NVIDIA TensorRT

API Reference | NVIDIA Docs

PR-08724-001_v8.4.0 Early Access (EA) | May 2022

Table of Contents

List of Tables

Chapter 1. Added, Deprecated, And Removed APIs

1.1. API Changes For TensorRT 8.4.0 EA

The following tables show which APIs were added, deprecated, and removed for the NVIDIA® TensorRT™ 8.4.0 EA release.

C++ changes

Table 1. New C++ APIs

New C++ APIs
setMemoryPoolLimit (IBuilderConfig::setMemoryPoolLimit)
getMemoryPoolLimit (IBuilderConfig::getMemoryPoolLimit)
MemoryPoolType
setMaxThreads (IBuilder::setMaxThreads, IRefitter::setMaxThreads, IRuntime::setMaxThreads)
getMaxThreads (IBuilder::getMaxThreads, IRefitter::getMaxThreads, IRuntime::getMaxThreads)
getBuilderPluginRegistry

Table 2. Deprecated C++ APIs

Deprecated C++ APIs
IFullyConnectedLayer
getMaxWorkspaceSize
setMaxWorkspaceSize

Python changes

Table 3. New Python APIs

New Python APIs
set_memory_pool_limit
get_memory_pool_limit
MemoryPoolType
max_threads property (Builder.max_threads, Refitter.max_threads, Runtime.max_threads)
get_builder_plugin_registry

Table 4. Deprecated Python APIs

Deprecated Python APIs
IFullyConnectedLayer
get_max_workspace_size
set_max_workspace_size

1.2. API Changes For TensorRT 8.2.0 EA

The following tables show which APIs were added, deprecated, and removed for the NVIDIA® TensorRT™ 8.2.0 EA release.

C++ changes

Table 5. New C++ APIs

New C++ APIs
IAssertionLayer
IConditionLayer
IEinsumLayer
IScatterLayer

Python changes

Table 6. New Python APIs

New Python APIs
IAssertionLayer
IConditionLayer
IEinsumLayer
IScatterLayer

1.3. API Changes For TensorRT 8.0.1

The following tables show which APIs were added, deprecated, and removed for the TensorRT 8.0.1 release.

C++ changes

Table 7. New C++ APIs

New C++ APIs
class IDequantizeLayer
class IQuantizeLayer
class ITimingCache
IBuilder::buildSerializedNetwork()
IBuilderConfig::getTimingCache()
IBuilderConfig::setTimingCache()
IGpuAllocator::reallocate()
INetworkDefinition::addDequantize()
INetworkDefinition::addQuantize()
INetworkDefinition::setWeightsName()
IPluginRegistry::deregisterCreator()
IRefitter::getMissingWeights()
IRefitter::getAllWeights()
IRefitter::setNamedWeights()
IResizeLayer::getCoordinateTransformation()

New C++ APIs
IResizeLayer::getNearestRounding()
IResizeLayer::getSelectorForSinglePixel()
IResizeLayer::setCoordinateTransformation()
IResizeLayer::setNearestRounding()
IResizeLayer::setSelectorForSinglePixel()
IScaleLayer::setChannelAxis()
enum ResizeCoordinateTransformation
enum ResizeMode
BuilderFlag::kSPARSE_WEIGHTS
TacticSource::kCUDNN
TensorFormat::kDLA_HWC4
TensorFormat::kDLA_LINEAR
TensorFormat::kHWC16

Table 8. Removed C++ APIs

Removed C++ APIs
Core Library
DimensionType
Dims::Type
class DimsCHW
class DimsNCHW
class IOutputDimensionFormula
class IPlugin
class IPluginFactory
class IPluginLayer
class IRNNLayer
IBuilder::getEngineCapability()
IBuilder::allowGPUFallback()
IBuilder::buildCudaEngine()
IBuilder::canRunOnDLA()
IBuilder::createNetwork()
IBuilder::getAverageFindIterations()

