**OpenFace安装教程**

1. 安装Ubuntu系统：

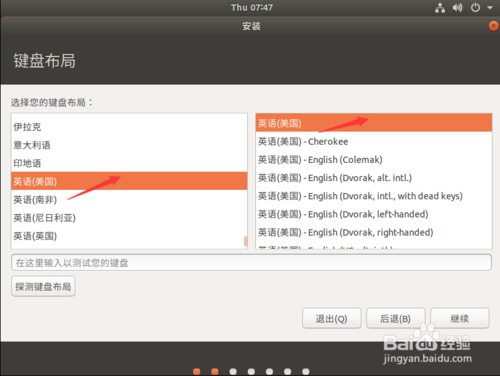
1.设置开机使用U盘启动，启动后会出现系统logo页面。



2.语言选择，直接选择英文，然后点击安装Ubuntu



3.键盘布局选择英语，然后点击继续。



4.选择正常安装，下面的可选安装根据自己情况选择。



5.这里安装类型选择“其他选项”。点击现在安装继续下一步。



6.在弹跳出的页面选择继续。



7.区域设置，默认即可。



8.为Ubuntu设置用户，设置完成后点击继续即可。



9.进入安装系统页面大概需要15-20分钟。耐心等待即可。





10.安装完毕后，点击重启，系统自动重启。



1. 安装darknet

1. 下载darknet的安装包：

$ git clone https://github.com/pjreddie/darknet

2. 进入darknet的安装目录下，并对其进行编译：

$ cd darknet

$ make

3. 下载weight文件：

$ wget https://pjreddie.com/media/files/yolov3.weights

4. 运行探测器：

$./darknet detect cfg/yolov3.cfg yolov3.weights data/dog.jpg

出现如下结果，即代表安装成功。

layer filters size input output

0 conv 32 3 x 3 / 1 416 x 416 x 3 -> 416 x 416 x 32 0.299 BFLOPs

1 conv 64 3 x 3 / 2 416 x 416 x 32 -> 208 x 208 x 64 1.595 BFLOPs

.......

105 conv 255 1 x 1 / 1 52 x 52 x 256 -> 52 x 52 x 255 0.353 BFLOPs

106 detection

truth\_thresh: Using default '1.000000'

Loading weights from yolov3.weights...Done!

data/dog.jpg: Predicted in 0.029329 seconds.

dog: 99%

truck: 93%

bicycle: 99%

1. 安装Openface，步骤如下：

1. python

　　Ubuntu 16.04桌面版自带python

2. git

$ sudo apt-get install git

3. 编译工具CMake

$ sudo apt-get install cmake

4. C++标准库安装

$ sudo apt-get install libboost-dev

$ sudo apt-get install libboost-python-dev

5. 下载OpenFace代码

$ git clone https://github.com/cmusatyalab/openface.git

6. OpenCV安装

$ sudo apt-get install libopencv-dev

$ sudo apt-get install python-opencv

7. 安装包管理工具pip

$ sudo apt install python-pip

更新pip,按上面安装不知道为什么是旧的版本，可能影响下面的操作。

$ pip install --upgrade pip

8. 安装依赖的 PYTHON库

$ cd openface

$ sudo pip install -r requirements.txt

$ sudo pip install dlib

$ sudo pip install matplotlib

9. 安装 luarocks—Lua 包管理器,提供一个命令行的方式来管理 Lua 包依赖、安装第三方 Lua 包等功能

$ sudo apt-get install luarocks

10. 安装 TORCH—科学计算框架，支持机器学习算法

$ git clone https://github.com/torch/distro.git ~/torch --recursive

$ cd torch

$ bash install-deps

如果出现”替换成software-properties-common“的错误

就用vim将“python-software-properties”替换为“software-properties-common”大约在178行

$ ./install.sh

如果出现以下错误

make[2]:\*\*\* [lib/THC/CMakeFiles/THC.dir/THC\_generated\_THCTensorMath.cu.o] Error 1

make[2]: \*\*\* Waiting for unfinished jobs....

