INSTALLING TENSORRT

SWE-SWDOCTRT-001-INST_vTensorRT 6.0.1 | November 2019

Installation Guide

TABLE OF CONTENTS

Chapter 1. OVERVIEW

The core of NVIDIA TensorRT is a C++ library that facilitates high performance inference on NVIDIA graphics processing units (GPUs). TensorRT takes a trained network, which consists of a network definition and a set of trained parameters, and produces a highly optimized runtime engine which performs inference for that network.

TensorRT provides API's via C++ and Python that help to express deep learning models via the Network Definition API or load a pre-defined model via the parsers that allows TensorRT to optimize and run them on an NVIDIA GPU. TensorRT applies graph optimizations, layer fusion, among other optimizations, while also finding the fastest implementation of that model leveraging a diverse collection of highly optimized kernels. TensorRT also supplies a runtime that you can use to execute this network on all of NVIDIA's GPU's from the Kepler generation onwards.

TensorRT also includes optional high speed mixed precision capabilities introduced in the Tegra X1, and extended with the Pascal, Volta, and Turing architectures.

Chapter 2. GETTING STARTED

Ensure you are familiar with the following installation requirements and notes.

- The Windows zip package for TensorRT does not provide Python support. Python may be supported in the future.
- If you are using the TensorRT Python API and PyCUDA isn't already installed on your system, see Installing PyCUDA. If you encounter any issues with PyCUDA usage, you may need to recompile it yourself. For more information, see Installing PyCUDA on Linux.
- Ensure you are familiar with the Release Notes. The current version of the release notes can be found online at TensorRT Release Notes.
- Verify that you have the CUDA Toolkit installed; versions 9.0, 10.0, 10.1 update 2, and 10.2 are supported.
- The TensorFlow to TensorRT model export requires TensorFlow 1.14.0.
- The PyTorch examples have been tested with PyTorch 1.1.0, but may work with older versions.
- If the target system has both TensorRT and one or more training frameworks installed on it, the simplest strategy is to use the same version of cuDNN for the training frameworks as the one that TensorRT ships with. If this is not possible, or for some reason strongly undesirable, be careful to properly manage the side-by-side installation of cuDNN on the single system. In some cases, depending on the training framework being used, this may not be possible without patching the training framework sources.
- The **`libnvcaffe_parser.so`** library functionality from previous versions is included in **`libnvparsers.so`** since TensorRT 5.0. The installed symbolic link for **`libnvcaffe_parser.so`** is updated to point to the new **`libnvparsers.so`** library. The static library **`libnvcaffe_parser.a`** is also symbolically linked to **`libnvparsers_static.a`**.
- The installation instructions below assume you want the full TensorRT; both the C++ and TensorRT Python APIs. In some environments and use cases, you may not want

to install the Python functionality. In which case, simply don't install the Debian or RPM packages labeled Python or the **`whl`** files. None of the C++ API functionality depends on Python. You would need to install the UFF **`whl`** file if you want to export UFF files from TensorFlow models.

Chapter 3. DOWNLOADING TENSORRT

Ensure you are a member of the NVIDIA Developer Program. If not, follow the prompts to gain access.

1. Go to: https://developer.nvidia.com/tensorrt.
2. Click **Download Now**.
3. Select the version of TensorRT that you are interested in.
4. Select the check-box to agree to the license terms.
5. Click the package you want to install. Your download begins.

Chapter 4. INSTALLING TENSORRT

You can choose between the following installation options when installing TensorRT; Debian or RPM packages, a tar file, or a zip file.

The Debian and RPM installations automatically install any dependencies, however, it:

- requires **`sudo`** or root privileges to install
- provides no flexibility as to which location TensorRT is installed into
- requires that the CUDA Toolkit and cuDNN have also been installed using Debian or RPM packages.
- does not allow more than one minor version of TensorRT to be installed at the same time

The tar file provides more flexibility, such as installing multiple versions of TensorRT at the same time. However, you need to ensure that you have the necessary dependencies already installed and you must manage **`LD_LIBRARY_PATH`** yourself. For more information, see Tar File Installation.

The zip file is the only option currently for Windows. It does not support any other platforms besides Windows. Ensure that you have the necessary dependencies already installed. For more information, see Zip File Installation.

TensorRT versions: TensorRT is a product made up of separately versioned components. The version on the product conveys important information about the significance of new features while the library version conveys information about the compatibility or incompatibility of the API. The following table shows the versioning of the TensorRT components.

Table 1 Versioning of TensorRT components

Product or Component		Previously Released Version	Current Version	Version Description
TensorRT product		5.1.5	6.0.1	+1.0 when significant new capabilities are added. +0.1 when capabilities have been improved.
nvinfer libraries, headers, samples, and documentation.		5.1.5	6.0.1	+1.0 when the API or ABI changes in a non-compatible way. +0.1 when the API or ABI changes are backward compatible
UFF	**uff-converter-tf** Debian and RPM packages	5.1.5	6.0.1	+0.1 while we are developing the core functionality. Set to 1.0 when we have all base functionality in place.
	uff-*.whl file	0.6.3	0.6.5	
graphsurgeon	**graphsurgeon-tf** Debian and RPM packages	5.1.5	6.0.1	+0.1 while we are developing the core functionality. Set to 1.0 when we have all base functionality in place.
	graphsurgeon-*.whl file	0.4.1	0.4.1	
libnvinfer python packages[1]	▸ **python-libnvinfer** ▸ **python-libnvinfer-dev**	5.1.5	6.0.1	+1.0 when the API or ABI changes in a non-compatible way.

