**Ubuntu 설치 및 Detectron 설치 과정**

1-1. ssd에 한번에 Ubuntu 설치 시  
<https://jimnong.tistory.com/676>  
=>볼륨 축소시 용량 그대로 가져감.(후에 용량 변경 불가)  
=>설치형식(windows10을 그대로 두고 Ubuntu 설치 : 우분투 설치시 파티션 안나눠도됨)

1-2. ssd에 windows, hdd에 Ubuntu 설치 시  
=> (ext4로 / 논리,100000MB, /boot 500MB, 스왑영역 8192MB, /home 논리,나머지) 파티션 분할 후, 설치  
=> UEFI 에서 부트로더 Other OS 선택 후 Ubuntu 접근 가능

2.우분투의 기본 드라이버 Nouveau 비활성화  
sudo apt --purge autoremove nvidia\*  
sudo apt-get remove nvidia\* && sudo apt autoremove  
sudo apt-get install dkms build-essential linux-headers-generic  
sudo vi /etc/modprobe.d/blacklist.conf  
------------blacklist.conf의 맨 밑에 추가할 내용--------------  
blacklist nouveau  
blacklist lbm-nouveau  
options nouveau modeset=0  
alias nouveau off  
alias lbm-nouveau off  
:wq 로 빠져나오기  
--------------------------------------------------------------  
echo options nouveau modeset=0 | sudo tee -a /etc/modprobe.d/nouveau-kms.conf  
sudo update-initramfs -u    #커널 재빌드  
sudo reboot  
nvidia-settings #설치된 드라이버 확인  
nvidia-smi   =======> 드라이버 버전, CUDA 버전 확인  
sudo service lightdm stop

3.CUDA+CUDNN 설치(CUDA에 NVIDIA Driver 내장되어 있음)  
<https://developer.nvidia.com/cuda-10.1-download-archive-base?target_os=Linux&target_arch=x86_64&target_distro=Ubuntu&target_version=1804&target_type=deblocal> =>(CUDA 10.1 설치)  
<https://developer.nvidia.com/rdp/cudnn-download>  
=>(CUDNN설치 Library for Linux로)  
  
sudo dpkg -i cuda-repo-ubuntu1804-10-1-local-10.1.105-418.39\_1.0-1\_amd64.deb  
sudo apt-key add /var/cuda-repo-10-1-local-10.1.105-418.39/7fa2af80.pub  
sudo apt-get update  
sudo apt-get -y install cuda  
sudo reboot  
--------cudnn Library for Linux 설치, 압축해제 후 cuda가 있는 경로로 이동--------  
sudo cp include/cudnn.h /usr/local/cuda-10.1/include  
sudo cp lib64/libcudnn\* /usr/local/cuda-10.1/lib64/  
sudo chmod a+r /usr/local/cuda-10.1/lib64/libcudnn\*  
-------------- bashrc 에 추가 ---------------------------------------------------------  
gedit ~/.bashrc  
export PATH=$PATH:/usr/local/cuda-10.1  
export PATH=/usr/local/cuda-10.1/bin${PATH:+:${PATH}}  
export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:/usr/local/cuda-10.1/lib64  
source ~/.bashrc   #편집내용 저장  
nvcc --version    #버전 확인  
---------------------------------------------------------------------------  
cat /usr/local/cuda-10.1/include/cudnn.h | grep CUDNN\_MAJOR -A 2  
=> 출력 화면에 CUDNN\_MAJOR가 포함되어 있으면 제대로 설치된 것  
  
