

# CUDA version

---

[TOC]

## 0. cuda版本确认

查看系统安装的 CUDA 路径（确认是否存在），如果已安装，通常会在 `/usr/local/` 目录下有对应的文件夹：

```
ls -l /usr/local | grep cuda
```

```
ls -l /usr/local | grep cuda
lrwxrwxrwx  1 root root   22 11月 17 10:32 cuda -> /etc/alternatives/cuda
drwxr-xr-x 17 root root 4096 11月  5 17:44 cuda-11.8
drwxr-xr-x 17 root root 4096 11月  5 17:39 cuda-12.4
drwxr-xr-x 17 root root 4096 11月 17 10:31 cuda-12.6
drwxr-xr-x 17 root root 4096 11月  6 11:06 cuda-13.0
```

## 1. 环境确认以及CUDA下载

确保nvidia驱动已经正确安装，使用`nvidia-smi`确认，如果没有安装请跳转[Nvidia驱动安装](#)文档进行安装

选择合适的版本进行下载 [CUDA Toolkit Archive](#)

一般选择 **runfile (local)** 方式进行下载，例如(图片以13.0为例)

```
wget
https://developer.download.nvidia.com/compute/cuda/13.0.1/local_installers/
cuda_13.0.1_580.82.07_linux.run
```

# CUDA Toolkit 13.0 Update 1 Downloads

### Select Target Platform

Click on the green buttons that describe your target platform. Only supported platforms will be shown. By downloading and using the software, you agree to fully comply with the terms and conditions of the [CUDA EULA](#).

Operating System	Linux	Windows						
Architecture	x86_64	arm64-sbsa						
Distribution	Amazon-Linux	Azure-Linux	Debian	Fedora	KylinOS	OpenSUSE	Oracle-Linux	RHEL
	Rocky	SLES	Ubuntu	WSL-Ubuntu				
Version	22.04	24.04						
Installer Type	deb (local)	deb (network)	runfile (local)					

Download Installer for Linux Ubuntu 24.04 x86\_64

The base installer is available for download below.

> CUDA Toolkit Installer

Installation Instructions:

```
$ wget https://developer.download.nvidia.com/compute/cuda/13.0.1/local_installers/cuda_13.0.1_580.82.07_linux.run
$ sudo sh cuda_13.0.1_580.82.07_linux.run
```

Additional installation options are detailed [here](#).

下载之后到指定文件夹后进行安装，具体安装参考[系统级CUDA安装管理](#)

## 2. 系统级CUDA安装管理

以11.8和12.4为例（注意包名和cu的是否对得上）

例如我下载的包名为**cuda\_11.8.0\_520.61.05\_linux.run**,那么我执行指令为

```
sudo sh cuda_11.8.0_520.61.05_linux.run --toolkit --silent --override --toolkitpath=/usr/local/cuda-11.8
```

后面**cuda-11.8**需要对应包的版本

```
# 安装多个CUDA版本到系统目录
sudo sh cuda_11.8.0_520.61.05_linux.run --toolkit --silent --override --toolkitpath=/usr/local/cuda-11.8
sudo sh cuda_12.4.0_550.54.15_linux.run --toolkit --silent --override --toolkitpath=/usr/local/cuda-12.4

# 注册到update-alternatives
sudo update-alternatives --install /usr/local/cuda cuda /usr/local/cuda-
```

```
11.8 118
sudo update-alternatives --install /usr/local/cuda cuda /usr/local/cuda-
12.4 124

# 设置默认版本（可选）
sudo update-alternatives --set cuda /usr/local/cuda-12.4
```