Product or Component		Previously Released Version	Current Version	Version Description
	▸ `python3-libnvinfer` ▸ `python3-libnvinfer-dev` Debian and RPM packages			+0.1 when the API or ABI changes are backward compatible.
	`tensorrt.whl` file	5.1.5	6.0.1	

4.1. Debian Installation

This section contains instructions for a developer installation and an app server installation.

This installation method is for new users or users who want the complete installation, including samples and documentation. For advanced users who are already familiar with TensorRT and want to get their application running quickly or to setup automation, follow the network repo installation instructions (see Using The NVIDIA Machine Learning Network Repo For Debian Installation).

Developer Installation: The following instructions sets up a full TensorRT development environment with samples, documentation and both the C++ and Python API.

Attention If only the C++ development environment is desired, you can modify the following instructions and simply not install the Python packages.

Before issuing the following commands, you'll need to replace `ubuntu1x04`, `cudax.x`, `trt6.x.x.x` and `yyyymmdd` with your specific OS version, CUDA version, TensorRT version and package date. The following commands are examples for `amd64`, however, the commands are identical for `ppc64el`.

1. Download the TensorRT local repo file that matches the Ubuntu version and CPU architecture that you are using.
2. Install TensorRT from the Debian local repo package.

```
$ sudo dpkg -i
nv-tensorrt-repo-ubuntu1x04-cudax.x-trt6.x.x.x-ga-yyyymmdd_1-1_amd64.deb
$ sudo apt-key add /var/nv-tensorrt-repo-cudax.x-trt6.x.x.x-ga-
yyyymmdd/7fa2af80.pub
```

```
$ sudo apt-get update
$ sudo apt-get install tensorrt
```

If using Python 2.7:

```
$ sudo apt-get install python-libnvinfer-dev
```

The following additional packages will be installed:

```
  python-libnvinfer
```

If using Python 3.x:

```
$ sudo apt-get install python3-libnvinfer-dev
```

The following additional packages will be installed:

```
  python3-libnvinfer
```

If you plan to use TensorRT with TensorFlow:

```
$ sudo apt-get install uff-converter-tf
```

The **graphsurgeon-tf** package will also be installed with the above command.

3. Verify the installation.

```
$ dpkg -l | grep TensorRT
```

You should see something similar to the following:

```
ii  graphsurgeon-tf 6.0.1-1+cuda10.2 amd64 GraphSurgeon for TensorRT package
ii  libnvinfer-bin  6.0.1-1+cuda10.2 amd64 TensorRT binaries
ii  libnvinfer-dev  6.0.1-1+cuda10.2 amd64 TensorRT development libraries
 and headers
ii  libnvinfer-doc  6.0.1-1+cuda10.2 all TensorRT documentation
ii  libnvinfer-plugin-dev 6.0.1-1+cuda10.2 amd64 TensorRT plugin libraries
ii  libnvinfer-plugin6 6.0.1-1+cuda10.2 amd64 TensorRT plugin libraries
ii  libnvinfer-samples 6.0.1-1+cuda10.2 all TensorRT samples
ii  libnvinfer6  6.0.1-1+cuda10.2 amd64 TensorRT runtime libraries
ii  libnvonnxparsers-dev  6.0.1-1+cuda10.2 amd64 TensorRT ONNX libraries
ii  libnvonnxparsers6 6.0.1-1+cuda10.2 amd64 TensorRT ONNX libraries
ii  libnvparsers-dev 6.0.1-1+cuda10.2 amd64 TensorRT parsers libraries
ii  libnvparsers6 6.0.1-1+cuda10.2 amd64 TensorRT parsers libraries
ii  python-libnvinfer 6.0.1-1+cuda10.2 amd64 Python bindings for TensorRT
ii  python-libnvinfer-dev 6.0.1-1+cuda10.2 amd64 Python development package
 for TensorRT
ii  python3-libnvinfer 6.0.1-1+cuda10.2 amd64 Python 3 bindings for TensorRT
ii  python3-libnvinfer-dev 6.0.1-1+cuda10.2 amd64 Python 3 development
 package for TensorRT
ii  tensorrt  6.0.1.x-1+cuda10.2  amd64 Meta package of TensorRT
ii  uff-converter-tf 6.0.1-1+cuda10.2 amd64 UFF converter for TensorRT
 package
```

App Server Installation: When setting up servers which will host TensorRT powered applications, you can simply install any of the following:

- the **libnvinfer6** package (C++), plus any additional library packages you require, or

- the **python-libnvinfer** package (Python 2.7), or
- the **python3-libnvinfer** package (Python 3.x).

Issue the following commands if you want to run an application that was built with TensorRT using the Debian package, for example:

```
$ sudo dpkg -i
nv-tensorrt-repo-ubuntu1x04-cudax.x-trt6.x.x.x-ga-yyyymmdd_1-1_amd64.deb
$ sudo apt-key add /var/nv-tensorrt-repo-cudax.x-trt6.x.x.x-ga-
yyyymmdd/7fa2af80.pub

$ sudo apt-get update
$ sudo apt-get install libnvinfer6
```

4.1.1. Using The NVIDIA Machine Learning Network Repo For Debian Installation

When only the C++ libraries and headers or Python are required, you can install TensorRT from the NVIDIA Machine Learning network repository.

This installation method is for advanced users who are already familiar with TensorRT and want to get their application running quickly or to setup automation. New users or users who want the complete installation, including samples and documentation, should follow the local repo installation instructions (see Debian Installation).

It's suggested that you setup the NVIDIA CUDA network repository first before setting up the NVIDIA Machine Learning network repository to satisfy package dependencies. We provide some example commands below to accomplish this task. For more information, see the NVIDIA CUDA Installation Guide for Linux.

