1 NPU 驱动和固件安装

1.1 确认操作系统

1. 执行如下命令，查询服务器当前运行环境的操作系统架构及版本。

   uname -m && cat /etc/*release

   显示如下，表示当前运行环境操作系统为 Ubuntu 18.04.1。

```
x86_64
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=18.04
DISTRIB_CODENAME=bionic
DISTRIB_DESCRIPTION="Ubuntu 18.04.1 LTS"
NAME="Ubuntu"
VERSION="18.04.1 LTS (Bionic Beaver)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 18.04.1 LTS"
VERSION_ID="18.04"
```

1.2 获取软件包

驱动：Atlas-200-npu-driver_21.0.4_linux-x86_64.run

固件：Atlas-200-npu-firmware_1.80.22.2.220.run

1.3 创建运行用户

首先介绍安装用户和运行用户概念，安装用户为安装驱动和固件所使用的用户，运行用户为驱动固件安装完成后，后续运行推理或训练业务时启动运行驱动和

固件的用户。

驱动和固件的安装用户必须为 root 用户，驱动和固件的运行用户可以为 root 用户或者非 root 用户，如果后续用户对运行业务有用户权限控制或者多用户隔离的需求，可创建非 root 用户作为后续运行用户。本章节介绍创建运行用户的步骤。

若用户首次安装时，使用.zip 格式软件包，系统会自动创建用户组和用户 HwHiAiUser，无需手动创建。请跳过此章节。

表 1-1 创建运行用户

运行用户	操作步骤

• root用户
• 非root用户（推荐）
  ▪ HwHiAiUser
  ▪ 非HwHiAiUser
 	
说明：
如果使用root用户作为运行用户，则需要与--install-for-all参数配合使用，详情请参见参数说明/常用命令，该场景下权限控制可能存在安全风险；如果使用非root用户作为运行用户，则创建的该用户不能为root用户属组。
	• 如果创建的用户是HwHiAiUser，安装软件包时无需指定运行用户，默认即为HwHiAiUser。
	• 如果创建的用户是非HwHiAiUser（含root），安装软件包时必须指定运行用户（通过--install-username=*username* --install-usergroup=*usergroup*参数指定）。因此如果对运行用户名称没有特殊要求，建议使用HwHiAiUser。
	请参见如下方法创建运行用户。
	1. 以root用户登录服务器。
	2. 执行如下命令，创建运行用户。
	`# groupadd usergroup`
	`# useradd -g usergroup -d /home/username -m username -s /bin/bash`

1.4 安装驱动

1 确认安装方式

表 1-2 环境检查

序号	检查项	检查命令	检查说明
1	操作系统内核版本	**uname -r**	如果 Host 系统中安装过 run 包，内核升级后会出现 run 包启动异常的问题。出现该问题时请参见升级系统内核版本时未卸载 run 包导致 run 包不可用解决。
2	GCC	**gcc -v**	• 若使用二进制文件直接安

序号	检查项	检查命令	检查说明
	版本		装驱动包，安装无须检查 GCC 版本。 • 若使用源码编译安装驱动包，GCC 版本须不低于《版本配套表》中要求的版本。
3	查询系统是否安装过软件包	**lsmod|grep drv_pcie_host**	• 如无内容表示未安装过软件包。可以直接安装软件包。 • 如有内容，表示安装过软件包。需要先卸载驱动包后，再安装新版本软件包。卸载驱动包具体请参见卸载驱动。
5（可选）	驱动安装过程中所需的 Linux 工具	NA	请根据表 6-2 检查所需的 Linux 工具并进行安装。如系统中缺失，安装过程中也会有回显信息提示，可以根据提示信息进行安装。

2 安装驱动（适用于.run 格式）

本章节介绍宿主机上的安装方法，本章节以*{product name}*-npu-driver_*x.x.x*_linux-*{arch}*.run 包为例说明，具体操作以客户实际 host 系统对应驱动包为准。

注意事项

- 在软件包安装或升级过程中，不允许对 Host 或 Device 进行复位或下电操作，否则会出现 Device 无法正常启动或者升级失败。

操作步骤

须知

首次安装请按照“**驱动->固件**”的顺序；覆盖安装或升级请按照“**固件->驱动**”的顺序，分别安装软件包。固件安装请参见安装固件（适用于.run 格式）。

