matrix.

Key Features and Enhancements

This Optimized Deep Learning Framework release includes the following key features and enhancements.

- NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 19.02 is based on 1.4.0.rc2.
- Latest version of DALI 0.6.1 Beta
- Added Jupyter and JupyterLab software in our packaged container.
- Latest version of jupyter_client 5.2.4
- Latest version of jupyter_core 4.4.0
- Added an image classification example in Gluon.
- Multiple enhancements to Gluon training speed with models hybridized with static.alloc=True setting.
- Added Python bindings for NVTX and CUDA profiler in the mxnet.cuda_utils package.
- Ubuntu 16.04 with January 2019 updates

Tensor Core Examples

These examples focus on achieving the best performance and convergence from NVIDIA Volta Tensor Cores by using the latest deep learning example networks for training. This container includes the following Tensor Core examples.

- An implementation of the ResNet50 model. The ResNet50 v1.5 model is a slightly modified version of the original ResNet50 v1 model that trains to a greater accuracy.

Known Issues

- The Apache MXNet KVStore GPU peer-to-peer communication tree discovery, as of release 18.09, is not compatible with DGX-1V. Only users that set the environment variable
MXNET_KVSTORE_USETREE=1 will experience issues, which will be resolved in a subsequent release. Issue tracked under 13341.

- The default setting of the environment variable MXNET_GPU_COPY_NTHREADS=1 in the container may not be optimal for all networks. Networks with a high ratio of parameters and computation, like AlexNet, may achieve greater multi-GPU training speeds with the setting MXNET_GPU_COPY_NTHREADS=2. Users are encouraged to try this setting for their own use case.

- If using or upgrading to a 3-part-version driver, for example, a driver that takes the format of xxx.yy.zz, you will receive a Failed to detect NVIDIA driver version. message. This is due to a known bug in the entry point script’s parsing of 3-part driver versions. This message is non-fatal and can be ignored. This will be fixed in the 19.04 release.
Chapter 20. Release 19.01

The container image for NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 19.01, is available.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in `/opt/mxnet`. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu** 16.04 including **Python 3.5**
- **NVIDIA CUDA 10.0.130** including **CUDA® Basic Linear Algebra Subroutines library™ (cuBLAS) 10.0.130**
- **NVIDIA CUDA® Deep Neural Network library™ (cuDNN) 7.4.2**
- **NCCL 2.3.7** (optimized for **NVLink™**)
- **ONNX exporter 0.1** for CNN classification models

Note: The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the **master branch**.

- **Amazon Labs Sockeye sequence-to-sequence framework 1.18.61** (for machine translation)
- **OpenMPI 3.1.3**
- **Horovod 0.15.1**
- **TensorRT 5.0.2**
- **DALI 0.6 Beta**
- Tensor Core optimized example:
  - **ResNet50 v1.5**
Driver Requirements

Release 19.01 is based on CUDA 10, which requires NVIDIA Driver release 410.xx. However, if you are running on Tesla (Tesla V100, Tesla P4, Tesla P40, or Tesla P100), you may use NVIDIA driver release 384. For more information, see CUDA Compatibility and Upgrades.

GPU Requirements

Release 19.01 supports CUDA compute capability 6.0 and higher. This corresponds to GPUs in the Pascal, Volta, and Turing families. Specifically, for a list of GPUs that this compute capability corresponds to, see CUDA GPUs. For additional support details, see Deep Learning Frameworks Support Matrix.

Key Features and Enhancements

This Optimized Deep Learning Framework release includes the following key features and enhancements.

- NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 19.01 is based on 1.4.0.rc0.
- Latest version of DALI 0.6 Beta
- Latest version of NVIDIA cuDNN 7.4.2
- Latest version of Sockeye
- Latest version of OpenMPI 3.1.3
- Improvements to out-of-the-box Horovod performance
- Added ResNet50 v1.5 Tensor Core optimized example.
- Ubuntu 16.04 with December 2018 updates

Tensor Core Examples

These examples focus on achieving the best performance and convergence from NVIDIA Volta Tensor Cores by using the latest deep learning example networks for training. This container includes the following Tensor Core examples.

- An implementation of the ResNet50 model. The ResNet50 v1 is a modified version of the original ResNet50 v1 model.

Known Issues

- The Apache MXNet KVStore GPU peer-to-peer communication tree discovery, as of release 18.09, is not compatible with DGX-1V. Only users that set the environment variable MXNET_KVSTORE_USETREE=1 will experience issues, which will be resolved in a subsequent release. Issue tracked under 13341.
- The default setting of the environment variable MXNET_GPU_COPY_NTHREADS=1 in the container may not be optimal for all networks. Networks with a high ratio of parameters

NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet
and computation, like AlexNet, may achieve greater multi-GPU training speeds with the setting `MXNET_GPU_COPY_NTHREADS=2`. Users are encouraged to try this setting for their own use case.

