

安装ssh:

安装ssh: root@ubuntu: apt-get install openssh-server

安装后需要启动ssh

root@ubuntu: /etc/init.d/ssh restart

允许root登录需要修改配置信息:

root@ubuntu: vi /etc/ssh/sshd_config

```
SyslogFacility AUTH
LogLevel INFO

# Authentication:
LoginGraceTime 120
PermitRootLogin prohibit-password
StrictModes yes

RSAAuthentication yes
PubkeyAuthentication yes
#AuthorizedKeysFile      %h/.ssh/authorized_keys
```

修改为:

```
LogLevel INFO

# Authentication:
LoginGraceTime 120
PermitRootLogin yes
StrictModes yes

RSAAuthentication yes
PubkeyAuthentication yes
#AuthorizedKeysFile      %h/.ssh/authoriz

# Don't read the user's ~/.rhosts and ~/.shosts
```

修改保存后再重启ssh

root@ubuntu: /etc/init.d/ssh restart

Ubuntu 平台的防火墙关闭命令

1、关闭防火墙:

#sudo ufw disable

关闭了防火墙,并取消了开机自启动。

2、查看防火墙状态:

#sudo ufw status

3、开启防火墙:

#sudo ufw enable

挂载硬盘,使用DISKS,修改挂载点,修改权限多用户可访问。

权限里面加user项,任何用户可以自动挂载。

/mnt/datadisk/

/mnt/codedisk/

使用链接方式: ln -s /mnt/datadisk/ /home/wangwei/

Ubuntu下samba配置和使用

首先需要安装samba程序,部分Ubuntu镜像已经自带;执行如下命令即可

sudo apt-get install samba

```
sudo apt-get install smbclient
```

第二步需要配置samba服务器，使用如下命令打开samba的配置文件

```
sudo vi /etc/samba/smb.conf
```

加入配置信息

```
[zhoubin]
comment = zhoubin's directory
path = /home/zhoubin
writable = yes
valid users = zhoubin
```

这其中比较重要的参数为path和valid users, path指定了samba服务器的根目录，可以任意指定合法路径；valid users表示可以访问samba服务器的合法用户；用户在配置时注意需要将路径改为自己需要的路径。

由于在上文中配置的samba服务器根文件路径在我的Ubuntu上还不存在，所以我需要在对应路径下创建文件夹，并修改其权限

```
mkdir share
chmod 777 share
```

接下来需要为samba服务器添加用户了，由于我的配置中使用的是名为"zhang"的用户(该用户其实就是我的登录用户)，所以我需要通过如下命令来为samba添加此用户并设置密码

```
sudo smbpasswd -a zhang
```

配置完成后需要重启samba服务器，使用如下命令重启samba服务器

```
sudo /etc/init.d/samba restart
```

Ubuntu16.04下安装显卡驱动，cuda，cudnn

下载显卡驱动

首先到nvidia官网下载好对应显卡驱动，进入网站<https://www.nvidia.cn/Download/Find.aspx?lang=cn>选好自己对应显卡型号和对应操作系统及位数，因为NVIDIA显卡驱动版本是向下兼容的，如果不能找到CUDA对应的旧驱动版本可以安装比默认版本略高的版本。

禁用nouveau

Ubuntu16.04默认安装了显卡驱动nouveau，首先需要禁用nouveau，不然会造成驱动间的冲突

1.编辑blacklist.conf

输入命令 `sudo vim /etc/modprobe.d/blacklist.conf`

在文件末尾插入以下内容：

```
blacklist nouveau
options nouveau modeset=0
```

2.更新系统

`sudo update-initramfs -u` 更新当前最新的

`sudo update-initramfs -c -k 4.4.0-31-generic` 更新指定的

3.重启系统

`sudo reboot`

4.验证nouveau是否已禁用

`lsmod | grep nouveau`

若没有任何信息输出，则证明了nouveau已禁用

进入显卡驱动安装

按下ctrl+alt+F1进入命令行模式

1.关闭图形界面

```
sudo service lightdm stop
```

2.卸载原有驱动(若没有安装NVIDIA驱动可跳过)

```
sudo apt-get remove nvidia-*
```

3.安装驱动

```
sudo chmod a+x NVIDIA-Linux-x86_64-396.18.run
```

```
sudo ./NVIDIA-Linux-x86_64-396.18.run -no-x-check -no-nouveau-check -no-opengl-files //一定要禁用opengl，一定要禁用opengl，一定要禁用opengl，重要的事情说三遍；禁用才不会出现循环登录属性：
```

-no-x-check：安装驱动时关闭X检查服务(可选)

-no-nouveau-check：安装驱动时禁用nouveau检查(可选)

-no-opengl-files：不安装OpenGL文件

4.安装驱动时的选项

一.The distribution-provided pre-install script failed! Are you sure you want to continue?

