



# NVIDIA TensorRT

Support Matrix | NVIDIA Docs

# Table of Contents

Chapter 1. Features for Platforms and Software.....	1
Chapter 2. Layers and Features.....	3
Chapter 3. Layers and Precision.....	8
Chapter 4. Hardware and Precision.....	11
Chapter 5. Layers for Flow-Control Constructs.....	13
Chapter 6. Compute Capability Per Platform.....	16
Chapter 7. Software Versions Per Platform.....	17
Chapter 8. ONNX Operator Support.....	18

# Chapter 1. Features for Platforms and Software

This section lists the supported NVIDIA® TensorRT™ features based on which platform and software.

Table 1. List of Supported Features per Platform

	Linux x86-64	Windows x64	Linux ppc64le	Linux AArch64
	8.4.x	8.4.x	8.0.x	8.4.x
<a href="#">Supported NVIDIA CUDA® versions</a>	<a href="#">11.7 update 1</a> <sup>1</sup> <a href="#">11.6 update 2</a> <a href="#">11.5 update 2</a> <sup>2</sup> <a href="#">11.4 update 4</a> <a href="#">11.3 update 1</a> <sup>3</sup> <a href="#">11.2 update 2</a> <sup>4</sup> <a href="#">11.1 update 1</a> <sup>5</sup>	<a href="#">11.7 update 1</a> <sup>7</sup> <a href="#">11.6 update 2</a> <a href="#">11.5 update 2</a> <sup>8</sup> <a href="#">11.4 update 4</a> <sup>9</sup> <a href="#">11.3 update 1</a> <sup>10</sup> <a href="#">11.2 update 2</a> <sup>11</sup> <a href="#">11.1 update 1</a> <sup>12</sup>	<a href="#">11.3 update 1</a>	<a href="#">11.6 update 2</a> <a href="#">11.4</a>

<sup>1</sup> CUDA 11.7 added a feature called Lazy loading, however, this feature is not supported by TensorRT 8.4 because the CUDA 11.x binaries were built with CUDA Toolkit 11.6.

<sup>7</sup> CUDA 11.7 added a feature called Lazy loading, however, this feature is not supported by TensorRT 8.4 because the CUDA 11.x binaries were built with CUDA Toolkit 11.6.

<sup>2</sup> These CUDA versions are supported using a single build, built with CUDA toolkit 11.6. It is compatible with all CUDA 11.x versions and only requires driver 450.x.

<sup>8</sup> These CUDA versions are supported using a single build, built with CUDA toolkit 11.6. It is compatible with all CUDA 11.x versions and only requires driver 450.x.

<sup>9</sup> These CUDA versions are supported using a single build, built with CUDA toolkit 11.6. It is compatible with all CUDA 11.x versions and only requires driver 450.x.

<sup>3</sup> These CUDA versions are supported using a single build, built with CUDA toolkit 11.6. It is compatible with all CUDA 11.x versions and only requires driver 450.x.


<sup>10</sup> These CUDA versions are supported using a single build, built with CUDA toolkit 11.6. It is compatible with all CUDA 11.x versions and only requires driver 450.x.

<sup>4</sup> These CUDA versions are supported using a single build, built with CUDA toolkit 11.6. It is compatible with all CUDA 11.x versions and only requires driver 450.x.

<sup>11</sup> These CUDA versions are supported using a single build, built with CUDA toolkit 11.6. It is compatible with all CUDA 11.x versions and only requires driver 450.x.

<sup>5</sup> These CUDA versions are supported using a single build, built with CUDA toolkit 11.6. It is compatible with all CUDA 11.x versions and only requires driver 450.x.