首次安装过程中如果使用了--install for all 参数，覆盖安装中也需要保留--install for all 参数。

1. 将获取软件包准备的安装包上传至 Linux 系统任意目录下（如/opt）。
2. 使用 PuTTY 登录服务器的 OS 命令行。
3. 执行如下命令，切换至 root 用户。

   su - root

4. 执行如下命令，进入软件包所在路径（如/opt）。

   cd /*opt*

5. 执行如下命令，增加软件包的可执行权限。

   chmod +x *{product name}*-npu-driver_*x.x.x*_linux-*{arch}*.run

6. 执行如下命令，校验 run 安装包的一致性和完整性。

   ./*{product name}*-npu-driver_*x.x.x*_linux-*{arch}*.run **--check**

   若出现如下回显信息，表示软件包校验成功。

```
Verifying archive integrity...  100%   SHA256 checksums are OK. All good.
```

说明

软件包校验使用 SHA256 加密方式；校验过程中若提示：./*{product name}*-npu-driver_*x.x.x*_linux-*{arch}*.run does not contain an embedded MD5 checksum.和./*{product name}*-npu-driver_*x.x.x*_linux-*{arch}*.run does not contain a CRC checksum，表示没有使用 MD5 和 CRC 加密方式，可忽略。

7. 执行**./***{product name}*-npu-driver_*x.x.x*_linux-*{arch}*.run **--full** 命令，完成驱动安装，软件包默认安装路径为“/usr/local/Ascend”。

说明

- 安装详细日志路径：/var/log/ascend_seclog/ascend_install.log。
- 安装后软件包的安装路径、安装命令以及运行用户信息记录路径为“/etc/ascend_install.info”。
- 安装 run 包驱动时，会将动态库 libdcmi.so 和头文件 dcmi_interface_api.h 拷贝到“/usr/local/dcmi/”目录下。
- 若创建运行用户是非 HwHiAiUser，安装驱动软件包时需要指定运行用户（通过**--install-username=***username* **--install-usergroup=***usergroup* 参数指定）。
- 如果指定 root 用户为运行用户，则需要与--install-for-all 参数配合使用，如下所示，该场景下权限控制可能存在安全风险。

```
--install-username=root  --install-usergroup=root
--install-for-all
```

- 如果服务器无 BMC 时，只能采用默认路径安装驱动。如果采用指定路径安装，ipmi_si 驱动加载时会出现阻塞，导致磁盘挂载慢，影响 npu 正常运行。
- Device 侧系统类日志通过 msnpureport 工具传输到 Host 侧进行查看。导出操作以及导出日志的存储路径具体请参见对应产品和版本的《黑匣子日志参考》的“msnpureport 工具使用”章节。容

> 器内不支持查看 Device 侧系统类日志，也不支持通过 msnpureport 工具导出 Device 侧系统类日志。

若用户需要指定安装路径，如以“/test/HiAI/”为例。

可执行**./***{product name}*-npu-driver_*x.x.x*_linux-*{arch}*.run **--full --install-path=**/test/HiAI/命令，完成安装。

- 若指定路径不存在，则安装时会自动创建目录，若有多层目录，则只有最后一层目录不存在时会自动创建。
- 若指定路径已存在：
  - 若该路径下所有层级目录属主为 root 用户，则请确保所有层级目录权限至少为 755。若不满足要求，请修改路径权限：**chmod 755 路径**
  - 若该路径下有一层目录属主为非 root 用户，则请用户自行修改为 root 属主，并确保所有层级目录权限为 755。若不满足要求，请修改路径属主为 root：**chown root: 群组名 路径**

若系统出现如下关键回显信息，则表示驱动安装成功。

```
Driver package installed successfully!
```

8. 根据系统提示信息决定是否重启系统，若需要重启，请执行以下命令；否则，请跳过此步骤。

   reboot

9. 执行 **npu-smi info** 查看驱动加载是否成功。

若出现如下回显信息，说明加载成功。否则，说明加载失败。请联系华为技术支持处理。

```
+------------------------------------------------------------------------------------------+
| npu-smi 21.0.4                       Version: 21.0.4.B030                                |
+-------------------+-----------------+-------------------------------------------------+
| NPU     Name      | Health          | Power(W)                                        |
Temp(C)               Hugepages-Usage(page)  |
| Chip    Device    | Bus-Id          | AICore(%)                                       |
Memory-Usage(MB)                                |
+===================+=================+=================================================+
| 0       310       | OK              | 12.8                                            |
49             0        / 970                   |
| 0       0         | NA              | 0                                               |
1556 / 4096                                     |
+===================+=================+=================================================+
```