1. Install the NVIDIA CUDA network repository installation package.

   ```
   $ wget https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1x04/
   x86_64/cuda-repo-ubuntu1x04_x.y.z-1_amd64.deb
   $ sudo dpkg -i cuda-repo-*.deb
   ```

   Where:

   - OS version: **ubuntu1x04** is **1404**, **1604** or **1804**
   - CUDA version: **x.y.z** is **9.0.176**, **10.0.130**, **10.1.243** or **10.2.89**

2. Install the NVIDIA Machine Learning network repository installation package. Choose the **wget** command below that matches the Ubuntu version you are using.

   ```
   $ wget https://developer.download.nvidia.com/compute/machine-learning/repos/
   ubuntu1404/x86_64/nvidia-machine-learning-repo-ubuntu1404_4.0-2_amd64.deb
   $ wget https://developer.download.nvidia.com/compute/machine-learning/repos/
   ubuntu1604/x86_64/nvidia-machine-learning-repo-ubuntu1604_1.0.0-1_amd64.deb
   $ wget https://developer.download.nvidia.com/compute/machine-learning/repos/
   ubuntu1804/x86_64/nvidia-machine-learning-repo-ubuntu1804_1.0.0-1_amd64.deb

   $ sudo dpkg -i nvidia-machine-learning-repo-*.deb
   $ sudo apt-get update
   ```

3. Install the TensorRT package that fits your particular needs.
   a) For only running TensorRT C++ applications:

```
$ sudo apt-get install libnvinfer6 libnvonnxparsers6 libnvparsers6
 libnvinfer-plugin6
```

   b) For also building TensorRT C++ applications:

```
$ sudo apt-get install libnvinfer-dev libnvonnxparsers-dev
 libnvparsers-dev libnvinfer-plugin-dev
```

   c) For running TensorRT Python applications:

```
$ sudo apt-get install python-libnvinfer python3-libnvinfer
```

4. When using the NVIDIA Machine Learning network repository, Ubuntu will be default install TensorRT for the latest CUDA version. The following commands will install **libnvinfer6** for an older CUDA version and hold the **libnvinfer6** package at this version. Replace **6.x.x** with your version of TensorRT and **cuda10.x** with your CUDA version for your install.

```
$ version="6.x.x-1+cuda10.x"
$ sudo apt-get install libnvinfer6=${version} libnvonnxparsers6=${version}
 libnvparsers6=${version} libnvinfer-plugin6=${version} libnvinfer-dev=
${version} libnvonnxparsers-dev=${version} libnvparsers-dev=${version}
 libnvinfer-plugin-dev=${version} python-libnvinfer=${version} python3-
libnvinfer=${version}
```

```
$ sudo apt-mark hold libnvinfer6 libnvonnxparsers6 libnvparsers6 libnvinfer-
plugin6 libnvinfer-dev libnvonnxparsers-dev libnvparsers-dev libnvinfer-
plugin-dev python-libnvinfer python3-libnvinfer
```

   If you want to upgrade to the latest version of TensorRT or the latest version of CUDA, then you can unhold the **libnvinfer6** package using the following command.

```
$ sudo apt-mark unhold libnvinfer6 libnvonnxparsers6 libnvparsers6
 libnvinfer-plugin6 libnvinfer-dev libnvonnxparsers-dev libnvparsers-dev
 libnvinfer-plugin-dev python-libnvinfer python3-libnvinfer
```

   You may need to repeat these steps for **libcudnn7** to prevent cuDNN from being updated to the latest CUDA version. Refer to the TensorRT Release Notes for the specific version of cuDNN that was tested with your version of TensorRT. Example commands for downgrading and holding the cuDNN version can be found in Upgrading From TensorRT 5.x.x To TensorRT 6.x.x. See the cuDNN Installation Guide for additional information.

   If both the NVIDIA Machine Learning network repository and a TensorRT local repository are enabled at the same time you may observe package conflicts with either TensorRT or cuDNN. You will need to configure APT so that it prefers local packages over network packages. You can do this by creating a new file at **/etc/apt/preferences.d/local-repo** with the following lines:

```
Package: *
Pin: origin ""
```

```
Pin-Priority: 1001
```

This preference change will affect more than just TensorRT in the unlikely event that you have other repositories which are also not downloaded over HTTP(S). To revert APT to its original behavior simply remove the newly created file.

4.2. RPM Installation

This section contains instructions for installing TensorRT from an RPM package.

This installation method is for new users or users who want the complete installation, including samples and documentation. For advanced users who are already familiar with TensorRT and want to get their application running quickly or to setup automation, follow the network repo installation instructions (see Using The NVIDIA Machine Learning Network Repo For RPM Installation).

- Before issuing the following commands, you'll need to replace **cudax.x**, **trt6.x.x.x**, and **yyyymmdd** with your specific CUDA version, TensorRT version, and package date.
- If you want to install the Python 3 RPM packages, you must first enable the EPEL repository. For more information about enabling EPEL, see the RPM package instructions in the CUDA Quick Start Guide.
- The following example commands are for **x86_64**, but the commands should be identical for **ppc64le**.