- If using or upgrading to a 3-part-version driver, for example, a driver that takes the format of `xxx.yy.zz`, you will receive a `Failed to detect NVIDIA driver version.` message. This is due to a known bug in the entry point script’s parsing of 3-part driver versions. This message is non-fatal and can be ignored. This will be fixed in the 19.04 release.

The container image for NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 18.12, is available.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu** 16.04 including **Python 3.5**
- **NVIDIA CUDA 10.0.130** including CUDA® Basic Linear Algebra Subroutines library™ (cuBLAS) 10.0.130
- **NVIDIA CUDA® Deep Neural Network library™ (cuDNN)** 7.4.1
- **NCCL 2.3.7** (optimized for NVLink™)
- **ONNX exporter 0.1** for CNN classification models

**Note:** The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- **Amazon Labs Sockeye sequence-to-sequence framework 1.18.28** (for machine translation)
- **OpenMPI 3.1.2**
- **Horovod 0.15.1**
- **TensorRT 5.0.2**
- **DALI 0.5.0 Beta**

Driver Requirements

Release 18.12 is based on CUDA 10, which requires **NVIDIA Driver** release 410.xx. However, if you are running on Tesla (Tesla V100, Tesla P4, Tesla P40, or Tesla P100), you may use NVIDIA driver release 384. For more information, see **CUDA Compatibility and Upgrades**.
GPU Requirements

Release 18.12 supports CUDA compute capability 6.0 and higher. This corresponds to GPUs in the Pascal, Volta, and Turing families. Specifically, for a list of GPUs that this compute capability corresponds to, see CUDA GPUs. For additional support details, see Deep Learning Frameworks Support Matrix.

Key Features and Enhancements

This Optimized Deep Learning Framework release includes the following key features and enhancements.

▪ NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 18.12 is based on 1.3.0, with all upstream changes from the MXNet master branch up to and including PR 13069.

▪ Improved handling of float32 datatype in examples/image-classification/train_imagenet_runner.

▪ Enabled NVIDIA Tools Extension SDK (NVTX) instrumentation.

▪ Improved speed of metrics computation during training, especially in the case of using TopKAccuracy metric.

▪ Latest version of DALI 0.5.0 Beta.

▪ Ubuntu 16.04 with November 2018 updates

Known Issues

▪ The Apache MXNet KVStore GPU peer-to-peer communication tree discovery, as of release 18.09, is not compatible with DGX-1V. Only users that set the environment variable MXNET_KVSTORE_USETREE=1 will experience issues, which will be resolved in a subsequent release. Issue tracked on https://github.com/apache/incubator-mxnet/issue/13341.

▪ The default setting of the environment variable MXNET_GPU_COPY_NTHREADS=1 in the container may not be optimal for all networks. Networks with a high ratio of parameters and computation, like AlexNet, may achieve greater multi-GPU training speeds with the setting MXNET_GPU_COPY_NTHREADS=2. Users are encouraged to try this setting for their own use case.
Chapter 22. Release 18.11

The container image for NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 18.11, is available.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu** 16.04 including **Python 3.5**
- **NVIDIA CUDA 10.0.130** including CUDA® Basic Linear Algebra Subroutines library™ (cuBLAS) 10.0.130
- **NVIDIA CUDA® Deep Neural Network library™ (cuDNN) 7.4.1**
- **NCCL** 2.3.7 [optimized for NVLink™]
- **ONNX exporter 0.1** for CNN classification models

![Note: The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.]

- **Amazon Labs Sockeye sequence-to-sequence framework 1.18.28** [for machine translation]
- **OpenMPI 3.1.2**
- **Horovod 0.15.1**
- **TensorRT 5.0.2**
- **DALI 0.4.1 Beta**

Driver Requirements

Release 18.11 is based on CUDA 10, which requires NVIDIA Driver release 410.xx. However, if you are running on Tesla [Tesla V100, Tesla P4, Tesla P40, or Tesla P100], you may use NVIDIA driver release 384. For more information, see [CUDA Compatibility and Upgrades](#).
Key Features and Enhancements

This Optimized Deep Learning Framework release includes the following key features and enhancements.

- **NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet** container image version 18.11 is based on 1.3.0, with all upstream changes from the Apache MXNet master branch up to and including PR 12537.

- Added fused `BatchNormAddRelu` operator to the Apache MXNet Symbol package (accessible via `mx.sym.BatchNormAddRelu`), which performs BatchNorm operation on data, sums the result with a tensor and performs Relu activation on the result of the sum. Currently it is limited to FP16 data type and NHWC data layout.