说明

- 加载失败，可以执行 **dmesg** 命令查看 linux 启动日志。如果出现 */安装路径*/driver/device/davinci_mini.fd copy err 信息，请参考卸载驱动（适用于.run 格式）章节卸载驱动，并使用默认路径重新进行驱动安装。
- 回显中“npu-smi”后面的字段为 npu-smi 工具版本号；“Version:”后面的字段为驱动版本号。

1.5 安装固件

安装固件（适用于.run 格式）

首次安装请按照“**驱动->固件**”的顺序；覆盖安装或升级请按照“**固件->驱动**”的顺序，分别安装软件包，驱动安装请参见安装驱动，本章节以*{product name}*-npu-firmware_*<version>*.run 为例介绍宿主机上.run 格式固件包的安装方法。

注意事项

- 在软件包安装或升级过程中，不允许对 Host 或 Device 进行复位或下电操作，否则会出现 Device 无法正常启动或者升级失败。

操作步骤

1. 将获取软件包准备的安装包上传至 Linux 系统任意目录下（如/opt）。
2. 使用 PuTTY 登录服务器的 OS 命令行。
3. 执行如下命令，切换至 root 用户。

   su - root

4. 执行如下命令，进入软件包所在路径（如/opt）。

   cd */opt*

5. 执行如下命令，增加软件包的可执行权限。

   chmod +x *{product name}-npu-firmware_ <version>.run*

6. 执行如下命令，校验 run 安装包的一致性和完整性。

   ./ *{product name}-npu-firmware_ <version>.run* **--check**

   出现如下回显信息，表示软件包校验成功。

```
Verifying archive integrity...  100%   SHA256 checksums are OK. All good.
```

说明

软件包校验使用 SHA256 加密方式；校验过程中若提示：*./{product name}-npu-firmware_ <version>.run* does not contain an embedded MD5 checksum.和*./{product name}-npu-firmware_ <version>.run* does not contain a CRC checksum，表示没有使用 MD5 和 CRC 加密方式，可忽略。

7. 执行如下安装命令完成安装。

   ./*{product name}-npu-firmware_ <version>.run* **--full**

   若系统出现如下关键回显信息，表示固件安装成功。

```
Firmware package installed successfully!
```

说明

- 安装固件软件包不支持指定运行用户名和用户组，共用驱动的运行用户名和用户组。
- 安装固件软件包不支持指定路径，共用驱动的安装路径。
- 软件包默认安装路径：/usr/local/Ascend
- 安装详细日志路径：/var/log/ascend_seclog/ascend_install.log。
- 安装后软件包的安装路径、安装命令以及运行用户信息记录路径：/etc/ascend_install.info
- Device 侧系统类日志通过 msnpureport 工具传输到 Host 侧进行查看。导出操作以及导出日志的存储路径具体请参见对应产品和版本的《黑匣子日志参考》的“msnpureport 工具使用”章节。容器内不支持查看 Device 侧系统类日志，也不支持通过 msnpureport 工具导出 Device 侧系统类日志。

8. 根据系统提示信息决定是否立即重启系统，若需要重启，请执行以下命令；否则，请跳过此步骤。

   reboot

9. 执行如下命令查看芯片固件版本号。若与目标版本一致，则说明安装成功。

   */usr/local/Ascend/***driver/tools/upgrade-tool --device_index -1 --component -1 --version**

   > **说明**
   >
   > 如果安装驱动时指定安装路径，则命令中的“*/usr/local/Ascend*”请根据实际情况替换。

1.6 卸载驱动（适用于.run 格式）

目前发布的 run 包在宿主机上的卸载方式相同。本章以*{product name}*-npu-driver_*x.x.x*_linux-*{arch}*.run 为例介绍.run 格式驱动包卸载相关操作，具体请根据实际情况进行替换。

