NVIDIA cuDNN

Installation Guide | NVIDIA Docs

Table of Contents

Chapter 1. Installing cuDNN on Linux

1.1. Prerequisites

For the latest compatibility software versions of the OS, NVIDIA CUDA, the CUDA driver, and the NVIDIA hardware, refer to the NVIDIA cuDNN Support Matrix.

1.1.1. Installing NVIDIA Graphics Drivers

Install up-to-date NVIDIA graphics drivers on your Linux system.

1. Go to: NVIDIA download drivers
2. Select the GPU and OS version from the drop-down menus.
3. Download and install the NVIDIA graphics driver as indicated on that web page. For more information, select the ADDITIONAL INFORMATION tab for step-by-step instructions for installing a driver.
4. Restart your system to ensure that the graphics driver takes effect.

1.1.2. Installing the CUDA Toolkit for Linux

Refer to the following instructions for installing CUDA on Linux, including the CUDA driver and toolkit: NVIDIA CUDA Installation Guide for Linux.

1.1.3. Installing Zlib

For Ubuntu users, to install the zlib package, run:

```
sudo apt-get install zlib1g
```

For RHEL users, to install the zlib package, run:

```
sudo yum install zlib
```

1.2. Downloading cuDNN for Linux

In order to download cuDNN, ensure you are registered for the NVIDIA Developer Program.

1. Go to: NVIDIA cuDNN home page.
2. Click Download.
3. Complete the short survey and click Submit.
4. Accept the Terms and Conditions. A list of available download versions of cuDNN displays.
5. Select the cuDNN version that you want to install. A list of available resources displays.

1.3. Installing on Linux

The following steps describe how to build a cuDNN dependent program. Choose the installation method that meets your environment needs. For example, the tar file installation applies to all Linux platforms. The Debian package installation applies to Debian 11, Ubuntu 18.04, Ubuntu 20.04, and 22.04. The RPM package installation applies to RHEL7, RHEL8, and RHEL9.

In the following sections:

▶ your CUDA directory path is referred to as `/usr/local/cuda/`

▶ your cuDNN download path is referred to as `<cudnnpath>`

1.3.1. Tar File Installation

Before issuing the following commands, you must replace `X.Y` and `v8.x.x.x` with your specific CUDA and cuDNN versions and package date.

1. Navigate to your `<cudnnpath>` directory containing the cuDNN tar file.
2. Unzip the cuDNN package.

   ```
   $ tar -xvf cudnn-linux-$arch-8.x.x.x_cudaX.Y-archive.tar.xz
   ```
   Where `$arch` is `x86_64`, `sbsa`, or `ppc64le`.

3. Copy the following files into the CUDA toolkit directory.

   ```
   $ sudo cp cudnn-*-archive/include/cudnn*.h /usr/local/cuda/include
   $ sudo cp -P cudnn-*-archive/lib/libcudnn* /usr/local/cuda/lib64
   $ sudo chmod a+r /usr/local/cuda/include/cudnn*.h /usr/local/cuda/lib64/libcudnn*
   ```

1.3.2. Debian Local Installation

Download the Debian local repository installation package. Before issuing the following commands, you must replace `X.Y` and `8.x.x.x` with your specific CUDA and cuDNN versions.

1. Navigate to your `downloads` directory containing the cuDNN Debian local installer file.
2. Enable the local repository.

```
sudo dpkg -i cudnn-local-repo-$distro-8.x.x.x_1.0-1_amd64.deb
```
or
```
sudo dpkg -i cudnn-local-repo-$distro-8.x.x.x_1.0-1_arm64.deb
```

   Where `$distro` is `ubuntu1804`, `ubuntu2004`, `ubuntu2204`, or `debian11`.
3. Import the CUDA GPG key.

```
sudo cp /var/cudnn-local-repo-*/cudnn-local-*-keyring.gpg /usr/share/keyrings/
```
4. Refresh the repository metadata.

```
sudo apt-get update
```
5. Install the runtime library.

```
sudo apt-get install libcudnn8=8.x.x.x-1+cudaX.Y
```
6. Install the developer library.

```
sudo apt-get install libcudnn8-dev=8.x.x.x-1+cudaX.Y
```
7. Install the code samples.

```
sudo apt-get install libcudnn8-samples=8.x.x.x-1+cudaX.Y
```

1.3.3.   RPM Local Installation

Download the RPM local repository installation package. Before issuing the following commands, you must replace `X.Y` and `8.x.x.x` with your specific CUDA and cuDNN versions.