选择 continue to install。

二.Would you like to register the kernel module sources with DKMS? This will allow DKMS to automatically build a new module, if you install a different kernel later?

是否安装DMS，选择 No。

三.Nvidia's 32-bit compatibility libraries?

是否兼容32位库，选择 No。

四.Would you like to run the nvidia-xconfig utility to automatically update your x configuration so that the NVIDIA x driver will be used when you restart x? Any pre-existing x config file will be backed up.

选择 Yes

提示安装完成之后

5.输入命令挂载驱动

```
modprobe nvidia
```

6.验证NVIDIA驱动安装成功

输入命令

```
nvidia-smi
```

出现上图，则安装显卡驱动成功

7.重新打开图形界面

```
sudo service lightdm restart
```

8.重新进入界面

ctrl+alt+F7进入桌面，输入密码，若没有循环登录则证明安装驱动完全了，不然需要卸载驱动重新上述步骤

安装CUDA

进入<https://developer.nvidia.com/cuda-toolkit-archive>下载对应驱动的CUDA版本

安装脚本

```
sudo bash cuda_9.0.176_384.81_linux.run --no-opengl-libs
```

```
sudo chmod +x ~/cuda_10.0.130_410.48_linux.run
sudo ./cuda_10.0.130_410.48_linux.run
# 稍后会出现很多提示信息，可以长按空格键直接跳过
```

安装选项

Do you accept the previously read EULA?
accept/decline/quit: accept

Install NVIDIA Accelerated Graphics Driver for Linux-x86_64 410.48?
(y)es/(n)o/(q)uit: no

Install the CUDA 10.0 Toolkit?
(y)es/(n)o/(q)uit: yes

Enter Toolkit Location
[default is /usr/local/cuda-10.0]:

Do you want to install a symbolic link at /usr/local/cuda?
(y)es/(n)o/(q)uit: no

Install the CUDA 10.0 Samples?
(y)es/(n)o/(q)uit: no

安装CUDNN

进入网站<https://developer.nvidia.com/rdp/cudnn-archive>下载相应版本的CUDNN

进入CUDNN文件夹，解压下载好的cudnn包

```
sudo tar -xzf cudnn-10.0-linux-x64-v7.6.4.38.tgz
```

```
sudo cp cuda/include/cudnn.h /usr/local/cuda-10.0/include
```

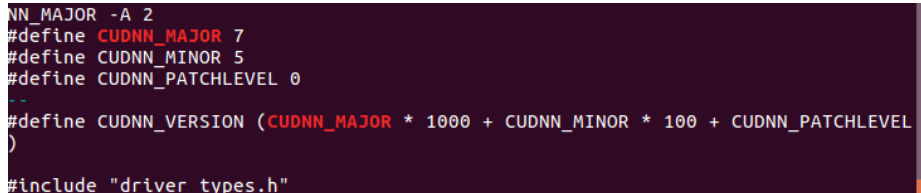
```
sudo cp cuda/lib64/libcudnn* /usr/local/cuda-10.0/lib64
```

```
sudo chmod a+r /usr/local/cuda-10.0/include/cudnn.h
```

```
sudo chmod a+r /usr/local/cuda-10.0/lib64/libcudnn*
```

查看cudnn版本

```
cat /usr/local/cuda-10.0/include/cudnn.h | grep CUDNN_MAJOR -A 2
```



出现上图则证明CUDNN安装成功

安装anaconda3

<https://blog.csdn.net/AlphaWun/article/details/90229812>

首先是阅读许可申明，可以一直按Enter键，然后问是否同意许可，输入yes，接着问Anaconda安装的路径，直接按Enter键会安装到默认的路径，也就是当前用户的目录下，这样也就只有当前用户可以使用Anaconda，要多用户共享安装，选择其他路径。在linux下安装第三方多用户共享使用的软件一般都安装在 /usr/local 目录下，输入路径后回车

```

Do you accept the license terms? [yes|no]
[no] >>>
Please answer 'yes' or 'no':
>>> yes

Anaconda3 will now be installed into this location:
/home/ubuntu/anaconda3

- Press ENTER to confirm the location
- Press CTRL-C to abort the installation
- Or specify a different location below
[/home/ubuntu/anaconda3] >>> /usr/local/anaconda3

```

输入要安装的路径

安装完后在/etc/profile文件中配置环境变量，在/etc/profile文件末尾加入下面命令

```
export PATH=/usr/local/anaconda3/bin:$PATH
```

