DLProf Viewer

User Guide

Table of Contents

Chapter 1. DLProf Viewer Overview

1.1. Overview

The DLProf Viewer makes it easy to visualize the performance of your models by showing Top 10 operations that took the most time, eligibility of Tensor Core operations and Tensor Core usage, as well as interactive iteration reports.

1.2. What's New in 1.8.0

- This is the last release of DLProf Viewer.
- Compatible with DLProf CLI v1.8.0.

1.3. Features

This release includes these commands and features:

- Panelized Dashboard Summary View: A summary view comprising several panels that provide a quick overview of the performance results.
- Top-level Key Metrics: The Summary view displays several key metrics that are used to quickly gauge the quality of the performance, including Average Iteration Time and Tensor Core Utilization.
- Top 10 GPU Ops Node: A table in the Summary view lists the top 10 Op Nodes with the most time spent on the GPU.
- Expert Systems Panel: This panel displays any issues detected by the DLProf Expert Systems, along with suggestions on how to address the issues and improve the models performance.
- Iteration Summary Panel: This panel visually displays iterations. Users can quickly see how many iterations are in the model, the iterations that were aggregated/profiled, and the durations of tensor core kernels in each iteration.
- Interactive Tables: All tables in detailed views are completely interactive, allowing the use to sort, filter, and paginate the display.

- Interoperable Tables: Several views have the ability to drill down for more information. Selecting a row in one table will populate the next table with performance information pertaining to the selection.
- Client/server architecture:
  - All of the data is now in a relational database.
  - Load times have improved for all views.
- GPUs View: Displays GPU utilization and GPU properties of all GPUs used during profiling.

Chapter 2. Quick Start

2.1. Installing Using Python Wheel

The DLPROF Viewer can be easily installed from the NVIDIA PY index.

Install `nvidia-pyindex`.

```
$ pip install nvidia-pyindex
```

Next install the DLProf Viewer using pip:

```
$ pip install nvidia-dlprofviewer
```

2.2. Using the NGC Docker Container

The following steps are required to use pre-built NGC containers:

- Ensure you have access and are logged into NGC. For step-by-step instructions, see the NGC Getting Started Guide.
- Install Docker and nvidia-docker. For DGX users, see Preparing to use NVIDIA Containers. For users other than DGX, see nvidia-docker installation documentation.

Use docker pull to get the TensorFlow container from NGC:

```
$ docker pull nvcr.io/nvidia/tensorflow:xx.yy>-tf1-py3
```

Assuming the training data for the model is available in /full/path/to/training/data, you can launch the container with the following command:

```
$ docker run --rm --gpus=1 --shm-size=1g --ulimit memlock=-1 \
--ulimit stack=67108864 -it -p8000:8000 -v/full/path/to/training/data:/data \
nvcr.io/nvidia/tensorflow:<xx.yy>-tf1-py3
```

2.3. Generating DLProf Database

The DLProf database is created directly from the Deep Learning Profiler. See the Deep Learning Profiler User Guide for information on how to generate the database.

2.4. Starting DLProf Viewer

The DLProf Viewer is installed in all containers that DLProf is installed in the NVIDIA GPU Cloud (NGC). The container must be run with the -p8000:8000 option to open port 8000 for the DLProf Viewer. Use any port such as 8001, 8002, etc.

Usage:

```
Usage: dlprofviewer [-h] [--version] [-b ADDRESS] [-p PORT] database

positional arguments:
  database              Specify path to DLProf database.

optional arguments:
  -h, --help            Show this help message and exit
  --version             Show program's version number and exit
  -b ADDRESS, --bind ADDRESS
                        Specify alternate bind address. Use '0.0.0.0' to serve to
 the entire local network. [default: localhost]
  -p PORT, --port PORT  Specify alternate port. [default: 8000]
```

The DLProf Viewer server is launched directly from the container:

```
$ dlprofviewer dlprof_dldb.sqlite
```

Where `dlprof_dldb.sqlite` is the path to the DLProf database.

Note: Using this example limits the visibility of this viewer on the network to just 'localhost'. See the Bind section for information on how to increase the visibility of this DLProf database on your network.

2.5. Bind

If you want anyone on your local network to be able to access your dlprofviewer server in their browser, launch with -b 0.0.0.0:

```
$ dlprofviewer -b 0.0.0.0 dlprof_dldb.sqlite
```

Once the server is running, the DLProf app can be viewed in a browser using the URL:

```
http://<machine IP Address>:8000
```

2.6. Port

You can also specify a custom port with `-p PORT`, e.g. (port 8001):

```
$ dlprofviewer -b 0.0.0.0 -p 8001 dlprof_dldb.sqlite
```

Then use your browser to go to this URL:

```
http://<machine IP Address>:8000
```

2.7. How to Launch DLProf Viewer

The DLProf Viewer is installed in all containers that DLProf is installed in the NVIDIA GPU Cloud (NGC). The container must be run with the -p8000:8000 option to open port 8000 for the DLProf Viewer. Use any port such as 8001, 8002, etc.

Server Location	Browser and Server on the same machine	Browser and Server on different machines
Server inside docker	▶ Run docker with -p=<port>:<port>. Substitute <port> for the port you want to use ▶ Run dlprofviewer with -b \`hostname -i\` ▶ Go to http://localhost:<port> in browser	▶ Run docker with -p=<port>:<port>. Substitute <port> for the port you want to use ▶ Run dlprofviewer with -b 0.0.0.0 ▶ Go to http://<server machine's ip>:<port> in browser
Server outside docker	▶ No -b option needed when running dlprofviewer ▶ Go to http://localhost:<port> in browser	▶ Run dlprofviewer with -b 0.0.0.0 ▶ Go to http://<server machine's ip>:<port> in browser

2.8. Backwards Compatibility

Some older versions of the DLProf database can be used as input to the latest dlprofviewer:

- A DLProf database's initial version is the version of DLProf used to create it.
- DLProf databases v1.3+ can be used as input to DLProf CLI v1.3+
  - The DLProf database version should still be less than or equal to the dlprofviewer version.
- DLProf databases v1.1 and v1.2 cannot be used as input to dlprofviewer v1.3+.
- When an older DLProf database is used as input, some minor changes will be made to it to allow it to be used correctly with the latest dlprofviewer.
- Backwards compatibility gets "reset" each major release.
  - dlprofviewer v2.x will not support DLProf databases v1.y.

Chapter 3. DLProf Viewer

This section describes each of the available views in the DLProf Viewer.

3.1. Terms and Definitions

Term	Definition
Aggregation	The rollup of metrics, given an iteration start, iteration stop, and key node.
Op Node	A node in the graph where an operation is performed on the incoming tensor.
Model, Graph, Network	<synonyms>

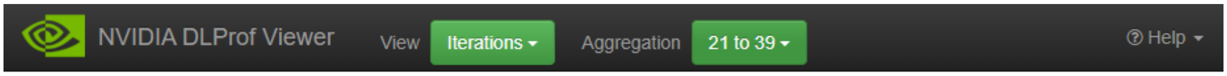
3.2. Pane Overview

The DLProf Viewer user interface is divided into two panes:

Pane	Definintion
Content Pane	Otherwise known as the VIEWS pane, this is where you will see all of the different pieces of the profiled neural network.
Details Pane	This pane is to display additional details about the information selected in the content pane. Let's say you're looking at the Kernel Summary panel in the content pane, and you want to see more information about the kernels. Click on the ⤢ button (tooltip = 'Show Kernel Details'). The detailed kernel information will appear in the Details Pane. Click the 'X' button to remove the details pane.

3.3. Navigation Bar

This standard navigation bar is a navigation header that is placed at the top of the viewer. It contains a few buttons like most standard web sites. It does not scroll off the screen so these buttons are always available.



Note:

Some screen shots in this documentation might have a pulldown menu called **Domain** in the Navigation Bar. This feature has been removed from the DLProf Viewer.

UI Controls

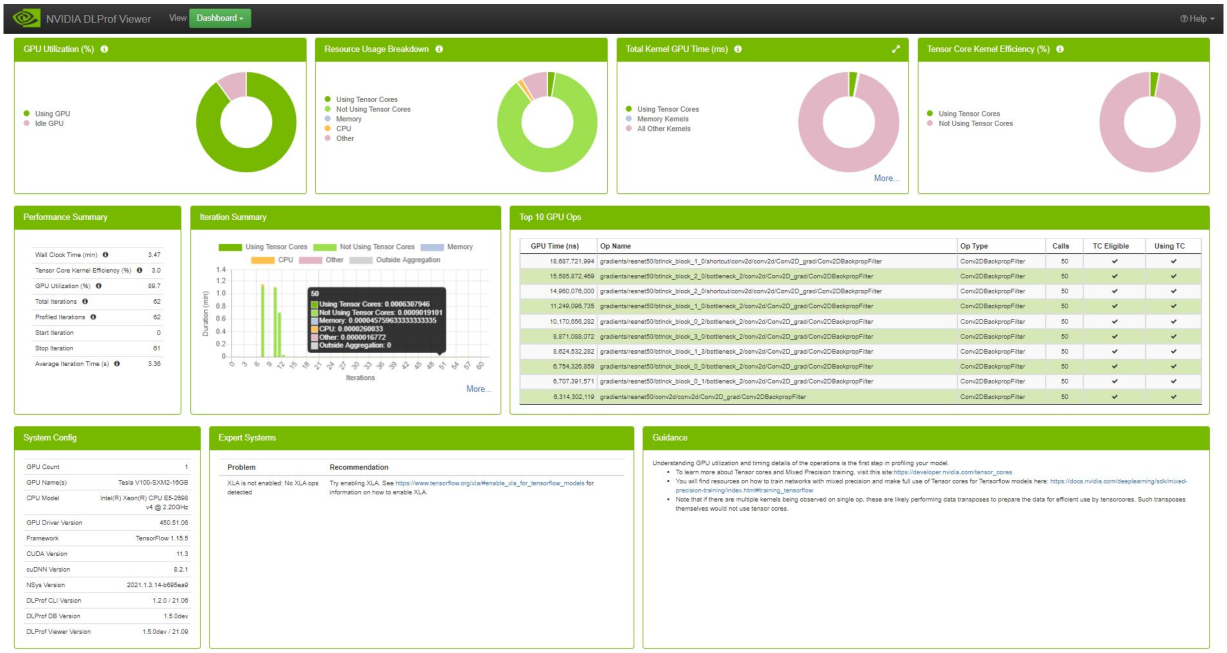
Control	Definition
NVIDIA DLProf Viewer button	This is the branding button. When clicked, the user is navigated to the Dashboard view.
View	This dropdown provides a list of available views. Clicking on a name in the dropdown loads that view in the main display panel. Available views are: ▶ Dashboard ▶ Op Type Summary ▶ Group Ops ▶ Ops and Kernels ▶ Kernels by Iteration ▶ Kernels by Op ▶ Iterations ▶ GPUs (when more than one GPU exists in profile run)
Aggregation (optional)	This optional dropdown appears when a network has been re-aggregated (ie, aggregated more than once). An aggregation is a combination of iteration start, iteration stop, and key node. The iter stop and iter stop values are listed in the dropdown as seen above. To see the corresponding key node, click the dropdown and hover over each aggregation in the list.. Aggregations can be created in the Iterations view. The workflow and user interface controls are documented in the Iterations view section.
Help / Deep Learning Profiler	This link navigates users to the online version of the DLProf User Guide.
Help / Viewer	This link navigates users to the online version of this document

Help / Contact Us	Let us know! If you have a comment, question, or suggestion, click this link. It will launch your default email software with the TO address already filled in. Just fill in the Subject line, type your email message, then click send.

3.4. Content Pane

3.4.1. Dashboard

The Dashboard view provides a high level summary of the performance results in a panelized view. This view serves as a starting point in analyzing the results and provides several key metrics.



Wall Clock Time (min)	3.47
Tensor Core Kernel Efficiency (%)	3.0
GPU Utilization (%)	89.7
Total Iterations	62
Profiled Iterations	62
Start Iteration	0
Stop Iteration	61
Average Iteration Time (s)	3.36

50
Using Tensor Cores: 0.0006307946
Not Using Tensor Cores: 0.0009019101
Memory: 0.000045759633333333335
CPU: 0.0000260033
Other: 0.0000016772
Outside Aggregation: 0

GPU Time (ns)	Op Name	Op Type	Calls	TC Eligible	Using TC
18,687,721,994	gradients/resnet50/btlnck_block_1_0/shortcut/conv2d/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	50	✓	✓
15,585,872,469	gradients/resnet50/btlnck_block_2_0/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	50	✓	✓
14,960,076,000	gradients/resnet50/btlnck_block_2_0/shortcut/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	50	✓	✓
11,249,096,735	gradients/resnet50/btlnck_block_1_0/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	50	✓	✓
10,170,656,282	gradients/resnet50/btlnck_block_0_2/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	50	✓	✓
8,871,088,072	gradients/resnet50/btlnck_block_3_0/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	50	✓	✓
8,624,532,282	gradients/resnet50/btlnck_block_1_3/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	50	✓	✓
6,754,326,859	gradients/resnet50/btlnck_block_0_0/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	50	✓	✓
6,707,391,571	gradients/resnet50/btlnck_block_0_1/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	50	✓	✓
6,314,302,119	gradients/resnet50/conv2d/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	50	✓	✓

GPU Count	1
GPU Name(s)	Tesla V100-SXM2-16GB
CPU Model	Intel(R) Xeon(R) CPU E5-2698 v4 @ 2.20GHz
GPU Driver Version	450.51.06
Framework	TensorFlow 1.15.5
CUDA Version	11.3
cuDNN Version	8.2.1
NSys Version	2021.1.3.14-b695ea9
DLProf CLI Version	1.2.0 / 21.06
DLProf DB Version	1.5.0dev
DLProf Viewer Version	1.5.0dev / 21.09

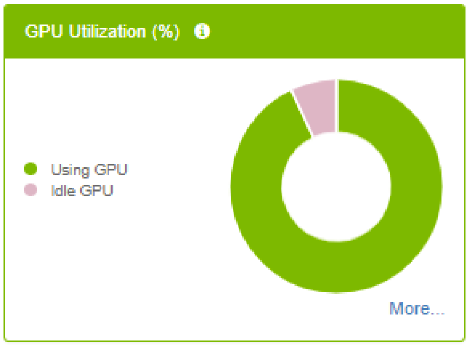
Problem	Recommendation
XLA is not enabled: No XLA ops detected	Try enabling XLA. See https://www.tensorflow.org/xla/#enable_xla_for_tensorflow_models for information on how to enable XLA.

Understanding GPU utilization and timing details of the operations is the first step in profiling your model.

- To learn more about Tensor cores and Mixed Precision training, visit this site:https://developer.nvidia.com/tensor_cores
- You will find resources on how to train networks with mixed precision and make full use of Tensor cores for Tensorflow models here: https://docs.nvidia.com/deeplearning/sdk/mixed-precision-training/index.html#training_tensorflow
- Note that if there are multiple kernels being observed on single op, these are likely performing data transposes to prepare the data for efficient use by tensorcores. Such transposes themselves would not use tensor cores.

3.4.1.1. GPU Utilization Panel

The GPU Idle panel visually indicates the percentage of GPU utilization time during execution of aggregated iterations. Hovering over a slice in the chart will show the numeric percentage.



Fields

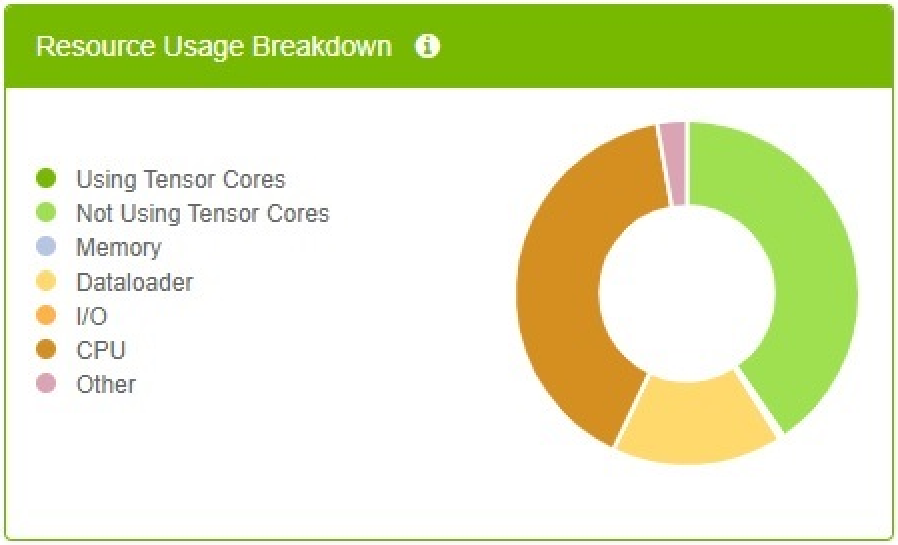
Legend Label	Definition
Using GPU	The average GPU utilization percentage across all GPUs.
Idle GPU	The average GPU idle percentage across of all GPUs.