- Added `MXNET_EXEC_ENABLE_ADDTO` environment variable, which when set to 1 increases performance for some networks.

- Increased performance of `Batchnorm` and `Batchnorm+Relu` operators in FP16 and NHWC data format.

- Added support for multi-node via Horovod integration. Currently you can use it by specifying `horovod` type of KVStore.

- Added `MXNET_UPDATE_ON_KVSTORE` environment variable, which controls whether to update parameters using KVStore (default is 1 for KVStore device and 0 for KVStore horovod).

- Added aggregation of SGD updates which increases performance when update on KVStore is disabled.

- Increased performance when training with small batch sizes.

- Fixed a bug that prevented matrix multiplications to overlap with other computation, which increases performance for some networks.

- Fixed an issue which prevented score function to respect not-full batches of data.

- Added `resnet-v1b` as possible network in the `train_imagenet_runner` script.

- Latest version of NCCL 2.3.7.

- Latest version of NVIDIA cuDNN 7.4.1.

- Latest version of TensorRT 5.0.2

- Latest version of DALI 0.4.1 Beta.

- Ubuntu 16.04 with October 2018 updates.

Known Issues

- The Apache MXNet KVStore GPU peer-to-peer communication tree discovery, as of release 18.09, is not compatible with DGX-1V. Only users that set the environment variable `MXNET_KVSTORE_USETREE=1` will experience issues, which will be resolved in a subsequent release.
Apache MXNet ResNet50 regresses in FP32 performance. This issue should be fixed in a later release.
Chapter 23. Release 18.10

The container image of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 18.10, is available.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu** 16.04 including **Python 3.5**
- **NVIDIA CUDA 10.0.130** including **CUDA® Basic Linear Algebra Subroutines library™ (cuBLAS) 10.0.130**
- **NVIDIA CUDA® Deep Neural Network library™ (cuDNN) 7.4.0**
- **NCCL 2.3.6** (optimized for **NVLink™**)
- **ONNX exporter 0.1** for CNN classification models

**Note:** The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- **Amazon Labs Sockeye sequence-to-sequence framework 1.18.28** (for machine translation)
- **OpenMPI 3.1.2**
- **TensorRT 5.0.0 RC**
- **DALI 0.4 Beta**

Driver Requirements

Release 18.10 is based on CUDA 10, which requires **NVIDIA Driver** release 410.xx. However, if you are running on Tesla (Tesla V100, Tesla P4, Tesla P40, or Tesla P100), you may use NVIDIA driver release 384. For more information, see **CUDA Compatibility and Upgrades**.
Key Features and Enhancements

This Optimized Deep Learning Framework release includes the following key features and enhancements.

- **NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet** container image version 18.10 is based on 1.3.0, with all upstream changes from the Apache MXNet master branch up to and including PR 12537.
- Latest version of NCCL 2.3.6.
- Latest version of DALI 0.4 Beta.
- Added support for OpenMPI 3.1.2
- The known issue in the prior release regarding the variable maximum GPU global memory usage has been fixed. You should now see lower and stable global memory usage from run to run, and across GPUs in multi-GPU training.
- Ubuntu 16.04 with September 2018 updates

Known Issues

- The Apache MXNet KVStore GPU peer-to-peer communication tree discovery, as of release 18.09, is not compatible with DGX-1V. Only users that set the environment variable MXNET_KVSTORE_USETREE=1 will experience issues, which will be resolved in a subsequent release.
- Apache MXNet ResNet50 regresses in FP32 performance. This issue should be fixed in a later release.
Chapter 24. Release 18.09

The container image of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 18.09, is available.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu** 16.04 including **Python 3.5**
- **NVIDIA CUDA 10.0** including **CUDA® Basic Linear Algebra Subroutines library™ (cuBLAS)** 10.0
- **NVIDIA CUDA® Deep Neural Network library™ (cuDNN)** 7.3.0
- **NCCL** 2.3.4 (optimized for NVLink™)
- **ONNX exporter 0.1** for CNN classification models

**Note:** The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- **Amazon Labs Sockeye sequence-to-sequence framework 1.18.28** (for machine translation)
- **TensorRT 5.0.0 RC**
- **DALI 0.2 Beta**

Driver Requirements

Release 18.09 is based on CUDA 10, which requires NVIDIA Driver release 410.xx. However, if you are running on Tesla (Tesla V100, Tesla P4, Tesla P40, or Tesla P100), you may use NVIDIA driver release 384. For more information, see **CUDA Compatibility and Upgrades**.

Key Features and Enhancements

This Optimized Deep Learning Framework release includes the following key features and enhancements.
NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 18.09 is based on 1.3.0, with all upstream changes from the Apache MXNet master branch up to the creation point of the v1.3.x branch [PR 12301], plus all substantive cherry-picks from master that were included in the v1.3.0 release.