^Clib/THC/CMakeFiles/THC.dir/build.make:105: recipe for target 'lib/THC/CMakeFiles/THC.dir/THC\_generated\_THCTensorCopy.cu.o' failed

make[2]:\*\* [lib/THC/CMakeFiles/THC.dir/THC\_generated\_THCTensorCopy.cu.o] Interrupt

lib/THC/CMakeFiles/THC.dir/build.make:140: recipe for target 'lib/THC/CMakeFiles/THC.dir/THC\_generated\_THCTensorMathPairwise.cu.o' failed

make[2]: \*\*\* [lib/THC/CMakeFiles/THC.dir/THC\_generated\_THCTensorMathPairwise.cu.o] Interrupt

CMakeFiles/Makefile2:172: recipe for target 'lib/THC/CMakeFiles/THC.dir/all' failed

make[1]: \*\*\* [lib/THC/CMakeFiles/THC.dir/all] Interrupt

Makefile:127: recipe for target 'all' failed

make: \*\*\* [all] Interrupt

先运行以下命令

export TORCH\_NVCC\_FLAGS="-D\_\_CUDA\_NO\_HALF\_OPERATORS\_\_"

再

./clean.sh

提示权限不足时，前面加sudo

再执行

./install.sh

使 torch7 设置的刚刚的环境变量生效

$ source ~/.bashrc

上一个错误链接：https://m.oldpan.me/archives/torch-install

11. 安装依赖的 LUA库

$ luarocks install dpnn

　 下面的为选装，有些函数或方法可能会用到

$ luarocks install image

$ luarocks install nn

$ luarocks install graphicsmagick

$ luarocks install torchx

$ luarocks install csvigo

12. 编译OpenFace代码

$ python setup.py build

$ sudo python setup.py install

13. 下载预训练后的数据

$ sh models/get-models.sh

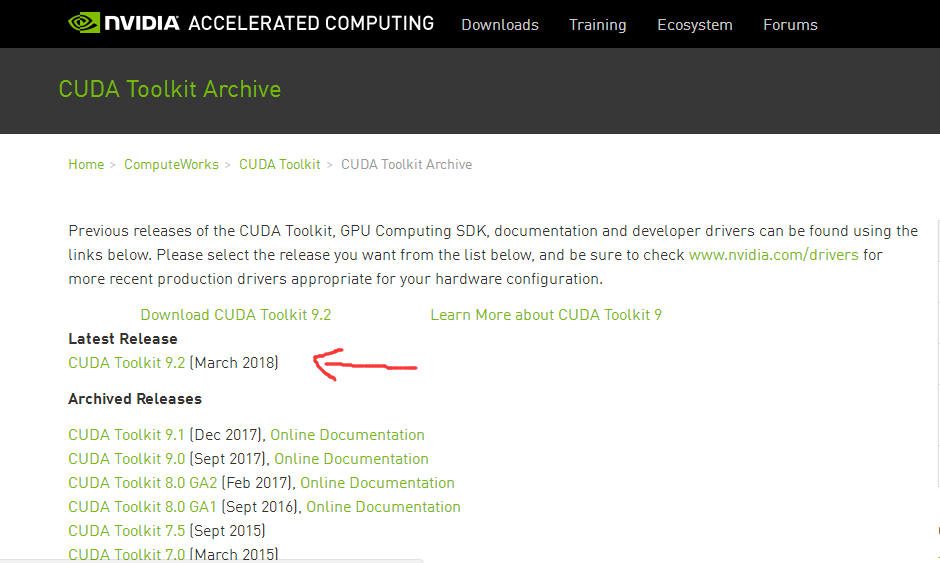
$ wget https://storage.cmusatyalab.org/openface-models/nn4.v1.t7 -O models/openface/nn4.v1.t7

14. 可以用compare.py(demo文件夹中)给出的示例检测两张脸的相近程度。

$ python demos/compare.py {3.jpg\*,4.jpg\*}

1. 安装CUDA

1. 先在官网https://developer.nvidia.com/cuda-toolkit-archive上下载最新的CUDA安装文件，例如cuda\_9.2.148\_396.37\_linux.run。



