

docker 报错:不能选择设备驱动 could not select device driver 的解决方法 (实测有效)

眼称: BooTurbo
 园龄: 4年7个月
 粉丝: 5
 关注: 0

3 4 10 11 17 18 24 25 31 1 7 8 7 8 9
14 15 16
21 22 23
28 29 30
4 5 6

搜索 找找看 谷歌搜索

Ubuntu(30) 计算机视觉(1 计算机视觉(18)
Windows系统(8)
Docker(7)
编辑器(6)
C++(4)
VSCode(3)
Python(3)
论文有关工具(3)
PyTorch(3)
更多

### 陪管档案

2023年5月(7) 2023年4月(6) 2023年2月(1) 2022年8月(3) 2022年7月(5) 2022年6月(3) 2022年5月(3) 2022年2月(1) 2022年2月(1) 2021年9月(1) 2020年12月(1 2020年11月(3 2020年8月(1) 2020年4月(5) 2019年12月(2)

**阅读排行榜**1. Ubuntu 20.04安装CUDA 11.0、cuDNN 8.
0.5、PyTorch 1.7.0(35231)
2. 安装系统时出现 Windows完括打开所需的
文件 C\Sources\install.wim 的解决办法(3320

8)
3. Win10系统修改主机名. 用户名标和密码。
以及C盘中的用户文件来名(20749)
4. 关于Protobuf摄错: If this call came from a pb2.py file, your generated code is out of date and must be repenerated with protoco = 3.190, (1355.99)
5. 训练种经网络的报错: can't convert cudaO device type tensor to numpy. Use Tensor. cup U1 to copy the tensor to host memory fir st.(13190)

**推荐排行榜**1. Ubuntu 20.4安英 UDA 11.0、cuDNN 8.
0.5、PyTorch 1.7.0(3)
2. Ubuntu 18.04安接 CUDA 10.1、cuDNN 7
6.5、PyTorch1.3(2)

B. PyTorch保存模型断点以及加载断点继续训练

4. Ubuntu18.04. 20.04 循环登陆无法讲入界 a. District 18.04、20.04 幅外登画元法亚八环 面,直至花屏,解决办法(实测有效)(1) 5. 神经网络训练时出现 无法获取维邦剪法的问题 tensorflow.python.framework.errors\_im pl.UnknownError: Failed to get convolution algorithm. This is probably because cuDN N failed to initialize(1)

是新平论

1. Re:关于protobuf报错: If this call came fr om a \_pb2.py file, your generated code is out of date and must be regenerated with protoc >= 3.19.0.

thanks

2. Re:安装系统时出现 Windows无法打开所需 的文件 C:\Sources\install.wim 的解决办法 太棒了,解决了问题

3. Re:Ubuntu 20.04安装CUDA 11.0、cuDNN 8.0.5, PyTorch 1.7.0

去掉sudo一样 4. Re:Ubuntu 20.04安装CUDA 11.0、cuDNN

你用管理员权限编辑 .bashrc 那保存以后那个 文件就变成管理员才能读写了啊

5. Re:神经网络训练时出现 无法获取卷积算法 的问题 tensorflow.python.framework.errors\_impl.UnknownError: Failed to get convolution algorithm. This is probably because cu DNN failed to initialize

Ubuntu安装完docker引擎后,在创建容器的时候指定 --gpus all,出现报错如下

:-\$ docker run --net=host -v \$HOME/W\_SiamFC\_imple:\$HOME/W\_SimaFC\_imple -it --gpus all --name SiamFC-repro nvidia/cuda:11.4.0

报错: docker: Error response from daemon: could not select device driver "" with capabilities: [[gpu]].

解决该问题还需要安装Nvidia-docker,本篇参照Nvidia官网。

NVIDIA Container Toolkit 在许多Linux发行版上都可用并且支持不同的容器引擎 (不只Docker)。

### 1、准备工作

开始前确保已安装NVIDIA驱动,

## (1) 平台要求:

## **Platform Requirements**

The list of prerequisites for running NVIDIA Container Toolkit is described below:

- 1. GNU/Linux x86 64 with kernel version > 3.10
- 2. Docker >= 19.03 (recommended, but some distributions may include older versions of Docker. The minimum supported
- 3. NVIDIA GPU with Architecture >= Kepler (or compute capability 3.0)
- 4. NVIDIA Linux drivers >= 418.81.07 (Note that older driver releases or branches are unsupported.)