The demonstrator of mixed precision ResNet-50 training using the NHWC data layout has been expanded to work now on the Turing architecture in addition to Volta.

- Latest version of cuDNN 7.3.0.
- Latest version of CUDA 10.0 which includes support for DGX-2, Turing, and Jetson Xavier.
- Latest version of cuBLAS 10.0.
- Latest version of NCCL 2.3.4.
- Latest version of TensorRT 5.0.0 RC.
- Latest version of DALI 0.2 Beta.
- Ubuntu 16.04 with August 2018 updates

**Known Issues**

The multi-threaded nature of Apache MXNet model execution may result in a variable maximum usage of GPU global memory, as discussed in earlier release notes. Users that experience sporadic out-of-GPU-memory errors should experiment with setting the environment variable `MXNET_GPU_WORKER_NTHREADS=1` as a possible remedy. We anticipate the need for this experimentation will be removed in our next release.
Chapter 25. Release 18.08

The container image of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 18.08, is available.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in `/opt/mxnet`. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu 16.04**

  **Note:** Container image 18.08-py2 contains Python 2.7; 18.08-py3 contains Python 3.5.

- **NVIDIA CUDA 9.0.176** [see Errata section and 2.1] including **CUDA® Basic Linear Algebra Subroutines library™ (cuBLAS) 9.0.425**
- **NVIDIA CUDA® Deep Neural Network library™ (cuDNN) 7.2.1**
- **NCCL 2.2.13** [optimized for NVLink™]
- **ONNX exporter 0.1** for CNN classification models

  **Note:** The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- **Amazon Labs Sockeye sequence-to-sequence framework 1.18.28** [for machine translation]
- **TensorRT 4.0.1**
- **DALI 0.1.2 Beta**

Driver Requirements

Release 18.08 is based on CUDA 9, which requires NVIDIA Driver release 384.xx.
Key Features and Enhancements

This Optimized Deep Learning Framework release includes the following key features and enhancements.

‣ **NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet** container image version 18.08 is based on 1.2.0, with all upstream changes from the Apache MXNet master branch up to and including PR 11545.

‣ Latest version of **cuDNN 7.2.1**.

‣ Latest version of **DALI 0.1.2 Beta**.

‣ New demonstrator of increased mixed-precision ResNet-50 training speeds on Volta when processed end-to-end in the **NHWC** data layout. We are working to PR the code improvements to upstream Apache MXNet. To evaluate in the meantime, type `/opt/mxnet/examples/image_classification/trainImagenet_runner --batch-size N`. Substitute 256 for N on systems with GPUs having 32GB global memory (or 192 with 16GB GPUs) and prepare the imagenet database as directed in `nvidia-examples/imagenet_preparations`. Training images should be pre-resized to 480px shorter side and validation should be pre-resized to 256px shorter side. The script expects RecordIO files to be present in the `/data/imagenet/train-480-val-256-recordio/` directory.

‣ Ubuntu 16.04 with July 2018 updates

Announcements

Starting with the next major version of the CUDA release, we will no longer provide updated Python 2 containers and will only update Python 3 containers.

Known Issues

The multi-threaded nature of Apache MXNet model execution may result in a variable maximum usage of GPU global memory. Users that experience sporadic out-of-GPU-memory errors should experiment with setting the environment variable `MXNET_GPU_WORKER_NTHREADS=1` as a possible remedy. We anticipate the need for this experimentation will be removed in a subsequent release.
Chapter 26. Release 18.07

The container image of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 18.07, is available.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu** 16.04

  ![Note: Container image 18.07-py2 contains Python 2.7; 18.07-py3 contains Python 3.5.]

- NVIDIA CUDA 9.0.176 (see Errata section and 2.1) including CUDA® Basic Linear Algebra Subroutines library™ (cuBLAS) 9.0.425
- NVIDIA CUDA® Deep Neural Network library™ (cuDNN) 7.1.4
- NCCL 2.2.13 (optimized for NVLink™)
- ONNX exporter 0.1 for CNN classification models

  ![Note: The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.]

- Amazon Labs Sockeye sequence-to-sequence framework 1.18.23 (for machine translation)
- TensorRT 4.0.1
- DALI 0.1 Beta

Driver Requirements

Release 18.07 is based on CUDA 9, which requires NVIDIA Driver release 384.xx.
Key Features and Enhancements

This Optimized Deep Learning Framework release includes the following key features and enhancements.

- NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 18.07 is based on 1.2.0, with all upstream changes from the Apache MXNet master branch up to and including PR 11302.
- Added support for DALI 0.1 Beta.
- Latest version of CUDA® Basic Linear Algebra Subroutines library™ (cuBLAS) 9.0.425.
- Ubuntu 16.04 with June 2018 updates

Announcements

Starting with the next major version of the CUDA release, we will no longer provide updated Python 2 containers and will only update Python 3 containers.