	Linux x86-64	Windows x64	Linux ppc64le	Linux AArch64
	8.4.x	8.4.x	8.0.x	8.4.x
	<a href="#">11.0 update 1</a> <sup>6</sup>	<a href="#">11.0 update 1</a> <sup>13</sup>		
	<a href="#">10.2</a>	<a href="#">10.2</a>		
<a href="#">Supported cuBLAS versions</a>	11.10.3.66 11.9.2.110 11.7.4.6 11.6.5.2 11.5.1.109 11.4.1.1043 11.3.0.106 11.2.0.252 10.2.3.254	11.10.3.66 11.9.2.110 11.7.4.6 11.6.5.2 11.5.1.109 11.4.1.1043 11.3.0.106 11.2.0.252 10.2.3.254	11.5.1.109	11.9.2.110 11.6.5.2
<a href="#">Supported cuDNN versions</a>	<a href="#">cuDNN 8.4.1</a>	<a href="#">cuDNN 8.4.1</a>	<a href="#">cuDNN 8.2.1</a>	<a href="#">cuDNN 8.3.3</a>
TensorRT Python API	Yes	Yes	Yes	Yes
NvUffParser	Yes	Yes	Yes	Yes
NvOnnxParser	Yes	Yes	Yes	Yes
<a href="#">Loops</a>	Yes	Yes	Yes	Yes


 **Note:**

- ▶ Serialized engines are not portable across platforms or TensorRT versions.
- ▶ Refer to the minimum compatible driver versions in the [NVIDIA CUDA Release Notes](#) for specific [NVIDIA Driver](#) versions.

<sup>6</sup> These CUDA versions are supported using a single build, built with CUDA toolkit 11.6. It is compatible with all CUDA 11.x versions and only requires driver 450.x.  
<sup>13</sup> These CUDA versions are supported using a single build, built with CUDA toolkit 11.6. It is compatible with all CUDA 11.x versions and only requires driver 450.x.

# Chapter 2. Layers and Features

The section lists the supported TensorRT layers and each of the features.

 **Note:**

- ▶ Supports broadcast indicates support for broadcast in this layer. This layer allows its two input tensors to be of dimensions [1, 5, 4, 3] and [1, 5, 1, 1], and its output is [1, 5, 4, 3]. The second input tensor has been broadcast in the innermost two dimensions.
- ▶ Supports broadcast across batch indicates support for broadcast across the batch dimension. “NA” in this column means it is not allowed in networks with an implicit batch dimension.

Table 2. List of Supported Features per TensorRT Layer

Layer	Dimensions of input tensor	Dimensions of output tensor	Does the operation apply to only the innermost 3 dimensions?	Supports broadcast	Supports broadcast across batch
<a href="#">IActivationLayer</a>	0-7 dimensions	0-7 dimensions	No	No	No
<a href="#">IAssertionLayer</a>	0-1 dimensions	No output	No	No	No
<a href="#">IConcatenationLayer</a>	1-7 dimensions	1-7 dimensions	No	No	No
<a href="#">IConditionLayer</a>	Zero	No output	No	No	No
<a href="#">IConstantLayer</a>	Has no inputs	0-7 dimensions	No	No	Always
<a href="#">IConvolutionLayer &gt; 2D Convolution</a>	Three or more dimensions	Three or more dimensions	Yes	No	No

Layer	Dimensions of input tensor	Dimensions of output tensor	Does the operation apply to only the innermost 3 dimensions?	Supports broadcast	Supports broadcast across batch
<a href="#">IConvolutionLayer &gt; 3D Convolution</a>	Four or more dimensions	Four or more dimensions	No	No	No
<a href="#">IDeconvolutionLayer &gt; 2D Deconvolution</a>	Three or more dimensions	Three or more dimensions	Yes	No	No
<a href="#">IDeconvolutionLayer &gt; 3D Deconvolution</a>	Four or more dimensions	Four or more dimensions	No	No	No
<a href="#">IDequantizeLayer</a>	Two or more dimensions	Two or more dimensions	Yes	No	No
<a href="#">IEinsumLayer</a>	0-7 dimensions	0-7 dimensions	No	No	Yes
<a href="#">IElementWiseLayer</a>	0-7 dimensions	0-7 dimensions	No	Yes	Yes
<a href="#">IFillLayer</a>	One dimension	0-7 dimensions	No	Not Applicable	Not Applicable
<a href="#">IFullyConnectedLayer</a>	Three or more dimensions	Three or more dimensions	Yes	No	No
<a href="#">IGatherLayer</a>	<ul style="list-style-type: none"> <li>▶ Input1: 1-7 dimensions</li> <li>▶ Input2: 0-7 dimensions</li> </ul>	0-7 dimensions	No	No	Yes
<a href="#">IIdentityLayer</a>	0-7 dimensions	0-7 dimensions	No	No	No
<a href="#">IIfConditionalOutputLayer</a>	0-7 dimensions	0-7 dimensions	No	No	No
<a href="#">IIfConditionalInputLayer</a>	0-7 dimensions	0-7 dimensions	No	No	No