UI Controls

Control	Definition
Legend Label	Toggle between hiding and showing legend entry in chart.
More...	Show drop-down menu of more views (only visible when more than one GPU was used during profiling).

3.4.1.2. Resource Usage Breakdown Panel

The panel provides a breakdown of profile activity into resource categories.



Fields

Legend Label	Definition

Using Tensor Cores	Accumulated duration of all kernels using tensor cores.
Not Using Tensor Cores	Accumulated duration of all kernels not using tensor cores.
Memory	Accumulated duration of all memory operations.
Dataloader	Accumulated duration of dataloader operations.
I/O	Accumulated duration of I/O operations.
CPU	Accumulated duration of all CPU operations.
Other	All time not in any other category.

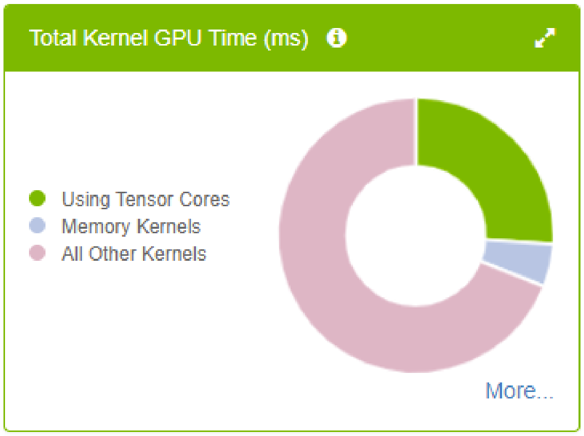
UI Controls

Control	Definition
Legend Label	Toggle between hiding and showing legend entry in chart.

3.4.1.3. Total Kernel GPU Time Panel

The panel provides key metrics about the kernels in the network aggregated over the specific iteration range.

- Hovering over a slice in the chart will show the aggregated GPU time.
- Clicking a legend item will toggle its visualization in the chart.



Fields

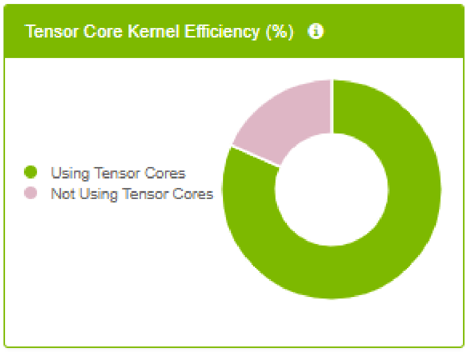
Legend Label	Definition
Using Tensor Cores	Aggregates the total GPU time for all kernels using Tensor Cores.
Memory Kernels	Aggregates the total GPU time for all memory-related kernels.
All Other Kernels	Aggregates the total GPU time for all remaining kernel types.

UI Controls

Control	Definition
⤢	Show Kernel Details panel in Details Pane.
Legend Label	Toggle between hiding and showing legend entry in chart.
More...	Show drop-down menu of more views.

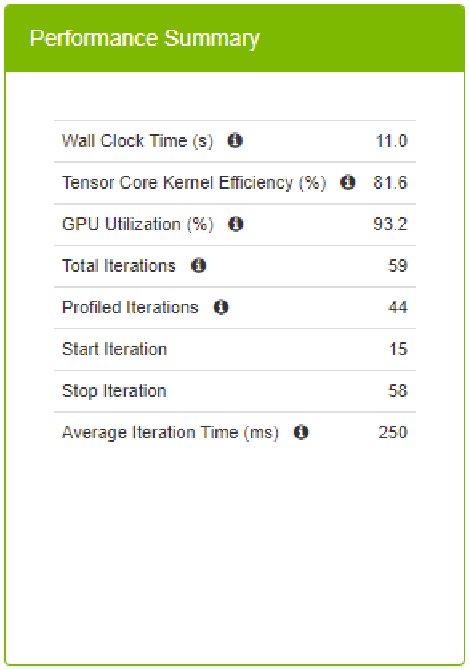
3.4.1.4. Tensor Core Kernel Efficiency Panel

- Hovering over a slice in the chart will show the percentage.
- Clicking a legend item will toggle its visualization in the chart.



3.4.1.5. Performance Summary Panel

The Performance Summary panel provides top level key metrics about the performance data aggregated over the specific iteration range. A helpful tooltip text will appear when hovering over the 'i' icon.



Performance Summary

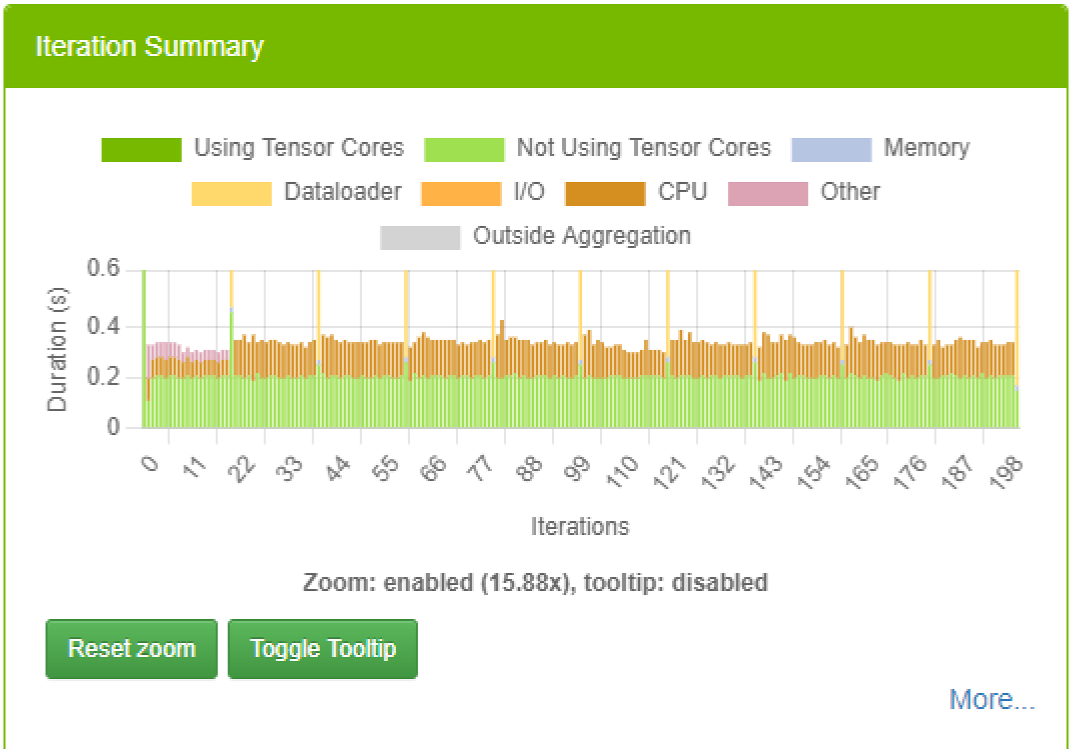
Wall Clock Time (s) ⓘ	11.0
Tensor Core Kernel Efficiency (%) ⓘ	81.6
GPU Utilization (%) ⓘ	93.2
Total Iterations ⓘ	59
Profiled Iterations ⓘ	44
Start Iteration	15
Stop Iteration	58
Average Iteration Time (ms) ⓘ	250

Field	Definition
Wall Clock Time	This is the total run time for the aggregation range, and is defined as the time between the start time of the first op in the starting iteration on the CPU and the end time of the last op in the final iteration on either the CPU or GPU, whichever timestamp is greatest.
Tensor Core Kernel Efficiency %	This high level metric represents the utilization of Tensor Core enabled kernels. Tensor Core operations can provide a performance improvement and should be used when possible. This metric is calculated by: [Total GPU Time for Tensor Core kernels] / [Total GPU Time for Tensor Core Eligible Ops] A 100% Tensor Core Utilization means that all eligible Ops are running only Tensor Core enabled kernels on the GPU. A 50% Tensor Core Utilization can mean anything from all eligible Ops are running Tensor Core kernels only half of the time to only half of all eligible Ops are running Tensor Core kernels only. This metric should be used with the Op Summary Panel to determine the quality of Tensor Core usage. Higher is better.
GPU Utilization %	Average GPU utilization across all GPUs. Higher is better.
Total Iterations	The total number of iterations found in the network.
Profiled Iterations	The total number of iterations used to aggregate the performance results. This number is calculated using 'Start Iteration' and 'Stop Iteration'.
Start Iteration	The starting iteration number used to generate performance results.
Stop Iteration	The ending iteration number used to generate performance results.

Average Iteration Time	The average iteration time is the total Wall Time divided by the number of iterations.

3.4.1.6. Iteration Summary panel

This panel visually displays iterations. Users can quickly see how many iterations are in the model, the iterations that were aggregated/profiled, and the durations of tensor core kernels in each iteration. The colors on this panel match the colors on all the other dashboard panels.

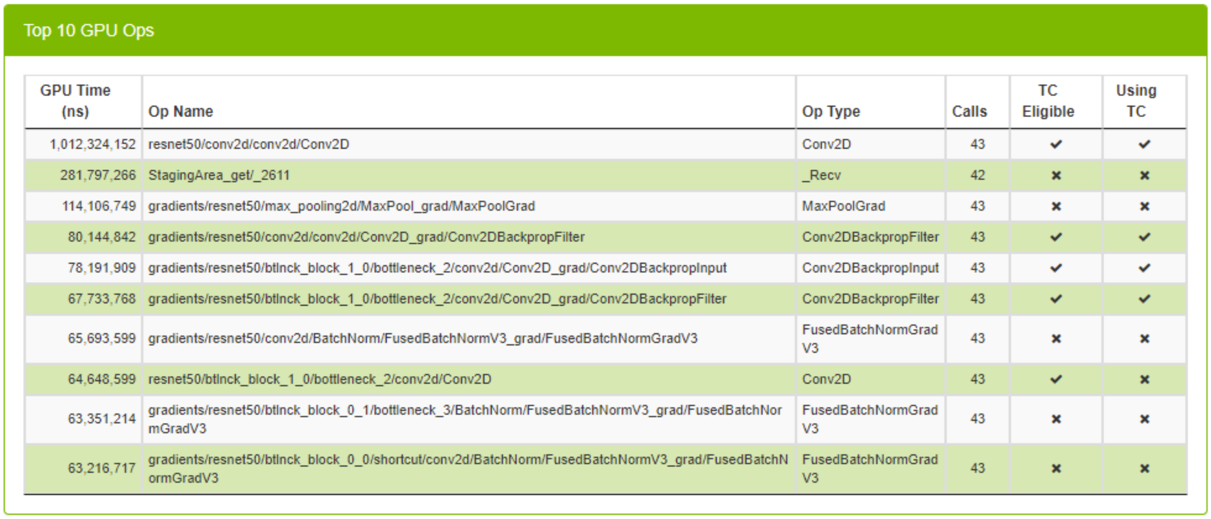


Zoom: enabled (15.88x), tooltip: disabled

For more information on this panel, see Iterations View.

3.4.1.7. Top 10 GPU Ops Panel

Top 10 GPU Ops table shows the top 10 operations with the largest execution times on the GPU. This table comes pre-sorted with the order of each row in descending GPU Time. The table is not sortable or searchable.

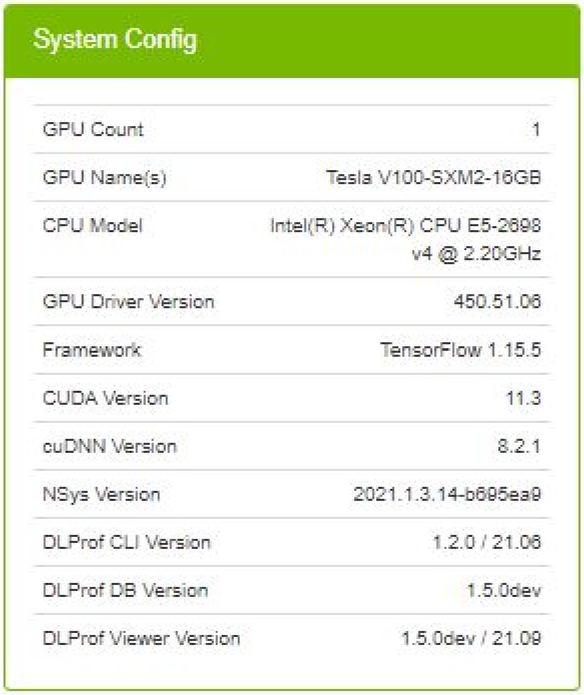


Top 10 GPU Ops

GPU Time (ns)	Op Name	Op Type	Calls	TC Eligible	Using TC
1,012,324,152	resnet50/conv2d/conv2d/Conv2D	Conv2D	43	✓	✓
281,797,266	StagingArea_get/_2611	_Recv	42	✗	✗
114,106,749	gradients/resnet50/max_pooling2d/MaxPool_grad/MaxPoolGrad	MaxPoolGrad	43	✗	✗
80,144,842	gradients/resnet50/conv2d/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	43	✓	✓
78,191,909	gradients/resnet50/btlnck_block_1_0/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropInput	Conv2DBackpropInput	43	✓	✓
67,733,768	gradients/resnet50/btlnck_block_1_0/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	43	✓	✓
65,693,599	gradients/resnet50/conv2d/BatchNorm/FusedBatchNormV3_grad/FusedBatchNormGradV3	FusedBatchNormGradV3	43	✗	✗
64,648,599	resnet50/btlnck_block_1_0/bottleneck_2/conv2d/Conv2D	Conv2D	43	✓	✗
63,351,214	gradients/resnet50/btlnck_block_0_1/bottleneck_3/BatchNorm/FusedBatchNormV3_grad/FusedBatchNormGradV3	FusedBatchNormGradV3	43	✗	✗
63,216,717	gradients/resnet50/btlnck_block_0_0/shortcut/conv2d/BatchNorm/FusedBatchNormV3_grad/FusedBatchNormGradV3	FusedBatchNormGradV3	43	✗	✗

Column	Definition
GPU Time	Shows total GPU time of all kernels across all GPUs.
Op Name	The name of the op.
Direction	The fprop/bprop direction of the op. (only visible on PyTorch runs).
Op Type	The type of the op.
Calls	The number of times the op was called.
TC Eligible	A true/false field indicating whether or not the op is eligible to use Tensor Core kernels.
Using TC	A true/false field indicating whether or not one of the kernels launched in this op is using Tensor Cores.