1. Download the TensorRT local repo file that matches the RHEL/CentOS version and CPU architecture you are using.
2. Install TensorRT from the RPM local repo package.

```
$ sudo rpm -Uvh nv-tensorrt-repo-rhel7-cudax.x-trt6.x.x.x-ga-
yyyymmdd-1-1.x86_64.rpm
$ sudo yum clean expire-cache
```

The packages which can be installed are:

```
graphsurgeon-tf.x86_64
libnvinfer-bin.x86_64
libnvinfer-devel.x86_64
libnvinfer-doc.x86_64
libnvinfer-plugin-devel.x86_64
libnvinfer-plugin6.x86_64
libnvinfer-samples.x86_64
libnvinfer6.x86_64
libnvonnxparsers-devel.x86_64
libnvonnxparsers6.x86_64
libnvparsers-devel.x86_64
libnvparsers6.x86_64
python-libnvinfer.x86_64
python-libnvinfer-devel.x86_64
```

```
python3-libnvinfer.x86_64
python3-libnvinfer-devel.x86_64
tensorrt.x86_64
uff-converter-tf.x86_64
```

Then, install TensorRT:

```
$ sudo yum install tensorrt
```

If using Python 2.7:

```
$ sudo yum install python-libnvinfer-devel
```

The following additional packages will be installed:

```
python-libnvinfer
```

If using Python 3:

```
$ sudo yum install python3-libnvinfer-devel
```

The following additional packages will be installed:

```
python3-libnvinfer
```

and for the UFF converter (only required if you plan to use TensorRT with TensorFlow):

```
$ sudo yum install uff-converter-tf
```

3. Verify the installation.
   a) Run:

   ```
   $ rpm -qa | grep tensorrt
   ```

   You should see something similar to the following:

   ```
   tensorrt-6.0.1.x-1.cuda10.2.x86_64
   ```

   b) Run:

   ```
   $ rpm -qa | grep -e libnvinfer -e libnv.*parsers
   ```

   You should see something similar to the following:

   ```
   libnvinfer-doc-6.0.1-1.cuda10.2.x86_64
   libnvinfer-plugin6-6.0.1-1.cuda10.2.x86_64
   libnvinfer-devel-6.0.1-1.cuda10.2.x86_64
   libnvinfer-bin-6.0.1-1.cuda10.2.x86_64
   libnvinfer6-6.0.1-1.cuda10.2.x86_64
   libnvinfer-samples-6.0.1-1.cuda10.2.x86_64
   libnvinfer-plugin-devel-6.0.1-1.cuda10.2.x86_64
   libnvonnxparsers6-6.0.1-1.cuda10.2.x86_64
   libnvonnxparsers-devel-6.0.1-1.cuda10.2.x86_64
   libnvparsers6-6.0.1-1.cuda10.2.x86_64
   libnvparsers-devel-6.0.1-1.cuda10.2.x86_64
   python3-libnvinfer-6.0.1-1.cuda10.2.x86_64
   python3-libnvinfer-devel-6.0.1-1.cuda10.2.x86_64
   python-libnvinfer-6.0.1-1.cuda10.2.x86_64
   python-libnvinfer-devel-6.0.1-1.cuda10.2.x86_64
   ```

   c) Run:

   ```
   $ rpm -qa | grep graphsurgeon-tf
   ```

You should see something similar to the following:

```
graphsurgeon-tf-6.0.1-1.cuda10.2.x86_64
```

d) Run:

```
$ rpm -qa | grep uff-converter-tf
```

You should see something similar to the following:

```
uff-converter-tf-6.0.1-1.cuda10.2.x86_64
```

4.2.1. Using The NVIDIA Machine Learning Network Repo For RPM Installation

When only the C++ libraries and headers or Python are required, you can install TensorRT from the NVIDIA Machine Learning network repository.

This installation method is for advanced users who are already familiar with TensorRT and want to get their application running quickly or to setup automation. New users or users who want the complete installation, including samples and documentation, should follow the local repo installation instructions (see RPM Installation).

It's suggested that you setup the NVIDIA CUDA network repository first before setting up the NVIDIA Machine Learning network repository to satisfy package dependencies. We provide some example commands below to accomplish this task. For more information, see the NVIDIA CUDA Installation Guide for Linux.

1. Install the NVIDIA CUDA network repository installation package.

```
$ wget https://developer.download.nvidia.com/compute/cuda/repos/rhel7/
x86_64/cuda-repo-rhel7-x.y.z-1.x86_64.rpm
$ sudo rpm -Uvh cuda-repo-*.rpm
```

Where:

- CUDA version: `x.y.z` is `9.0.176`, `10.0.130`, `10.1.243` or `10.2.89`

2. Install the NVIDIA Machine Learning network repository installation package.

```
$ wget https://developer.download.nvidia.com/compute/machine-learning/repos/
rhel7/x86_64/nvidia-machine-learning-repo-rhel7-1.0.0-1.x86_64.rpm

$ sudo rpm -Uvh nvidia-machine-learning-repo-*.rpm
```

3. Install the TensorRT package that fits your particular needs.
   a) For only running TensorRT C++ applications:

```
$ sudo yum install libnvinfer6 libnvparsers6 libnvonnxparsers6
 libnvinfer-plugin6
```

   b) For also building TensorRT C++ applications:

```
$ sudo yum install libnvinfer-devel libnvparsers-devel libnvonnxparsers-
devel libnvinfer-plugin-devel
```

c) For running TensorRT Python applications:

```
$ sudo yum install python-libnvinfer python3-libnvinfer
```

4. When using the NVIDIA Machine Learning network repository, RHEL will by default install TensorRT for the latest CUDA version. The following commands will install **libnvinfer6** for an older CUDA version and hold the **libnvinfer6** package at this version. Replace **6.x.x** with your version of TensorRT and **cuda10.x** with your CUDA version for your install.

```
$ version="6.x.x-1.cuda10.x"
$ sudo yum downgrade libnvinfer6-${version} libnvparsers6-${version}
 libnvonnxparsers6-${version} libnvinfer-plugin6-${version} libnvinfer-
devel-${version} libnvparsers-devel-${version} libnvonnxparsers-devel-
${version} libnvinfer-plugin-devel-${version} python-libnvinfer-${version}
 python3-libnvinfer-${version}
```

```
$ sudo yum install yum-plugin-versionlock
$ sudo yum versionlock libnvinfer6 libnvparsers6 libnvonnxparsers6
 libnvinfer-plugin6 libnvinfer-devel libnvparsers-devel libnvonnxparsers-
devel libnvinfer-plugin-devel python-libnvinfer python3-libnvinfer
```