前提条件

未安装 run 安装包时不能使用卸载功能。

操作步骤

1. 使用 PuTTY 登录服务器的 OS 命令行。
2. 执行如下命令，切换至 root 用户。

   su - root

3. 支持以下两种卸载方式，用户根据实际情况选择其中一个方式卸载即可：

   - 在任意路径执行如下命令卸载软件包：

     *<install-path>***/driver/script/uninstall.sh**

说明

*<install-path>*表示软件包安装路径，可以执行 **cat /etc/ascend_install.info** 命令查询安装路径；请根据实际情况替换。

- 在软件包所在路径执行以下命令进行卸载：

  ./*{product name}*-npu-driver_*x.x.x*_linux-*{arch}*.run **--uninstall**

  注意

  若业务或容器占用了 davinci 节点，卸载过程中系统会提示“The davinci nodes are occupied by some processes, please stop processes and install or uninstall again, details in : /var/log/ascend_seclog/ascend_install.log”报错，可以选择使用**--force** 参数进行强制卸载，详细请参见参数说明/常用命令。但可能导致当前正在运行的业务受到影响。

  若出现如下关键回显信息，则表示驱动卸载成功。

  ```
  Driver package uninstalled successfully!
  ```

4. 根据系统提示信息决定是否重启服务器，若需要重启系统，请执行以下命令；否则，请跳过此步骤。

   reboot

   须知

   驱动卸载后，npu-smi 工具不可用。

1.7 卸载固件（适用于.run 格式）

驱动和固件的卸载没有先后顺序要求。驱动卸载请参见卸载驱动（适用于.run 格式）。本章以*{product name}*-npu-firmware_*<version>*.run 为例介绍固件卸载相关指导，具体请根据实际情况进行替换。

前提条件

未安装 run 安装包时不能使用卸载功能。

操作步骤

1. 使用 PuTTY 登录服务器的 OS 命令行。
2. 执行如下命令，切换至 root 用户。

   su - root

3. 支持以下两种方式，用户根据实际情况选择其中一个方式卸载即可：

   - 在任意路径执行如下命令卸载软件包：

     *<install-path>***/firmware/script/uninstall.sh**

     说明

     *<install-path>*表示软件包安装路径，可以执行 **cat /etc/ascend_install.info** 命令查询安装路径；请根据实际情况替换。

   - 在软件包所在路径执行以下命令进行卸载：

     ./*{product name}*-npu-firmware_*<version>*.run **--uninstall**

     若出现如下关键回显信息，则表示固件卸载成功。

     ```
     Firmware package uninstalled successfully!
     ```

4. 根据系统提示信息决定是否重启服务器，若需要重启系统，请执行以下命令；否则，请跳过此步骤。

   reboot

2 安装 CANN

2.1 安装依赖

检查源

安装过程需要下载相关依赖，请确保安装环境能够连接网络。如需配置网络代理，请参见配置系统网络代理（Linux）。

请在 root 用户下执行如下命令检查源是否可用。

```
apt-get update
```

如果命令执行报错或者后续安装依赖时等待时间过长甚至报错，则检查网络是否连接或者把“/etc/apt/sources.list”文件中的源更换为可用的源或使用镜像源（以配置华为镜像源为例，可参考华为开源镜像站）。

配置安装用户权限

可使用 root 或非 root 用户（该非 root 用户需与软件包安装用户保持一致）安装依赖，如果使用非 root 用户安装，可能需要用到提权命令，请用户自行获取所需的 sudo 权限。使用完成后请取消涉及高危命令的权限，否则有 sudo 提权风险。

安装依赖

检查系统是否安装 python 依赖以及 gcc 等软件。
分别使用如下命令检查是否安装 gcc，make 以及 python 依赖软件等。

```
gcc --version
g++ --version
make --version
cmake --version
dpkg -l zlib1g| grep zlib1g| grep ii
dpkg -l zlib1g-dev| grep zlib1g-dev| grep ii
dpkg -l libsqlite3-dev| grep libsqlite3-dev| grep ii
dpkg -l openssl| grep openssl| grep ii
```

```
dpkg -l libssl-dev| grep libssl-dev| grep ii
dpkg -l libffi-dev| grep libffi-dev| grep ii
dpkg -l unzip| grep unzip| grep ii
dpkg -l pciutils| grep pciutils| grep ii
dpkg -l net-tools| grep net-tools| grep ii
dpkg -l libblas-dev| grep libblas-dev| grep ii
dpkg -l gfortran| grep gfortran| grep ii
dpkg -l libblas3| grep libblas3| grep ii
dpkg -l libopenblas-dev| grep libopenblas-dev| grep ii
```