3.4.1.8. System Configuration Panel



System Config

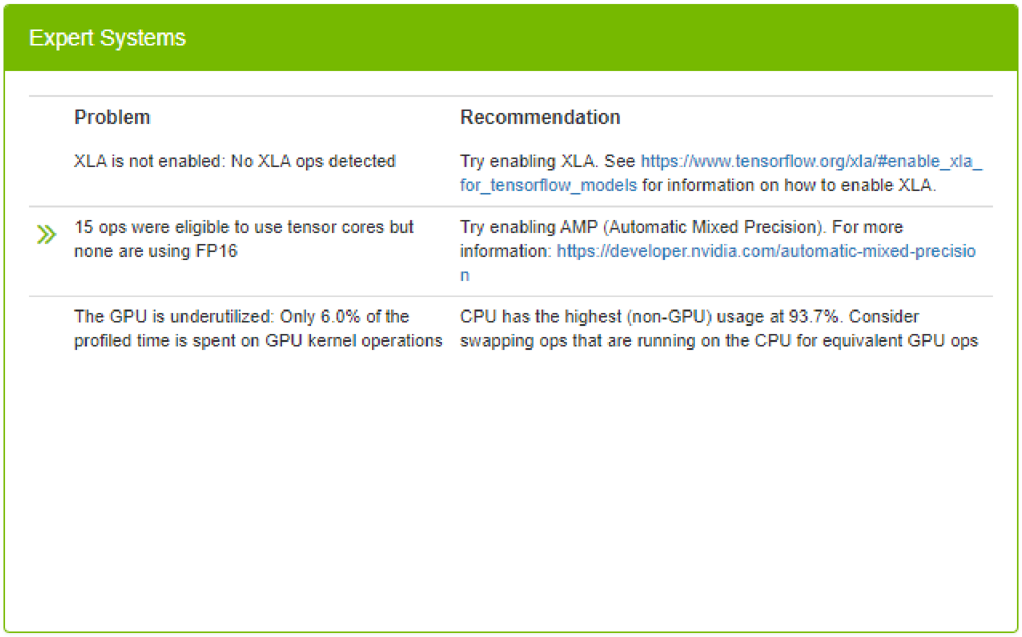
GPU Count	1
GPU Name(s)	Tesla V100-SXM2-16GB
CPU Model	Intel(R) Xeon(R) CPU E5-2698 v4 @ 2.20GHz
GPU Driver Version	450.51.06
Framework	TensorFlow 1.15.5
CUDA Version	11.3
cuDNN Version	8.2.1
NSys Version	2021.1.3.14-b695ea9
DLProf CLI Version	1.2.0 / 21.06
DLProf DB Version	1.5.0dev
DLProf Viewer Version	1.5.0dev / 21.09

Field	Definition
Profile Name	(Optional) Helpful label to describe the profiled network. The value in this field corresponds to the value supplied in the --profile_name command line argument in DLProf.
GPU Count	The number of GPU devices found on the computer during training.
GPU Name(s)	A list of the GPU devices found on the computer during training.
CPU Model	The model of the CPU on the computer during training.
GPU Driver Version	The version of the driver used for NVIDIA Graphics GPU.
Framework	The framework used to generate profiling data (eg, TensorFlow, PyTorch).
CUDA Version	The version of the CUDA parallel computing platform.
cuDNN Version	The version of CUDA Deep Neural Network used during training.
NSys Version	The version of Nsight Systems used during training.
DLProf Version	The version of the Deep Learning Profiler used to generate the data visualized in the DLProf Viewer.
DLProf DB Version	The version of the DLProf database.
DLProf Viewer Version	The version of the DLProf Viewer.

3.4.1.9. Expert Systems panel

The Expert Systems panel displays common issues detected in the profiled network and provides potential solutions and suggestions to address the issues. The panel will only show issues that have been detected by DLProf. For a full list of potential issues that DLProf looks for, see the Expert Systems section in the Deep Learning Profiler User Guide.

Clicking on the double-green arrows shows additional information about the detected problem.



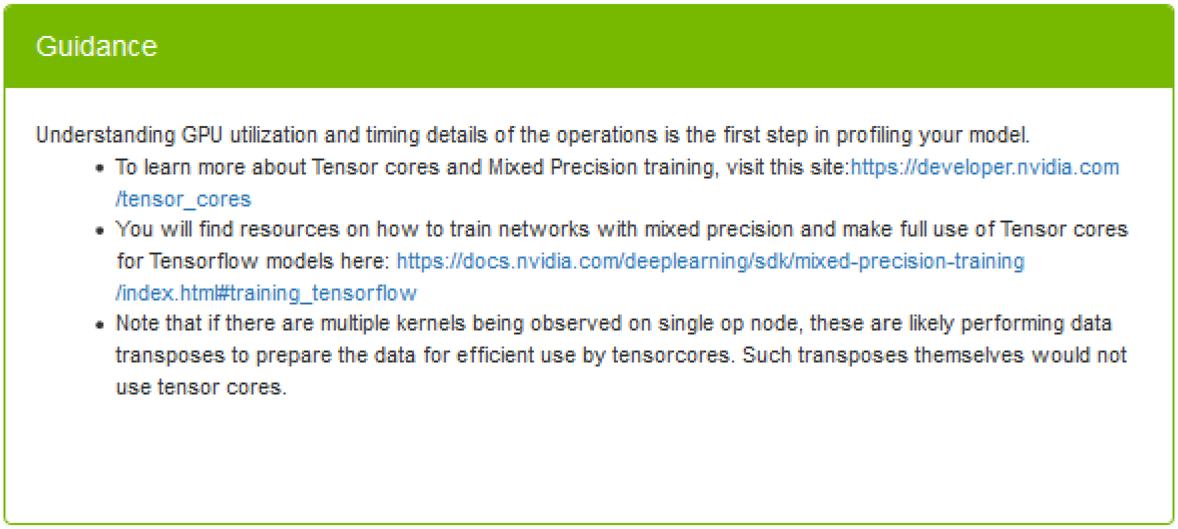
Expert Systems

	Problem	Recommendation
	XLA is not enabled: No XLA ops detected	Try enabling XLA. See https://www.tensorflow.org/xla/#enable_xla_for_tensorflow_models for information on how to enable XLA.
»	15 ops were eligible to use tensor cores but none are using FP16	Try enabling AMP (Automatic Mixed Precision). For more information: https://developer.nvidia.com/automatic-mixed-precision
	The GPU is underutilized: Only 6.0% of the profiled time is spent on GPU kernel operations	CPU has the highest (non-GPU) usage at 93.7%. Consider swapping ops that are running on the CPU for equivalent GPU ops

Column	Definition
Problem	The description of the scenario that DLProf detected when profiling the network.
>>	(Optional) When present, clicking on the double arrows will display a new view displaying the problem in detail.
Recommendation	A recommendation or actionable feedback, a tangible suggestion that the user can do to improve the network. Clicking on a hyperlink inside the recommendation will open a new tab in the browser.

3.4.1.10. Guidance Panel

This panel provides static guidance to the user to help the user learn more about Tensor Cores, Mixed Precision training. The panel has hyperlinks for further reading. Clicking on a hyperlink inside the Guidance Panel will open a new tab in the browser.



Guidance

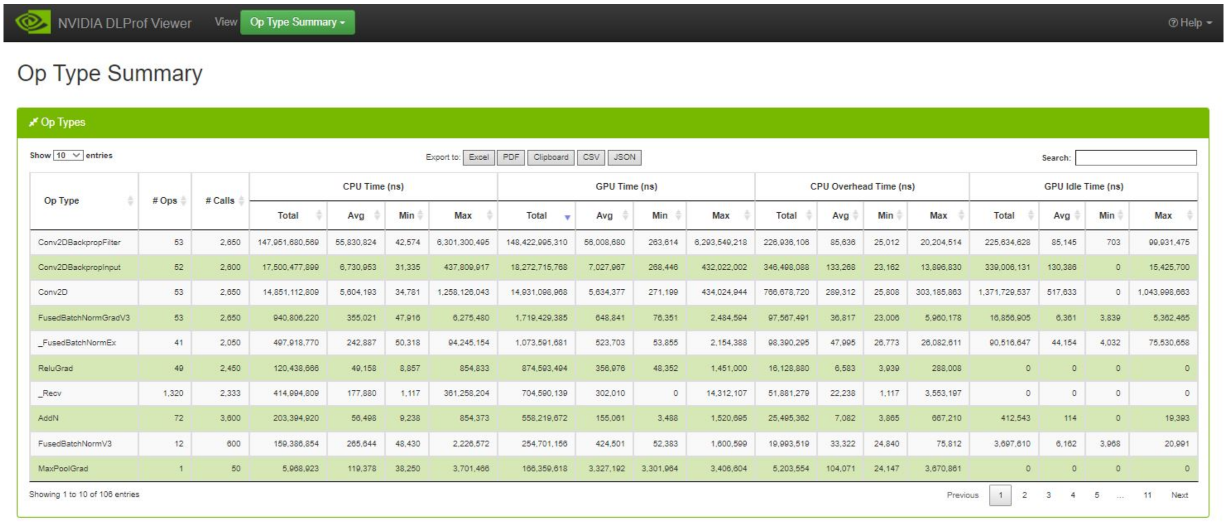
Understanding GPU utilization and timing details of the operations is the first step in profiling your model.

- To learn more about Tensor cores and Mixed Precision training, visit this site:https://developer.nvidia.com/tensor_cores
- You will find resources on how to train networks with mixed precision and make full use of Tensor cores for Tensorflow models here: https://docs.nvidia.com/deeplearning/sdk/mixed-precision-training/index.html#training_tensorflow
- Note that if there are multiple kernels being observed on single op node, these are likely performing data transposes to prepare the data for efficient use by tensorcores. Such transposes themselves would not use tensor cores.

3.4.2. Op Type Summary

This table aggregates metrics over all op types and enables users to see the performance of all the ops in terms of its types, such as Convolutions, Matrix Multiplications, etc.

See this description for all the features available in all DataTables.



Op Type Summary

Op Type	# Ops	# Calls	CPU Time (ns)				GPU Time (ns)				CPU Overhead Time (ns)				GPU Idle Time (ns)			
			Total	Avg	Min	Max	Total	Avg	Min	Max	Total	Avg	Min	Max	Total	Avg	Min	Max
Conv2DBackpropFilter	53	2,650	147,951,680,569	55,830,824	42,574	6,301,300,495	148,422,995,310	56,008,680	263,614	6,293,549,218	226,936,106	85,636	25,012	20,204,514	225,634,628	85,145	703	99,931,475
Conv2DBackpropInput	52	2,600	17,500,477,899	6,730,953	31,335	437,809,917	18,272,715,768	7,027,967	268,446	432,022,002	346,498,088	133,268	23,162	13,896,830	339,006,131	130,386	0	15,425,700
Conv2D	53	2,650	14,851,112,809	5,604,193	34,781	1,258,126,043	14,931,098,968	5,634,377	271,199	434,024,944	766,678,720	289,312	25,808	303,185,863	1,371,729,537	517,633	0	1,043,998,663
FusedBatchNormGradV3	53	2,650	940,806,220	355,021	47,916	6,275,480	1,719,429,385	648,841	76,351	2,484,594	97,567,491	36,817	23,006	5,960,178	16,856,905	6,361	3,839	5,362,465
_FusedBatchNormEx	41	2,050	497,918,770	242,887	50,318	94,245,154	1,073,591,681	523,703	53,855	2,154,388	98,390,295	47,995	26,773	26,082,611	90,516,647	44,154	4,032	75,530,658
ReluGrad	49	2,450	120,438,666	49,158	8,857	854,833	874,593,494	356,976	48,352	1,451,000	16,128,880	6,583	3,939	288,008	0	0	0	0
_Recv	1,320	2,333	414,994,809	177,880	1,117	361,258,204	704,590,139	302,010	0	14,312,107	51,881,279	22,238	1,117	3,553,197	0	0	0	0
AddN	72	3,600	203,394,920	56,498	9,238	854,373	558,219,672	155,061	3,488	1,520,695	25,495,362	7,082	3,865	667,210	412,543	114	0	19,393
FusedBatchNormV3	12	600	159,386,854	265,644	48,430	2,226,572	254,701,156	424,501	52,383	1,600,599	19,993,519	33,322	24,840	75,812	3,697,610	6,162	3,968	20,991
MaxPoolGrad	1	50	5,968,923	119,378	38,250	3,701,466	166,359,618	3,327,192	3,301,964	3,406,604	5,203,554	104,071	24,147	3,670,861	0	0	0	0

Showing 1 to 10 of 106 entries

Op Type Data Table

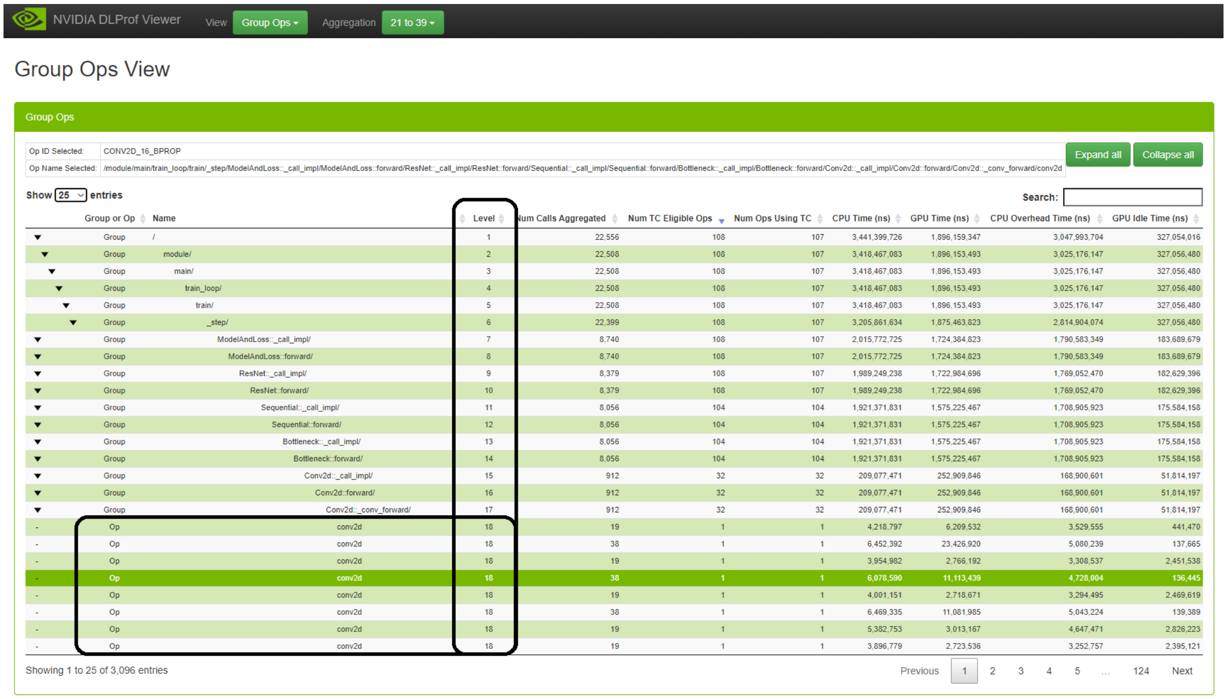
Column name	Description
Op Type	The operation type.
No. Ops	The number of ops that have the Op Type above.
No. Calls	Number of instances that the operation was called / executed.

Column name	Description
Total CPU Time (ns)	The total CPU time of all instances of this op type.
Avg. CPU Time (ns)	The average CPU time of all instances of this op type.
Min CPU Time (ns)	The minimum CPU time found amongst all instances of this op type.
Max CPU Time (ns)	The maximum CPU time found amongst all instances of this op type.
Total GPU Time (ns)	The total GPU time of all instances of this op type.
Avg. GPU Time (ns)	The average GPU time of all instances of this op type.
Min GPU Time (ns)	The minimum GPU time found amongst all instances of this op type.
Max GPU Time (ns)	The maximum GPU time found amongst all instances of this op type.
Total CPU Overhead Time (ns)	The total CPU overhead of all instances of this op type.
Avg. CPU Overhead Time (ns)	The average CPU overhead of all instances of this op type.
Min CPU Overhead Time (ns)	The minimum CPU overhead found amongst all instances of this op type.
Max CPU Overhead Time (ns)	The maximum CPU overhead found amongst all instances of this op type.
Total GPU Idle Time (ns)	The total GPU idle time of all instances of this op type.
Avg. GPU Idle Time (ns)	The average GPU idle time of all instances of this op type.
Min GPU Idle Time (ns)	The minimum GPU idle time found amongst all instances of this op type.
Max GPU Idle Time (ns)	The maximum GPU idle time found amongst all instances of this op type.

3.4.3. Group Ops

This view displays all operations in a hierarchical view. This new view is based on the 'group node report' provided by the DLProf CLI. Whereas the textual report provided by the CLI is a CSV file, this view is interactive by supporting filtering, sorting, and pagination. This information is interactive, it provides a powerful mechanism to quickly find ops and groups that consume too much duration.

See this description for all the features available in all DataTables.