If you want to upgrade to the latest version of TensorRT or the latest version of CUDA, then you can unhold the **libnvinfer6** package using the following command.

```
$ sudo yum versionlock delete libnvinfer6 libnvparsers6 libnvonnxparsers6
 libnvinfer-plugin6 libnvinfer-devel libnvparsers-devel libnvonnxparsers-
devel libnvinfer-plugin-devel python-libnvinfer python3-libnvinfer
```

You may need to repeat these steps for **libcudnn7** to prevent cuDNN from being updated to the latest CUDA version. Refer to the TensorRT Release Notes for the specific version of cuDNN that was tested with your version of TensorRT. Example commands for downgrading and holding the cuDNN version can be found in Upgrading From TensorRT 5.x.x To TensorRT 6.x.x. See the cuDNN Installation Guide for additional information.

4.3. Tar File Installation

Before issuing the following commands, you'll need to replace **6.x.x.x** with your specific TensorRT version. The following commands are examples.

1. Install the following dependencies, if not already present:
   - CUDA 9.0, 10.0, 10.1 update 2, or 10.2
   - cuDNN 7.6.5
   - Python 2 or Python 3 (Optional)
2. Download the TensorRT tar file that matches the Linux distribution you are using.
3. Choose where you want to install TensorRT. This tar file will install everything into a subdirectory called **TensorRT-6.x.x.x**.
4. Unpack the tar file.

```
$ tar xzvf TensorRT-6.x.x.x.<os>.<arch>-gnu.cuda-x.x.cudnn7.x.tar.gz
```

Where:

- **6.x.x.x** is your TensorRT version
- **<os>** is:
  - **Ubuntu-14.04**
  - **Ubuntu-16.04**
  - **Ubuntu-18.04**
  - **CentOS-7.6**
- **<arch>** is **x86_64** or **ppc64le**
- **cuda-x.x** is CUDA version CUDA version **9.0**, **10.0**, **10.1**, or **10.2**
- **cudnn7.x** is cuDNN version **7.6**

This directory will have sub-directories like **lib**, **include**, **data**, etc…

```
$ ls TensorRT-6.x.x.x
bin  data  doc  graphsurgeon  include  lib  python  samples  targets
 TensorRT-Release-Notes.pdf  uff
```

5. Add the absolute path to the TensorRT **lib** directory to the environment variable **LD_LIBRARY_PATH**:

```
$ export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:<eg:TensorRT-6.x.x.x/lib>
```

6. Install the Python TensorRT wheel file.

```
$ cd TensorRT-6.x.x.x/python
```

If using Python 2.7:

```
$ sudo pip2 install tensorrt-6.x.x.x-cp27-none-linux_x86_64.whl
```

If using Python 3.x:

```
$ sudo pip3 install tensorrt-6.x.x.x-cp3x-none-linux_x86_64.whl
```

7. Install the Python UFF wheel file. This is only required if you plan to use TensorRT with TensorFlow.

```
$ cd TensorRT-6.x.x.x/uff
```

If using Python 2.7:

```
$ sudo pip2 install uff-0.6.5-py2.py3-none-any.whl
```

If using Python 3.x:

```
$ sudo pip3 install uff-0.6.5-py2.py3-none-any.whl
```

In either case:

```
$ which convert-to-uff
/usr/local/bin/convert-to-uff
```

8. Install the Python **graphsurgeon** wheel file.

```
$ cd TensorRT-6.x.x.x/graphsurgeon
```

If using Python 2.7:

```
$ sudo pip2 install graphsurgeon-0.4.1-py2.py3-none-any.whl
```

If using Python 3.x:

```
$ sudo pip3 install graphsurgeon-0.4.1-py2.py3-none-any.whl
```

9. Verify the installation:
   a) Ensure that the installed files are located in the correct directories. For example, run the **`tree -d`** command to check whether all supported installed files are in place in the **`lib`**, **`include`**, **`data`**, etc… directories.
   b) Build and run one of the shipped samples, for example, **`sampleMNIST`** in the installed directory. You should be able to compile and execute the sample without additional settings. For more information about sampleMNSIT, see the "Hello World" For TensorRT sample.
   c) The Python samples are in the **`samples/python`** directory.

4.4. Zip File Installation

Ensure that you have the following dependencies installed.

- CUDA 9.0, 10.0, or 10.1 update 2
- cuDNN 7.6.5

This section contains instructions for installing TensorRT from a zip package on Windows 10.

1. Download the TensorRT zip file that matches the Windows version you are using.
2. Choose where you want to install TensorRT. The zip file will install everything into a subdirectory called **`TensorRT-6.x.x.x`**. This new sub directory will be referred to as **`<installpath>`** in the steps below.
3. Unzip the **`TensorRT-6.x.x.x.Windows10.x86_64.cuda-x.x.cudnnx.x.zip`** file to the location that you chose. Replace:
   a) **`6.x.x.x`** with the TensorRT version
   b) **`cuda-x.x`** with the CUDA version, and
   c) **`cudnnx.x`** with the cuDNN version for your particular download.
4. Add the TensorRT library files to your system **`PATH`**. There are two ways to accomplish this task:
   a) Leave the DLL files where they were unzipped and add **`<installpath>/lib`** to your system **`PATH`**. You can add a new path to your system **`PATH`** using the steps below.