1. Navigate to your `downloads` directory containing the cuDNN RPM local installer file.
2. Enable the local repository.

```
sudo rpm -i cudnn-local-repo-$distro-8.x.x.x-1.0-1.x86_64.rpm
```
or
```
sudo rpm -i cudnn-local-repo-$distro-8.x.x.x-1.0-1.aarch64.rpm
```

   Where `$distro` is `rhel7`, `rhel8`, or `rhel9`.
3. Refresh the repository metadata.

```
sudo yum clean all
```
4. Install the runtime library.

```
sudo yum install libcudnn8-8.x.x.x-1.cudaX.Y
```
5. Install the developer library.

```
sudo yum install libcudnn8-devel-8.x.x.x-1.cudaX.Y
```
6. Install the code samples.

```
sudo yum install libcudnn8-samples-8.x.x.x-1.cudaX.Y
```

1.3.4.   Package Manager Installation

The Package Manager installation interfaces with your system's package manager.

If the actual installation packages are available online, then the package manager will automatically download them and install them.

1.3.4.1. Ubuntu/Debian Network Installation

These are the installation instructions for Debian 11, Ubuntu 18.04, Ubuntu 20.04, and 22.04 users.

1. Enable the network repository. Perform the steps described in the NVIDIA CUDA Installation Guide for Ubuntu or the NVIDIA CUDA Installation Guide for Debian.

> Note: For the `$distro/$arch` noted in the above links, refer to the NVIDIA cuDNN Support Matrix for the `$distro/$arch` supported versions, as cuDNN's Support Matrix might differ from CUDA.

Where `$distro/$arch` should be replaced by one of the following:

- ► `ubuntu1804/x86_64`
- ► `ubuntu2004/x86_64`
- ► `ubuntu2204/x86_64`
- ► `debian11/x86_64`
- ► `ubuntu2004/sbsa`
- ► `ubuntu2204/sbsa`

2. Install the cuDNN library and cuDNN samples:

```
sudo apt-get install libcudnn8=${cudnn_version}-1+${cuda_version}
sudo apt-get install libcudnn8-dev=${cudnn_version}-1+${cuda_version}
sudo apt-get install libcudnn8-samples=${cudnn_version}-1+${cuda_version}
```

Where:

- ► `${cudnn_version}` is `8.9.7.*`
- ► `${cuda_version}` is `cuda12.2` or `cuda11.8`

1.3.4.2. RHEL Network Installation

These are the installation instructions for RHEL7, RHEL8, and RHEL9 users.

1. Enable the repository:

```
sudo yum-config-manager --add-repo https://developer.download.nvidia.com/compute/cuda/
repos/$distro/$arch/cuda-$distro.repo
sudo yum clean all
```

Where `$distro/$arch` should be replaced by one of the following:

- ► `rhel7/x86_64`
- ► `rhel8/x86_64`
- ► `rhel9/x86_64`
- ► `rhel8/ppc64le`
- ► `rhel8/sbsa`
- ► `rhel9/sbsa`

2. Install the cuDNN library and cuDNN samples:

```
sudo yum install libcudnn8-${cudnn_version}-1.${cuda_version}
sudo yum install libcudnn8-devel-${cudnn_version}-1.${cuda_version}
sudo yum install libcudnn8-samples-${cudnn_version}-1.${cuda_version}
```

Where:

▶ `${cudnn_version}` is `8.9.7.*`

▶ `${cuda_version}` is `cuda12.2` or `cuda11.8`

1.4. Verifying the Install on Linux

To verify that cuDNN is installed and is running properly, compile the `mnistCUDNN` sample located in the `/usr/src/cudnn_samples_v8` directory in the Debian file.

1. Copy the cuDNN samples to a writable path.

```
$cp -r /usr/src/cudnn_samples_v8/ $HOME
```

2. Go to the writable path.

```
$ cd  $HOME/cudnn_samples_v8/mnistCUDNN
```

3. Compile the `mnistCUDNN` sample.

```
$make clean && make
```

4. Run the `mnistCUDNN` sample.

```
$ ./mnistCUDNN
```

If cuDNN is properly installed and running on your Linux system, you will see a message similar to the following:

```
Test passed!
```

1.5. Upgrading from cuDNN 7.x.x to cuDNN 8.x.x

Since version 8 can coexist with previous versions of cuDNN, if the user has an older version of cuDNN such as v6 or v7, installing version 8 will not automatically delete an older revision. Therefore, if the user wants the latest version, install cuDNN version 8 by following the installation steps.