Group Ops View

Op ID Selected: CONV2D_16_BPROP

Op Name Selected: /module/main/train_loop/train/_step/ModelAndLoss::_call_impl/ModelAndLoss::forward/ResNet::_call_impl/ResNet::forward/Sequential::_call_impl/Sequential::forward/Bottleneck::_call_impl/Bottleneck::forward/Conv2d::_call_impl/Conv2d::forward/Conv2d::_conv_forward/conv2d

Show 25 entries

	Group or Op	Name	Level	Num Calls Aggregated	Num TC Eligible Ops	Num Ops Using TC	CPU Time (ns)	GPU Time (ns)	CPU Overhead Time (ns)	GPU Idle Time (ns)
▼	Group	/	1	22,556	108	107	3,441,399,726	1,896,159,347	3,047,993,704	327,054,016
▼	Group	module/	2	22,508	108	107	3,418,467,083	1,896,153,493	3,025,176,147	327,056,480
▼	Group	main/	3	22,508	108	107	3,418,467,083	1,896,153,493	3,025,176,147	327,056,480
▼	Group	train_loop/	4	22,508	108	107	3,418,467,083	1,896,153,493	3,025,176,147	327,056,480
▼	Group	train/	5	22,508	108	107	3,418,467,083	1,896,153,493	3,025,176,147	327,056,480
▼	Group	_step/	6	22,399	108	107	3,205,861,634	1,875,463,823	2,814,904,074	327,056,480
▼	Group	ModelAndLoss::_call_impl/	7	8,740	108	107	2,015,772,725	1,724,384,823	1,790,583,349	183,689,679
▼	Group	ModelAndLoss::forward/	8	8,740	108	107	2,015,772,725	1,724,384,823	1,790,583,349	183,689,679
▼	Group	ResNet::_call_impl/	9	8,379	108	107	1,989,249,238	1,722,984,696	1,769,052,470	182,629,396
▼	Group	ResNet::forward/	10	8,379	108	107	1,989,249,238	1,722,984,696	1,769,052,470	182,629,396
▼	Group	Sequential::_call_impl/	11	8,056	104	104	1,921,371,831	1,575,225,467	1,708,905,923	175,584,158
▼	Group	Sequential::forward/	12	8,056	104	104	1,921,371,831	1,575,225,467	1,708,905,923	175,584,158
▼	Group	Bottleneck::_call_impl/	13	8,056	104	104	1,921,371,831	1,575,225,467	1,708,905,923	175,584,158
▼	Group	Bottleneck::forward/	14	8,056	104	104	1,921,371,831	1,575,225,467	1,708,905,923	175,584,158
▼	Group	Conv2d::_call_impl/	15	912	32	32	209,077,471	252,909,846	168,900,601	51,814,197
▼	Group	Conv2d::forward/	16	912	32	32	209,077,471	252,909,846	168,900,601	51,814,197
▼	Group	Conv2d::_conv_forward/	17	912	32	32	209,077,471	252,909,846	168,900,601	51,814,197
-	Op	conv2d	18	19	1	1	4,218,797	6,209,532	3,529,555	441,470
-	Op	conv2d	18	38	1	1	6,452,392	23,426,920	5,080,239	137,665
-	Op	conv2d	18	19	1	1	3,954,982	2,766,192	3,308,537	2,451,538
-	Op	conv2d	18	38	1	1	6,078,590	11,113,439	4,728,004	136,445
-	Op	conv2d	18	19	1	1	4,001,151	2,718,671	3,294,495	2,469,619
-	Op	conv2d	18	38	1	1	6,469,335	11,081,985	5,043,224	139,389
-	Op	conv2d	18	19	1	1	5,382,753	3,013,167	4,647,471	2,826,223
-	Op	conv2d	18	19	1	1	3,896,779	2,723,536	3,252,757	2,395,121

Showing 1 to 25 of 3,096 entries

The intersection of box black highlighting boxes above depicts the first eight of 32 ops in the level 17 group called 'Conv2d::_conv_forward'. The values in the upper levels of the network graph are the sums of all the ops in the lower levels.

Definitions

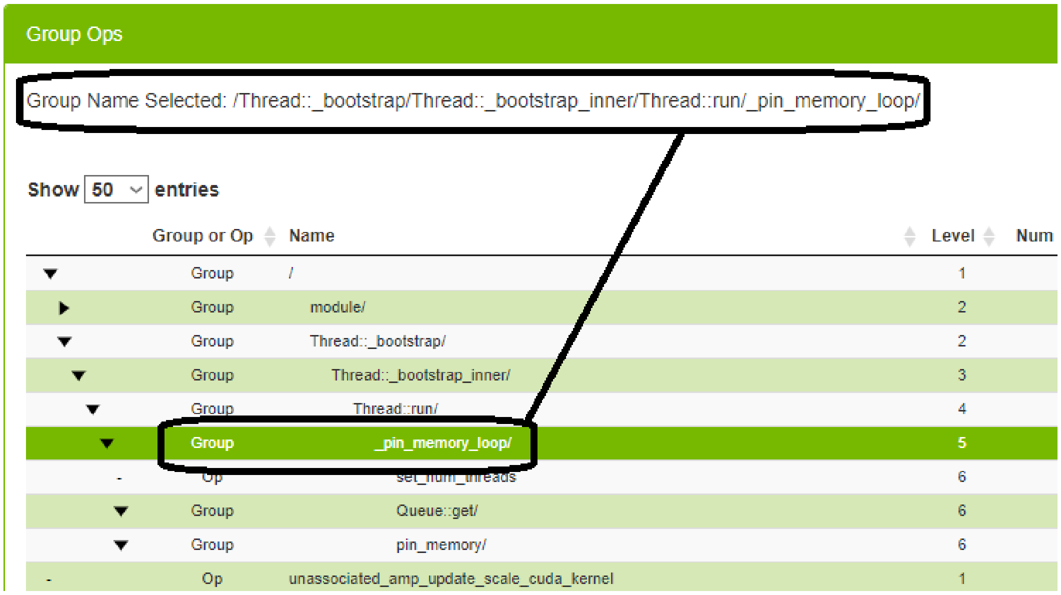
Column	Definition
Group	An entity in the tree that represents a collection of ops.
Op	A normal operation in the network graph.
Level	Represents the depth of the group or op. Level one is the root. The higher the level value the deeper/lower the group or op is in the network graph.
▼ or ▶	Depict whether Group Op is Expanded or Collapsed. The indentation of the arrows depicts the level (or depth) of the group. **Note:** There is a bug in the 3rd party treetable control whereby a group level 7 or higher appears in column one.

Column	Definition
Group or Op	One of two values: Group or Op.
Name	The name of the selected row. A group has an appending slash. The indentation depicts the level (or depth) of the name in the network graph.
Level	Represents the depth of the group or op.
All other columns	▶ If an op, the values represent counts and durations for that individual op . ▶ If a group, the values represent a summation of all the encapsulated ops

UI Controls

Control	Definition
Expand All Collapse All	Expand or collapse all Groups. Expand All will show those Groups and Ops at level one. Collapse All will show all Groups and Ops.

Selection Panel

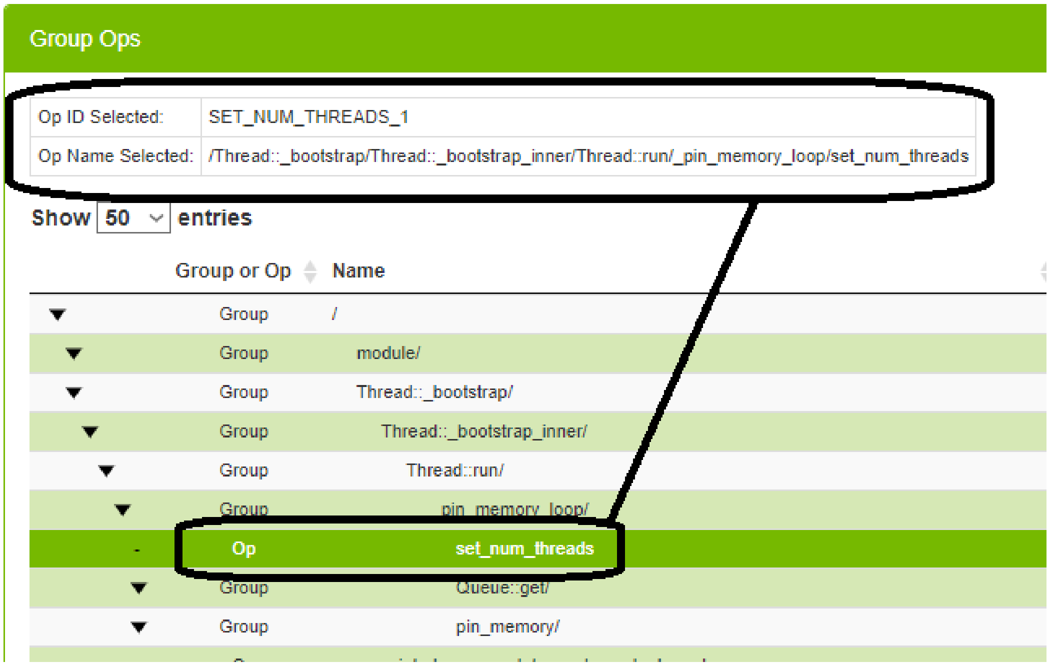
Selection	Definition
Group	

Group Ops

Group Name Selected: /Thread::_bootstrap/Thread::_bootstrap_inner/Thread::run/_pin_memory_loop/

Show 50 entries

	Group or Op	Name	Level	Num
▼	Group	/	1	
▶	Group	module/	2	
▼	Group	Thread::_bootstrap/	2	
▼	Group	Thread::_bootstrap_inner/	3	
▼	Group	Thread::run/	4	
▼	Group	_pin_memory_loop/	5	
-	Op	set_num_threads	6	
▼	Group	Queue::get/	6	
▼	Group	pin_memory/	6	
-	Op	unassociated_amp_update_scale_cuda_kernel	1	

Op Name and Op ID match	(Not pictured: same as 'Group' above except the label is 'Op Name Selected')
Op Name and Op ID do not match	

Group Ops

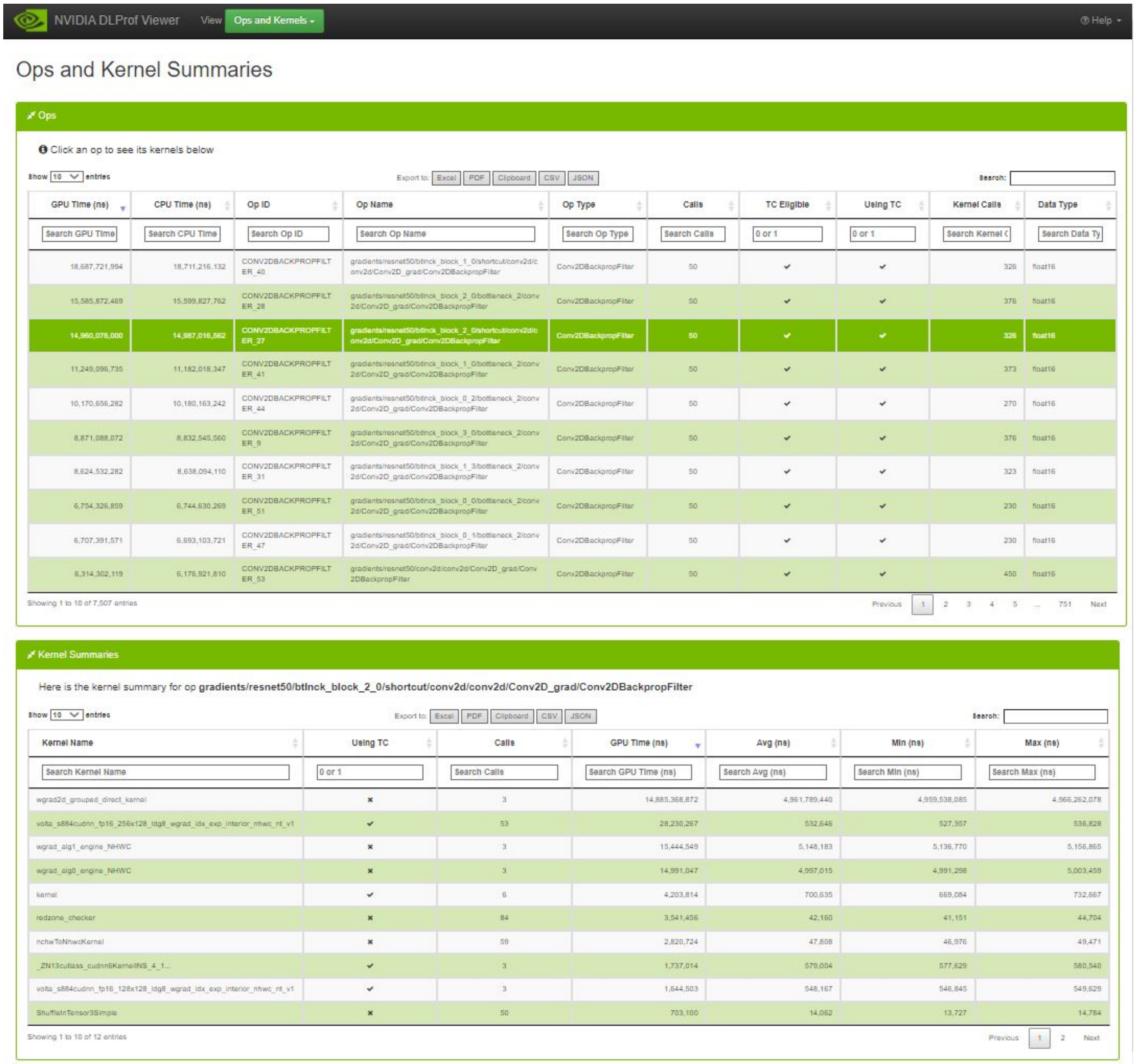
Op ID Selected:	SET_NUM_THREADS_1
Op Name Selected:	/Thread::_bootstrap/Thread::_bootstrap_inner/Thread::run/_pin_memory_loop/set_num_threads

Show 50 entries

	Group or Op	Name
▼	Group	/
▼	Group	module/
▼	Group	Thread::_bootstrap/
▼	Group	Thread::_bootstrap_inner/
▼	Group	Thread::run/
▼	Group	pin_memory_loop/
-	Op	set_num_threads
▼	Group	Queue::get/
▼	Group	pin_memory/

3.4.4. Ops and Kernels

This view enables users to view, search, sort all ops and their corresponding kernels in the entire network.