  
4. OpenCV 설치(★3.4.0 버전인 것 꼭 확인★)  
pkg-config --modversion opencv #opencv 설치 확인  
-----------2.4버전이 설치되어 있는 경우 삭제 후 진행---------------------  
sudo apt-get purge  libopencv\* python-opencv  
sudo apt-get autoremove  
sudo find /usr/local/ -name "\*opencv\*" -exec rm -i {} \;  
----------------------------------------------------------------------  
sudo apt-get update  
sudo apt-get upgrade  
sudo apt-get install build-essential cmake    
-------------위 명령에서 libcudnn.so.7 is not a symbolic link 뜨면----------------------------  
/usr/local/cuda-10.1/lib64$ ls -al  
sudo ldconfig -v  
sudo apt-get install build-essential cmake    
----------------------------------------------------------------------------------------------  
sudo apt-get install pkg-config  
sudo apt-get install libjpeg-dev libtiff5-dev libpng-dev  
sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libxvidcore-dev libx264-dev libxine2-dev  
sudo apt-get install libv4l-dev v4l-utils  
sudo apt-get install libgstreamer1.0-dev libgstreamer-plugins-base1.0-dev  
sudo apt-get install libgtk2.0-dev  
sudo apt-get install mesa-utils libgl1-mesa-dri libgtkgl2.0-dev libgtkglext1-dev    
sudo apt-get install libatlas-base-dev gfortran libeigen3-dev  
sudo apt-get install python2.7-dev python3-dev python-numpy python3-numpy  
  
mkdir opencv  
cd opencv  
wget -O opencv.zip <https://github.com/opencv/opencv/archive/3.4.0.zip>  
unzip opencv.zip  
wget -O opencv\_contrib.zip <https://github.com/opencv/opencv_contrib/archive/3.4.0.zip>  
unzip opencv\_contrib.zip  
ls -d \*/       #디렉토리 생성 확인  
cd opencv-3.4.0/  
mkdir build  
cd build  
  
-----------------------------------------------------------------------------------------  
★cmake시 유의할점 1: python library경로에 [libpython3.6m.so](http://libpython3.6m.so/)있는지 꼭 확인(안하면 큰일남)  
     만약 설정했는데도 안바꼈으면 cmakecache 파일 삭제후 다시 cmake★  
★cmake시 유의할점2: Qt5 오류  
apt-file search Qt5CoreConfig.cmake  
sudo apt install apt-file  
sudo apt install qtbase5-dev  
-----------------------------------------------------------------------------------------  
  
cmake -D CMAKE\_BUILD\_TYPE=RELEASE \  
-D CMAKE\_INSTALL\_PREFIX=/usr/local \  
-D WITH\_TBB=OFF \  
-D WITH\_IPP=OFF \  
-D WITH\_1394=OFF \  
-D BUILD\_opencv\_cudacodec=Off \  
-D BUILD\_WITH\_DEBUG\_INFO=OFF \  
-D BUILD\_DOCS=OFF \  
-D INSTALL\_C\_EXAMPLES=ON \  
-D INSTALL\_PYTHON\_EXAMPLES=ON \  
-D BUILD\_EXAMPLES=OFF \  
-D BUILD\_TESTS=OFF \  
-D BUILD\_PERF\_TESTS=OFF \  
-D WITH\_QT=ON \  
-D WITH\_OPENGL=ON \  
-D OPENCV\_EXTRA\_MODULES\_PATH=../../opencv\_contrib-3.4.0/modules \  
-D WITH\_V4L=ON  \  
-D WITH\_FFMPEG=ON \  
-D WITH\_XINE=ON \  
-D BUILD\_NEW\_PYTHON\_SUPPORT=ON \  
-D PYTHON2\_INCLUDE\_DIR=/usr/include/python2.7 \  
-D PYTHON2\_NUMPY\_INCLUDE\_DIRS=/usr/lib/python2.7/dist-packages/numpy/core/include/ \  
-D PYTHON2\_PACKAGES\_PATH=/usr/lib/python2.7/dist-packages \  
-D PYTHON2\_LIBRARY= /usr/lib/x86\_64-linux-gnu/[libpython2.7.so](http://libpython2.7.so/) \  
-D PYTHON3\_INCLUDE\_DIR=/usr/include/python3.6m \  
-D PYTHON3\_NUMPY\_INCLUDE\_DIRS=/usr/lib/python3/dist-packages/numpy/core/include/ \  
-D PYTHON3\_PACKAGES\_PATH=/usr/lib/python3.6/dist-packages \  
-D PYTHON3\_LIBRARY=/usr/lib/x86\_64-linux-gnu/[libpython3.6m.so](http://libpython3.6m.so/) \  
../  
  