2. 以vim的方式打开文件blacklist.conf，命令如下：

$ sudo vim /etc/modprobe.d/blacklist.conf

将下面几行写到文件中：

blacklist nouveau

blacklist lbm-nouveau

options nouveau modeset=0

alias nouveau off

alias lbm-nouveau off

3. 调用指令禁止nouveau。

$ echo options nouveau modeset=0 | sudo tee -a /etc/modprobe.d/nouveau-kms.conf

4. 建立新的内核

$ sudo update-initramfs -u

5. 重启电脑，在登录界面直接按Ctrl+Alt+F1进入控制台，输入用户名和密码登陆后，执行如下命令，关闭图形界面。

$ sudo service gdm stop

$ sudo service gdm3 stop

6. 切换到CUDA安装文件的目录，进行安装。

$ cd cuda

$ sudo sh cuda\_9.2.148\_396.37\_linux.run

这里会出现More（0%），意思是你要阅读这些协议，连续按空格键可跳过，按照提示输入accept，第一个安装的时候，是安装图形驱动的，选择yes，安装英伟达驱动。后面也全选yes，并建议选择默认安装路径。

可能出现的问题有：

(1) cuda:Missing recommended library: libGLU.so,libX11.so,libXi.so,libXmu.so

解决方案：原因是缺少相关的依赖库,安装相应库就解决了：

$ sudo apt-get install freeglut3-dev build-essential libx11-devlibxmu-dev libxi-dev libgl1-mesa-glx libglu1-mesa libglu1-mesa-dev

再次安装,就不再提示了。

7. 环境变量的配置

用gedit命令打开bashrc文件

$ sudo gedit ~./bashrc

将以下内容写到文件尾部。

export PATH=/usr/local/cuda-8.0/bin${PATH:+:${PATH}}

export LD\_LIBRARY\_PATH=/usr/local/cuda-8.0/lib64${LD\_LIBRARY\_PATH:+:${LD\_LIBRARY\_PATH}}

保存后，更新文件

$ source ~/.bashrc

1. 配置cuDNN

1. 先在官网https://developer.nvidia.com/rdp/cudnn-archive上下载最新的cuDNN文件,例如cudnn-9.2-linux-x64-v7.1，并解压。

2. 进入解压后的目录后，执行如下命令：

$ cd cuda/include

$ sudo cp cudnn.h /usr/local/cuda/include #复制头文件

3. 再将进入lib64目录下的动态文件进行复制和链接：

$ cd ..

$ cd lib64

$ sudo cp lib\* /usr/local/cuda/lib64/ #复制动态链接库

$ cd /usr/local/cuda/lib64/

$ sudo rm -rf libcudnn.so libcudnn.so.5 #删除原有动态文件

$ sudo ln -s libcudnn.so.5.1.10 libcudnn.so.5 #生成软衔接

$ sudo ln -s libcudnn.so.5 libcudnn.so #生成软链接

$ sudo ldconfig #使配置生效

4. 测试CUDA的samples

$ cd /usr/local/cuda-8.0/samples/1\_Utilities/deviceQuery

$ sudo make

$ sudo ./deviceQuery

如果显示一些关于GPU的信息，怎说明安装成功。

1. 使用nginx+nginx-rtmp-module+ffmpeg搭建流媒体服务器

1、下载nginx，地址http://nginx.org/en/download.html，选择最新版本下载、解压。目前最新版是nginx-1.15.2。

2、为了增加对rtmp的支持，下载nginx-rtmp-module，地址：https://github.com/arut/nginx-rtmp-module#example-nginxconf，这个是个开源项目。解压后，为了和我在网上看到的教程同步，我改了文件夹名字，将其改成了nginx-rtmp-module。然后将其放到/home/user/目录下。

3、进入到nginx-1.15.2文件夹目录下，执行如下命令：

$ ./configure --prefix=/usr/local/nginx --add-module=/home/user/nginx-rtmp-module --with-http\_ssl\_module

4、待上述命令执行完成之后，执行如下命令安装：

$ make

可能出现的问题有：

(1)“make[1]: Leaving directory ‘/usr/local/nginx/’”

解决方案：如果你走到make这一步的时候只出现了一行“make[1]: Leaving directory `/usr/local/nginx-1.12.1；”提示，不用管它，继续走下一步

make install

5、make完成之后，执行：

$ make install

6、希望能顺利完成以上步骤。安装完成后,打开nginx的配置文件nginx.conf进行配置:在root前提下执行下面命令打开nginx.conf

$ gedit /usr/local/nginx/conf/nginx.conf

7、nginx.conf中增加rtmp的支持，参考如下网址：https://blog.csdn.net/xdwyyan/article/details/43198985 中“下面是修改过的文件，以后可能还会修改更新”下的文件

8、保存完配置文件之后，启动nginx服务

$ /usr/local/nginx/sbin/nginx -c /usr/local/nginx/conf/nginx.conf

9、当服务启动成功的话，在本机浏览器输入：http://localhost/，出现如下字

样，即代表服务成功启动。

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.