### 3. 自动化环境配置脚本

创建通用的配置函数：

```
# 为vm1reocn环境创建配置
mkdir -p ~/anaconda3/envs/vm1reocn/etc/conda/activate.d
mkdir -p ~/anaconda3/envs/vm1reocn/etc/conda/deactivate.d

# 激活脚本
cat > ~/anaconda3/envs/vm1reocn/etc/conda/activate.d/cuda_hook.sh << 'EOF'
#!/bin/bash
echo "激活 CUDA 11.8 环境"

# 设置CUDA相关环境变量
export CUDA_HOME=/usr/local/cuda-11.8
export PATH=${CUDA_HOME}/bin:${PATH}
export LD_LIBRARY_PATH=${CUDA_HOME}/lib64:${LD_LIBRARY_PATH}

# 设置GCC版本
export CC=/usr/bin/gcc-9
export CXX=/usr/bin/g++-9

# 可选：备份当前设置（用于恢复）
export OLD_CUDA_HOME=${CUDA_HOME:-}
export OLD_PATH=${PATH}
export OLD_LD_LIBRARY_PATH=${LD_LIBRARY_PATH}
export OLD_CC=${CC:-}
export OLD_CXX=${CXX:-}
EOF

# 退出脚本
cat > ~/anaconda3/envs/vm1reocn/etc/conda/deactivate.d/cuda_hook.sh <<
'EOF'
#!/bin/bash
echo "退出 CUDA 11.8 环境"

# 恢复环境变量
if [ -n "${OLD_CUDA_HOME}" ]; then
    export CUDA_HOME=${OLD_CUDA_HOME}
else
    unset CUDA_HOME
fi

export PATH=${OLD_PATH}
```

```
export LD_LIBRARY_PATH=${OLD_LD_LIBRARY_PATH}

if [ -n "${OLD_CC}" ]; then
    export CC=${OLD_CC}
else
    unset CC
fi

if [ -n "${OLD_CXX}" ]; then
    export CXX=${OLD_CXX}
else
    unset CXX
fi

# 清理备份变量
unset OLD_CUDA_HOME OLD_PATH OLD_LD_LIBRARY_PATH OLD_CC OLD_CXX
EOF

chmod +x ~/anaconda3/envs/vm1reocn/etc/conda/activate.d/cuda_hook.sh
chmod +x ~/anaconda3/envs/vm1reocn/etc/conda/deactivate.d/cuda_hook.sh
```

```
# vm1rs环境 (CUDA 12.4 + GCC 11)
mkdir -p ~/anaconda3/envs/vm1rs/etc/conda/activate.d
mkdir -p ~/anaconda3/envs/vm1rs/etc/conda/deactivate.d

cat > ~/anaconda3/envs/vm1rs/etc/conda/activate.d/cuda_hook.sh << 'EOF'
#!/bin/bash
echo "激活 CUDA 12.4 环境"

export CUDA_HOME=/usr/local/cuda-12.4
export PATH=${CUDA_HOME}/bin:${PATH}
export LD_LIBRARY_PATH=${CUDA_HOME}/lib64:${LD_LIBRARY_PATH}

export CC=/usr/bin/gcc-11
export CXX=/usr/bin/g++-11

export OLD_CUDA_HOME=${CUDA_HOME:-}
export OLD_PATH=${PATH}
export OLD_LD_LIBRARY_PATH=${LD_LIBRARY_PATH}
export OLD_CC=${CC:-}
export OLD_CXX=${CXX:-}
EOF

cat > ~/anaconda3/envs/vm1rs/etc/conda/deactivate.d/cuda_hook.sh << 'EOF'
#!/bin/bash
echo "退出 CUDA 12.4 环境"

if [ -n "${OLD_CUDA_HOME}" ]; then
    export CUDA_HOME=${OLD_CUDA_HOME}
else
    unset CUDA_HOME
```

```

fi

export PATH=${OLD_PATH}
export LD_LIBRARY_PATH=${OLD_LD_LIBRARY_PATH}

if [ -n "${OLD_CC}" ]; then
    export CC=${OLD_CC}
else
    unset CC
fi

if [ -n "${OLD_CXX}" ]; then
    export CXX=${OLD_CXX}
else
    unset CXX
fi

unset OLD_CUDA_HOME OLD_PATH OLD_LD_LIBRARY_PATH OLD_CC OLD_CXX
EOF

chmod +x ~/anaconda3/envs/vm1rs/etc/conda/activate.d/cuda_hook.sh
chmod +x ~/anaconda3/envs/vm1rs/etc/conda/deactivate.d/cuda_hook.sh

```

## 4. GCC/G++版本环境配置

```

# 安装必要的GCC版本
sudo apt update
sudo apt install gcc-9 g++-9 gcc-10 g++-10 gcc-11 g++-11

# 注册GCC到update-alternatives
sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-9 9
sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-10 10
sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-11 11
sudo update-alternatives --install /usr/bin/g++ g++ /usr/bin/g++-9 9
sudo update-alternatives --install /usr/bin/g++ g++ /usr/bin/g++-10 10
sudo update-alternatives --install /usr/bin/g++ g++ /usr/bin/g++-11 11

# 配置vm1reocn环境 (CUDA 11.8 + GCC 9)
setup_cuda_env "vm1reocn" "11.8" "9"

# 配置vm1rs环境 (CUDA 12.4 + GCC 11)
setup_cuda_env "vm1rs" "12.4" "11"

```

## 5. 验证脚本

创建验证函数检查环境配置：

```

check_cuda_env() {
    echo "=== CUDA环境检查 ==="

```

```

    echo "当前CUDA版本:"
    nvcc --version 2>/dev/null || echo "nvcc未找到"
    echo -e "\nCUDA链接位置:"
    ls -l /usr/local/cuda
    echo -e "\nGCC版本:"
    gcc --version | head -n1
    echo -e "\n环境变量:"
    echo "CUDA_HOME=$CUDA_HOME"
    echo "PATH包含CUDA: $(echo $PATH | grep -o "cuda[^\:]*" | head -n1)"
    echo "CC=$CC"
    echo "CXX=$CXX"
}

```

## 6. 使用示例

```

# 激活环境并验证
conda activate vm1reocn
check_cuda_env

# 切换到另一个环境
conda deactivate
conda activate vm1rs
check_cuda_env

```

## 7. 添加boot权限

使用环境变量而非**sudo**（避免密码输入）：

```

# 在激活脚本中使用软链接而非update-alternatives
ln -sf /usr/local/cuda-$cuda_version
$CONDA_PREFIX/etc/conda/activate.d/cuda
export PATH=$CONDA_PREFIX/etc/conda/activate.d/cuda/bin:$PATH

```

创建管理工具脚本 `cuda_manager.sh`（可选）：

```

#!/bin/bash
list_cuda_versions() {
    echo "已安装的CUDA版本:"
    ls -d /usr/local/cuda-* 2>/dev/null
}

switch_global_cuda() {
    sudo update-alternatives --config cuda
}

```