Known Issues

- Some of the unit tests available in /opt/mxnet/tests/python/{gpu,unittest}/*.py require the SciPy Python library. For those that want to run the unit tests, first install SciPy by `pip install scipy`. There is no longer a need to specifically request the 1.0 version of SciPy.
- The multi-threaded nature of Apache MXNet model execution may result in a variable maximum usage of GPU global memory. Users that experience sporadic out-of-GPU-memory errors should experiment with setting the environment variable `MXNET_GPU_WORKER_NTHREADS=1` as a possible remedy. We anticipate the need for this experimentation will be removed in a subsequent release.
Chapter 27. Release 18.06

The container image of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 18.06, is available.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in `/opt/mxnet`. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu** 16.04

  - **Note:** Container image 18.06-py2 contains Python 2.7; 18.06-py3 contains Python 3.5.

  - **NVIDIA CUDA** 9.0.176 (see Errata section and 2.1) including CUDA® Basic Linear Algebra Subroutines library™ (cuBLAS) 9.0.333 (see section 2.3.1)
  - **NVIDIA CUDA® Deep Neural Network library™ (cuDNN) 7.1.4
  - **NCCL** 2.2.13 (optimized for NVLink™)
  - **ONNX exporter 0.1** for CNN classification models

  - **Note:** The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

  - **Amazon Labs Sockeye sequence-to-sequence framework 1.18.22** (for machine translation)
  - **TensorRT 4.0.1**

Driver Requirements

Release 18.06 is based on CUDA 9, which requires NVIDIA Driver release 384.xx.

Key Features and Enhancements

This Optimized Deep Learning Framework release includes the following key features and enhancements.
NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 18.06 is based on 1.2.0. Specifically, container image 18.06 has merged all commits on upstream Apache MXNet master, up to the creation point of the v1.2.0 branch, and all commits on that branch up to the 1.2.0 tag.

- Container includes TensorRT 4.0.1
- TensorRT integration examples for in-framework inference can be found in /workspace/examples/tensorrt-integration. This includes a LeNet-5 unit test and a ResNet-50 example.
- Support added for DALI iterators.
- Ubuntu 16.04 with May 2018 updates

Announcements

Starting with the next major version of CUDA release, we will no longer provide updated Python 2 containers and will only update Python 3 containers.

Known Issues

Some of the unit tests available in /opt/mxnet/tests/python/{gpu,unittest}/*.py require the SciPy Python library. For those that want to run the unit tests, first install the 1.0 version of SciPy by typing `pip install scipy==1.0`.

Note: The latest SciPy release, version 1.1, is not compatible with the unit tests.
Chapter 28. Release 18.05

The container image of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 18.05, is available.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu 16.04**

  - **Note:** Container image 18.05-py2 contains Python 2.7; 18.05-py3 contains Python 3.5.

- NVIDIA CUDA 9.0.176 [see Errata section and 2.1] including CUDA® Basic Linear Algebra Subroutines library™ (cuBLAS) 9.0.333 [see section 2.3.1]

- NVIDIA CUDA® Deep Neural Network library™ (cuDNN) 7.1.2

- NCCL 2.1.15 (optimized for NVLink™)

- ONNX exporter 0.1 for CNN classification models

  - **Note:** The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- Amazon Labs Sockeye sequence-to-sequence framework 1.18.13 [for machine translation]

Driver Requirements

Release 18.05 is based on CUDA 9, which requires NVIDIA Driver release 384.xx.

Key Features and Enhancements

This Optimized Deep Learning Framework release includes the following key features and enhancements.
NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 18.05 is based on Apache MXNet 1.1.0.

For this month, no upstream merges as we work toward incorporating the upcoming Apache MXNet 1.2.0 release.

Ubuntu 16.04 with April 2018 updates

Announcements

Starting with the next major version of CUDA release, we will no longer provide Python 2 containers and will only maintain Python 3 containers.

Known Issues

Those wishing to run the NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet unit tests under /opt/mxnet/tests/python should install SciPy using pip install scipy==1.0, as the recently available SciPy v1.1 is not compatible with all the unit tests. For more information, see Broken test_sparse_operator.test_sparse_mathematical_core with scipy 1.1.0.
The container image of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 18.04, is available.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu 16.04**

  Note: Container image 18.04-py2 contains Python 2.7; 18.04-py3 contains Python 3.5.