修改完这个文件使用 以下命令在不用重启系统的情况下使修改的内容生效

```
source /etc/profile
```

配好环境变量后，查看是否安装成功

```

wangql@lab1206:/home/data$ python
Python 3.7.3 (default, Mar 27 2019, 22:11:17)
[GCC 7.3.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>

```

如上图，已成功安装Anaconda Python3.7环境。

使用其他用户登录Ubuntu，同样可以看到上面的效果。这样，多用户就可以共享使用Anaconda了。

2、在 Anaconda下创建多用户共享的tensorflow环境

安装好Anaconda环境后，接下来创建一个新的tensorflow gpu环境。

创建多个用户共享使用的tensorflow环境，要在root用户下创建环境。如果在普通用户下创建，那只有该用户可以使用环境。在root用户下输入以下命令创建tensorflow gpu环境

```
# conda create --name tensorflow_gpu python=3.7
```

输入上面的命令可能会提示conda：未找到命名。这是，输入 export

PATH=/usr/local/anaconda3/bin:\$PATH 即可，然后继续输入上面一行命名就可以创建环境了。

```

root@lab1206:/home/wangql# conda create --name tensorflow_gpu python=3.7
conda: 未找到命令
root@lab1206:/home/wangql# export PATH=/usr/local/anaconda3/bin:$PATH
root@lab1206:/home/wangql#

```

创建完环境后，Ubuntu其他用户就看到看到刚才创建的tensorflow环境，多用户也就可以共享该环境了，不用每个用户都创建一遍。

```

(base) ubuntu@lab1206:~$ conda info --env
WARNING: The conda.compat module is deprecated and will be removed in a future release.
# conda environments:
#
base                  * /usr/local/anaconda3
tensorflow_gpu        /usr/local/anaconda3/envs/tensorflow_gpu

```

这样，多用户共享的环境就装好了，接下来在该环境中安装tensorflow-gpu。

2.2、安装tensorflow-gpu

添加清华镜像，加快下载速度。输入以下命令

```
# conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkg/free/
```

```
# conda config --set show_channel_urls yes
```

找不到命令可以输入环境

```
export PATH=/usr/local/anaconda3/bin:$PATH
```

进入刚才创建的tensorflow-gpu环境

```
# source activate tensorflow_gpu (linux下+source, windows下无需+source)
```

```
source deactivate
```

```
conda info --envs
```

安装tensorflow-gpu

```
# conda install tensorflow-gpu
```

安装keras-gpu

```
# conda install keras-gpu
```

ubuntu16.04 查看内核，升级内核，删除内核，切换内核

<https://blog.csdn.net/u011304615/article/details/70919711>

1:查看内核列表

```
sudo dpkg --get-selections |grep linux-image
```

2:查看当前使用的内核

```
uname -r
```

4.4.0-21-generic

3:升级/安装内核

```
sudo apt-get install linux-image-4.4.0-75-generic
```

3:删除内核

tip：删除当前版本重启会使用低一级的已安装内核，如果是最后一个内核版本删除之后重启会进入BIOS界面

```
sudo apt-get remove linux-image-4.4.0-75-generic
```

4:切换内核

<http://blog.csdn.net/u011304615/article/details/70920171>

1.该命令显示内核的启动顺序

```
zgw@zgw-ThinkPad:~$ grep menuentry /boot/grub/grub.cfg
```

```
修改/etc/default/grub
```

```
GRUB_DEFAULT="Advantce Ubuntu>Ubuntu, with Linux 3.2.0-23-generic"
```

然后使用命令sudo update-grub 根据warning命令进行修改。

docker和nvidia-docker安装和环境设置。

https://blog.csdn.net/weixin_42749767/article/details/82934294

安装docker

GPU driver安装

nvidia官网下载安装对应型号的显卡驱动：链接

如果安装成功，在终端中输入lspci | grep -i nvidia，会显示自己的NVIDIA GPU版本信息

CUDA安装

实验室服务器是ubuntu 18.04版本，可以直接sudo apt install nvidia-cuda-toolkit安装

docker安装

- 安装必要的一些系统工具
- sudo apt-get -y install apt-transport-https ca-certificates curl software-properties-common
- 安装GPG证书
- curl -fsSL http://mirrors.aliyun.com/docker-ce/linux/ubuntu/gpg | sudo apt-key add -
- 写入软件源信息
- sudo add-apt-repository "deb [arch=amd64] http://mirrors.aliyun.com/docker-ce/linux/ubuntu \$(lsb_release -cs) stable"