若分别返回如下信息则说明已经安装，进入下一步（以下回显仅为示例，请以实际情况为准）。

```
gcc (Ubuntu 7.3.0-3ubuntu1~18.04) 7.3.0
g++ (Ubuntu 7.3.0-3ubuntu1~18.04) 7.3.0
GNU Make 4.1
cmake version 3.10.2
zlib1g:arm64      1:1.2.11.dfsg-0ubuntu2 arm64          compression library - runtime
zlib1g-dev:arm64 1:1.2.11.dfsg-0ubuntu2 arm64          compression library - development
libsqlite3-dev:arm64 3.22.0-1ubuntu0.3 arm64          SQLite 3 development files
openssl    1.1.1-1ubuntu2.1~18.04.6 arm64     Secure Sockets Layer toolkit - cryptographic utility
libssl-dev:arm64 1.1.1-1ubuntu2.1~18.04.6 arm64       Secure Sockets Layer toolkit - development files
libffi-dev:arm64 3.2.1-8          arm64          Foreign Function Interface library (development files)
unzip              6.0-21ubuntu1 arm64          De-archiver for .zip files
pciutils           1:3.5.2-1ubuntu1 arm64          Linux PCI Utilities
net-tools          1.60+git20161116.90da8a0-1ubuntu1 arm64          NET-3 networking toolkit
libblas-dev:arm64 3.7.1-4ubuntu1 arm64          Basic Linear Algebra Subroutines 3, static library
gfortran           4:7.4.0-1ubuntu2.3 arm64          GNU Fortran 95 compiler
```

```
libblas3:arm64 3.7.1-4ubuntu1 arm64        Basic Linear Algebra Reference implementations, shared library
libopenblas-dev:arm64 0.2.20+ds-4   arm64       Optimized BLAS (linear algebra) library (development files)
```

否则请执行如下安装命令（如果只有部分软件未安装，则如下命令修改为只安装还未安装的软件即可）：

说明

- 如果使用 root 用户安装依赖，请将**步骤 1 至步骤 2** 命令中的 sudo 删除。
- 如果 python 及其依赖是使用非 root 用户安装，则需要执行 **su - *username*** 命令切换到非 root 用户继续执行**步骤 1 至步骤 3**。
- libsqlite3-dev 需要在 python 安装之前安装，如果用户操作系统已经安装满足版本要求的 python 环境，在此之后再安装 libsqlite3-dev，则需要重新编译 python 环境。

```
sudo apt-get install -y gcc g++ make cmake zlib1g zlib1g-dev openssl libsqlite3-dev libssl-dev libffi-dev unzip pciutils net-tools libblas-dev gfortran libblas3 libopenblas-dev
```

检查系统是否安装满足版本要求的 python 开发环境（具体要求请参见依赖列表，此步骤以环境上需要使用 python 3.7.*x* 为例进行说明）。
在当前安装用户下执行命令 **python3 --version**。

- 如果返回信息满足 python 版本要求（3.7.0~ 3.7.11），则直接进入步骤 3。
- 如果返回信息不满足满足 python 版本要求（3.7.0~ 3.7.11）：
  - 当环境上 root 用户下已安装满足要求的 python 版本（3.7.0~ 3.7.11）时，如果当前安装用户需要复用该 python，则可以直接参考设置 python3.7.5 环境变量配置环境变量。
  - 当环境上未安装满足要求的 python 版本（3.7.0~ 3.7.11）时，则可以参考如下步骤安装 python。

  使用 wget 下载 python3.7.5 源码包，可以下载到安装环境的任意目录，命令为：

  ```
  wget https://www.python.org/ftp/python/3.7.5/Python-3.7.5.tgz
  ```

  进入下载后的目录，解压源码包，命令为：

  ```
  tar -zxvf Python-3.7.5.tgz
  ```

  进入解压后的文件夹，执行配置、编译和安装命令：

  ```
  cd Python-3.7.5

  ./configure --prefix=/usr/local/python3.7.5 --enable-loadable-sqlite-extensions --enable-shared

  make
  ```

```
sudo make install
```

其中“--prefix”参数用于指定 python 安装路径，用户根据实际情况进行修改。“--enable-shared”参数用于编译出 libpython3.7m.so.1.0 动态库。“--enable-loadable-sqlite-extensions”参数用于加载 libsqlite3-dev 依赖。