Layer	Dimensions of input tensor	Dimensions of output tensor	Does the operation apply to only the innermost 3 dimensions?	Supports broadcast	Supports broadcast across batch
<a href="#">IIteratorLayer</a>	1-7 dimensions	0-6 dimensions	No	No	Not Applicable
<a href="#">ILoopOutputLayer</a>	0-7 dimensions	0-7 dimensions	No	No	Not Applicable
<a href="#">ILRNLayer</a>	Three or more dimensions	Three or more dimensions	Yes	No	No
<a href="#">IMatrixMultiplyLayer</a>	Two or more dimensions	Two or more dimensions	No	Yes	Yes
<a href="#">IPaddingLayer</a>	Three or more dimensions	Three or more dimensions	Yes	No	No
<a href="#">IParametricReLULayer</a>	1-7 dimensions	1-7 dimensions	No	No	No
<a href="#">IPluginV2Layer</a>	User defined	User defined	User defined	User defined	User defined
<a href="#">IPoolingLayer &gt; 2D Pooling</a>	Three or more dimensions	Three or more dimensions	Yes	Yes	Yes
<a href="#">IPoolingLayer &gt; 3D Pooling</a>	Four or more dimensions	Four or more dimensions	No	Yes	Yes
<a href="#">IQuantizeLayer</a>	Two or more dimensions	Two or more dimensions	Yes	No	No
<a href="#">IRaggedSoftMaxLayer</a>	<ul style="list-style-type: none"> <li>▶ Input: Two dimensions</li> <li>▶ Bounds: Two dimensions</li> </ul>	Two or more dimensions	No	No	Yes
<a href="#">IRecurrenceLayer</a>	0-7 dimensions	0-7 dimensions	No	No	Not Applicable
<a href="#">IReduceLayer</a>	1-7 dimensions	0-7 dimensions	No	No	No
<a href="#">IResizeLayer</a>	1-7 dimensions	1-7 dimensions	No	No	No

Layer	Dimensions of input tensor	Dimensions of output tensor	Does the operation apply to only the innermost 3 dimensions?	Supports broadcast	Supports broadcast across batch
<a href="#">IRNNv2Layer</a>	<ul style="list-style-type: none"> <li>▶ Data/Hidden/Cell: Two or more dimensions</li> <li>▶ SeqLen: Zero or more dimensions</li> </ul>	Data/Hidden/Cell: Two or more dimensions	No	No	No
<a href="#">IScaleLayer</a>	Three or more dimensions	Three or more dimensions	Yes	No	No
<a href="#">IScatterLayer</a>	0-7 dimensions	0-7 dimensions	No	No	No
<a href="#">ISelectLayer</a>	0-7 dimensions	0-7 dimensions	No	Yes	Not Applicable
<a href="#">IShapeLayer</a>	One or more dimensions	One dimension	No	No	Not Applicable
<a href="#">IShuffleLayer</a>	0-7 dimensions	0-7 dimensions	No	No	No
<a href="#">ISliceLayer</a>	1-7 dimensions	1-7 dimensions	No	No	Yes
<a href="#">ISoftMaxLayer</a>	1-7 dimensions	1-7 dimensions	No	No	Yes
<a href="#">ITopKLayer</a>	1-7 dimensions	<ul style="list-style-type: none"> <li>▶ Output1: 1-7 dimensions</li> <li>▶ Output2: 1-7 dimensions</li> </ul>	Yes	No	Yes
<a href="#">ITripLimitLayer</a>	Zero dimensions	Has no outputs	No	No	Not Applicable



<b>Layer</b>	<b>Dimensions of input tensor</b>	<b>Dimensions of output tensor</b>	<b>Does the operation apply to only the innermost 3 dimensions?</b>	<b>Supports broadcast</b>	<b>Supports broadcast across batch</b>
<a href="#">UnaryLayer</a>	1-7 dimensions	1-7 dimensions	No	No	No

For more information about each of the TensorRT layers, see [TensorRT Layers](#).

# Chapter 3. Layers and Precision

The section lists the TensorRT layers and the precision modes that each layer supports. It also lists the ability of the layer to run on Deep Learning Accelerator (DLA).