Ops and Kernel Summaries

Ops

Click an op to see its kernels below

Show 10 entries | Export to: Excel PDF Clipboard CSV JSON | Search:

GPU Time (ns)	CPU Time (ns)	Op ID	Op Name	Op Type	Calls	TC Eligible	Using TC	Kernel Calls	Data Type
Search GPU Time	Search CPU Time	Search Op ID	Search Op Name	Search Op Type	Search Calls	0 or 1	0 or 1	Search Kernel C	Search Data Ty
18,687,721,994	18,711,216,132	CONV2DBACKPROPFILTER_40	gradients/resnet50/btlnck_block_1_0/shortcut/conv2d/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	50	✓	✓	326	float16
15,585,872,469	15,599,827,762	CONV2DBACKPROPFILTER_28	gradients/resnet50/btlnck_block_2_0/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	50	✓	✓	376	float16
14,960,076,000	14,987,016,562	CONV2DBACKPROPFILTER_27	gradients/resnet50/btlnck_block_2_0/shortcut/conv2d/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	50	✓	✓	326	float16
11,249,096,735	11,182,018,347	CONV2DBACKPROPFILTER_41	gradients/resnet50/btlnck_block_1_0/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	50	✓	✓	373	float16
10,170,656,282	10,180,163,242	CONV2DBACKPROPFILTER_44	gradients/resnet50/btlnck_block_0_2/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	50	✓	✓	270	float16
8,871,088,072	8,832,545,560	CONV2DBACKPROPFILTER_9	gradients/resnet50/btlnck_block_3_0/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	50	✓	✓	376	float16
8,624,532,282	8,638,094,110	CONV2DBACKPROPFILTER_31	gradients/resnet50/btlnck_block_1_3/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	50	✓	✓	323	float16
6,754,326,859	6,744,630,269	CONV2DBACKPROPFILTER_51	gradients/resnet50/btlnck_block_0_0/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	50	✓	✓	230	float16
6,707,391,571	6,693,103,721	CONV2DBACKPROPFILTER_47	gradients/resnet50/btlnck_block_0_1/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	50	✓	✓	230	float16
6,314,302,119	6,176,921,810	CONV2DBACKPROPFILTER_53	gradients/resnet50/conv2d/conv2d/Conv2D_grad/Conv2DBackpropFilter	Conv2DBackpropFilter	50	✓	✓	450	float16

Showing 1 to 10 of 7,507 entries

Previous 1 2 3 4 5 ... 751 Next

Kernel Summaries

Here is the kernel summary for op **gradients/resnet50/btlnck_block_2_0/shortcut/conv2d/conv2d/Conv2D_grad/Conv2DBackpropFilter**

Show 10 entries | Export to: Excel PDF Clipboard CSV JSON | Search:

Kernel Name	Using TC	Calls	GPU Time (ns)	Avg (ns)	Min (ns)	Max (ns)
Search Kernel Name	0 or 1	Search Calls	Search GPU Time (ns)	Search Avg (ns)	Search Min (ns)	Search Max (ns)
wgrad2d_grouped_direct_kernel	✗	3	14,885,368,872	4,961,789,440	4,959,538,085	4,966,262,078
volta_s884cudnn_fp16_256x128_ldg8_wgrad_idx_exp_interior_nhwc_nt_v1	✓	53	28,230,267	532,646	527,357	536,828
wgrad_alg1_engine_NHWC	✗	3	15,444,549	5,148,183	5,136,770	5,156,865
wgrad_alg0_engine_NHWC	✗	3	14,991,047	4,997,015	4,991,298	5,003,459
kernel	✓	6	4,203,814	700,635	669,084	732,667
redzone_checker	✗	84	3,541,456	42,160	41,151	44,704
nchwToNhwcKernel	✗	59	2,820,724	47,808	46,976	49,471
_ZN13cutlass_cudnn6KernelINS_4_1...	✓	3	1,737,014	579,004	577,629	580,540
volta_s884cudnn_fp16_128x128_ldg8_wgrad_idx_exp_interior_nhwc_nt_v1	✓	3	1,644,503	548,167	546,845	549,629
ShuffleInTensor3Simple	✗	50	703,100	14,062	13,727	14,784

Showing 1 to 10 of 12 entries

Previous 1 2 Next

3.4.4.1. Ops Data Table

When a row is selected in the Ops table, a summary of each kernel for that op is displayed in the bottom table.

See this description for all the features available in all DataTables.

Entry	Description
GPU Time (ns)	Cumulative time executing all GPU kernels launched for the op.
CPU Time (ns)	Cumulative time executing all op instances on the CPU.
Op Name	The name of the op.

Entry	Description
Direction	The fprop/bprop direction of the op. (only visible on PyTorch runs).
Op Type	The operation of the op.
Calls	The number of times the op was called.
TC Eligible	A true/false field indicating whether or not the op is eligible to use Tensor Core kernels. To filter, type '1' for true, and '0' for false.
Using TC	A true/false field indicating whether or not one of the kernels launched in this op is using Tensor Cores. To filter, type '1' for true, and '0' for false.
Kernel Calls	The number of kernels launched in the op.
Data Type	The data type of this op (eg, float16, int64, int32)
Stack Trace	The stack trace of the op. (only visible on PyTorch runs). If the contents of this cell is more than 100 characters, a 'See More' hyperlink appears. When clicked, the full contents of the cell appears. When the cell is expanded, the hyperlink text is changed to 'See Less'. When clicked, the cell collapses back to the first 100 characters.

3.4.4.2. Kernel Summaries Data Table

See this description for all the features available in all DataTables.

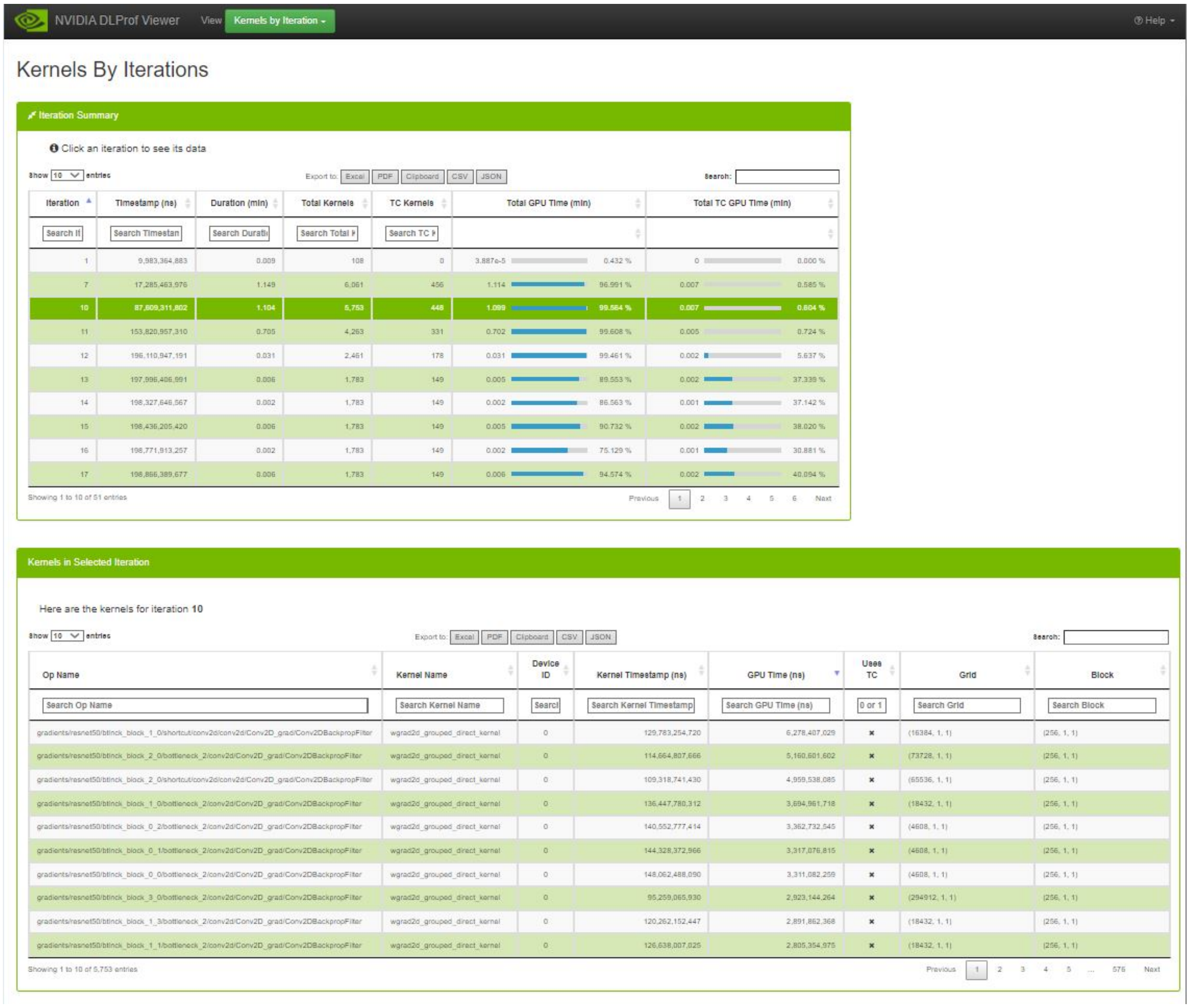
Kernels Summary table

Entry	Description
Kernel name	The full name of the kernel.
Using TC	A true/false field indicating whether or not the kernel is actually using a Tensor Core. To filter, type '1' for true, and '0' for false.
Calls	The number of times this kernel was launched.
GPU Time (ns)	The aggregate duration of each time the kernel was launched.

Entry	Description
Avg (ns)	The average duration of each time this kernel was launched.
Min (ns)	The minimum duration of all kernel launches.
Max (ns)	The maximum duration of all kernel launches.

3.4.5. Kernels by Iteration

The Kernels by Iterations view shows operations and their execution time for every iteration. At a glance, you can compare iterations with respect to time as well as Kernels executed.


NVIDIA DLProf Viewer
View
Kernels by Iteration
Help
Export to: Excel PDF Clipboard CSV JSON
Search:
Previous 1 2 3 4 5 6 Next
Export to: Excel PDF Clipboard CSV JSON
Search:
Previous 1 2 3 4 5 ... 576 Next

Kernels By Iterations

Iteration Summary

Click an iteration to see its data

Show 10 entries

Iteration	Timestamp (ns)	Duration (min)	Total Kernels	TC Kernels	Total GPU Time (min)	Total TC GPU Time (min)
Search It	Search Timestan	Search Durati	Search Total K	Search TC K		
1	9,983,364,883	0.009	108	0	3.887e-5 0.432 %	0 0.000 %
7	17,285,463,976	1.149	6,061	456	1.114 96.991 %	0.007 0.585 %
10	87,609,311,802	1.104	5,753	448	1.099 99.564 %	0.007 0.604 %
11	153,820,957,310	0.705	4,263	331	0.702 99.608 %	0.005 0.724 %
12	196,110,947,191	0.031	2,461	178	0.031 99.461 %	0.002 5.637 %
13	197,996,406,991	0.006	1,783	149	0.005 89.553 %	0.002 37.339 %
14	198,327,646,567	0.002	1,783	149	0.002 86.563 %	0.001 37.142 %
15	198,436,205,420	0.006	1,783	149	0.005 90.732 %	0.002 38.020 %
16	198,771,913,257	0.002	1,783	149	0.002 75.129 %	0.001 30.881 %
17	198,866,389,677	0.006	1,783	149	0.006 94.574 %	0.002 40.094 %

Showing 1 to 10 of 51 entries

Kernels in Selected Iteration

Here are the kernels for iteration **10**

Show 10 entries

Op Name	Kernel Name	Device ID	Kernel Timestamp (ns)	GPU Time (ns)	Uses TC	Grid	Block
Search Op Name	Search Kernel Name	Searcl	Search Kernel Timestamp	Search GPU Time (ns)	0 or 1	Search Grid	Search Block
gradients/resnet50/btlnck_block_1_0/shortcut/conv2d/conv2d/Conv2D_grad/Conv2DBackpropFilter	wgrad2d_grouped_direct_kernel	0	129,783,254,720	6,278,407,029	✖	(16384, 1, 1)	(256, 1, 1)
gradients/resnet50/btlnck_block_2_0/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	wgrad2d_grouped_direct_kernel	0	114,664,807,666	5,160,601,602	✖	(73728, 1, 1)	(256, 1, 1)
gradients/resnet50/btlnck_block_2_0/shortcut/conv2d/conv2d/Conv2D_grad/Conv2DBackpropFilter	wgrad2d_grouped_direct_kernel	0	109,318,741,430	4,959,538,085	✖	(65536, 1, 1)	(256, 1, 1)
gradients/resnet50/btlnck_block_1_0/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	wgrad2d_grouped_direct_kernel	0	136,447,780,312	3,694,961,718	✖	(18432, 1, 1)	(256, 1, 1)
gradients/resnet50/btlnck_block_0_2/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	wgrad2d_grouped_direct_kernel	0	140,552,777,414	3,362,732,545	✖	(4608, 1, 1)	(256, 1, 1)
gradients/resnet50/btlnck_block_0_1/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	wgrad2d_grouped_direct_kernel	0	144,328,372,966	3,317,076,815	✖	(4608, 1, 1)	(256, 1, 1)
gradients/resnet50/btlnck_block_0_0/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	wgrad2d_grouped_direct_kernel	0	148,062,488,090	3,311,082,259	✖	(4608, 1, 1)	(256, 1, 1)
gradients/resnet50/btlnck_block_3_0/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	wgrad2d_grouped_direct_kernel	0	95,259,065,930	2,923,144,264	✖	(294912, 1, 1)	(256, 1, 1)
gradients/resnet50/btlnck_block_1_3/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	wgrad2d_grouped_direct_kernel	0	120,262,152,447	2,891,862,368	✖	(18432, 1, 1)	(256, 1, 1)
gradients/resnet50/btlnck_block_1_1/bottleneck_2/conv2d/Conv2D_grad/Conv2DBackpropFilter	wgrad2d_grouped_direct_kernel	0	126,638,007,025	2,805,354,975	✖	(18432, 1, 1)	(256, 1, 1)

Showing 1 to 10 of 5,753 entries

3.4.5.1. Iteration Summary Data Table

To see the kernels for a specific iteration, click a row in the top table. The 'Kernels in Selected Iteration' table will be filled with the kernels from the selection iteration.

See this description for all the features available in all DataTables.

Column	Description
Iteration	The iteration interval numbers.
Timestamp	The exact time when this iteration started.
Duration	The length of time it took for this iteration to execute. Units on this column are dynamic.
Total Kernels	The number of GPU kernels called during this iteration.
TC Kernels	The number of GPU Tensor Core kernels called during the iteration.
Average GPU Time (μs)	Average execution time for all kernels across all GPUs during the iteration. The contents of this cell are broken into multiple pieces: 1. The value is the average execution time of this iteration across all GPUs. 2. The progress meter visually indicates how much of this iteration executed on the GPU. 3. The percentage is the average GPU time of all kernels across all GPUs over the total iteration time. 4. Hovering over this cell displays a tooltip text with helpful information. Units on this column are dynamic.
Average TC GPU Time (μs)	Average execution time for all Tensor Core kernels across all GPUs during the iteration. The contents of this cell are broken into multiple pieces: 1. The value is the average execution time of Tensor Core kernels during this iteration across all GPUs. 2. The progress meter visually indicates how much of this iteration executed Tensor Core kernels. 3. The percentage is the average GPU time of kernels using Tensor Core across all GPUs over the total iteration time. 4. Hovering over this cell displays a tooltip text with helpful information. Units on this column are dynamic.

3.4.5.2. Kernels in Selected Iteration

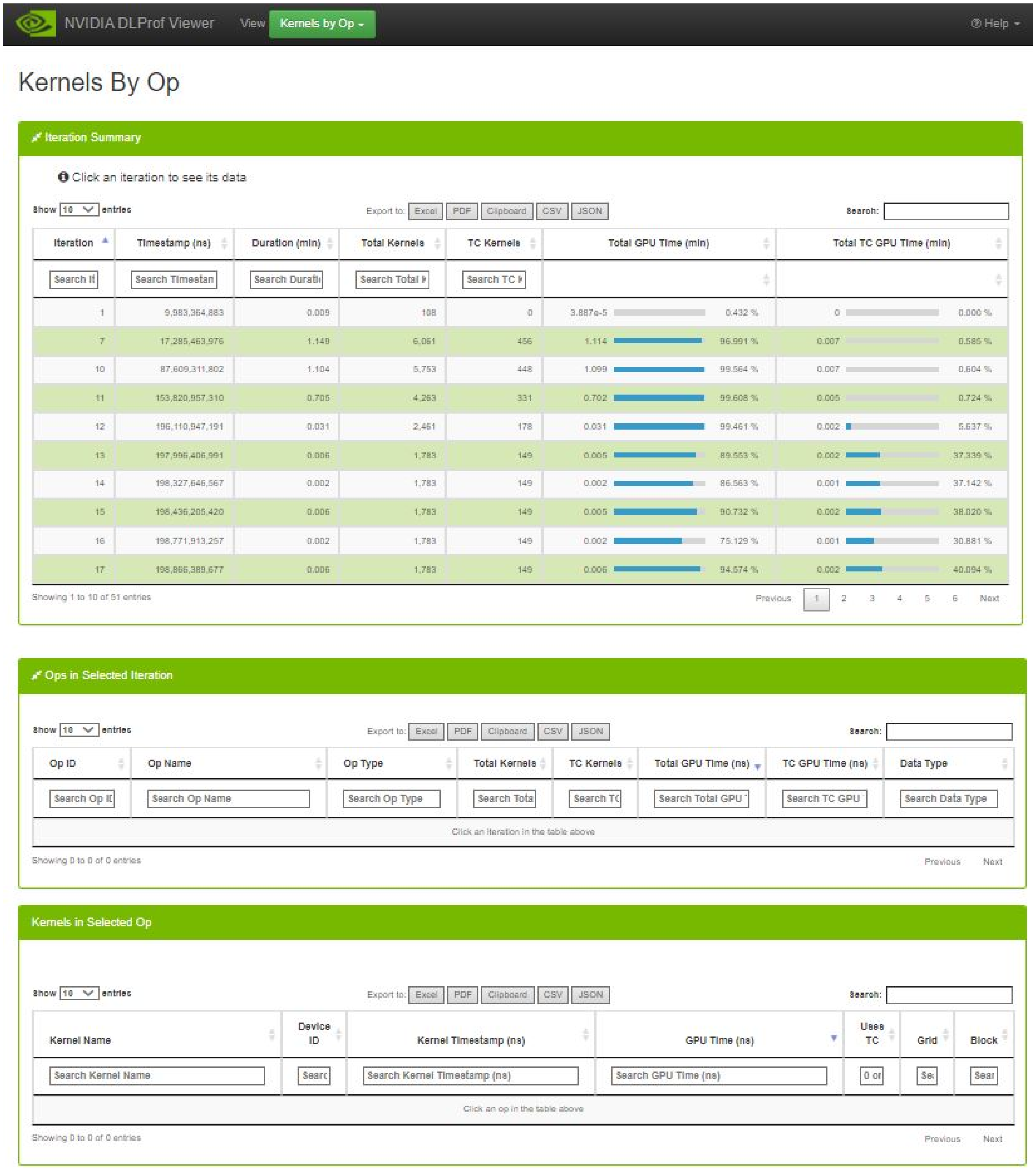
See this description for all the features available in all DataTables.