Commercial support is available at nginx.com.

Thank you for using nginx.

1. 部署项目DarknetCPP

1. 安装qt5

$ sudo apt-get install qt5-default qtcreator

2. 将项目文件夹DarknetCPP拷贝到用户主目录下

3. 启动nginx流媒体服务器，首先检查nginx的配置文件是否正确

$ ./nginx -t

如果出现以下信息，继续执行启动命令。

nginx.conf syntax is ok

nginx.conf test is successful

启动nginx流媒体服务器：

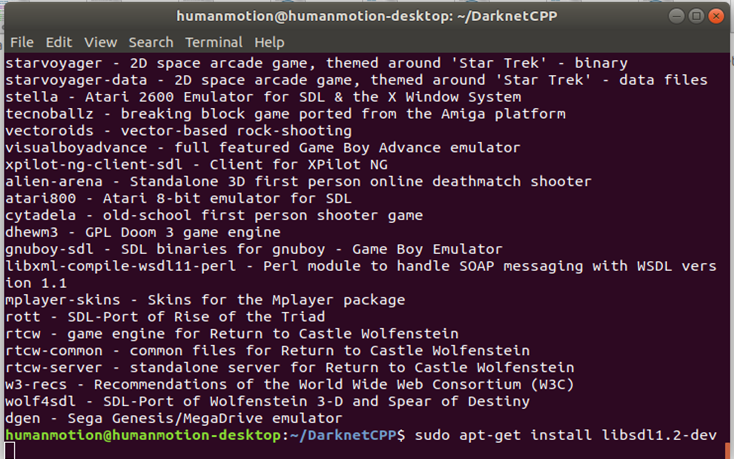
$ sudo ./nginx

4. 进入项目文件夹下，对项目进行编译：

$ cd DarknetCPP

$ make -j8

可能出现的问题有,如下图：



解决方案：输入以下命令，安装sdl库。

$ sudo apt-get install libsdl2-dev