For more information about additional constraints, see [DLA Supported Layers](#).

For more information about each of the TensorRT layers, see [TensorRT Layers](#). To view a list of the specific attributes that are supported by each layer, refer to the [NVIDIA TensorRT API Reference](#) documentation.

Table 3. List of Supported Precision Modes per TensorRT Layer

Layer	FP32	FP16	INT8	INT32	Bool	DLA FP16	DLA INT8
<a href="#">IActivationLayer</a>	Yes	Yes	Yes	No	No	Yes <sup>14</sup>	Yes <sup>15</sup>
<a href="#">IAssertionLayer</a>	No	No	No	No	Yes	No	No
<a href="#">IConcatenationLayer</a>	Yes	Yes	Yes	Yes	No	Yes <sup>16</sup>	Yes <sup>5</sup>
<a href="#">IConditionLayer</a>	No	No	No	No	Yes	No	No
<a href="#">IConstantLayer</a>	Yes	Yes	Yes	Yes	No	No	No
<a href="#">IConvolutionLayer &gt; 2D Convolution</a>	Yes	Yes	Yes	No	No	Yes	Yes
<a href="#">IConvolutionLayer &gt; 3D Convolution</a>	Yes	Yes	Yes	No	No	No	No
<a href="#">IDeconvolutionLayer &gt; 2D Deconvolution</a>	Yes	Yes	Yes	No	No	Yes	Yes <sup>17</sup>

<sup>14</sup> Partial support. Yes for ReLU, Clipped ReLU, Leaky ReLU, Sigmoid, and TanH activation types only.

<sup>15</sup> Partial support. Yes for ReLU, Clipped ReLU, Leaky ReLU, Sigmoid, and TanH activation types only.

<sup>16</sup> Partial support. Yes for concatenation across c dimension only.

<sup>17</sup> Partial support. Yes for ungrouped deconvolutions and No for grouped.

Layer	FP32	FP16	INT8	INT32	Bool	DLA FP16	DLA INT8
<a href="#">IDeconvolutionLayer &gt; 3D Deconvolution</a>	Yes	Yes	No	No	No	No	No
<a href="#">IDequantizeLayer</a>	No	No	Yes	No	No	No	No
<a href="#">IEinsumLayer</a>	Yes	Yes	No	No	No	No	No
<a href="#">IElementWiseLayer</a>	Yes	Yes	Yes	Yes	Yes	Yes <sup>18</sup>	Yes <sup>19</sup>
<a href="#">IFillLayer</a>	Yes	No	No	Yes	No	No	No
<a href="#">IFullyConnectedLayer</a>	Yes	Yes	Yes	No	No	Yes	Yes
<a href="#">IGatherLayer</a>	Yes	Yes	No	Yes	No	No	No
<a href="#">IIdentityLayer</a>	Yes	Yes	Yes	Yes	No	No	No
<a href="#">IIfConditionalOutputLayer</a>	Yes	Yes	No	Yes	Yes	No	No
<a href="#">IIfConditionalInputLayer</a>	Yes	Yes	No	Yes	Yes	No	No
<a href="#">IIteratorLayer</a>	Yes	Yes	No	Yes	No	No	No
<a href="#">ILoopOutputLayer</a>	Yes	Yes	No	Yes	No	No	No
<a href="#">ILRNLayer</a>	Yes	Yes	Yes	No	No	Yes	No
<a href="#">IMatrixMultiplyLayer</a>	Yes	Yes	Yes <sup>20</sup>	No	No	No	No
<a href="#">IPaddingLayer</a>	Yes	Yes	Yes	No	No	No	No
<a href="#">IParametricReLULayer</a>	Yes	Yes	Yes	No	No	Yes	Yes
<a href="#">IPluginV2Layer</a>	Yes	Yes	Yes	No	No	No	No
<a href="#">IPoolingLayer &gt; 2D Pooling</a>	Yes	Yes	Yes	No	No	Yes <sup>21</sup>	Yes <sup>9</sup>
<a href="#">IPoolingLayer &gt; 3D Pooling</a>	Yes	Yes	No	No	No	No	No
<a href="#">IQuantizeLayer</a>	Yes	No	No	No	No	No	No
<a href="#">IRaggedSoftmaxLayer</a>	Yes	No	No	No	No	No	No
<a href="#">IRecurrenceLayer</a>	Yes	Yes	No	Yes	Yes	No	No
<a href="#">IReduceLayer</a>	Yes	Yes	Yes	Yes	No	No	No
<a href="#">IResizeLayer</a>	Yes	Yes	Yes	No	No	Yes	Yes

<sup>18</sup> Partial support. Yes for sum, sub, prod, min, and max elementwise operations only.