1. Press the **Windows** key and search for **environment variables** which should present you with the option **Edit the system environment variables** and click it.
2. Click **Environment Variables...** at the bottom of the window.
3. Under **System variables**, select **Path** and click **Edit...**.
4. Click either **New** or **Browse** to add a new item which contains **`<installpath>/lib`**.
5. Continue to click **OK** until all the newly opened windows are closed.
6. If your cuDNN libraries were not copied to the CUDA installation directory and instead left where they were unzipped, then repeat the above steps for the cuDNN **`bin`** directory.

b) Copy the DLL files from **`<installpath>/lib`** to your CUDA installation directory, for example **`C:\Program Files\NVIDIA GPU Computing Toolkit \CUDA\vX.Y\bin`**, where **`vX.Y`** is your CUDA version. The CUDA installer should have already added the CUDA path to your system **`PATH`**.

5. To verify that your installation is working you should open a Visual Studio Solution file from one of the samples, such as **`sampleMNIST`**, and confirm that you are able to build and run the sample.

   If you want to use TensorRT in your own project, ensure that the following is present in your Visual Studio Solution project properties:

   a) **`<installpath>/lib`** has been added to your **`PATH`** variable and is present under **VC++ Directories** > **Executable Directories.**

   b) **`<installpath>/include`** is present under **C/C++** > **General** > **AdditionalDirectories.**

   c) **`nvinfer.lib`** and any other **`LIB`** files that your project requires is present under **Linker** > **Input** > **Additional Dependencies.**

In order to build the included samples, you should have Visual Studio 2017 (https://visualstudio.microsoft.com/downloads/) installed. The community edition is sufficient to build the TensorRT samples.

6. Install the **`uff`** and **`graphsurgeon`** wheel packages. You must prepare the Python environment before installing **`uff`** and **`graphsurgeon`**.

   If using Python 2.7:

```
$python -m pip install <installpath>\graphsurgeon\graphsurgeon-0.4.1-
py2.py3-none-any.whl
$python -m pip install <installpath>\uff\uff-0.6.5-py2.py3-none-any.whl
```

   If using Python 3.x:

```
$python3 -m pip install <installpath>\graphsurgeon\graphsurgeon-0.4.1-
py2.py3-none-any.whl
$python3 -m pip install <installpath>\uff\uff-0.6.5-py2.py3-none-any.whl
```

4.5. Additional Installation Methods

Aside from installing TensorRT from the product package, you can also install TensorRT from the following locations:

TensorRT container

The TensorRT container provides an easy method for deploying TensorRT with all necessary dependencies already packaged in the container. For information about installing TensorRT via a container, see the TensorRT Container Release Notes.

JetPack

JetPack bundles all Jetson platform software, including TensorRT. Use it to flash your Jetson Developer Kit with the latest OS image, install NVIDIA SDKs, and jump-start your development environment. For information about installing TensorRT through JetPack, see the JetPack documentation.

For JetPack downloads, see Develop: Jetpack.

NVIDIA DriveWorks

With every release, TensorRT delivers features to make the DRIVE Development Platform an excellent computing platform for Autonomous Driving. For more information about installing TensorRT through DriveWorks, see the DriveWorks documentation.

For DriveWorks downloads, see NVIDIA Developer: Drive Downloads.

Chapter 5. UPGRADING TENSORRT

Upgrading TensorRT to the latest version is only supported when the currently installed TensorRT version is equal to or newer than the last two public releases. For example, TensorRT 6.0.x supports upgrading from TensorRT 5.0.x and TensorRT 5.1.x. If you want to upgrade from an unsupported version, then you should upgrade incrementally until you reach the latest version of TensorRT.

5.1. Ubuntu And Windows Users

5.1.1. Upgrading From TensorRT 5.x.x To TensorRT 6.x.x

These upgrade instructions are for Ubuntu and Windows users only. When upgrading from TensorRT 5.x.x to TensorRT 6.x.x, ensure you are familiar with the following.

Using a Debian file

- The Debian packages are designed to upgrade your development environment without removing any runtime components that other packages and programs might rely on. If you installed TensorRT 5.x.x via a Debian package and you upgrade to TensorRT 6.x.x, your documentation, samples, and headers will all be updated to the TensorRT 6.x.x content. After you have downloaded the new local repo, use **`apt-get`** to upgrade your system to the new version of TensorRT.

```
sudo dpkg -i nv-tensorrt-repo-ubuntu1x04-cudax.x-trt6.x.x.x-ga-
yyyymmdd_1-1_amd64.deb
sudo apt-get update
sudo apt-get install tensorrt libcudnn7
```

- If using Python 2.7:

```
sudo apt-get install python-libnvinfer-dev
```

- If using Python 3:

```
sudo apt-get install python3-libnvinfer-dev
```

- If you are using the **uff-converter** and/or **graphsurgeon,** then you should also upgrade those Debian packages to the latest versions.

```
sudo apt-get install uff-converter-tf graphsurgeon-tf
```