- 更新并安装 docker-ce

sudo apt-get -y update

sudo apt-get -y install docker-ce

- sudo service docker status #或者sudo systemctl status
- sudo docker run hello-world #测试Docker安装是否成功

1. nvidia-docker安装

- 如果之前安装过docker1.0版本，需要先删掉该版本和之前创建的容器
- docker volume ls -q -f driver=nvidia-docker | xargs -r -I{} -n1 docker ps -q -a -f volume={} | xargs -r docker rm -f
- sudo apt-get purge -y nvidia-docker
- 添加代码仓库
- curl -s -L https://nvidia.github.io/nvidia-docker/gpgkey | \
- sudo apt-key add -
- distribution=\$(. /etc/os-release;echo \$ID\$VERSION_ID)
- curl -s -L https://nvidia.github.io/nvidia-docker/\$distribution/nvidia-docker.list | \
- sudo tee /etc/apt/sources.list.d/nvidia-docker.list
- sudo apt-get update
- 安装docker 2
- sudo apt-get install -y nvidia-docker2
- sudo pkill -SIGHUP dockerd
- 测试
- docker run --runtime=nvidia --rm nvidia/cuda:10.0-base nvidia-smi
- 安装过程中遇到的问题

用nvidia docker进行训练

1. 拉取镜像（这里拉取了阿里云的一个镜像，里面自带了编译好的caffe，不过由于在实验室的宿主机上已经有编译好的caffe，可以直接将宿主机的目录挂载到容器中，这个后面有说）

2. sudo nvidia-docker pull registry.cn-hangzhou.aliyuncs.com/docker_learning_aliyun/caffe:v1

1. 查看拉取的镜像信息

2. sudo nvidia-docker images

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
nvidia/cuda	9.0-base	825690924f78	2 weeks ago	133MB
hello-world	latest	4ab4c602aa5e	4 weeks ago	1.84kB
registry.cn-hangzhou.aliyuncs.com/docker_learning_aliyun/caffe	v1	4e33692cc981	17 months ago	3.89GB

1. 利用拉取的镜像启动容器，并把宿主机的caffe目录挂载到容器上

2. `sudo nvidia-docker run -it -v $CAFFE_ROOT:/workspace 4e33(镜像id前4位即可) /bin/bash`

这样就启动了一个容器，并且把caffe目录挂载到了容器的/workspace下。这样操作的好处是，训练完的数据可以直接存放在宿主机，省略了从容器中拷贝的繁琐步骤。

Ubuntu16.04 添加 Docker用户组

将用户添加到docker用户组就不用每次都 `sudo`了。

首先创建用户组

```
sudo groupadd docker
```

将用户加如组

```
sudo gpasswd -a ${USER} docker //sudo gpasswd -a wx docker
```

重启服务

```
sudo service docker restart
```

切换当前会话到新组

```
newgrp - docker
```

```
*****
```

```
export PATH=/usr/local/cuda-10.0/bin:$PATH
```

```
export LD_LIBRARY_PATH=/usr/local/cuda-10.0/lib64:$LD_LIBRARY_PATH
```

```
export PATH=/usr/local/anaconda3/bin:$PATH
```

```
SWIG_PATH=/usr/local/share/swig/3.0.12
```

如何安装llvmlite llvm3.9

```
pip3 install llvmlite==0.16.0
```

<https://blog.csdn.net/zhangpeterx/article/details/92851562>

https://blog.csdn.net/melissa_cjt/article/details/74995527

1.依赖库安装

```
sudo apt-get install build-essential curl libcap-dev git cmake libncurses5-dev python-minimal
```

```
python-pip unzip zlib1g-dev
```

```
sudo apt-get update
```

```
sudo apt-get install clang-3.9 lldb-3.9
```

```
export PATH="/usr/lib/llvm-3.9/bin:$PATH"
```

报错是因为找不到llvm-config这个命令：

```
alias llvm-config="llvm-config-3.9"
```

```
export LLVM_CONFIG="/usr/bin/llvm-config-3.9"
```

python相关命令：

```
sudo apt install python-pip
```

```
pip install numpy -i https://mirrors.aliyun.com/pypi/simple/
```

```
pip install virtualenv==16.7.2 -i https://mirrors.aliyun.com/pypi/simple/
```

```
pip3 install --upgrade pip -i https://mirrors.aliyun.com/pypi/simple/
```

```
virtualenv -p /usr/bin/python3.5 venv
```

```
pip freeze > 1.txt
```

- 如果需要卸载1.txt中的所有包的命令为：

```
pip install -r 1.txt
```


- 如果需要安装1.txt中的所有包的命令为：

```
pip uninstall -r 1.txt
```