本手册以--prefix=/usr/local/python3.7.5 路径为例进行说明。执行配置、编译和安装命令后，安装包在/usr/local/python3.7.5 路径，libpython3.7m.so.1.0 动态库在/usr/local/python3.7.5/lib/libpython3.7m.so.1.0 路径。

设置 python3.7.5 环境变量。

```
#用于设置 python3.7.5 库文件路径

export LD_LIBRARY_PATH=/usr/local/python3.7.5/lib:$LD_LIBRARY_PATH

#如果用户环境存在多个 python3 版本，则指定使用 python3.7.5 版本

export PATH=/usr/local/python3.7.5/bin:$PATH
```

通过以上 export 方式设置环境变量，该种方式设置的环境变量只在当前窗口有效。您也可以通过将以上命令写入~/.bashrc 文件中，然后执行 **source ~/.bashrc** 命令，使上述环境变量永久生效。注意如果后续您有使用环境上其他 python 版本的需求，则不建议将以上命令写入到~/.bashrc 文件中。

安装完成之后，执行如下命令查看安装版本，如果返回相关版本信息，则说明安装成功。

```
python3 --version

pip3 --version
```

安装前请先使用 **pip3 list** 命令检查是否安装相关依赖，若已经安装，则请跳过该步骤；若未安装，则安装命令如下（如果只有部分软件未安装，则如下命令修改为只安装还未安装的软件即可）。

- 请在安装前配置好 pip 源，具体可参考配置 pip 源。
- 安装前，建议执行命令 **pip3 install --upgrade pip** 进行升级，避免因 pip 版本过低导致安装失败。
- 如下命令如果使用非 root 用户安装，需要在安装命令后加上**--user**，例如：**pip3 install attrs --user**，安装命令可在任意路径下执行。
- 要求 numpy 版本大于等于 1.14.3，如下命令以安装 numpy 1.17.2 为例。

```
pip3 install attrs

pip3 install numpy==1.17.2

pip3 install decorator

pip3 install sympy

pip3 install cffi

pip3 install pyyaml
```

```
pip3 install pathlib2
pip3 install psutil
pip3 install protobuf
pip3 install scipy
pip3 install requests
```

2.2 安装开发工具包

获取软件包：Ascend-cann-toolkit_5.1.RC1.alpha002_linux-x86_64.run

前提条件

- 请参见安装 OS 依赖完成安装前准备。
- 通过准备软件包章节获取开发套件包 Ascend-cann-toolkit_*xxx*.run。
- 安装开发套件包前请确保安装目录可用空间大于 7G，如不满足请清理空间或更换安装目录。

安装步骤

1. 以软件包的安装用户登录安装环境。
   若安装 OS 依赖中安装依赖的用户为 root 用户，则软件包的安装用户可自行指定；若安装 OS 依赖中安装依赖的用户为非 root 用户，请确保软件包的安装用户与该用户保持一致。
2. 将获取到的开发套件包上传到安装环境任意路径（如“/home/package”）。
3. 进入软件包所在路径。
4. 增加对软件包的可执行权限。
   chmod +x *软件包名*.**run**

   说明

   其中*软件包名*.**run** 表示开发套件包 Ascend-cann-toolkit_*{version}*_linux-*{arch}*.run，请根据实际包名进行替换。
5. 执行如下命令校验软件包安装文件的一致性和完整性。
   ./*软件包名*.**run --check**
6. 执行以下命令安装软件（以下命令支持**--install-path=**<*path*>等参数，具体参数说明请参见参数说明）。
   ./*软件包名*.**run --install**

   说明

    - 开发套件包支持不同用户在同一开发环境安装，但安装版本必须保持一致，不同用户所属的属组必须和驱动运行用户所属属组相同；如果不同，请用户自行添加到驱动运行用户属组。
    - 如果以 root 用户安装，**不允许安装在非 root 用户目录下**。
    - 如果用户未指定安装路径，则软件会安装到默认路径下，默认安装路径如下。
        - root 用户：“/usr/local/Ascend”
        - 非 root 用户：“*${HOME}*/Ascend”
          其中${HOME}为当前用户目录。