- After you upgrade, ensure you have a directory **/usr/src/tensorrt** and the corresponding version shown by the **dpkg -l tensorrt** command is **6.x.x.x**.
- The **libnvinfer5** package will not be removed until you use: **sudo apt-get autoremove**
- If installing a Debian package on a system where the previously installed version was from a tar file, note that the Debian package will not remove the previously installed files. Unless a side-by-side installation is desired, it would be best to remove the older version before installing the new version to avoid compiling against outdated libraries.
- If you are currently or were previously using the NVIDIA Machine Learning network repository, then it may conflict with the version of **libcudnn7** that is expected to be installed from the local repository for TensorRT. The following commands will change **libcudnn7** to version 7.6.x.x, which is supported and tested with TensorRT 6.x.x, and hold the **libcudnn7** package at this version. Replace **cuda10.x** with the appropriate CUDA version for your install.

```
sudo apt-get install libcudnn7=7.6.x.x-1+cuda10.x \
  libcudnn7-dev=7.6.x.x-1+cuda10.x
sudo apt-mark hold libcudnn7 libcudnn7-dev
```

Using a tar file

- If you are upgrading using the tar file installation method, then install TensorRT into a new location. Tar file installations can support multiple use cases including having a full installation of TensorRT 5.x.x with headers and documentation side-by-side with a full installation of TensorRT 6.x.x. If the intention is to have the new version of TensorRT replace the old version, then the old version should be removed once the new version is verified.
- If installing a tar file on a system where the previously installed version was from a Debian package, note that the tar file installation will not remove the previously installed packages. Unless a side-by-side installation is desired, it would be best to remove the previously installed **libnvinfer5,libnvinfer-dev,** and **libnvinfer-samples** packages to avoid confusion.

Using a zip file

- If you are upgrading using the zip file installation method, then install TensorRT into a new location. Zip file installations can support multiple use cases including having a full installation of TensorRT 5.x.x with headers and documentation side-by-side with a full installation of TensorRT 6.x.x. If the intention is to have the new version of TensorRT replace the old version, then the old version should be removed once the new version is verified.

- After unzipping the new version of TensorRT you will need to either update the **PATH** environment variable to point to the new install location or copy the DLL files to the location where you previously installed the TensorRT libraries. Refer to Zip File Installation for more information about setting the **PATH** environment variable.

5.2. RedHat And CentOS Users

5.2.1. Upgrading From TensorRT 5.x.x To TensorRT 6.x.x

These upgrade instructions are for Red Hat Enterprise Linux (RHEL) and CentOS users only. When upgrading from TensorRT 5.x.x to TensorRT 6.x.x, ensure you are familiar with the following.

Using an RPM file

- The RPM packages are designed to upgrade your development environment without removing any runtime components that other packages and programs might rely on. If you installed TensorRT 5.x.x via an RPM package and you want to upgrade to TensorRT 6.x.x, your documentation, samples, and headers will all be updated to the TensorRT 6.x.x content. After you have downloaded the new local repo, issue:

```
sudo rpm -Uvh nv-tensorrt-repo-rhel7-cudax.x-trt6.x.x.x-ga-
yyyymmdd-1-1.x86_64.rpm
sudo yum clean expire-cache
sudo yum install tensorrt libcudnn7
```

- If using Python 2.7:

```
sudo yum install python-libnvinfer-devel
```

- If using Python 3:

```
sudo yum install python3-libnvinfer-devel
```

- If using **uff-converter** and/or **graphsurgeon**:

```
sudo yum install uff-converter-tf graphsurgeon-tf
```

- After you upgrade, ensure you see the **/usr/src/tensorrt** directory and the corresponding version shown by the **rpm -qa tensorrt** command is **6.x.x.x**.
- If you are currently or were previously using the NVIDIA Machine Learning network repository, then it may conflict with the version of **libcudnn7** that is expected to be installed from the local repository for TensorRT. The following commands will change **libcudnn7** to version 7.6.x.x, which is supported and tested with TensorRT 6.x.x, and hold the **libcudnn7** package at this version. Replace **cuda10.x** with the appropriate CUDA version for your install.

```
sudo yum downgrade libcudnn7-7.6.x.x-1.cuda10.x \
  libcudnn7-devel-7.6.x.x-1.cuda10.x
sudo yum install yum-plugin-versionlock
```

```
sudo yum versionlock libcudnn7 libcudnn7-devel
```

Chapter 6. UNINSTALLING TENSORRT

To uninstall TensorRT using the untarred file, simply delete the tar files and reset **LD_LIBRARY_PATH** to its original value.

To uninstall TensorRT using the zip file, simply delete the unzipped files and remove the newly added path from the **PATH** environment variable.

To uninstall TensorRT using the Debian or RPM packages, follow these steps:

1. Uninstall **libnvinfer6** which was installed using the Debian or RPM packages.

   ```
   $ sudo apt-get purge "libnvinfer*"
   ```

   Or

   ```
   $ sudo yum erase "libnvinfer*"
   ```

2. Uninstall **uff-converter-tf** and **graphsurgeon-tf**, which were also installed using the Debian or RPM packages.

   ```
   $ sudo apt-get purge "graphsurgeon-tf"
   ```

   Or

   ```
   $ sudo yum erase "graphsurgeon-tf"
   ```

   The **uff-converter-tf** will also be removed with the above command.

   You can use the following command to uninstall **uff-converter-tf** and not remove **graphsurgeon-tf**, however, it is no longer required.

   ```
   $ sudo apt-get purge "uff-converter-tf"
   ```

   Or

   ```
   $ sudo yum erase "uff-converter-tf"
   ```

   You can later use **autoremove** to uninstall **graphsurgeon-tf** as well.