<sup>19</sup> Partial support. Yes for sum, sub, prod, min, and max elementwise operations only.

<sup>20</sup> Partial support. Yes for the case the second input is build-time constant and the first input is not transposed - either produced by a Shuffle layer or `opA == kTRANSPPOSE`.

<sup>21</sup> Partial support. Yes for max and average padding inclusive pooling type only.

Layer	FP32	FP16	INT8	INT32	Bool	DLA FP16	DLA INT8
<a href="#">IRNNv2Layer</a>	Yes	Yes	No	No	No	No	No
<a href="#">IScaleLayer</a>	Yes	Yes	Yes	No	No	Yes <sup>22</sup>	Yes <sup>10</sup>
<a href="#">IScatterLayer</a>	Yes	Yes	Yes	Yes	No	No	No
<a href="#">ISelectLayer</a>	Yes	Yes	No	Yes	Yes	No	No
<a href="#">IShapeLayer</a>	Yes <sup>23</sup>	Yes	Yes	Yes	Yes	No	No
<a href="#">IShuffleLayer</a>	Yes	Yes	Yes	Yes	No	Yes <sup>24</sup>	Yes <sup>25</sup>
<a href="#">ISliceLayer</a>	Yes	Yes	No <sup>26</sup>	Yes	No	Yes	No
<a href="#">ISoftMaxLayer</a>	Yes	Yes	No	No	No	Yes	No
<a href="#">ITopKLayer</a>	Yes	Yes	No	No	No	No	No
<a href="#">ITripLimitLayer</a>	Yes	Yes	No	Yes	Yes	No	No
<a href="#">IUnaryLayer</a>	Yes <sup>27</sup>	Yes	Yes	Yes	Yes	No	No



Note: DLA with FP16/INT8 precision with some restrictions on layer parameters.

<sup>22</sup> Partial support. DLA does not support power on the scale layer.

<sup>23</sup> Output is always INT32.

<sup>24</sup> Partial support in TensorRT 8.4.12 only.

<sup>25</sup> Partial support in TensorRT 8.4.12 only.

<sup>26</sup> Partial support. Yes for unstrided Slice and No for strided.

<sup>27</sup> Datatype support is limited to the type of unary operation used. Refer to the [Unary Layer](#) for more information.

# Chapter 4. Hardware and Precision

The following table lists NVIDIA hardware and which precision modes that each hardware supports. TensorRT supports all NVIDIA hardware with capability SM 5.0 or higher. It also lists the availability of DLA on this hardware. Refer to the following tables for the specifics.



Note: Support for CUDA compute capability version 3.0 has been removed. Support for CUDA compute capability versions below 5.0 may be removed in a future release and is now deprecated.

Table 4. Supported Hardware

<a href="#">CUDA Compute Capability</a>	<b>Example Device</b>	<b>TF32</b>	<b>FP32</b>	<b>FP16</b>	<b>INT8</b>	<b>FP16 Tensor Cores</b>	<b>INT8 Tensor Cores</b>	<b>DLA</b>
8.7	NVIDIA DRIVE AGX Orin™	Yes	Yes	Yes	Yes	Yes	Yes	Yes
8.6	NVIDIA A10	Yes	Yes	Yes	Yes	Yes	Yes	No
8.0	NVIDIA A100/ GA100 GPU	Yes	Yes	Yes	Yes	Yes	Yes	No
7.5	NVIDIA T4	No	Yes	Yes	Yes	Yes	Yes	No
7.2	Jetson AGX Xavier	No	Yes	Yes	Yes	Yes	Yes	Yes
7.0	NVIDIA V100	No	Yes	Yes	Yes	Yes	No	No