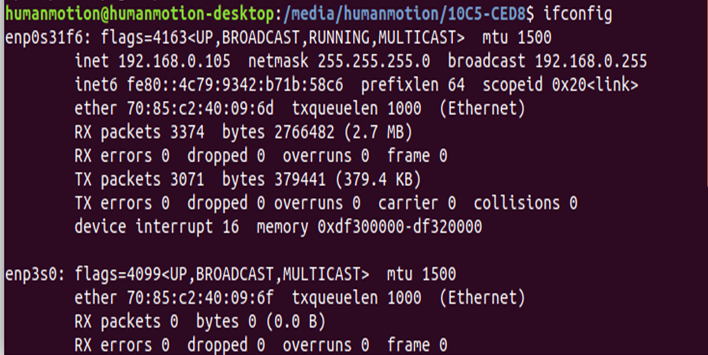
5. 在qt5中打开项目，并运行。

可能出现的问题有：

（1）终端出现了“Failed to load module ‘canberra-gtk-module’”的错误提示。

解决方案：

6. 在终端输入ifconfig,查看网卡的IP地址，例如192.168.0.105，如下图。



在代码中进行配置后，打开谷歌浏览器，输入地址http://192.168.0.105/index-helmet.html 即可。

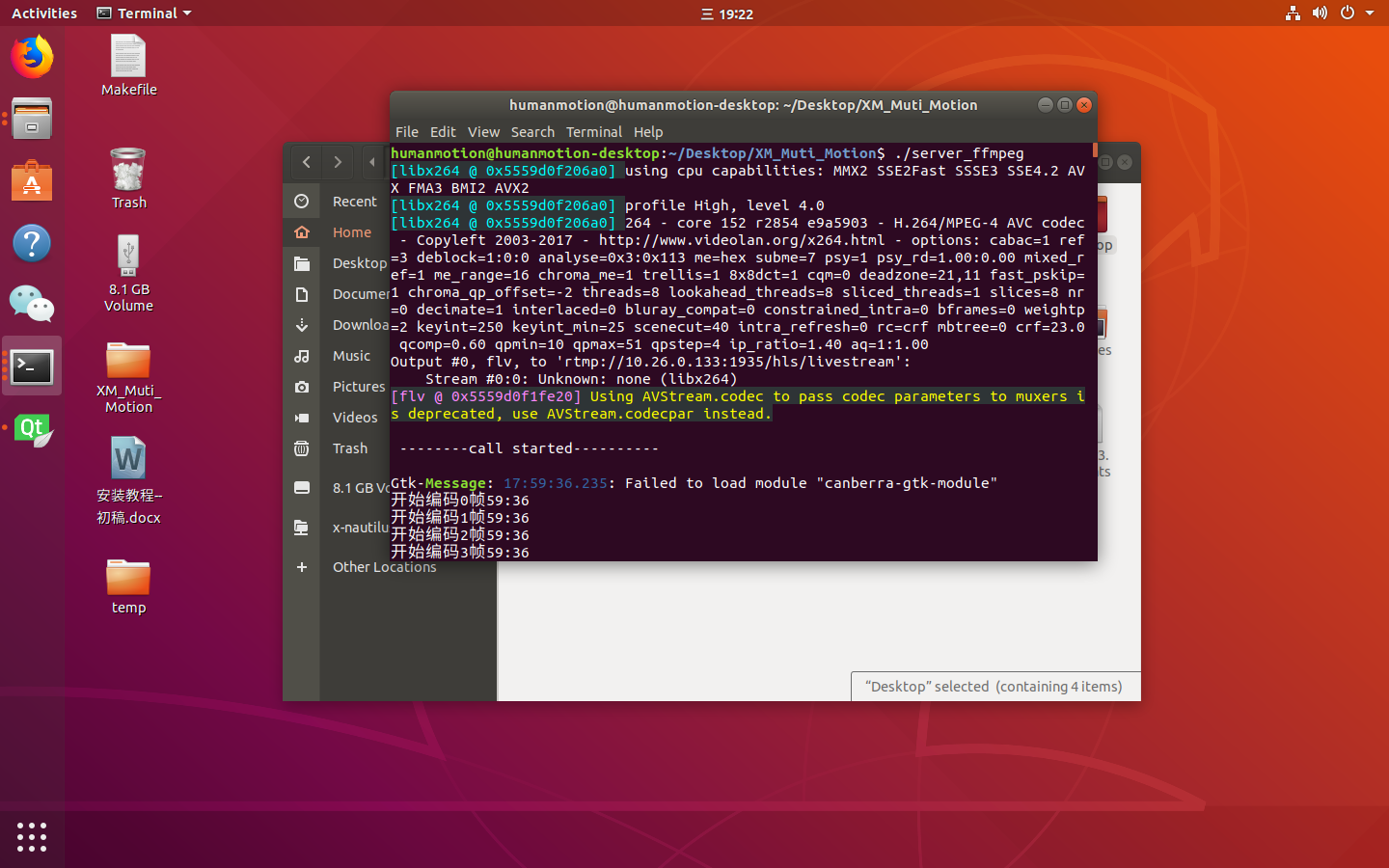
1. 部署项目XM\_Muti\_Motion

1. 将项目文件夹XM\_Muti\_Motion拷贝到任意文件夹下（例如：桌面）。

2. 终端进入项目文件夹下，启动ffmpeg服务器：

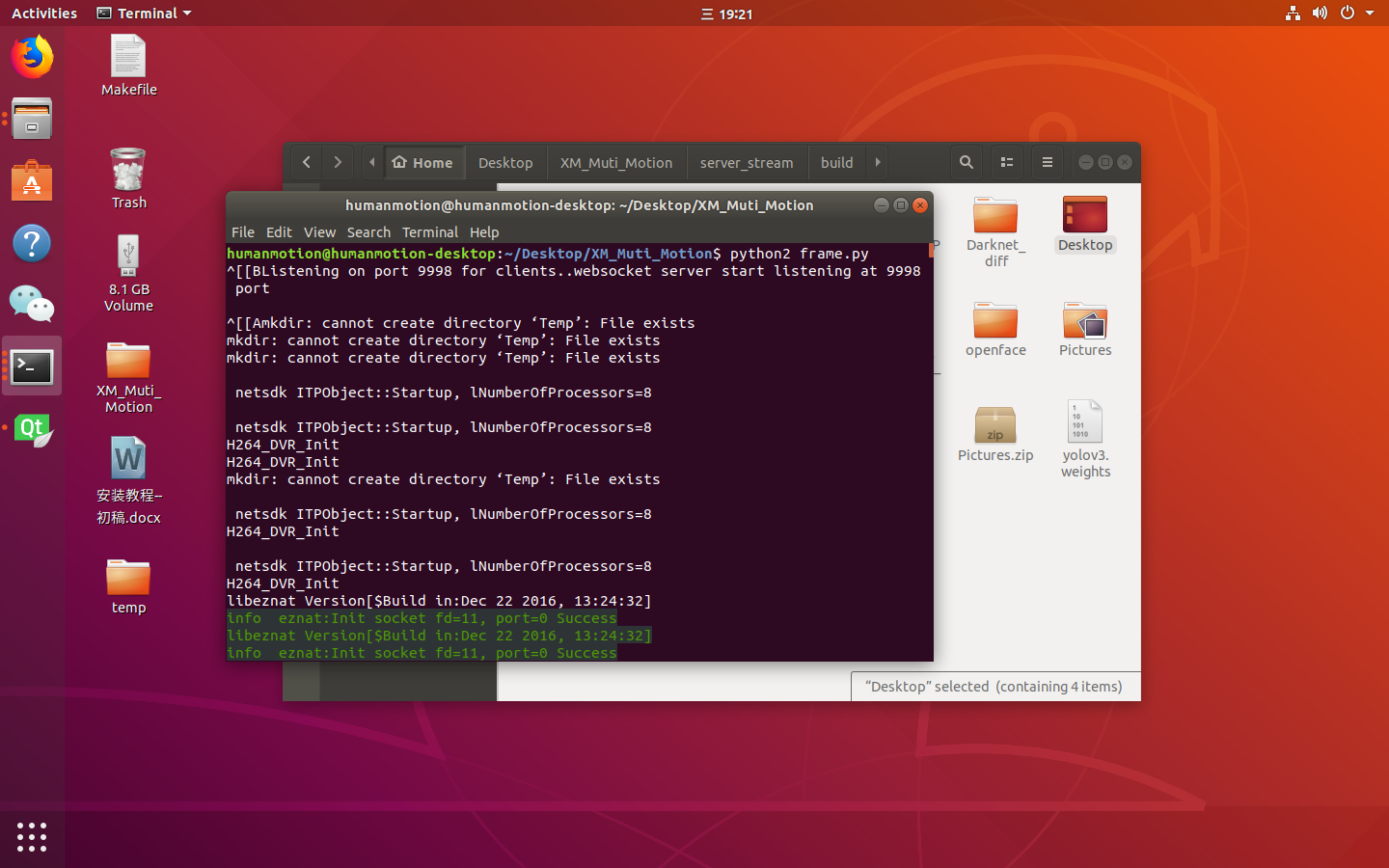
$ cd /Desktop/XM\_Muti\_Motion

$ ./server\_ffpmeg



3. 运行项目：

$ python2 frame.py



4. 关闭项目：

首先，Ctrl+Z 挂起项目，然后Ctrl+D退出。

参考文件如下：

1. darknet的安装: https://pjreddie.com/darknet/yolo/

2. openface的安装：

https://blog.csdn.net/weixinhum/article/details/77046873

3. cuda及cuDNN的安装：

https://www.cnblogs.com/maxuewei2/p/8974243.html

https://blog.csdn.net/lhx\_998/article/details/76135936

https://blog.csdn.net/dihuanlai9093/article/details/79253963/

4. 使用nginx+nginx-rtmp-module+ffmpeg搭建流媒体服务器:

https://blog.csdn.net/xdwyyan/article/details/43198985