- **NVIDIA CUDA 9.0.176** (see Errata section and 2.1) including **CUDA® Basic Linear Algebra Subroutines library™ (cuBLAS) 9.0.333** (see section 2.3.1)
- **NVIDIA CUDA® Deep Neural Network library™ (cuDNN) 7.1.1**
- **NCCL 2.1.15** (optimized for **NVLink™**)
- **ONNX exporter 0.1** for CNN classification models

  Note: The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the **master branch**.

- **Amazon Labs Sockeye sequence-to-sequence framework 1.18.1** (for machine translation)

Driver Requirements

Release 18.04 is based on CUDA 9, which requires NVIDIA Driver release 384.xx.

Key Features and Enhancements

This NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet release includes the following key features and enhancements.
NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 18.04 is based on NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet 1.1.0.

For this month, no upstream merges as we develop a performant approach to Apache MXNet’s recent operator refactoring to a stateless imperative style.

ResNet-50 performance improvement based on the automatic fusion of `add_relu` and `copy_split` backward pass.

Latest version of NCCL 2.1.15

Ubuntu 16.04 with March 2018 updates

Announcements

Starting with the next major version of CUDA release, we will no longer provide Python 2 containers and will only maintain Python 3 containers.

Known Issues

There are no known issues in this release.
Chapter 30. Release 18.03

The container image of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 18.03, is available.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu** 16.04

  - **Note:** Container image 18.03-py2 contains Python 2.7; 18.03-py3 contains Python 3.5.

- **NVIDIA CUDA 9.0.176** (see Errata section and 2.1) including CUDA® Basic Linear Algebra Subroutines library™ (cuBLAS) 9.0.333 [see section 2.3.1]

- **NVIDIA CUDA® Deep Neural Network library™ (cuDNN) 7.1.1**

- **NCCL** 2.1.2 (optimized for NVLink™)

- **ONNX exporter 0.1** for CNN classification models

  - **Note:** The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- **Amazon Labs Sockeye sequence-to-sequence framework 1.17.4** (for machine translation)

Driver Requirements

Release 18.03 is based on CUDA 9, which requires NVIDIA Driver release 384.xx.

Key Features and Enhancements

This NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet release includes the following key features and enhancements.
NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 18.03 is based on Apache MXNet 1.1.0.

Incorporated all upstream changes from the NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet master branch, specifically, PR 9749.

Added compute-graph optimizations for improved ResNet performance.

Latest version of cuBLAS 9.0.333

Latest version of cuDNN 7.1.1

Ubuntu 16.04 with February 2018 updates

Announcements

Starting with the next major version of CUDA release, we will no longer provide Python 2 containers and will only maintain Python 3 containers.

Known Issues

There are no known issues in this release.
Chapter 31. Release 18.02

The container image of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 18.02, is available.

NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 18.02 is based on Apache MXNet 1.1.0.

Contents of Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu** 16.04
- **NVIDIA CUDA** 9.0.176 including:
  - CUDA® Basic Linear Algebra Subroutines library™ (cuBLAS) 9.0.282 Patch 2 which is installed by default
  - cuBLAS 9.0.234 Patch 1 as a debian file. Installing Patch 1 by issuing the `dpkg -i /opt/cuda-cublas-9-0_9.0.234-1_amd64.deb` command is the workaround for the known issue described below.
- **NVIDIA CUDA® Deep Neural Network library™ (cuDNN) 7.0.5**
- **NCCL 2.1.2** (optimized for NVLink™)
- **ONNX exporter 0.1** for CNN classification models

  Note: The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the master branch.

- **Amazon Labs Sockeye sequence-to-sequence framework 1.17.0** (for machine translation)
Driver Requirements

Release 18.02 is based on CUDA 9, which requires NVIDIA Driver release 384.xx.

Key Features and Enhancements

This NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet release includes the following key features and enhancements.

- Latest version of cuBLAS
- Ubuntu 16.04 with January 2018 updates

Known Issues

cuBLAS 9.0.282 regresses RNN seq2seq FP16 performance for a small subset of input sizes. This issue should be fixed in the next update. As a workaround, install cuBLAS 9.0.234 Patch 1 by issuing the `dpkg -i /opt/cuda-cublas-9-0_9.0.234-1_amd64.deb` command.
Chapter 32. Release 18.01

The container image of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 18.01, is available.

**NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet** container image version 18.01 is based on Apache MXNet 1.0.0.

**Contents of the Optimized Deep Learning Framework container**

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in `/opt/mxnet`. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu 16.04**

  - **Note:** Container image 18.01-py2 contains **Python 2.7**; 18.01-py3 contains **Python 3.5**.

  - NVIDIA CUDA 9.0.176 including **CUDA® Basic Linear Algebra Subroutines library™ (cuBLAS)** 9.0.282
  - NVIDIA CUDA® **Deep Neural Network library™ (cuDNN)** 7.0.5
  - **NCCL** 2.1.2 (optimized for **NVLink™**)
  - **ONNX exporter 0.1** for CNN classification models

  - **Note:** The ONNX exporter is being continuously improved. You can try the latest changes by pulling from the **master branch**.