Removed C++ APIs
IBuilder::getDebugSync()
IBuilder::getDefaultDeviceType()
IBuilder::getDeviceType()
IBuilder::getDLACore()
IBuilder::getFp16Mode()
IBuilder::getHalf2Mode()
IBuilder::getInt8Mode()
IBuilder::getMaxWorkspaceSize()
IBuilder::getMinFindIterations()
IBuilder::getRefittable()
IBuilder::getStrictTypeConstraints()
IBuilder::isDeviceTypeSet()
IBuilder::reset()
IBuilder::resetDeviceType()
IBuilder::setAverageFindIterations()
IBuilder::setDebugSync()
IBuilder::setDefaultDeviceType()
IBuilder::setDeviceType()
IBuilder::setDLACore()
IBuilder::setEngineCapability()
IBuilder::setFp16Mode()
IBuilder::setHalf2Mode()
IBuilder::setInt8Calibrator()
IBuilder::setInt8Mode()
IBuilder::setMaxWorkspaceSize()
IBuilder::setMinFindIterations()
IBuilder::setRefittable()
IBuilder::setStrictTypeConstraints()
ICudaEngine::getWorkspaceSize()
IMatrixMultiplyLayer::getTranspose()
IMatrixMultiplyLayer::setTranspose()
INetworkDefinition::addMatrixMultiply()

Removed C++ APIs
`INetworkDefinition::addPlugin()`
`INetworkDefinition::addPluginExt()`
`INetworkDefinition::addRNN()`
`INetworkDefinition::getConvolutionOutputDimensionsFormula()`
`INetworkDefinition::getDeconvolutionOutputDimensionsFormula()`
`INetworkDefinition::getPoolingOutputDimensionsFormula()`
`INetworkDefinition::setConvolutionOutputDimensionsFormula()`
`INetworkDefinition::setDeconvolutionOutputDimensionsFormula()`
`INetworkDefinition::setPoolingOutputDimensionsFormula()`
`ITensor::getDynamicRange()`
`TensorFormat::kNHWC8`
`TensorFormat::NCHW`
`TensorFormat::kNC2HW2`
Caffe Parser
`class IPluginFactory`
`class IPluginFactoryExt`
`setPluginFactory()`
`setPluginFactoryExt()`
UFF Parser
`class IPluginFactory`
`class IPluginFactoryExt`
`setPluginFactory()`
`setPluginFactoryExt()`

Table 9. Removed Plugins

Removed Plugins
`class INvPlugin`
`createLReLUPlugin()`
`createClipPlugin()`
`PluginType`
`struct SoftmaxTree`

For plugins based on `IPluginV2DynamicExt` and `IPluginV2IOExt`, certain methods with legacy function signatures (derived from `IPluginV2` and `IPluginV2Ext` base classes) which were deprecated and marked for removal in TensorRT 8.0 will no longer be available. Plugins using these interface methods must stop using them or implement the versions with updated signatures, as applicable.

Table 10. Unsupported plugin methods removed in TensorRT 8.0

Removed Plugins
`IPluginV2DynamicExt::canBroadcastInputAcrossBatch()`
`IPluginV2DynamicExt::isOutputBroadcastAcrossBatch()`
`IPluginV2DynamicExt::getTensorRTVersion()`
`IPluginV2IOExt::configureWithFormat()`
`IPluginV2IOExt::getTensorRTVersion()`

Table 11. Updated versions for supported plugin methods

Removed Plugin	Replaced with
	`IPluginV2DynamicExt::configurePlugin()`
	`IPluginV2DynamicExt::enqueue()`
	`IPluginV2DynamicExt::getOutputDimensions()`
	`IPluginV2DynamicExt::getWorkspaceSize()`
	`IPluginV2IOExt::configurePlugin()`
`IPluginV2DynamicExt::supportsFormat()`	`IPluginV2DynamicExt::supportsFormatCombination()`
`IPluginV2IOExt::supportsFormat()`	`IPluginV2IOExt::supportsFormatCombination()`

Python changes

Table 12. New Python APIs

New Python APIs
`class IDequantizeLayer`
`class IQuantizeLayer`
`class ITimingCache`
`Builder.build_serialized_network()`
`IBuilderConfig.get_timing_cache()`

New Python APIs
IBuilderConfig.set_timing_cache()
IGpuAllocator.reallocate()
INetworkDefinition.add_dequantize()
INetworkDefinition.add_quantize()
INetworkDefinition.set_weights_name()
IPluginRegistry.deregister_creator()
Refitter.get_all_weights()
Refitter.get_missing_weights()
Refitter::set_named_weights()
IResizeLayer.coordinate_transformation
IResizeLayer.nearest_rounding
IResizeLayer.selector_for_single_pixel
IScaleLayer.channel_axis
enum ResizeCoordinateTransformationDoc
enum ResizeMode
BuilderFlag.SPARSE_WEIGHTS
TacticSource.CUDNN
TensorFormat.DLA_HWC4
TensorFormat.DLA_LINEAR
TensorFormat.HWC16