cat /proc/cpuinfo | grep processor | wc -l  #CPU 코어수 확인  
make -j8   #여기서 시간 엄청 오래 걸림(대략 40~50분), -j 뒤에 코어 수 삽입  
sudo make install  
cat /etc/ld.so.conf.d/\*   #/usr/local/lib 가 있는지 확인  
=>없으면 sudo sh -c 'echo '/usr/local/lib' > /etc/ld.so.conf.d/opencv.conf' 추가  
sudo ldconfig  
pkg-config --modversion opencv  #다시 opencv 설치 확인하면 출력이 됨 짜잔-!  
pkg-config --libs --cflags opencv  #문제없이 나오면 설치완료된것  
ls /usr/local/share/OpenCV/samples/   #예제코드들 설치된 것 확인  
#cv2 import 되는지 확인 (★매우 중요★ 안되면 opencv 버전 설치 잘못한 것)  
python  
import cv2  
cv2.\_\_version\_\_  
  
#필요없어진 opencv 소스코드 디렉토리 삭제  
cd  
rm -rf opencv  
  
  
5. Caffe2 설치  
<https://caffe2.ai/docs/getting-started.html?platform=ubuntu&configuration=prebuilt>  
=>Ubuntu -> Build From Source 에서  
sudo apt-get update  
sudo apt-get install -y --no-install-recommends \  
      build-essential \  
      cmake \  
      git \  
      libgoogle-glog-dev \  
      libgtest-dev \  
      libiomp-dev \  
      libleveldb-dev \  
      liblmdb-dev \  
      libopencv-dev \  
      libopenmpi-dev \  
      libsnappy-dev \  
      libprotobuf-dev \  
      openmpi-bin \  
      openmpi-doc \  
      protobuf-compiler \  
      python-dev \  
      python-pip                            
pip install --user \  
      future \  
      numpy \  
      protobuf \  
      typing \  
      hypothesis  
★★★pip 버전 업그레이드(tip : pip ==> 19.3.1 일 때 성공했음)★★★  
pip list (sudo로 하면 절대안됨)    or   pip --version ======> pip 버전 확인  
pip install --upgrade pip   =======> pip 업그레이드  
which pip  #pip 위치 확인  
sudo which pip   #sudo pip 위치 확인  
  
--------------------------★★★권한 없을 때★★★------------------------------------------  
ls -al   ===========> cd ~ 에서 모든 파일 권한보기  
(필요한건가??sudo chown -R kimdaeun:kimdaeun .cache)  
sudo chmod -R 777 .cache      ======>권한 없으면 .cache와 .local에 모든권한 주기  
ls -al     ================> 다시 권한 확인해보기  
--------------------------------------------------------------------------------------------  
  
  
------------------------------- 오류 발생 시 해결법 -----------------------------------------  
★Command "python setup.py egg\_info" failed with error code 1 in /tmp/pip-build-vp3xzo/future/  
ImportError: No module named setuptools 오류뜨면 => sudo pip install setuptools  
★hypothesis 에러 뜨면  
pip install -Iv hypothesis==3.35.0  
---------------------------------------------------------------------------------------------  
  
  
# for Ubuntu 16.04(Ubuntu 18.04에서도 되는 걸로 확인)  
sudo apt-get install -y --no-install-recommends \  
      libgflags-dev \  
      cmake  
  
git clone <https://github.com/pytorch/pytorch.git> && cd pytorch  
git submodule update --init --recursive  
python setup.py install          #시간 오래 걸림  
#Success 출력시 성공한 것  
cd ~ && python -c 'from caffe2.python import core' 2>/dev/null && echo "Success" || echo "Failure"          
# 0 이상의 숫자가 출력되면 성공한 것  
python -c 'from caffe2.python import workspace; print(workspace.NumCudaDevices())'  
python caffe2/python/operator\_test/activation\_ops\_test.py  #failed 상관없음  
  