Column	Description
Op Name	The name of the op that launched the kernel.
Kernel Name	The name of the kernel.
Device ID	the device ID of the kernel.
Kernel Timestamp (ns)	The timestamp for when the CUDA API call was made for this kernel in the CPU thread. Useful to see the order in which the kernels were called.
GPU Time (ns)	The time spent executing the kernel on the GPU.
Users TC	A true/false field indicating whether or not the kernel uses Tensor Cores. To filter, type '1' for true, and '0' for false.
Grid	The grid size of the kernel.
Block	The block size of the kernel.

3.4.6. Kernels by Op

The Kernels by Op view is a variation of the Kernels by Iterations view. It has the capability to filter the list of kernels by iterations and op.

See this description for all the features available in all DataTables.



Kernels By Op

Iteration Summary

Click an iteration to see its data

Show 10 entries   Export to: Excel PDF Clipboard CSV JSON   Search:

Iteration	Timestamp (ns)	Duration (min)	Total Kernels	TC Kernels	Total GPU Time (min)	Total TC GPU Time (min)
Search It	Search Timestan	Search Durati	Search Total K	Search TC K		
1	9,983,364,883	0.009	108	0	3.887e-5 0.432 %	0 0.000 %
7	17,285,463,976	1.149	6,061	456	1.114 96.991 %	0.007 0.585 %
10	87,609,311,802	1.104	5,753	448	1.099 99.564 %	0.007 0.604 %
11	153,820,957,310	0.705	4,263	331	0.702 99.608 %	0.005 0.724 %
12	196,110,947,191	0.031	2,461	178	0.031 99.461 %	0.002 5.637 %
13	197,996,406,991	0.006	1,783	149	0.005 89.553 %	0.002 37.339 %
14	198,327,646,567	0.002	1,783	149	0.002 86.563 %	0.001 37.142 %
15	198,436,205,420	0.006	1,783	149	0.005 90.732 %	0.002 38.020 %
16	198,771,913,257	0.002	1,783	149	0.002 75.129 %	0.001 30.881 %
17	198,866,389,677	0.006	1,783	149	0.006 94.574 %	0.002 40.094 %

Showing 1 to 10 of 51 entries   Previous 1 2 3 4 5 6 Next

Ops in Selected Iteration

Show 10 entries   Export to: Excel PDF Clipboard CSV JSON   Search:

Op ID	Op Name	Op Type	Total Kernels	TC Kernels	Total GPU Time (ns)	TC GPU Time (ns)	Data Type
Search Op ID	Search Op Name	Search Op Type	Search Tota	Search TC	Search Total GPU '	Search TC GPU	Search Data Type
Click an iteration in the table above							

Showing 0 to 0 of 0 entries   Previous Next

Kernels in Selected Op

Show 10 entries   Export to: Excel PDF Clipboard CSV JSON   Search:

Kernel Name	Device ID	Kernel Timestamp (ns)	GPU Time (ns)	Uses TC	Grid	Block
Search Kernel Name	Searc	Search Kernel Timestamp (ns)	Search GPU Time (ns)	0 or	Se	Sear
Click an op in the table above						

Showing 0 to 0 of 0 entries   Previous Next

3.4.6.1. Iteration Summary Data Table

Selecting an iteration in the Iteration Summary table will populate the Ops in Selected Iteration table with all the profile data for the ops from the selected iteration. Selecting an op in the Ops in Selected Iteration table will populate the Kernels in Selected Op table with the list of kernels and timing data executed for the selected Op and Iteration.

See this description for all the features available in all DataTables.

Column	Description
Iteration	The iteration interval numbers
Timestamp	The exact time when this iteration started.
Duration	The length of time it took for this iteration to execute. Units on this column are dynamic.
Total Kernels	The number of GPU kernels called during this iteration.
TC Kernels	The number of GPU Tensor Core kernels called during the iteration.
Total GPU Time (µs)	Total execution time for all kernels across all GPUs during the iteration. The contents of this cell are broken into multiple pieces: 1. The value is the average execution time of this iteration across all GPUs. 2. The progress meter visually indicates how much of this iteration executed on the GPU. 3. The percentage is the average GPU time of all kernels across all GPUs over the total iteration time. 4. Hovering over this cell displays a tooltip text with helpful information. Units on this column are dynamic.
Total TC GPU Time (µs)	Total execution time for all Tensor Core kernels across all GPUs during the iteration. The contents of this cell are broken into multiple pieces: 1. The value is the average execution time of Tensor Core kernels during this iteration across all GPUs. 2. The progress meter visually indicates how much of this iteration executed Tensor Core kernels. 3. The percentage is the average GPU time of kernels using Tensor Core across all GPUs over the total iteration time. 4. Hovering over this cell displays a tooltip text with helpful information. Units on this column are dynamic.

3.4.6.2. Ops in Selected Iteration Table

Ops in Selected Iteration

See this description for all the features available in all DataTables.

Column	Description
Op Name	The name of the op that launched the kernel.
Direction	The fprop/bprop direction of the op. (only visible on PyTorch runs).
Op Type	The type of the op.
Total Kernels	The number of GPU kernels called during this iteration.
TC Kernels	The number of GPU Tensor Core kernels called during the iteration.
Total GPU Time (ns)	Cumulative execution time for all kernels on the GPU during the op.
TC GPU Time (ns)	Cumulative execution time for all Tensor Core kernels on the GPU during the op.
Data Type	The data type of this op (eg, float16, int64, int32).
Stack Trace	The stack trace of the op. (only visible on PyTorch runs) If the contents of this cell is more than 100 characters, a 'See More' hyperlink appears. When clicked, the full contents of the cell appears. When the cell is expanded, the hyperlink text is changed to 'See Less'. When clicked, the cell collapses back to the first 100 characters.

3.4.6.3. Kernels Selected Iteration / Op Combination Table

Kernels in Selected Iteration / Op combination

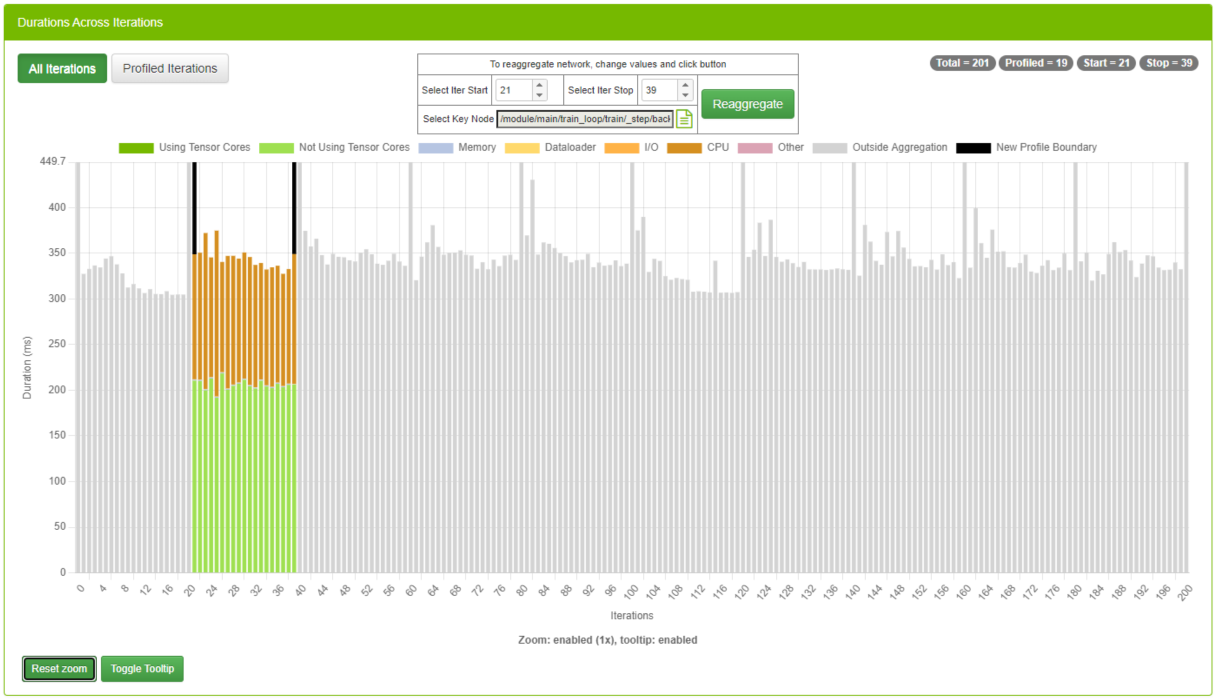
See this description for all the features available in all DataTables.

Column	Description
Kernel Name	The name of the kernel.
Device ID	The device ID of the kernel.

Column	Description
Kernel Timestamp (ns)	The timestamp for when the CUDA API call was made for this kernel in the CPU thread. Useful to see the order in which the kernels were called.
GPU Time (ns)	The time spent executing the kernel on the GPU.
Uses TC	A true/false field indicating whether or not the kernel uses Tensor Cores. To filter, type '1' for true, and '0' for false.
Grid	The grid size of the kernel.
Block	The block size of the kernel.

3.4.7. Iterations view

This view displays iterations visually. Users can quickly see how many iterations are in the model, the iterations that were aggregated/profiled, and the accumulated durations of tensor core kernels in each iteration.



To reaggregate network, change values and click button

Select Iter Start: 21

Select Iter Stop: 39

Select Key Node: /module/main/train_loop/train/_step/back

Total = 201 Profiled = 19 Start = 21 Stop = 39

Zoom: enabled (1x), tooltip: enabled

Fields

Field	Definition
X-Axis	The iterations of the model.
Y-Axis	Duration of iterations. The unit of this axis is dynamic (e.g., ms, ns, s, etc).

Zoom and Tooltip Label	Near the bottom/center, this label displays the current zoom factor, and the toggle state of the tooltip window.
Badges	In the top right, four badges display iteration specifics of the profiled network. ▶ Total ▶ Profiled ▶ Start ▶ Stop The values inside the start and stop badges come directly from DLProf: see the --iter_start and --iter_stop command line options.
Legend	▶ Dark Green: Accumulated duration of all kernels using tensor cores ▶ Light Green: Accumulated duration of all kernels not using tensor cores ▶ Blue: Accumulated duration of all memory operations ▶ Dark Orange: Accumulated duration of all CPU operations ▶ Orange: Accumulated duration of all I/O operations. ▶ Light Orange: Accumulated duration of all dataloder operations. ▶ Pink: All time not in any other category. ▶ Gray: Duration of iteration outside the profile. ▶ Black: User-positioned aggregation boundary. See 'Select Iter Start' and 'Select Iter Stop' below.

UI Controls

Control	Definition
Toggle Buttons	▶ All Iterations: Show all the iterations in the entire network ▶ Profiled Iterations: Zooms in to only show the iterations that were profiled
Chart Controls	▶ Reset Zoom: Resets zoom back to 1.0x ▶ Toggle Tooltip: Shows/hides the tooltip window when hovering over bars on the bar chart.
Hover	When mouse hovers over an iteration, a popup window appears displaying the values of all the constituents of the iteration.
Chart Mouse Wheel	▶ Mouse wheel up: Zoom in ▶ Mouse wheel down: Zoom out
Chart pan	Click and drag the chart to move it up or down.
Select Iter Start	Specification of a new "iteration start" value. Used for re-aggregating the data inside the viewer. The value can be changed by typing into the field, by clicking the up/down arrow spinners, or by hovering the cursor over the field and spinning the mouse wheel. When this value changes, the left black bar will move accordingly.

Select Iter Stop	Specification of a new "iteration stop" value. Used for re-aggregating the data inside the viewer. The value can be changed by typing into the field, by clicking the up/down arrow spinners, or by hovering the cursor over the field and spinning the mouse wheel. When this value changes, the right black bar will move accordingly.
Select Key Node	1. The existing key node value is displayed in a read-only field. 2. Sometimes key nodes can be very long. If a long key node is selected, the first few characters of the key node will appear in the field. The entire length of the key node field will appear when the cursor hovers over the field. 3. Click the Key Node Picker button to view and pick a new key node. See the dialog below.
Reaggregate	Instructs the DLProf back-end server to reaggregate the network with the specified parameters.

Workflow

Sometimes the original aggregation parameters on the DLProf command-line specified an iteration start value, an iteration stop value, or even key node that yielded a suboptimal representation of the network's profile (such as including warm-up nodes). This feature allows the user to change those values and re-aggregate.

Here is an example:

1. Notice in the Iterations View screenshot above:

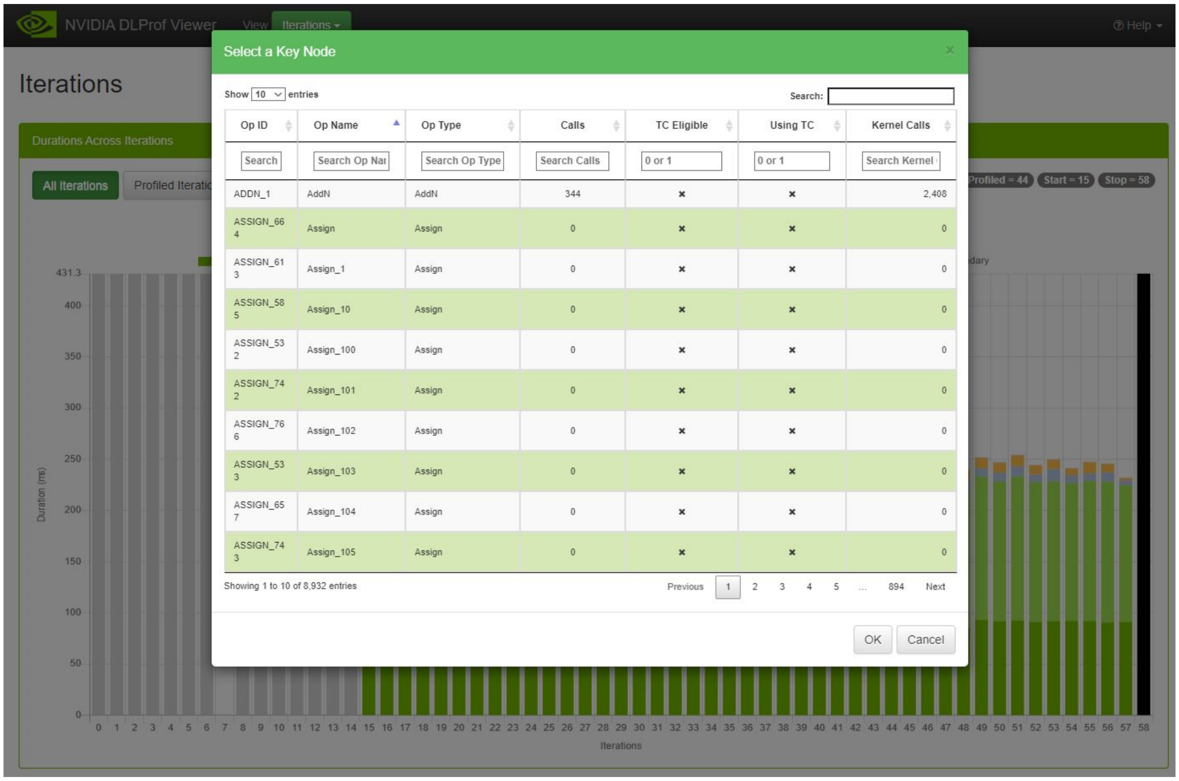
    a). Iteration eight contains no work, and

    b). Iteration nine through fifteen (among others) contain way too much work. The aggregation values throughout the viewer contain those ops and kernels and potentially skews the results.