  安装完成后，若显示如下信息，则说明软件安装成功：

  ```
  [INFO] xxx install success
  ```

  xxx 表示安装的实际软件包名。

CANN 软件提供进程级环境变量设置脚本，供用户在进程中引用，以自动完成环境变量设置。用户进程结束后自动失效。示例如下（以 root 用户默认安装路径为例）：

```
# 安装 toolkit 包时配置
source /usr/local/Ascend/ascend-toolkit/set_env.sh
```

用户也可以通过修改~/.bashrc 文件方式设置永久环境变量，操作如下：

1. 以运行用户在任意目录下执行 **vi ~/.bashrc** 命令，打开**.bashrc** 文件，在文件最后一行后面添加上述内容。
2. 执行**:wq!**命令保存文件并退出。
3. 执行 **source ~/.bashrc** 命令使其立即生效。

2.3 安装运行环境 nnrt

准备软件包：Ascend-cann-nnrt_5.1.RC1.alpha002_linux-x86_64.run

- 运行用户：实际运行推理业务的用户。
- 安装用户：实际安装软件包的用户。
    - 若使用 root 用户安装，支持所有用户运行相关业务。
    - 若使用非 root 用户安装，则安装及运行用户必须相同。
        - 已有非 root 用户，则无需再次创建。
        - 若想使用新的非 root 用户，则需要先创建该用户，请参见如下方法创建。

说明

- 如果安装驱动时未携带“--install-for-all”，并且 CANN 软件包运行用户为非 root，则该 CANN 软件包运行用户所属的属组必须和驱动运行用户所属属组相同；如果不同，请用户自行添加到驱动运行用户属组。
- 运行用户不建议为 root 用户属组，权限控制可能存在安全风险，请谨慎使用。

创建非 root 用户操作方法如下，如下命令请以 root 用户执行。

1. 创建非 root 用户。该用户需要和 Driver 的运行用户（默认为 HwHiAiUser）同属组，即如下 *usergroup* 字段需要替换成 Driver 的运行用户属组（默认为 HwHiAiUser）。
2. **groupadd** *usergroup*

   useradd -g *usergroup* **-d /home/***username* **-m** *username* **-s /bin/bash**
3. 设置非 root 用户密码。
   passwd *username*

说明

- 创建完运行用户后， 请勿关闭该用户的登录认证功能。
- 设置的口令需符合口令复杂度要求（请参见口令复杂度要求）。密码有效期为 90 天，您可以在/etc/login.defs 文件中修改有效期的天数，或者通过 chage 命令来设置用户的有效期，详情请参见设置用户有效期。

安装离线推理引擎包

前提条件

- 请参见准备安装及运行用户完成安装前准备。
- 在安装软件前请确保安装环境已安装推理卡或 AI 加速模块的驱动和固件。
- 通过准备软件包章节获取离线推理引擎包 nnrt。

安装步骤

获取并安装离线推理引擎包，详细安装步骤如下。

1. 以软件包的安装用户登录安装环境。
2. 将获取到的离线推理引擎包上传到安装环境任意路径（如“/home/package”）。
3. 进入软件包所在路径。
4. 增加对软件包的可执行权限。
   chmod +x *软件包名***.run**

   说明

   其中*软件包名***.run** 表示软件包名，例如离线推理引擎包 Ascend-cann-nnrt_*{version}*_linux-*{arch}*.run。请根据实际包名进行替换。

5. 执行如下命令校验软件包安装文件的一致性和完整性。
   ./*软件包名***.run --check**
6. 执行以下命令安装软件（以下命令支持**--install-path=**<*path*>等参数，具体参数说明请参见参数说明）。
   ./*软件包名***.run --install**

   说明

   - 离线推理引擎包支持不同用户在同一运行环境安装，但安装版本必须保持一致，不同用户所属的属组也必须和驱动运行用户所属属组相同；如果不同，请用户自行添加到驱动运行用户属组。
   - 如果以 root 用户安装，**不允许安装在非 root 用户目录下**。
   - 如果用户未指定安装路径，则软件会安装到默认路径下，默认安装路径如下。
     - root 用户：“/usr/local/Ascend”
     - 非 root 用户：“*${HOME}*/Ascend”
       其中${HOME}为当前用户目录。

