# Docker下ACT学习与推理安装文档及配置说明v1.0

# docker镜像包：

V1.0 仅带环境，需要挂载源码

V1.1 不需要挂载源码，需要提供数据。源码路径为/ACT

使用docker-compose.act.yml启动docker容器，参考5.2-5.3

# 安装Docker环境

docker安装参考 https://blog.csdn.net/educth/article/details/144138879

1. 卸载旧有版本（如有）

sudo apt-get remove docker docker-engine docker.io containerd runc

1. 更新 apt 索引并安装依赖

sudo apt-get update  
sudo apt-get install \  
 ca-certificates \  
 curl \  
 gnupg \  
 lsb-release

1. 添加 Docker 官方的 GPG 密钥（国内容易遇到网络问题，可以使用手动下载的方式）

#联网添加密钥  
sudo mkdir -p /etc/apt/keyrings  
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | \  
 sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg  
  
#概率遇到报错：gpg: no valid OpenPGP data found.执行测试：  
curl -I https://download.docker.com/linux/ubuntu/gpg  
#若无法收到200 OK，说明网络有问题。  
#手动下载并转换密钥  
sudo mkdir -p /etc/apt/keyrings  
curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o docker.gpg  
sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg docker.gpg  
  
#检查 docker.gpg 文件内容是否非空：  
ls -lh docker.gpg  
rm docker.gpg

1. 设置 Docker 仓库

echo \  
 "deb [arch=$(dpkg --print-architecture) \  
 signed-by=/etc/apt/keyrings/docker.gpg] \  
 https://download.docker.com/linux/ubuntu \  
 $(lsb\_release -cs) stable" | \  
 sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

1. 再次更新 apt 索引

sudo apt-get update

1. 安装 Docker Engine

sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

1. 验证 Docker 是否安装成功

docker -v

1. 将当前用户加入docker组：

sudo groupadd docker  
sudo usermod -aG docker $USER  
# 重新登录或重启宿主机后生效，或使用以下命令  
sudo chmod a+rw /var/run/docker.sock

1. （可选）docker换源、修改docker存储路径

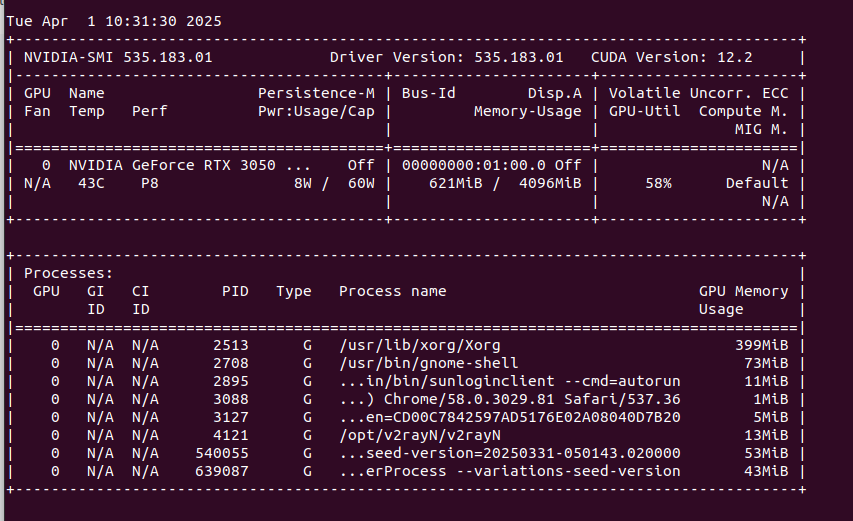
# 在 Docker 的配置文件 /etc/docker/daemon.json中加入以下内容，如果文件不存在或内容不正确，可以手动创建或修改。  
{  
 "registry-mirrors": [  
 "https://dockerproxy.com",  
 "https://docker.m.daocloud.io",  
 "https://cr.console.aliyun.com",  
 "https://ccr.ccs.tencentyun.com",  
 "https://hub-mirror.c.163.com",  
 "https://mirror.baidubce.com",  
 "https://docker.nju.edu.cn",  
 "https://docker.mirrors.sjtug.sjtu.edu.cn",  
 "https://registry.docker-cn.com"  
 ],  
 "runtimes": {  
 "nvidia": {  
 "path": "nvidia-container-runtime",  
 "runtimeArgs": []  
 }  
 },  
 "default-runtime": "nvidia"  
 "data-root":"/data/docker"  
}