2. You can change the iteration start and iteration stop values in a number of ways: typing and clicking the up and down spinners. The best way is to hover over the field and spin the mouse wheel.
3. Some platforms do not have a default key node for a neural network. Sometimes the predefined key node is suboptimal. By clicking on the 'Select Key Node' picker, a key node can be selected.

    This is a full-featured panel that allows for filtering, sorting, and pagination to find a Key Node. See the view called Ops and Kernels for more details.

    Note: There are fewer columns here than in the view, but the usability is the same.



Select a Key Node

Show 10 entries Search:

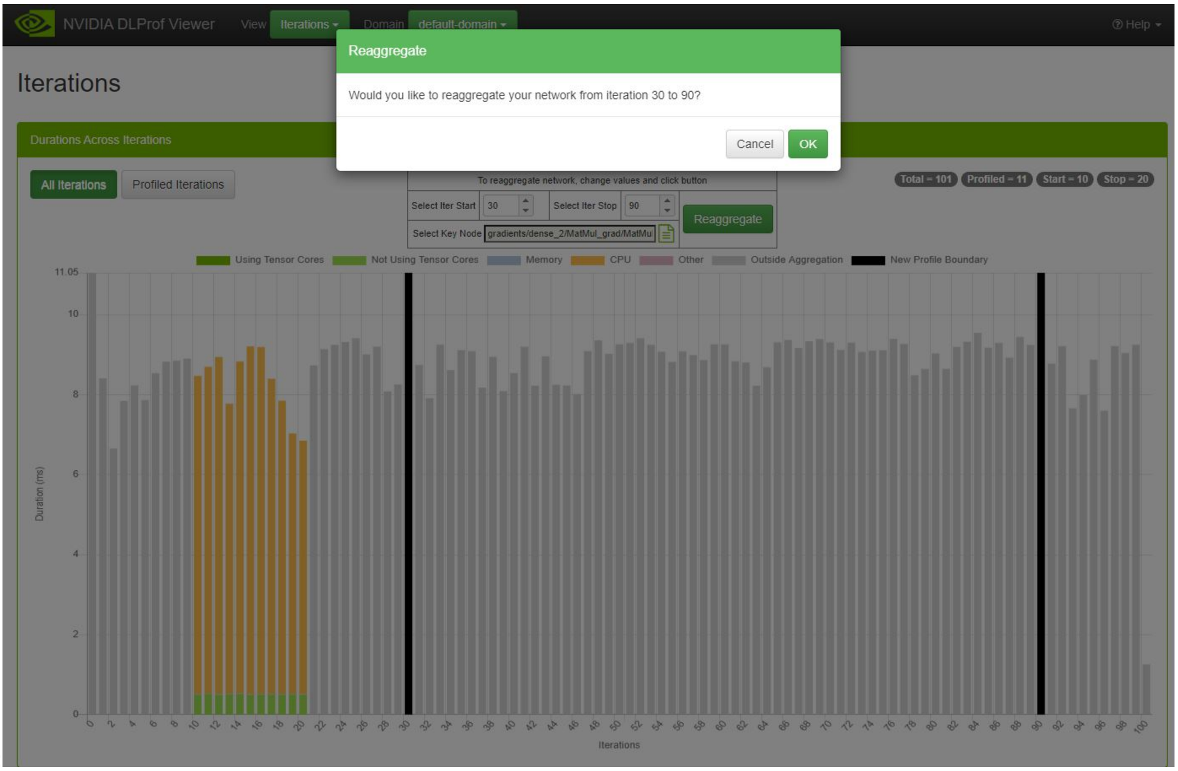
Op ID	Op Name	Op Type	Calls	TC Eligible	Using TC	Kernel Calls
Search	Search Op Nar	Search Op Type	Search Calls	0 or 1	0 or 1	Search Kernel
ADDN_1	AddN	AddN	344	✖	✖	2,408
ASSIGN_664	Assign	Assign	0	✖	✖	0
ASSIGN_613	Assign_1	Assign	0	✖	✖	0
ASSIGN_585	Assign_10	Assign	0	✖	✖	0
ASSIGN_532	Assign_100	Assign	0	✖	✖	0
ASSIGN_742	Assign_101	Assign	0	✖	✖	0
ASSIGN_766	Assign_102	Assign	0	✖	✖	0
ASSIGN_533	Assign_103	Assign	0	✖	✖	0
ASSIGN_657	Assign_104	Assign	0	✖	✖	0
ASSIGN_743	Assign_105	Assign	0	✖	✖	0

Showing 1 to 10 of 8,932 entries

Previous 1 2 3 4 5 … 894 Next

OK Cancel

4. Once any of these three fields change, click the Reaggregate button. A confirmation dialog is displayed:



Reaggregate

Would you like to reaggregate your network from iteration 30 to 90?

Cancel OK

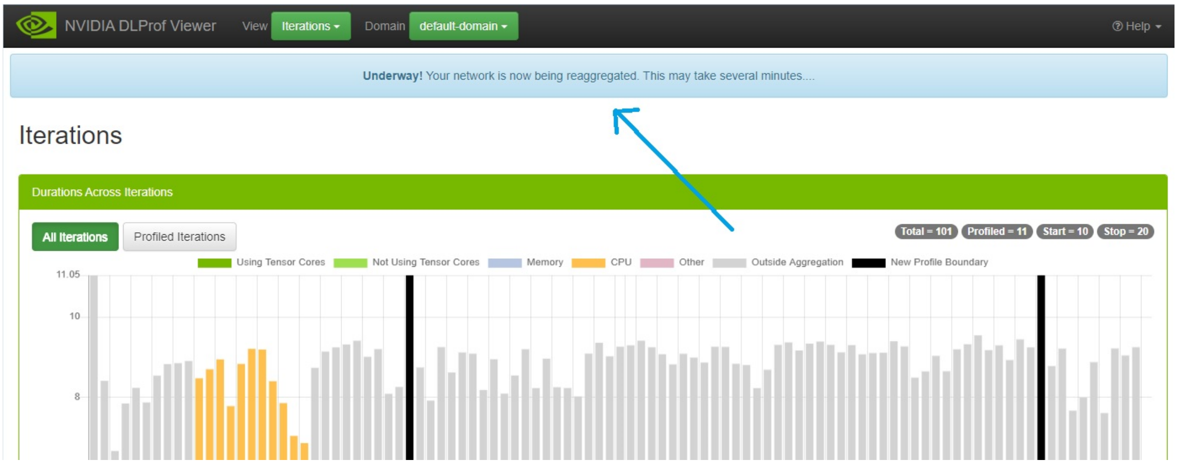
To reaggregate network, change values and click button

Select Iter Start: 30 Select Iter Stop: 90 Reaggregate

Select Key Node: gradients/dense_2/MatMul_grad/MatMul

5. After confirmation, a message is sent to the back-end DLProf server to reaggregate the profile with these values. This reaggregation process

could potentially take minutes, so the following panel appears above all panels:

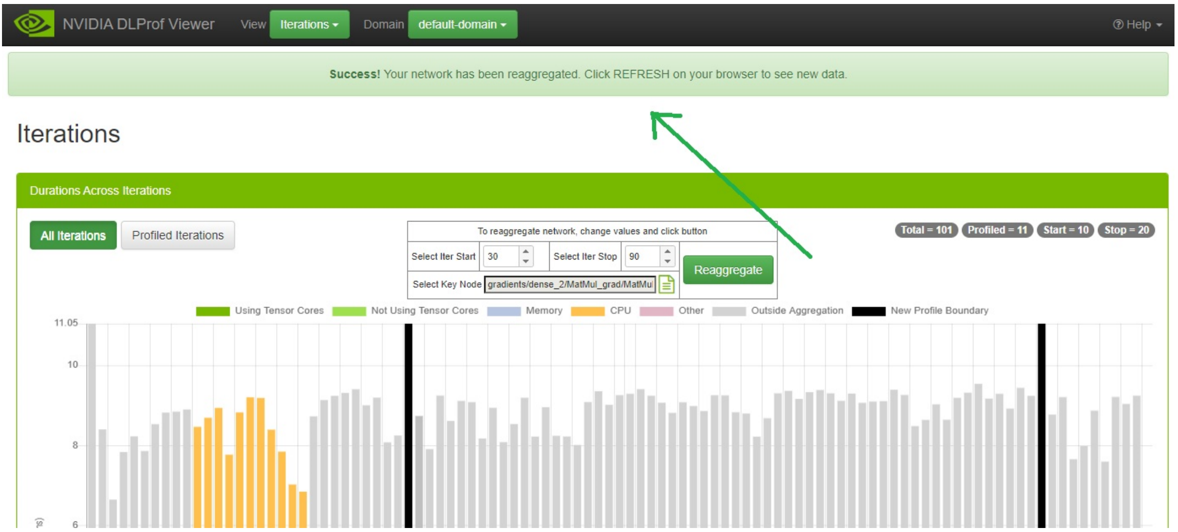


Underway! Your network is now being reaggregated. This may take several minutes....

The viewer is fully functional while the re-aggregation is taking place. The “Underway!” panel will remain in view above all other views and panels until reaggregation is complete.

> **Note:** Do not click the browser’s BACK or REFRESH button. If either are accidentally, the “Underway!” panel will no longer appear. The re-aggregation will continue regardless.

6. When the reaggregation is complete, a “Success!” panel will appear like this:



Success! Your network has been reaggregated. Click REFRESH on your browser to see new data.

To reaggregate network, change values and click button

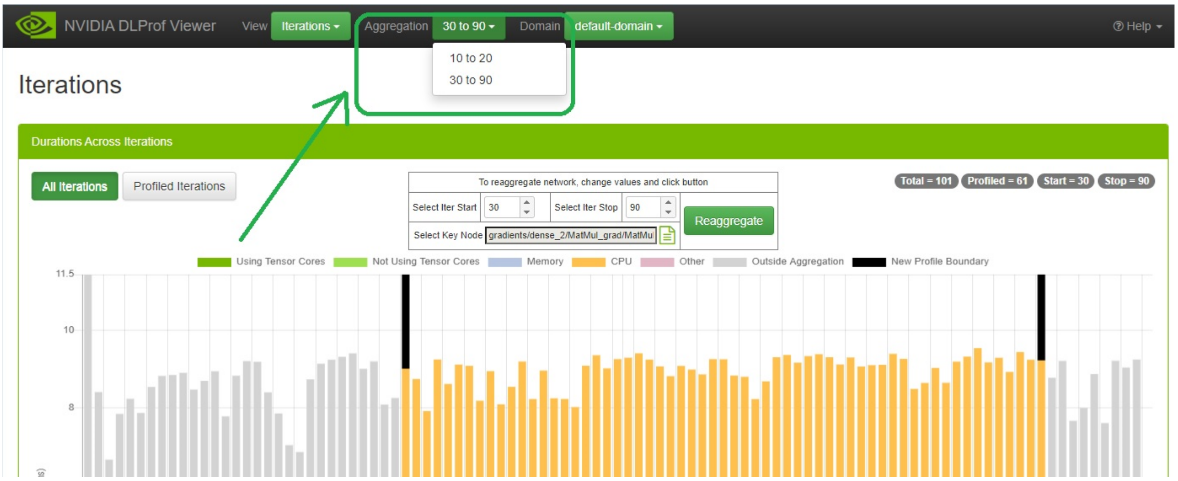
Select Iter Start: 30

Select Iter Stop: 90

Select Key Node: gradients/dense_2/MatMul_grad/MatMul

7. At this point, clicking the REFRESH button will load all the newly reaggregated data into the viewer. If it’s not already there, the

dropdown called Aggregation is displayed in the Navigation Bar:



To reaggregate network, change values and click button

Select Iter Start: 30

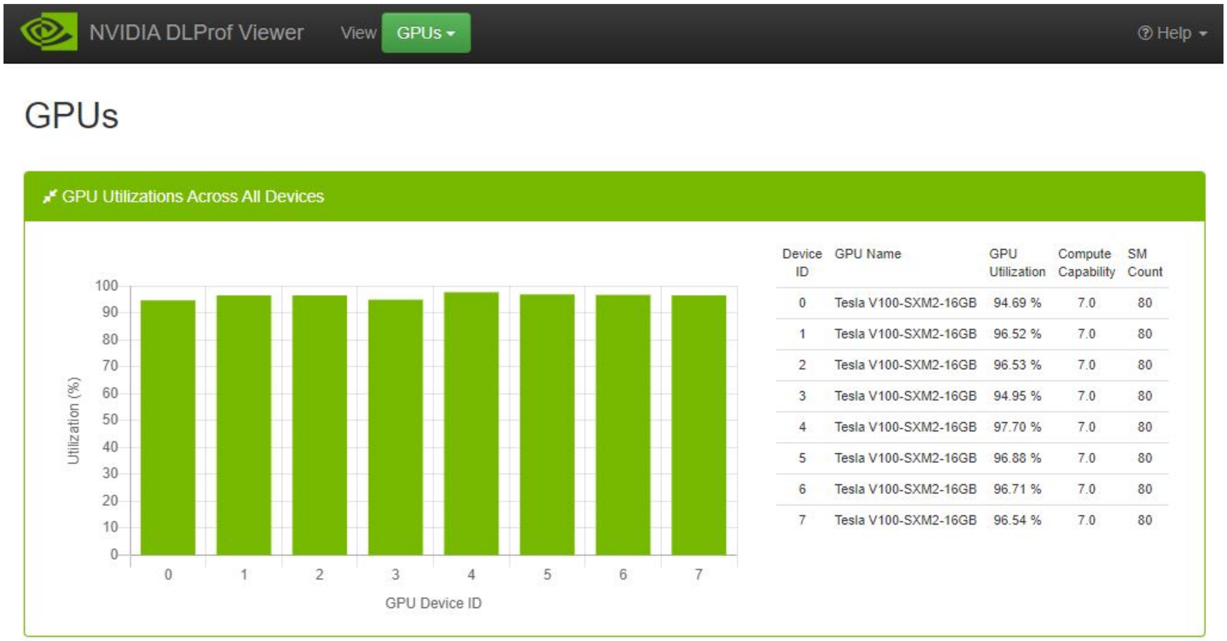
Select Iter Stop: 90

Select Key Node: gradients/dense_2/MatMul_grad/MatMul

3.4.8. GPUs View

This view shows the utilization of all GPUs during the profile run. It is broken down into two different but related elements:

- **Bar Chart** - Quick visualization where you can see the GPU utilization for every GPU used during the profile. This view appears only when more than one GPU is used in a profile.
- **Table** - This table shows a little more detail about each GPU device, including its name, Compute Capability, and SM count.