Your driver version might limit your CUDA capabilities. Newer NVIDIA drivers are backwards-compatible with CUDA Toolk versions, but each new version of CUDA requires a minimum driver version. Running a CUDA container requires a machin with at least one CUDA-capable GPU and a driver compatible with the CUDA toolkit version you are using. The machine running the CUDA container only requires the NVIDIA driver, the CUDA toolkit doesn't have to be installed. The CUDA otes includes a table of the minimum driver and CUDA Toolkit versions.

- (2) Docker安装见上篇 Ubuntu 20.04安装Docker及相关设置
- 2. 安装NVIDIA Container Toolkit
- (1) 创建包仓库和GPG key:

```
$ distribution=$(. /etc/os-release;echo $ID$VERSION ID) \
        && curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey | sudo gpg --dearmor -o /usr/share/keyrings/nvidia-container-too
lkit-keyring.gpg \
        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
 :-$ distribution=$(./etc/os-release;echo $ID$VERSION_ID) \
&& curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey | sudo gpg --dearmor -o /usr/share/keyrings/nvidia-container-toolkit-keyr
ing.gpg \
> && curl -s -L https://nvidia.github.io/libnvidia-container/$distribution/libnvidia-container.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
[sudo] password for :
```

. deb [signed-by=/usr/share/keyrings/nvidia-container-toolkit-keyring.gpg] https://nvidia.github.io/libnvidia-container/stable/ubuntu18.04/\$(AR

(2) 更新包列表之后安装 nvidia-docker2 包(以及依赖)

```
sudo apt-get update
sudo apt-get install -y nvidia-docker2
```

```
-- $ sudo apt-get update
Get:1 https://nvidia.github.io/libnvidia-container/stable/ubuntul8.04/amd64 InRelease [1,484 B]
Hit:2 http://ftp.jaist.ac.jp/pub/Linux/ubuntu focal InRelease
Get:3 http://ftp.jaist.ac.jp/pub/Linux/ubuntu focal-updates InRelease [114 kB]
Hit:4 https://download.docker.com/linux/ubuntu focal InRelease
Get:5 https://deb.opera.com/opera-stable stable InRelease [2,590 B
Get:6 http://dl.google.com/linux/chrome/deb stable InRelease [1,811 B]
Get:7 http://ftp.jaist.ac.jp/pub/Linux/ubuntu focal-backports InRelease [108 kB]
Get:8 http://ftp.jaist.ac.jp/pub/Linux/ubuntu focal-security InRelease [114 kB]
Get:9 https://nvidia.github.io/libnvidia-container/stable/ubuntu18.04/amd64 Packages [18.7 kB]
```

```
:-$ sudo apt-get install -y nvidia-docker2

Reading package lists... Done

Building dependency tree

Reading state information... Done

The following packages were automatically installed and are no longer required:
    gconf-service gconf-service-backend gconf2 gconf2-common libart-2.0-2 libbonobo2-0 libbonobo2-common libbonoboui2-0 libbonoboui2-common libfprint-2-tod1 libfwupdplugin1 libgconf-2-4 libglade2-0 libgnome-2-0 libgnome-keyring-common libgnome-keyring0 libgnome2-canvas-perl libgnome2-common libgnomew2-gconf-perl libgnome2-perl libgnome2-vfs-perl libgnome2-wnck-perl libgnomecanvas2-0 libgnomecanvas2-common libgnomewfs2-extra libgoo-canvas-perl libgocanvas-common libgtk2-appindicator-perl libgtk2-imageview-perl libgtk2-unique-perl libgtkimageview0 libllvm10 libllvm10:i386 liblvm11 liblvm11:i386 libnvidia-common-460 libnvidia-compute-460 libnvidia-compute-460:i386 libnvidia-decode-460 libnvidia-decode-460 libnvidia-decode-460 libnvidia-gl-460 libnvidia-fbc1-460 libnvidia-fbc1-460:i386 libnvidia-gl-460 libnvidia-fbc1-460 libnvidia-fbc1-460:i386 libnvidia-gl-460 libnvidia-fbc1-460 libnvidia-fbc1-460:i386 libnvidia-decode-460 libnvidia-fbc1-460 libnvidia-fbc1-460:i386 libnvidia-decode-460 libnvidia-dec
                                                                                                                                                                                                        :- $ sudo apt-get install -y nvidia-docker2
```