To upgrade from cuDNN v7 to v8, refer to the Package Manager Installation section and follow the steps for your OS.

To switch between v7 and v8 installations, issue `sudo update-alternatives --config libcudnn` and choose the appropriate cuDNN version.

1.6. Troubleshooting

Join the NVIDIA Developer Forum to post questions and follow discussions.

Chapter 2. Installing cuDNN on Windows

2.1. Prerequisites

For the latest compatibility software versions of the OS, CUDA, the CUDA driver, and the NVIDIA hardware, refer to the NVIDIA cuDNN Support Matrix.

2.1.1. Installing NVIDIA Graphic Drivers

Install up-to-date NVIDIA graphics drivers on your Windows system.

1. Go to: NVIDIA download drivers
2. Select the GPU and OS version from the drop-down menus.
3. Download and install the NVIDIA driver as indicated on that web page. For more information, select the ADDITIONAL INFORMATION tab for step-by-step instructions for installing a driver.
4. Restart your system to ensure that the graphics driver takes effect.

2.1.2. Installing the CUDA Toolkit for Windows

Refer to the following instructions for installing CUDA on Windows, including the CUDA driver and toolkit: NVIDIA CUDA Installation Guide for Windows.

2.2. Downloading cuDNN for Windows

In order to download cuDNN, ensure you are registered for the NVIDIA Developer Program.

1. Go to: NVIDIA cuDNN home page.
2. Click Download.
3. Complete the short survey and click Submit.

4. Accept the Terms and Conditions. A list of available download versions of cuDNN displays.
5. Select the cuDNN version that you want to install. A list of available resources displays.
6. Download the cuDNN package for Windows (zip).

2.3.   Installing on Windows

The following steps describe how to build a cuDNN dependent program.

You must replace `8.x` and `8.x.y.z` with your specific cuDNN version.

Package installation (zip)

In the following steps, the package directory path is referred to as `<packagepath>`.

1. Navigate to your `<packagepath>` directory containing the cuDNN package.
2. Unzip the cuDNN package.
   ```
   cudnn-windows-x86_64-*-archive.zip
   ```
3. Copy the following files from the unzipped package into the NVIDIA cuDNN directory.
   a). Copy `bin\cudnn*.dll` to `C:\Program Files\NVIDIA\CUDNN\v8.x\bin`.

   b). Copy `include\cudnn*.h` to `C:\Program Files\NVIDIA\CUDNN\v8.x\include`.

   c). Copy `lib\cudnn*.lib` to `C:\Program Files\NVIDIA\CUDNN\v8.x\lib`.
4. Set the following environment variable to point to where cuDNN is located. To access the value of the `$(PATH)` environment variable, perform the following steps:
   a). Open a command prompt from the Start menu.

   b). Type `Run` and hit Enter.

   c). Issue the `control sysdm.cpl` command.

   d). Select the Advanced tab at the top of the window.

   e). Click Environment Variables at the bottom of the window.

   f). Add the NVIDIA cuDNN `bin` directory path to the PATH variable:
   ```
   Variable Name: PATH
   Value to Add: C:\Program Files\NVIDIA\CUDNN\v8.x\bin
   ```
5. Add cuDNN to your Visual Studio project.
   a). Open the Visual Studio project, right-click on the project name in Solution Explorer, and choose Properties.

   b). Click VC++ Directories and append `C:\Program Files\NVIDIA\CUDNN\v8.x\include` to the Include Directories field.

   c). Click Linker > General and append `C:\Program Files\NVIDIA\CUDNN\v8.x\lib` to the Additional Library Directories field.

   d). Click Linker > Input and append `cudnn.lib` to the Additional Dependencies field and click OK.

2.4.   Upgrading cuDNN

Navigate to the directory containing cuDNN and delete the old cuDNN `bin`, `lib`, and `header` files. Remove the path to the directory containing cuDNN from the `$(PATH)` environment variable. Reinstall a newer cuDNN version by following the steps in Installing on Windows.

2.5.   Troubleshooting

Join the NVIDIA Developer Forum to post questions and follow discussions.