  - Amazon Labs Sockeye sequence-to-sequence framework 1.16.2 [for machine translation]

**Driver Requirements**

Release 18.01 is based on CUDA 9, which requires NVIDIA **Driver** release 384.xx.
Key Features and Enhancements

This NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet release includes the following key features and enhancements.

‣ Addition of Python 3 package
‣ Enhanced-performance cuDNN-based batched 1D convolutions (merged to upstream)
‣ Added MXNet-to-ONNX exporter for classification of CNN models (tested with LeNet-5, ResNet-50, etc.).
‣ Added the Sockeye sequence-to-sequence framework, along with a German-to-English translation model, based on the WMT’15 dataset and translation task. This model’s launch script should reproduce the OpenNMT reference model when trained until convergence.
‣ Latest version of cuBLAS
‣ Latest version of cuDNN
‣ Latest version of NCCL
‣ Ubuntu 16.04 with December 2017 updates

Known Issues

cuBLAS 9.0.282 regresses RNN seq2seq FP16 performance for a small subset of input sizes. As a workaround, revert back to the 17.12 container.
Chapter 33. Release 17.12

The container image of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 17.12, is available.

NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 17.12 is based on Apache MXNet 1.0.0.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- Ubuntu 16.04
- NVIDIA CUDA 9.0.176 including CUDA® Basic Linear Algebra Subroutines library™ (cuBLAS) 9.0.234
- NVIDIA CUDA® Deep Neural Network library™ (cuDNN) 7.0.5
- NCCL 2.1.2 (optimized for NVLink™)

Driver Requirements

Release 17.12 is based on CUDA 9, which requires NVIDIA Driver release 384.xx.

Key Features and Enhancements

This NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet release includes the following key features and enhancements.

- Both the nccl and nccl_allreduce KVStore options now have the same, improved performance.
- Latest version of CUDA
- Latest version of cuDNN
- Latest version of NCCL
- Ubuntu 16.04 with November 2017 updates
Known Issues

There are no known issues in this release.
Chapter 34. Release 17.11

The container image of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 17.11, is available.

**NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet** container image version 17.11 is based on **Apache MXNet 0.12.0**.

**Contents of the Optimized Deep Learning Framework container**

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- Ubuntu 16.04
- NVIDIA CUDA 9.0.176 including CUDA® Basic Linear Algebra Subroutines library™ (cuBLAS) 9.0.234
- NVIDIA CUDA® Deep Neural Network library™ (cuDNN) 7.0.4
- NCCL 2.1.2 (optimized for NVLink™)

**Driver Requirements**

Release 17.11 is based on CUDA 9, which requires NVIDIA Driver release 384.xx.

**Key Features and Enhancements**

This NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet release includes the following key features and enhancements.

- Added Sockeye to the container, including NCCL kvstore option
- Enabled mx.sym.batch_dot to use FP16
- Latest version of CUDA
- Latest version of cuDNN
- Latest version of NCCL
- Ubuntu 16.04 with October 2017 updates
Known Issues

There are no known issues in this release.
Chapter 35. Release 17.10

The NVIDIA container image of NVIDIA Optimized Deep Learning Framework, powered by
Apache MXNet, release 17.10, is available.

NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image
version 17.10 is based on Apache MXNet 0.11.0.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep
Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to
the Python path.

The container also includes the following:

- Ubuntu 16.04
- NVIDIA CUDA® 9.0
- NVIDIA CUDA® Deep Neural Network library™ (cuDNN) 7.0.3
- NCCL 2.0.5 (optimized for NVLink™)

Driver Requirements

Release 17.10 is based on CUDA 9, which requires NVIDIA Driver release 384.xx.

Key Features and Enhancements

This NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet release
includes the following key features and enhancements.

- Mixed precision support for all optimizers
- New image input pipeline with faster speed and support for global shuffling after each
  epoch when used with IndexedRecordIO format.
- Latest version of CUDA
- Latest version of cuDNN
- Latest version of NCCL
- Ubuntu 16.04 with September 2017 updates
Known Issues

There are no known issues in this release.
Chapter 36. Release 17.09

The container image of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 17.09, is available.

NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 17.09 is based on Apache MXNet 0.11.0.rc3.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu** 16.04
- **NVIDIA CUDA®** 9.0
- **NVIDIA CUDA® Deep Neural Network library™ (cuDNN)** 7.0.2
- **NCCL** 2.0.5 (optimized for NVLink™)

Driver Requirements

Release 17.09 is based on CUDA 9, which requires NVIDIA Driver release 384.xx.