  
6.COCO API 설치  
# COCOAPI=/path/to/clone/cocoapi     #COCOAPI 설치 경로 설정  
git clone <https://github.com/cocodataset/cocoapi.git> $COCOAPI    # $COCOAPI는 경로 부분  
cd $COCOAPI/PythonAPI  
sudo make install        # Install into global site-packages  
(위 명령에서 x86\_64-linux-gnu-gcc: error: pycocotools/\_mask.c: 그런 파일이나 디렉터리가 없습니다 오류 떴을 때  
sudo pip install cython)  
---------------------------------------------------------------------------  
# Alternatively, if you do not have permissions or prefer  
# not to install the COCO API into global site-packages  
sudo python setup.py install --user  
(위 명령에서 syntax오류 발생했을 때 pip install matplotlib==2.1.0 --user)

7.Detectron 설치  
# DETECTRON=/path/to/clone/detectron  
git clone <https://github.com/facebookresearch/detectron> $DETECTRON  
pip install -r $DETECTRON/requirements.txt  
cd $DETECTRON && make  
python $DETECTRON/detectron/tests/test\_spatial\_narrow\_as\_op.py

8.demo 확인  
python tools/infer\_simple.py \  
    --cfg configs/12\_2017\_baselines/e2e\_mask\_rcnn\_R-101-FPN\_2x.yaml \  
    --output-dir /tmp/detectron-visualizations \  
    --image-ext jpg \  
    --wts <https://dl.fbaipublicfiles.com/detectron/35861858/12_2017_baselines/e2e_mask_rcnn_R-101-FPN_2x.yaml.02_32_51.SgT4y1cO/output/train/coco_2014_train:coco_2014_valminusminival/generalized_rcnn/model_final.pkl> \  
    demo  
=> output-dir 경로에 디텍팅 결과 생성됨

9.COCO Dataset 설치(coco\_2014\_minival로 테스트 시)  
-minival == val2017  
-valminusminival + 2014 train == train2017  
-coco\_2014\_train:coco\_2014\_valminusminival == train2017  
-train=>트레이닝 할 이미지의 대부분, val=>트레이닝 할 이미지 일부분, test=>테스트용 이미지  
-------------------------------------------------------------------------------------  
<------gsutil(Google Storage 유틸리티) 설치------->  
sudo apt install curl  
curl [https://sdk.cloud.google.com](https://sdk.cloud.google.com/) | bash  
source ~/.bashrc           =========> bashrc 저장  
<------val2017 설치------->  
mkdir cocodataset  
cd cocodataset  
mkdir val2017  
gsutil -m rsync gs://[images.cocodataset.org/val2017](http://images.cocodataset.org/val2017) val2017  
mkdir anns  
gsutil -m rsync gs://[images.cocodataset.org/annotations](http://images.cocodataset.org/annotations) anns  
(위 명령에서 오류뜨면 wget <http://images.cocodataset.org/annotations/annotations_trainval2017.zip>)  
cd val2017  
unzip ../anns/annotations\_trainval2017.zip  
  
  
10. coco dataset으로 test 하는 법  
------------★sample 소스-------------------------------------------------------------  
cd detectron  
python tools/test\_net.py \  
    --cfg configs/12\_2017\_baselines/e2e\_mask\_rcnn\_R-101-FPN\_2x.yaml \  
    --multi-gpu-testing \  
    TEST.WEIGHTS <https://dl.fbaipublicfiles.com/detectron/35861858/12_2017_baselines/e2e_mask_rcnn_R-101-FPN_2x.yaml.02_32_51.SgT4y1cO/output/train/coco_2014_train:coco_2014_valminusminival/generalized_rcnn/model_final.pkl> \  
    NUM\_GPUS 8     #GPU 코어 수  
--------------------------------------------------------------------------------------  
python tools/infer\_simple.py \  
    --cfg configs/12\_2017\_baselines/e2e\_mask\_rcnn\_R-101-FPN\_2x.yaml \  
    --output-dir /tmp/detectron-visualizations \  
    --image-ext jpg \  
    --wts  <https://dl.fbaipublicfiles.com/detectron/35861858/12_2017_baselines/e2e_mask_rcnn_R-101-FPN_2x.yaml.02_32_51.SgT4y1cO/output/train/coco_2014_train:coco_2014_valminusminival/generalized_rcnn/model_final.pkl> \  
    test/test    #####테스트 해 볼 이미지폴더 경로  
  