<a href="#">CUDA Compute Capability</a>	<b>Example Device</b>	<b>TF32</b>	<b>FP32</b>	<b>FP16</b>	<b>INT8</b>	<b>FP16 Tensor Cores</b>	<b>INT8 Tensor Cores</b>	<b>DLA</b>
6.1	NVIDIA P4	No	Yes	No	Yes	No	No	No
6.0	NVIDIA P100	No	Yes	Yes	No	No	No	No
5.2	NVIDIA M4	No	Yes	No	No	No	No	No
5.0	Quadro K2200	No	Yes	No	No	No	No	No

### Deprecated Hardware

Table 5. List of Supported Precision Mode per Hardware

<a href="#">CUDA Compute Capability</a>	<b>Example Device</b>	<b>FP32</b>	<b>FP16</b>	<b>INT8</b>	<b>FP16 Tensor Cores</b>	<b>INT8 Tensor Cores</b>	<b>DLA</b>
3.7	NVIDIA K80	Yes	No	No	No	No	No
3.5	NVIDIA K40	Yes	No	No	No	No	No

### Removed Hardware

Table 6. List of Supported Precision Mode per Hardware

<a href="#">CUDA Compute Capability</a>	<b>Example Device</b>	<b>FP32</b>	<b>FP16</b>	<b>INT8</b>	<b>FP16 Tensor Cores</b>	<b>INT8 Tensor Cores</b>	<b>DLA</b>
3.0	NVIDIA K10	Yes	No	No	No	No	No

---

# Chapter 5. Layers for Flow-Control Constructs

The following table lists the TensorRT layers that can be used as interior layers in TensorRT flow-control constructs.

Currently, TensorRT supports loop constructs (using `ILoopLayer`) and ternary conditional constructs (using `IIfConditionalLayer`). Interior layers are layers that include the body of a loop or one of the two branches of an if-conditional.

An `ILoopLayer` interior layer may contain other loops and if-conditionals. An `IIfConditionalLayer` branch may contain other if-conditionals and loops.

Flow-control constructs do not support INT8 calibration and interior-layers cannot employ implicit-quantization (INT8 is supported only in explicit-quantization mode).

Table 7. List of TensorRT Layers that are Supported as Interior Layers of Flow-control Constructs

Layer	Supported
<a href="#">IActivationLayer</a>	Yes, when the operation is one of: <code>kRELU</code> , <code>kSIGMOID</code> , <code>kTANH</code> , <code>kELU</code>
<a href="#">IAssertionLayer</a>	Yes
<a href="#">IConcatenationLayer</a>	Yes
<a href="#">IConditionLayer</a>	Yes (for nested conditionals)
<a href="#">IConstantLayer</a>	Yes
<a href="#">IConvolutionLayer &gt; 2D Convolution</a>	singleton channel and spatial dims, that are, the dimensions must be static or have a single value in each optimization profile
<a href="#">IConvolutionLayer &gt; 3D Convolution</a>	singleton channel and spatial dims
<a href="#">IDeconvolutionLayer &gt; 2D Deconvolution</a>	No
<a href="#">IDeconvolutionLayer &gt; 3D Deconvolution</a>	No
<a href="#">IDequantizeLayer</a>	No

Layer	Supported
<a href="#">IEinsumLayer</a>	Yes
<a href="#">IElementWiseLayer</a>	Yes
<a href="#">IFillLayer</a>	kRANDOM_UNIFORM only
<a href="#">IFullyConnectedLayer</a>	Yes
<a href="#">IGatherLayer</a>	Yes
<a href="#">IIdentityLayer</a>	Yes
<a href="#">IIfConditionalOutputLayer</a>	Yes (for nested conditionals)
<a href="#">IIfConditionalInputLayer</a>	Yes (for nested conditionals)
<a href="#">IIteratorLayer</a>	Yes (for nested loops)
<a href="#">ILoopOutputLayer</a>	Yes (for nested loops)
<a href="#">ILRNLayer</a>	No
<a href="#">IMatrixMultiplyLayer</a>	Yes
<a href="#">IPaddingLayer</a>	No
<a href="#">IParametricReluLayer</a>	No
<a href="#">IPluginV2Layer</a>	Yes
<a href="#">IPoolingLayer &gt; 2D Pooling</a>	No
<a href="#">IPoolingLayer &gt; 3D Pooling</a>	No
<a href="#">IQuantizeLayer</a>	No
<a href="#">IRaggedSoftMaxLayer</a>	No
<a href="#">IRecurrenceLayer</a>	Yes
<a href="#">IReduceLayer</a>	Yes
<a href="#">IResizeLayer</a>	No
<a href="#">IRNNv2Layer</a>	No
<a href="#">IScaleLayer</a>	Yes
<a href="#">IScatterLayer</a>	Yes
<a href="#">ISelectLayer</a>	Yes
<a href="#">IShapeLayer</a>	Yes
<a href="#">IShuffleLayer</a>	Yes
<a href="#">ISliceLayer</a>	Yes
<a href="#">ISoftMaxLayer</a>	Yes
<a href="#">ITopKLayer</a>	No
<a href="#">ITripLimitLayer</a>	Yes