# Docker 容器配置NVIDIA环境

Docker 容器配置NVIDIA环境参考 <https://blog.csdn.net/ydscc/article/details/146185450>

1. 检查 NVIDIA 驱动是否安装

#确保宿主机上已经安装了 NVIDIA 驱动，并且驱动版本与 CUDA 版本兼容  
nvidia-smi    
#如果输出显示了 GPU 信息和驱动版本，则说明驱动安装正常



#如果 nvidia-smi 命令找不到，可能需要安装 NVIDIA 驱动  
sudo apt update  
sudo apt install -y nvidia-driver  
#安装后需要然后重启宿主机。

1. 安装 NVIDIA Container Toolkit (Ubuntu / Debian 系统)

Docker 使用 NVIDIA Container Toolkit 来管理 GPU 设备。如果未安装或版本不兼容，可能会导致此问题。安装步骤如下：

# 添加 NVIDIA 容器工具包的仓库  
curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey | sudo gpg --dearmor -o /usr/share/keyrings/nvidia-container-toolkit-keyring.gpg \  
 && curl -s -L https://nvidia.github.io/libnvidia-container/stable/deb/nvidia-container-toolkit.list | \  
 sed 's#deb https://#deb [signed-by=/usr/share/keyrings/nvidia-container-toolkit-keyring.gpg] https://#g' | \  
 sudo tee /etc/apt/sources.list.d/nvidia-container-toolkit.list  
  
# 更新包列表并安装  
sudo apt-get update  
sudo apt-get install -y nvidia-container-toolkit

1. 更改Docker配置文件

# 检查 Docker 的配置文件 /etc/docker/daemon.json，确保其中包含以下内容，如果文件不存在或内容不正确，可以手动创建或修改：  
{  
 "registry-mirrors": [  
 "https://docker.m.daocloud.io",  
 "https://docker.imgdb.de",  
 "https://docker-0.unsee.tech",  
 "https://docker.hlmirror.com",  
 "https://cjie.eu.org"  
 ],  
 "runtimes": {  
 "nvidia": {  
 "path": "nvidia-container-runtime",  
 "runtimeArgs": []  
 }  
 },  
 "default-runtime": "nvidia"  
}

1. 重启 Docker 服务

sudo systemctl restart docker

1. 验证 Docker 是否识别 GPU

#方法一 如果输出为空，说明 Docker 未正确识别 NVIDIA 驱动  
docker info | grep -i nvidia

#方法二   
## 直接查找nvidia-container-cli，如果输出/usr/bin/nvidia-container-cli，则已安装  
which nvidia-container-cli  
## 运行下面命令测试，如果能看到设备信息，说明 nvidia-docker 支持正常。  
nvidia-container-cli info

#方法三 拉取官方镜像  
sudo docker run --rm --gpus all nvidia/cuda:12.2.0-base-ubuntu22.04 nvidia-smi

# 启用训练环境

1. 创建docker-compose.act.yml并准备好提供的docker image

version: "3.8"  
  
services:  
 act:  
 image: actimages:V1.0  
 container\_name: act  
 tty: false  
 stdin\_open: false  
 volumes:  
 - ../ACT:/ACT  
 - /dev/shm:/dev/shm  
 network\_mode: host  
 command: /bin/bash -c "sleep infinity"  
 restart: unless-stopped  
 runtime: nvidia  
 environment:  
 - NVIDIA\_VISIBLE\_DEVICES=all

1. 运行容器，并进入容器

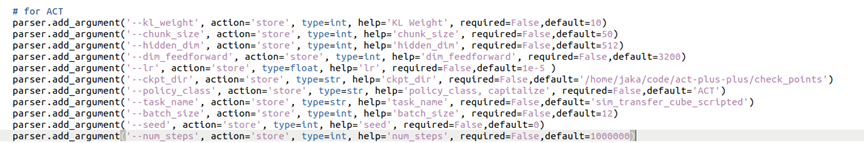
docker compose -f docker-compose.act.yml up -d  
docker exec -it ros2\_humble bash

1. 修改数据路径：打开/ACT/constans.py文件



首先定义DATA\_DIR的路径也就是数据集的路径，声明相机名称和数量（根据数据采集的结果来定义，名称与数据集里的相机名称要一致）。

1. 修改/ACT/train.py创建一个check\_points路径用于存储训练结果，并注释wandb相关内容



1. 运行train.py即可训练模型