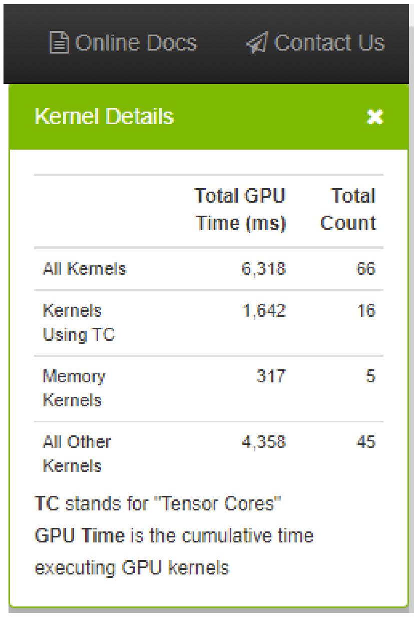
Device ID	GPU Name	GPU Utilization	Compute Capability	SM Count
0	Tesla V100-SXM2-16GB	94.69 %	7.0	80
1	Tesla V100-SXM2-16GB	96.52 %	7.0	80
2	Tesla V100-SXM2-16GB	96.53 %	7.0	80
3	Tesla V100-SXM2-16GB	94.95 %	7.0	80
4	Tesla V100-SXM2-16GB	97.70 %	7.0	80
5	Tesla V100-SXM2-16GB	96.88 %	7.0	80
6	Tesla V100-SXM2-16GB	96.71 %	7.0	80
7	Tesla V100-SXM2-16GB	96.54 %	7.0	80

3.4.9. Details Pane

This pane shows more details about a particular area of the system.

3.4.9.1. Kernel Details Panel

This panel provides key metrics about the kernels in the network aggregated over the specific iteration range.



	Total GPU Time (ms)	Total Count
All Kernels	6,318	66
Kernels Using TC	1,642	16
Memory Kernels	317	5
All Other Kernels	4,358	45

TC stands for "Tensor Cores"

GPU Time is the cumulative time executing GPU kernels

Fields

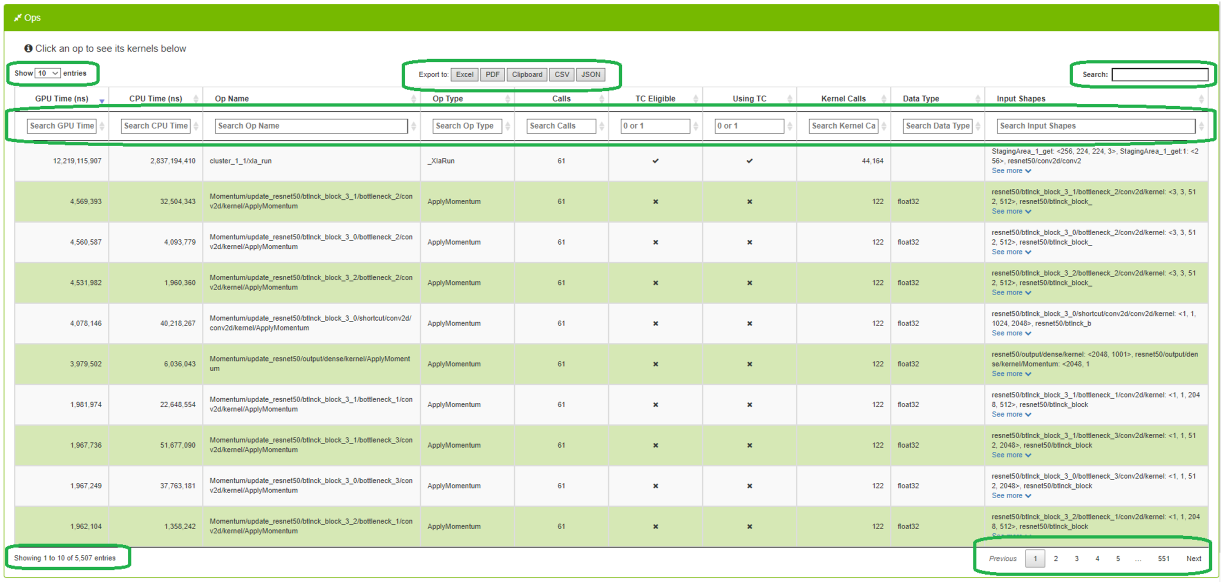
Field	Definition
All Kernels	Aggregates the total GPU time and count for all kernels in the network.
Kernels Using TC	Aggregates the total GPU time and count for all kernels using Tensor Cores.
Memory Kernels	Aggregates the total GPU time and count for all memory kernels.
All Other Kernels	Aggregates the total GPU time and count for all remaining kernel types.

UI Controls

Control	Definition
X	Closes the Details Pane.

3.5. Using Data Tables

DataTables are used in many views in the DLProf Viewer. The features in DataTables enable users to quickly find information. Below is a screenshot to see the location of the features, followed by a table describing the functionality of each feature.



Ops

Click an op to see its kernels below

Show 10 entries

Export to: Excel PDF Clipboard CSV JSON

Search:

GPU Time (ns)	CPU Time (ns)	Op Name	Op Type	Calls	TC Eligible	Using TC	Kernel Calls	Data Type	Input Shapes
Search GPU Time	Search CPU Time	Search Op Name	Search Op Type	Search Calls	0 or 1	0 or 1	Search Kernel Ca	Search Data Type	Search Input Shapes
12,219,115,907	2,837,194,410	cluster_1_1/xla_run	_XlaRun	61	✓	✓	44,164		StagingArea_1_get: <256, 224, 224, 3>, StagingArea_1_get:1: <256>, resnet50/conv2d/conv2 See more
4,569,393	32,504,343	Momentum/update_resnet50/btlnck_block_3_1/bottleneck_2/conv2d/kernel/ApplyMomentum	ApplyMomentum	61	✖	✖	122	float32	resnet50/btlnck_block_3_1/bottleneck_2/conv2d/kernel: <3, 3, 512, 512>, resnet50/btlnck_block_ See more
4,560,587	4,093,779	Momentum/update_resnet50/btlnck_block_3_0/bottleneck_2/conv2d/kernel/ApplyMomentum	ApplyMomentum	61	✖	✖	122	float32	resnet50/btlnck_block_3_0/bottleneck_2/conv2d/kernel: <3, 3, 512, 512>, resnet50/btlnck_block_ See more
4,531,982	1,960,360	Momentum/update_resnet50/btlnck_block_3_2/bottleneck_2/conv2d/kernel/ApplyMomentum	ApplyMomentum	61	✖	✖	122	float32	resnet50/btlnck_block_3_2/bottleneck_2/conv2d/kernel: <3, 3, 512, 512>, resnet50/btlnck_block_ See more
4,078,146	40,218,267	Momentum/update_resnet50/btlnck_block_3_0/shortcut/conv2d/conv2d/kernel/ApplyMomentum	ApplyMomentum	61	✖	✖	122	float32	resnet50/btlnck_block_3_0/shortcut/conv2d/conv2d/kernel: <1, 1, 1024, 2048>, resnet50/btlnck_b See more
3,979,502	6,036,043	Momentum/update_resnet50/output/dense/kernel/ApplyMomentum	ApplyMomentum	61	✖	✖	122	float32	resnet50/output/dense/kernel: <2048, 1001>, resnet50/output/dense/kernel/Momentum: <2048, 1 See more
1,981,974	22,648,554	Momentum/update_resnet50/btlnck_block_3_1/bottleneck_1/conv2d/kernel/ApplyMomentum	ApplyMomentum	61	✖	✖	122	float32	resnet50/btlnck_block_3_1/bottleneck_1/conv2d/kernel: <1, 1, 2048, 512>, resnet50/btlnck_block See more
1,967,736	51,677,090	Momentum/update_resnet50/btlnck_block_3_1/bottleneck_3/conv2d/kernel/ApplyMomentum	ApplyMomentum	61	✖	✖	122	float32	resnet50/btlnck_block_3_1/bottleneck_3/conv2d/kernel: <1, 1, 512, 2048>, resnet50/btlnck_block See more
1,967,249	37,763,181	Momentum/update_resnet50/btlnck_block_3_0/bottleneck_3/conv2d/kernel/ApplyMomentum	ApplyMomentum	61	✖	✖	122	float32	resnet50/btlnck_block_3_0/bottleneck_3/conv2d/kernel: <1, 1, 512, 2048>, resnet50/btlnck_block See more
1,962,104	1,358,242	Momentum/update_resnet50/btlnck_block_3_2/bottleneck_1/conv2d/kernel/ApplyMomentum	ApplyMomentum	61	✖	✖	122	float32	resnet50/btlnck_block_3_2/bottleneck_1/conv2d/kernel: <1, 1, 2048, 512>, resnet50/btlnck_block

Showing 1 to 10 of 5,507 entries

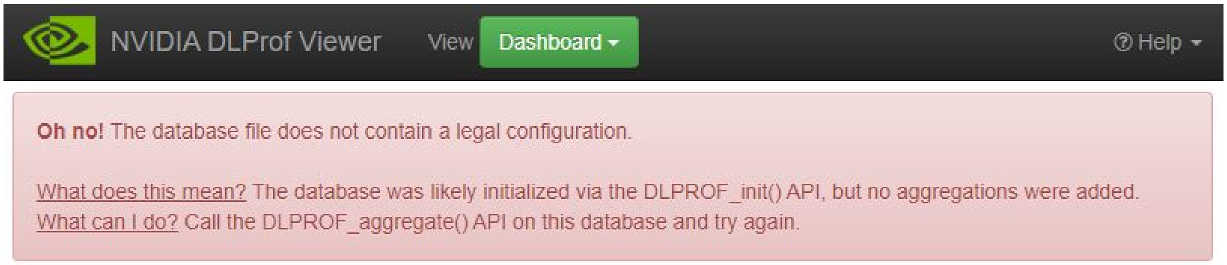
Previous 1 2 3 4 5 … 551 Next

Data Table Feature	Definition
Showing label	The label under the table (bottom left) will show a real-time count of rows in the table.
Search text box	Filter results by text search. Typing in this field will display only those rows that contain the text in the box. Adding or removing text in the Search box will update the "Showing…" label.
Column search	Typing into the text box under most column headers will display only those rows with the entered text. This is powerful since users can enter search criteria for multiple columns to zero-in on interesting rows. Columns with 'x' and 'check mark' are boolean fields. Users can enter "1" to show those rows with a 'check mark', and '0' to show only those rows with 'x'.
Sort toggle button	Clicking on a column header will sort the table. When clicked, all rows are sorted either ascending or descending. The initial sort on most numeral columns are descending.
Show Entries dropdown	This dropdown allows the user to display 10, 25, 50, or 100 rows. Changing the setting will update the "Showing…" label.
Pagination buttons	Previous, next, and page# navigation. Allows users to quickly page through large data sets. Uses 'Show Entries' setting and updates the "Showing…" label.
Export to buttons	Allows users to easily export the data in the datatable to well known formats. An optional profile name is displayed in the header of all exports when the `--profile_name` command-line argument is used in DLProf. Warning: A slight delay occurs when any of these buttons are clicked on large networks. Enable pop-ups in your browser to export to PDF.

Chapter 4. Troubleshooting FAQ

The database does not contain a legal configuration.

The DLProf Database cannot be found.

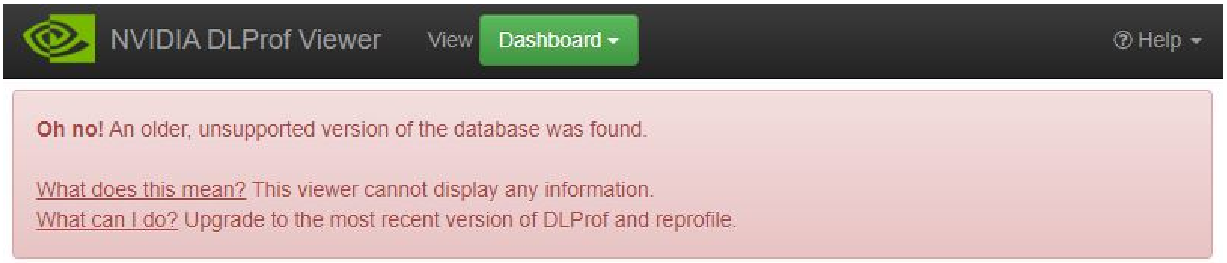


Oh no! The database file does not contain a legal configuration.

What does this mean? The database was likely initialized via the DLPROF_init() API, but no aggregations were added.
What can I do? Call the DLPROF_aggregate() API on this database and try again.

See Starting DLProf Viewer for more information.

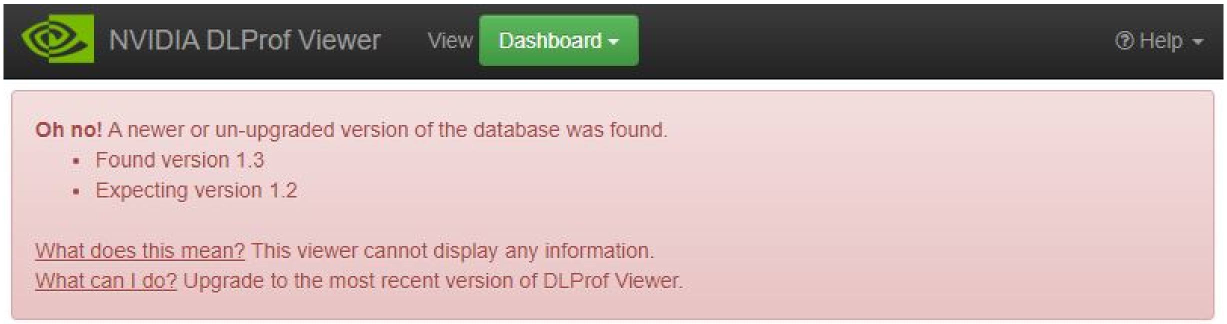
An older database was found.



Oh no! An older, unsupported version of the database was found.

What does this mean? This viewer cannot display any information.
What can I do? Upgrade to the most recent version of DLProf and reprofile.

A new database was found.

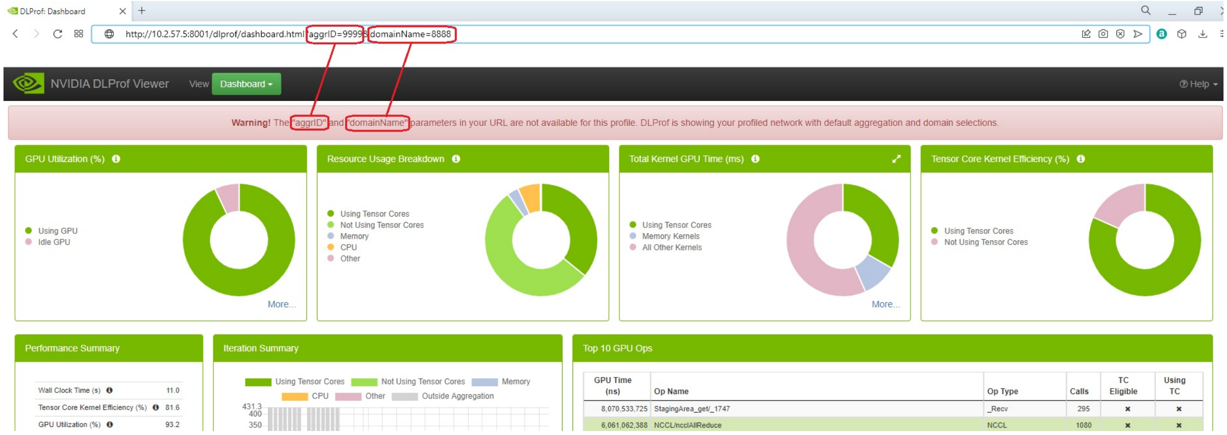


Oh no! A newer or un-upgraded version of the database was found.

- Found version 1.3
- Expecting version 1.2

What does this mean? This viewer cannot display any information.
What can I do? Upgrade to the most recent version of DLProf Viewer.

Bad parameters found in URL.



http://10.2.57.5:8001/dlprof/dashboard.html?aggrID=9999&domainName=8888

Warning! The "aggrID" and "domainName" parameters in your URL are not available for this profile. DLProf is showing your profiled network with default aggregation and domain selections.

Wall Clock Time (s)	11.0
Tensor Core Kernel Efficiency (%)	81.6
GPU Utilization (%)	93.2

GPU Time (ns)	Op Name	Op Type	Calls	TC Eligible	Using TC
8,070,533,725	StagingArea_get/_1747	_Recv	295	✖	✖
6,061,062,388	NCCL/ncclAllReduce	NCCL	1080	✖	✖

Notice

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

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

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

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

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

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

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

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

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

HDMI

OpenCL

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

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