Table 13. Removed Python APIs

Removed Python APIs
Core Library
class DimsCHW
class DimsNCHW
class IPlugin
class IPluginFactory
class IPluginLayer
class IRNNLayer
Builder.build_cuda_engine()
Builder.average_find_iterations

Removed Python APIs
Builder.debug_sync
Builder.fp16_mode
IBuilder.int8_mode
Builder.max_workspace_size
Builder.min_find_iterations
Builder.refittable
Builder.strict_type_constraints
ICudaEngine.max_workspace_size
IMatrixMultiplyLayer.transpose0
INetworkDefinition.add_matrix_multiply_deprecated()
INetworkDefinition.add_plugin()
INetworkDefinition.add_plugin_ext()
INetworkDefinition.add_rnn()
INetworkDefinition.convolution_output_dimensions_formula
INetworkDefinition.deconvolution_output_dimensions_formula
INetworkDefinition.pooling_output_dimensions_formula
ITensor.get_dynamic_range()
Dims.get_type()
TensorFormat.HWC8
TensorFormat.NCHW
TensorFormat.NCHW2
Caffe Parser
class IPluginFactory
class IPluginFactoryExt
CaffeParser.plugin_factory
CaffeParser.plugin_factory_ext
UFF Parser
class IPluginFactory
class IPluginFactoryExt
UffParser.plugin_factory
UffParser.plugin_factory_ext

Deprecated

For our deprecation policy, refer to the TensorRT Deprecation Policy section in the *TensorRT Developer Guide*.

Table 14. Deprecated APIs

Deprecated APIs	Replaced with
`nvinfer1::IResizeLayer::setAlignCorners`	`nvinfer1::IResizeLayer::setCoordinateTransformation`
`nvinfer1::IResizeLayer::getAlignCorners`	`nvinfer1::IResizeLayer::setSelectorForSinglePixel` `nvinfer1::IResizeLayer::setNearestRounding`

Chapter 2. C++ API

The NVIDIA® TensorRT™ C++ API allows developers to import, calibrate, generate and deploy networks using C++. Networks can be imported directly from ONNX. They may also be created programmatically by instantiating individual layers and setting parameters and weights directly.

Within the core C++ API in `NvInfer.h`, the following APIs are included:

- Builder API
- Execution API
- Network Definition API
- ONNX Parser API
- Plugin API

To view this API, see TensorRT C++ API.

For more information about the C++ API, including sample code, see *NVIDIA TensorRT Developer Guide*.

Chapter 3. Python API

The NVIDIA® TensorRT™ Python API enables developers in Python based development environments and those looking to experiment with TensorRT to easily parse models (for example, from ONNX) and generate and run PLAN files.

To view this API, see TensorRT Python API.

For more information about the Python API, including sample code, see TensorRT Developer Guide.

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.

ARM

ARM, AMBA and ARM Powered are registered trademarks of ARM Limited. Cortex, MPCore and Mali are trademarks of ARM Limited. "ARM" is used to represent ARM Holdings plc; its operating company ARM Limited; and the regional subsidiaries ARM Inc.; ARM KK; ARM Korea Limited.; ARM Taiwan Limited; ARM France SAS; ARM Consulting (Shanghai) Co. Ltd.; ARM Germany GmbH; ARM Embedded Technologies Pvt. Ltd.; ARM Norway, AS and ARM Sweden AB.

HDMI

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

Blackberry/QNX

Trademarks, including but not limited to BLACKBERRY, EMBLEM Design, QNX, AVIAGE, MOMENTICS, NEUTRINO and QNX CAR are the trademarks or registered trademarks of BlackBerry Limited, used under license, and the exclusive rights to such trademarks are expressly reserved.

Google

Android, Android TV, Google Play and the Google Play logo are trademarks of Google, Inc.

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

Trademarks

NVIDIA, the NVIDIA logo, and BlueField, CUDA, DALI, DRIVE, JetPack, Jetson AGX Xavier, Jetson Nano, Kepler, Maxwell, NGC, Nsight, Orin, Pascal, Quadro, Tegra, TensorRT, Triton, Turing and Volta are trademarks and/or registered trademarks of NVIDIA Corporation in the United States and other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

Copyright

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