Key Features and Enhancements

This NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet release includes the following key features and enhancements.

- Tensor Core support in convolutions, deconvolutions, and fully connected layers on Volta
- Support for mixed precision training with SGD optimizer
- Streamlined FP16 examples for image classification
- Optimized input pipeline for image processing
- Latest version of CUDA
- Latest version of cuDNN
- Latest version of NCCL
Ubuntu 16.04 with August 2017 updates

**Known Issues**

There are no known issues in this release.
Chapter 37. Release 17.07

The container image of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 17.07, is available.

NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 17.07 is based on Apache MXNet 0.10.0.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu 16.04**
- **NVIDIA CUDA® 8.0.61.2 including CUDA® Basic Linear Algebra Subroutines library™ (cuBLAS) Patch 2**
- **NVIDIA CUDA® Deep Neural Network library™ (cuDNN) 6.0.21**
- **NCCL 2.0.3 [optimized for NVLink™]**

Key Features and Enhancements

This NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet release includes the following key features and enhancements.

- Support for multi-precision SGD
- cuBLAS back-end for FullyConnected operation
- Ubuntu 16.04 with June 2017 updates

Known Issues

There are no known issues in this release.
Chapter 38. Release 17.06

The container image of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 17.06, is available.

NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 17.06 is based on Apache MXNet 0.10.0.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- Ubuntu 16.04
- NVIDIA CUDA® 8.0.61
- NVIDIA CUDA® Deep Neural Network library™ (cuDNN) 6.0.21
- NCCL 1.6.1 (optimized for NVLink™)

Key Features and Enhancements

This NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet release includes the following key features and enhancements.

- Implemented double buffering in ResNet v1 example
- Ubuntu 16.04 with May 2017 updates

Known Issues

There are no known issues in this release.
Chapter 39. Release 17.05

The container image of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 17.05, is available.

NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 17.05 is based on Apache MXNet 0.9.3a+.

Contents of the Optimized Deep Learning Framework container

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu** 16.04
- **NVIDIA CUDA®** 8.0.61
- **NVIDIA CUDA® Deep Neural Network library™ (cuDNN)** 6.0.21
- **NCCL** 1.6.1 (optimized for NVLink™)

Key Features and Enhancements

This NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet release includes the following key features and enhancements.

- Latest cuDNN release
- Improved IO pipeline for increased multi-GPU performance
- Optimized SGD weight update
- Added the `nccl_allreduce` option for gradient communication
- Added support for dilated deconvolutions
- Improved convolutional neural network (CNN) performance by removing unnecessary computations
- Added options to show the cuDNN algorithms that are chosen for convolutions
- Ubuntu 16.04 with April 2017 updates
Known Issues

There are no known issues in this release.
Chapter 40. Release 17.04

The container image of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 17.04, is available.

**NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet** container image version 17.04 is based on **Apache MXNet 0.9.3a+**.

**Contents of the Optimized Deep Learning Framework container**

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in `/opt/mxnet`. It is pre-built and installed to the Python path.

The container also includes the following:

- **Ubuntu 16.04**
- **NVIDIA CUDA® 8.0.61**
- **NVIDIA CUDA® Deep Neural Network library™ (cuDNN) 6.0.20**
- **NCCL 1.6.1** (optimized for **NVLink™**)  

**Key Features and Enhancements**

This NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet release includes the following key features and enhancements.

- Support for cuDNN accelerated dilated convolutions
- Ubuntu 16.04 with March 2017 updates

**Known Issues**

There are no known issues in this release.
Chapter 41. Release 17.03

The container image of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet, release 17.03, is available.

NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet container image version 17.03 is based on Apache MXNet 0.9.3.

Contents of the Optimized Deep Learning Framework

This container image contains the complete source of the version of NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet in /opt/mxnet. It is pre-built and installed to the Python path.

The container also includes the following:

- Ubuntu 16.04
- NVIDIA CUDA® 8.0.61
- NVIDIA CUDA® Deep Neural Network library™ (cuDNN) 6.0.20

Key Features and Enhancements

This NVIDIA Optimized Deep Learning Framework, powered by Apache MXNet release includes the following key features and enhancements.

- Ubuntu 16.04 with February 2017 updates
- Improved input pipeline for image processing
- Support for FP16 training of AlexNet
- Optimized embedding layer of CUDA kernels
- Optimized tensor broadcast and reduce CUDA kernels

Known Issues

There are no known issues in this release.
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.

VESA DisplayPort

DisplayPort and DisplayPort Compliance Logo, DisplayPort Compliance Logo for Dual-mode Sources, and DisplayPort Compliance Logo for Active Cables are trademarks owned by the Video Electronics Standards Association in the United States and other countries.

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.