(3) 重启Docker daemon 来完成设定默认运行时后的安装。

sudo systemctl restart docker

:-\$ sudo systemctl restart docker

(4) 通过运行一个基础的CUDA container测试安装是否有效。

sudo docker run --rm --gpus all nvidia/cuda:11.0.3-base-ubuntu20.04 nvidia-smi

```
得到如下输出表明成功。
:-$ sudo docker run --rm --gpus all nvidia/cuda:11.0.3-base-ubuntu20.04 nvidia-smi
Unable to find image 'nvidia/cuda:11.0.3-base-ubuntu20.04' locally
11.0.3-base-ubuntu20.04: Pulling from nvidia/cuda
d5fd17ec1767: Already exists
ea7643e57386: Already exists
622a04926279: Already exists
18fcb7509e42: Already exists
21e5db7c1fa2: Already exists
 digest: sha256:a76eddd79a7ec423decf72eaaa348a69a98362615aa10565011f439940142471
Status: Downloaded newer image for nvidia/cuda:11.0.3-base-ubuntu20.04
 ri May 27 09:27:54 2022
  NVIDIA-SMI 470.129.06 Driver Version: 470.129.06 CUDA Version: 11.4
                                                  Disp.A | Volatile Uncorr. ECC
Memory-Usage | GPU-Util Compute M.
                      Persistence-M| Bus-Id
  GPU Name
  Fan Temp Perf Pwr:Usage/Cap
                                                                                     MIG M.
       N/A
                                                                                     Default
                                                                                         N/A
   GPU GI
                                                                                  GPU Memory
                            PID Type Process name
          ID
                                                                                  Usage
    ______
```

3、最后再次创建容器指定--gpus all ,问题解决,容器创建成功。

```
:-$ docker run --net=host -v $HOME/W_SiamFC_imple:$HOME/W_SimaFC_imple -it --gpus all --name SiamFC-repro nvidia/cuda:11.4.0
-cudnn8-devel-ubuntu20.04 /bin/bash
root@ _____:/# ls -la
total 80
root@l
total 80
```

# Enjoy it!

标签: <u>Docker</u>, <u>Ubuntu</u>, <u>NVIDIA Container Toolkit</u>















« 上一篇: Ubuntu 20.04安装Docker及相关设置

posted @ 2022-05-27 19:27 BooTurbo 阅读(4480) 评论(0) 编辑 收藏 举报 刷新评论 刷新页面 返回顶部

0 0 €推荐

□ 登录后才能查看或发表评论,立即 登录 或者 逛逛 博客园首页

【推荐】基于阿里云免费资源,创建自己的独立博客,赢AirPods Pro礼品 【推荐】阿里云-云服务器省钱攻略: 五种权益,限时发放,不容错过 【推荐】天翼云818全民上云季,爆款云主机2核2G三个月仅47.4元 【推荐】SQL专家云: SQL Server 数据库可视化、智能化运维平台

編輯機等:
他用 MediatR 实现 CQRS
- 记一次、NET採股关系统 非托管泄露分析
- 订规则 另循环、扣不测的广告余额,记一个生产问题
- .NET 对象的分符布局
- 记录一次朝决数据库连接池连接泄露BUG

.NET Core 多线 (5) 常见性能问题 一种基于ChatGPT的高效吃瓜方式的探索和研究。 WPF 入门笔记 - 07 - MVVM示例 使用MediatR实现CQRS