Layer	Supported
<a href="#">UnaryLayer</a>	Yes, when the operation is one of: <code>kABS</code> , <code>kCEIL</code> , <code>kERF</code> , <code>kEXP</code> , <code>kFLOOR</code> , <code>kLOG</code> , <code>kNEG</code> , <code>kNOT</code> , <code>kRECIP</code> , <code>kROUND</code> , <code>kSIGN</code> , <code>kSQRT</code> , <code>kSIN</code> , <code>kCOS</code> , <code>kATAN</code>

---

# Chapter 6. Compute Capability Per Platform

The section lists the supported compute capability based on platform.

Table 8. Compute Capability per Platform

Platform	Compute capability
Linux x86-64	3.5, 3.7, 5.0, 5.2, 6.0, 6.1, 7.0, 7.5, 8.0 <sup>28</sup> , 8.6 <sup>29</sup>
Windows 10 x64	3.5, 3.7, 5.0, 5.2, 6.0, 6.1, 7.0, 7.5, 8.0 <sup>13</sup> , 8.6 <sup>14</sup>
CentOS 8.3 ppc64le	7.0, 7.5, 8.0, 8.6
Ubuntu 20.04 SBSA	7.0, 7.5, 8.0, 8.6
NVIDIA JetPack AArch64	5.3, 6.2, 7.2, 8.7

<sup>28</sup> Requires CUDA toolkit 11.0 or newer and a TensorRT CUDA 11.x build.

<sup>29</sup> Requires CUDA toolkit 11.1 or newer and a TensorRT CUDA 11.x build.

---

# Chapter 7. Software Versions Per Platform

The section lists the supported software versions based on platform.

Table 9. List of Supported Platforms per Software Version

Platform	Compiler version	Python version
Ubuntu 18.04 x86-64	<a href="#">gcc 8.3.1</a>	<a href="#">3.6</a>
Ubuntu 20.04 x86-64	<a href="#">gcc 8.3.1</a>	<a href="#">3.8</a>
CentOS 7.9 x86-64	<a href="#">gcc 8.3.1</a>	<a href="#">3.6</a>
CentOS 8.3 x86-64	<a href="#">gcc 8.3.1</a>	<a href="#">3.8</a>
SLES 15 x86-64	<a href="#">gcc 8.3.1</a>	Not Applicable
Windows 10 x64	<a href="#">MSVC 2017u8</a>	Not Applicable
CentOS 8.3 ppc64le	<a href="#">Clang 10.0.1</a>	<a href="#">3.8</a>
Ubuntu 20.04 SBSA	<a href="#">gcc 8.4.0</a>	<a href="#">3.8</a>
NVIDIA JetPack AArch64	<a href="#">gcc 9.3.0</a>	<a href="#">3.8</a>



Note: Python versions supported when using Debian or RPM packages. When using Python wheel files, versions 3.6, 3.7, 3.8, 3.9, and 3.10 are supported.

---

# Chapter 8. ONNX Operator Support

The ONNX operator support list for TensorRT can be found [here](#).

## 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 NON-INFRINGEMENT, 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

Copyright © 2020 BlackBerry Limited. All rights reserved.

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.

## Trademarks

NVIDIA, the NVIDIA logo, and BlueField, CUDA, DALI, DRIVE, Hopper, JetPack, Jetson AGX Xavier, Jetson Nano, 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

© 2018-2024 NVIDIA Corporation & affiliates. All rights reserved.