   ```
   $ sudo apt-get autoremove
   ```

   Or

```
$ sudo yum autoremove
```

3. Uninstall the Python TensorRT wheel file.

   If using Python 2.7:

   ```
   $ sudo pip2 uninstall tensorrt
   ```

   If using Python 3.x:

   ```
   $ sudo pip3 uninstall tensorrt
   ```

4. Uninstall the Python UFF wheel file.

   If using Python 2.7:

   ```
   $ sudo pip2 uninstall uff
   ```

   If using Python 3.x:

   ```
   $ sudo pip3 uninstall uff
   ```

5. Uninstall the Python GraphSurgeon wheel file.

   If using Python 2.7:

   ```
   $ sudo pip2 uninstall graphsurgeon
   ```

   If using Python 3.x:

   ```
   $ sudo pip3 uninstall graphsurgeon
   ```

Chapter 7. INSTALLING PYCUDA

Attention If you have to update your CUDA version on your system, do not install PyCUDA at this time. Perform the steps in Updating CUDA first, then install PyCUDA.

PyCUDA is used within Python wrappers to access NVIDIA's CUDA APIs. Some of the key features of PyCUDA include:

- Maps all of CUDA into Python.
- Enables run-time code generation (RTCG) for flexible, fast, automatically tuned codes.
- Added robustness: automatic management of object lifetimes, automatic error checking
- Added convenience: comes with ready-made on-GPU linear algebra, reduction, scan.
- Add-on packages for FFT and LAPACK available.
- Fast. Near-zero wrapping overhead.

To install PyCUDA first make sure **`nvcc`** is in your **`PATH`**, then issue the following command:

```
pip install 'pycuda>=2019.1.1'
```

If you encounter any issues with PyCUDA usage after installing PyCUDA with the above command, you may need to recompile it yourself. For more information, see Installing PyCUDA on Linux.

7.1. Updating CUDA

Existing installations of PyCUDA will not automatically work with a newly installed CUDA Toolkit. That is because PyCUDA will only work with a CUDA Toolkit that is already on the target system when PyCUDA was installed. This requires that PyCUDA

be updated after the newer version of the CUDA Toolkit is installed. The steps below are the most reliable method to ensure that everything works in a compatible fashion after the CUDA Toolkit on your system has been upgraded.

1. Uninstall the existing PyCUDA installation.
2. Update CUDA. For more information, see the CUDA Installation Guide.
3. Install PyCUDA. To install PyCUDA, issue the following command:

```
pip install 'pycuda>=2019.1.1'
```

Chapter 8. INSTALLING ONNX FOR PYTHON

The binary distribution of ONNX from PyPI in certain scenarios may be incompatible with the TensorRT Python bindings, specifically on Ubuntu 14.04 and RHEL 7.x. ONNX and TensorRT are both using **pybind11** to generate their Python bindings. If the STL implementations are incompatible, then importing both the ONNX and TensorRT Python modules at the same time will result in failure. To workaround this issue, build the ONNX Python module from its source.

Ensure that you:

- install the following packages natively on your system and not through a virtual environment such as Conda.
- have the following packages installed before proceeding.

- **cmake >= 3.2**
- **protobuf-compiler**
- **libprotoc-dev**

The **pip** command below will install or upgrade the ONNX Python module from its source to ensure compatibility with TensorRT, which was built using the distribution compiler. Replace the version below with the specific version of ONNX that is supported by your TensorRT release.

```
pip install --no-binary onnx 'onnx==1.5.0'
```

Chapter 9. TROUBLESHOOTING

For troubleshooting support refer to your support engineer or post your questions onto the NVIDIA Developer Forum.

Appendix A. APPENDIX

A.1. ACKNOWLEDGEMENTS

TensorRT uses elements from the following software, whose licenses are reproduced below:

Google Protobuf

This license applies to all parts of Protocol Buffers except the following:

- Atomicops support for generic gcc, located in **`src/google/protobuf/stubs/atomicops_internals_generic_gcc.h`**. This file is copyrighted by Red Hat Inc.
- Atomicops support for AIX/POWER, located in **`src/google/protobuf/stubs/atomicops_internals_power.h`**. This file is copyrighted by Bloomberg Finance LP.

Copyright 2014, Google Inc. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of Google Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF

MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR
CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED
TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA,
OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF
THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH
DAMAGE.

Code generated by the Protocol Buffer compiler is owned by the owner of the input file used when generating it. This code is not standalone and requires a support library to be linked with it. This support library is itself covered by the above license.

Google Flatbuffers

Apache License Version 2.0, January 2004 http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

   "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

   "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

   "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

   "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

   "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

   "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

    a. You must give any other recipients of the Work or Derivative Works a copy of this License; and
    b. You must cause any modified files to carry prominent notices stating that You changed the files; and
    c. You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
    d. If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

    You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.
5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.
6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.
7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY

KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.
9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright 2014 Google Inc.

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at: http://www.apache.org/licenses/LICENSE-2.0.

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

BVLC Caffe

COPYRIGHT

All contributions by the University of California:

Copyright (c) 2014, 2015, The Regents of the University of California (Regents) All rights reserved.

All other contributions:

Copyright (c) 2014, 2015, the respective contributors All rights reserved.

Caffe uses a shared copyright model: each contributor holds copyright over their contributions to Caffe. The project versioning records all such contribution and copyright details. If a contributor wants to further mark their specific copyright on a particular contribution, they should indicate their copyright solely in the commit message of the change when it is committed.

LICENSE

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

CONTRIBUTION AGREEMENT

By contributing to the BVLC/Caffe repository through pull-request, comment, or otherwise, the contributor releases their content to the license and copyright terms herein.

half.h

jQuery.js

jQuery.js is generated automatically under doxygen.

In all cases TensorRT uses the functions under the MIT license.

CRC

getopt.c

Sponsored in part by the Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory, Air Force Materiel Command, USAF, under agreement number F39502-99-1-0512.

This code is derived from software contributed to The NetBSD Foundation by Dieter Baron and Thomas Klausner.

Notice

Trademarks

Copyright

www.nvidia.com