   安装完成后，若显示如下信息，则说明软件安装成功：

   [INFO] *xxx* install success

   xxx 表示安装的实际软件包名。

3 合设 C++工程环境依赖安装

昇腾 AI 设备安装开发环境，同时将此环境作为运行环境的 samples 相关依赖安装 （推荐）

以下指导以普通用户 HwHiAiUser 安装 CANN 包为例说明；如果是 root 用户，请将安装准备中所有的${HOME}修改为/usr/local。

安装准备

请执行以下命令进行安装准备

```
# 以安装用户在任意目录下执行以下命令，打开.bashrc 文件。
vi ~/.bashrc
# 在文件最后一行后面添加如下内容。
export CPU_ARCH=`arch`
export THIRDPART_PATH=${HOME}/Ascend/thirdpart/${CPU_ARCH}  #代码编译时链接samples 所依赖的相关库文件
```

```
export LD_LIBRARY_PATH=${THIRDPART_PATH}/lib:$LD_LIBRARY_PATH  #运行时链接库文件
export INSTALL_DIR=${HOME}/Ascend/ascend-toolkit/latest #CANN 软件安装后的文件存储路径，根据安装目录自行修改
# 执行命令保存文件并退出。
:wq!
# 执行命令使其立即生效。
source ~/.bashrc
# 创建第 samples 相关依赖文件夹
mkdir -p ${THIRDPART_PATH}
# 下载源码并安装 git
cd ${HOME}
sudo apt-get install git
git clone https://gitee.com/ascend/samples.git
# 拷贝公共文件到 samples 相关依赖路径中
cp -r ${HOME}/samples/common ${THIRDPART_PATH}
```

如果是 200DK 场景还需要执行以下命令拷贝 media_mini 等 so 文件以及相关头文件，满足摄像头样例编译需要。

```
mkdir -p ${INSTALL_DIR}/driver
cp /usr/lib64/libmedia_mini.so ${INSTALL_DIR}/driver/
cp /usr/lib64/libslog.so ${INSTALL_DIR}/driver/
cp /usr/lib64/libc_sec.so ${INSTALL_DIR}/driver/
cp /usr/lib64/libmmpa.so ${INSTALL_DIR}/driver/
cp /usr/local/Ascend/include/peripheral_api.h ${INSTALL_DIR}/driver/
```

安装过程

安装 opencv

执行以下命令安装 opencv，如果代码中并没有使用 opencv 相关功能及函数，可以跳过此步骤。

```
sudo apt-get install libopencv-dev
```

安装 ffmpeg

执行以下命令源码安装 ffmpeg，由于 apt 安装的 ffmpeg 版本较低，所以需要源码安装。源码安装 ffmpeg 主要是为了 acllite 库的安装，如果代码中并没有使用 ffmpeg 或 acllite，可以跳过此步骤。

```
# 下载 ffmpeg
cd ${HOME}
wget http://www.ffmpeg.org/releases/ffmpeg-4.1.3.tar.gz --no-check-certificate
tar -zxvf ffmpeg-4.1.3.tar.gz
cd ffmpeg-4.1.3
# 安装 ffmpeg
./configure --enable-shared --enable-pic --enable-static --disable-x86asm --prefix=${THIRDPART_PATH}
make -j8
make install
```

安装 acllite 库

执行以下命令安装 acllite（注意，安装前需要先进行 ffmpeg 的源码安装）。如果代码中并没有使用 acllite 库相关功能及函数，可以跳过此步骤。

```
# 编译并安装 acllite
cd ${HOME}/samples/cplusplus/common/acllite/
make
make install
```

安装 presentagent

执行以下命令源码安装 protobuf 及 presentagent。如果代码中并没有使用 presentagent 相关功能及函数，可以跳过此步骤。

开始安装 protobuf 及 presentagent。

```
# 安装 protobuf 相关依赖
sudo apt-get install autoconf automake libtool
```

```
# 下载 protobuf 源码
cd ${HOME}
git clone -b 3.13.x https://gitee.com/mirrors/protobufsource.git protobuf
# 编译安装 protobuf
cd protobuf
./autogen.sh
./configure --prefix=${THIRDPART_PATH}
make clean
make -j8
sudo make install
# 进入 presentagent 源码目录并编译
cd ${HOME}/samples/cplusplus/common/presenteragent/proto
${THIRDPART_PATH}/bin/protoc presenter_message.proto --cpp_out=./
# 开始编译 presentagnet
cd ..
make -j8
make install
```