  
\* 성능을 높인 방법  
FPN(Feature Pyramid Network)  
Faster R-CNN 에 RoIAlign 을 사용하는 것만으로도 성능이 향상된다.  
Mask 를 추가적으로 구하는 것이 오히려 성능을 향상시킨다.  
마지막으로 ResNetXt 를 주 구조(backbone)로 사용했을 때 성능이 가장 높다.

11. 만든 데이터셋으로 학습시키는 방법

---------------- jq 를 이용해서 json file 줄 바꿔서 보는 법 ------------------

///// instances\_val2017.json 파일 변환하기 => 한줄로 뜨므로 줄 바꿔줘야함

sudo apt-get install jq #jq 다운로드

apt list jq #apt 리스트에서 jq 확인

jq --version #jq 버전 확인

cd /home/kimdaeun/cocodataset/anns/annotations\_trainval2017/annotations

cat instances\_val2017.json | jq

---------------------------------------------------------------------------

1) COCO API json format 으로 변환(디텍트론은 COCO API를 사용하기 때문)

http://cocodataset.org/#download => COCO dataset 사이트 참고

* 12번으로 이동

2) 데이터셋 목록에 추가

detectron/datasets/dataset\_catalog.py 이동 후 데이터셋 목록에 추가

3) 만든 데이터셋으로 test 진행

4) 객체 면적 구하는 법

cocoapi/PythonAPI/pycocotools/mask.py 의 area 함수를 이용해 객체 면적 구할 수 있음

* 13번으로 이동

12. Detectron에 활용할 객체 태깅 방법

1) Amazon SageMaker Ground Truth 프로그램 이용(유료)

=> cocodataset 형식으로 생성 가능 But, 금전적인 문제

2) labelme 이용(무료)

=> <https://github.com/wkentaro/labelme>

3) pycococreator 이용(무료)

=> <https://github.com/asyncbridge/pycococreator>

13. labelme 태깅 방법

# Ubuntu 14.04 / Ubuntu 16.04

# Python2

# sudo apt-get install python-qt4 # PyQt4

sudo apt-get install python-pyqt5 # PyQt5

sudo pip install labelme

# Python3

sudo apt-get install python3-pyqt5 # PyQt5

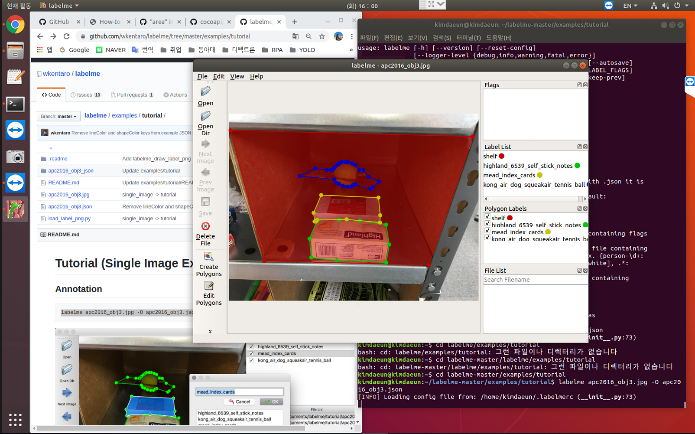
sudo pip3 install labelme

<https://github.com/wkentaro/labelme>

* labelme 다운받은 후

cd labelme-master/examples/tutorial

labelme apc2016\_obj3.jpg -O apc2016\_obj3.json

****

<labelme 태깅 test 화면>

14. mask 면적 구하는 방법

cocoapi/PythonAPI/pycocotools/mask.py

에서 area 함수를 통해 구할수있음

**\*추후 테스트 결과 사진\***

****