Chapter 3. Building a cuDNN Dependent Program

3.1. Including cuDNN's Dependencies

Because cuDNN uses symbols defined in external libraries, you need to ensure that the linker can locate these libraries while building a cuDNN dependent program. One way to achieve this is by explicitly specifying them on the linker command.

For linker dependencies for the dynamic cuDNN libs

Linux: Add `-lcublas -lcublasLt -lz` to the linker command.

Windows: Add `cublas.lib cublasLt.lib zlibwapi.lib` to the linker command.

Linker dependencies for the static cuDNN libs

Linux: Add `-lcublas_static -lcublasLt_static -lz -lculibos -lnvrtc_static -lnvrtc-builtins_static -lnvptxcompiler_static -lcudart_static` to the linker command.

Windows: Not applicable. Static cuDNN libs for Windows are not supported.

3.2. cuDNN's Inter-Library Dependencies

Since cuDNN is split into several libraries, dependencies between them need to be taken into account.

For example, when statically linking `libcudnn_cnn_infer_static.a` into an application, `libcudnn_ops_infer_static.a` is also needed, in this order (a dependent library followed by its dependency). Specifically:

```
-static -Wl,--whole-archive ${CUDNN_PATH}/libcudnn_cnn_infer_static.a ${CUDNN_PATH}/
libcudnn_ops_infer_static.a -Wl,--no-whole-archive
```

The chain of dependencies can be found in the NVIDIA cuDNN API Reference.

Chapter 4. Cross-Compiling cuDNN Samples

This section describes how to cross-compile cuDNN samples.

4.1. Linux AArch64 SBSA

Follow the steps in this section to cross-compile cuDNN samples on Linux AArch64. Linux AArch64 incorporates ARM® based CPU cores for Server Base System Architecture (SBSA).

4.1.1. Installing the CUDA Toolkit for Linux AArch64 SBSA

Before issuing the following commands, you must replace `x-y` with your specific CUDA version.

1. Download the Ubuntu package: `cuda*ubuntu*_amd64.deb`
2. Download the cross compile package: `cuda*-cross-sbsa*_all.deb`
3. Execute the following commands:

```
sudo dpkg -i cuda*ubuntu*_amd64.deb

sudo dpkg -i cuda-repo-cross-sbsa*_all.deb

sudo apt-get update

sudo apt-get install cuda-toolkit-x-y -y

sudo apt-get install cuda-cross-sbsa* -y
```

4.1.2. Installing cuDNN for Linux AArch64 SBSA

1. Download the cuDNN Ubuntu package for your preferred CUDA toolkit version.
   `cudnn-local-repo-*_amd64.deb`
2. Download the cross compile package.
   `cudnn-local-repo-cross-sbsa-*_all.deb`
3. Execute the following commands:

```
sudo dpkg -i cudnn-local-repo-*_amd64.deb

sudo apt-get update

sudo apt-get install libcudnn8 libcudnn8-dev libcudnn-samples -y

sudo dpkg -i cudnn-local-repo-cross-sbsa-*_all.deb

sudo apt-get update

sudo apt-get install libcudnn8-cross-sbsa -y
```

4. Install AArch64 host compiler.
```
sudo apt install g++-aarch64-linux-gnu
```

4.1.3.   Cross-Compiling cuDNN Samples for Linux AArch64 SBSA

1. Copy the `cudnn_samples_v8` directory to your home directory:
```
$ cp -r /usr/src/cudnn_samples_v8 $HOME
```

2. For each sample, execute the following commands:
```
$ cd $HOME/cudnn_samples_v8/(each sample)
$ sudo make TARGET_ARCH=aarch64 SBSA=1
```

Chapter 5.   Appendix

5.1.   ACKNOWLEDGEMENTS

NVIDIA would like to thank the following individuals and institutions for their contributions:

▶ This product includes zlib - a general purpose compression library https://zlib.net/ Copyright © 1995-2017 Jean-loup Gailly and Mark Adler

▶ This product includes zstr - a C++ zlib wrapper https://github.com/mateidavid/zstr Copyright © 2015 Matei David, Ontario Institute for Cancer Research

▶ This product includes RapidJSON - A fast JSON parser/generator for C++ with both SAX/DOM style API https://github.com/Tencent/rapidjson Copyright © 2015 THL A29 Limited, a Tencent company, and Milo Yip.

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

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