1. 개발 환경

: Darknet Build를 위한 개발환경 설정 문서이며, Ubuntu os를 기준으로 작성되었음. 또한 Nvidia 관련 프로그램들은 (CUDA, cuDNN, Nvidia Driver) RTX 3090 기준으로 작성되었으며, GPU가 다를 경우 버전확인이 필요함

1. 사용 OS: Ubuntu 18.04
2. GPU: Nvidia RTX 3090
3. Darknet Build를 위한 환경 설정
4. Build-essential 설치

$ sudo apt-get update

$ sudo apt-get install build-essential

1. Nvidia Driver 설치
   1. Nvidia 홈페이지에서 드라이버 검색 후 다운로드
      * https://www.nvidia.com/Download/index.aspx



드라이버 다운로드 폴더 이동 후 아래 명령어 실행하여 Nvidia Driver 설치

$ sudo sh NVIDIA-Linux-x86\_64-470.94.run

1. CUDA 설치 (11.2.1)

$wget https://developer.download.nvidia.com/compute/cuda/11.2.1/local\_installers/cuda\_11.2.1\_460.32.03\_linux.run

$ sudo sh cuda\_11.2.1\_460.32.03\_linux.run



* 다른 버전 사용할 경우 Nvidia 홈페이지에서 원하는 버전 검색 후 다운로드
  + - <https://developer.nvidia.com/cuda-toolkit-archive>

1. cuDNN 설치 (8.2.1)
   * + <https://developer.nvidia.com/rdp/cudnn-download> 홈페이지 회원 가입 후 cuDNN 다운로드
     + 앞에서 설치한 CUDA 버전에 맞는 cuDNN 다운로드





* + - cuDNN 다운로드 폴더로 이동

$ tar xvfz cudnn-11.3-linux-x64-v8.2.1.32.tgz

* + - Header 및 library 이동

$ sudo cp cuda/include/cudnn\*.h /usr/local/cuda/include

$ sudo cp cuda/lib64/libcudnn\* /usr/local/cuda/lib64

$ sudo chmod a+r /usr/local/cuda/include/cudnn\*.h /usr/local/cuda/lib64/libcudnn\*

1. OpenCV 설치 (4.2)
   1. OpenCV 컴파일을 위한 패키지 설치

$   sudo apt-get update

$ sudo apt-get install pkg-config

: pkg-config 프로그램 컴파일 및 링크시 필요한 라이브러리에 대한 정보를 메타파일로부터 가져오는데 사용

$  sudo apt-get install libjpeg-dev libtiff5-dev libpng-dev

: 특정 포맷의 이미지 파일 불러오거나 기록하기 위한 패키지

$ sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libxvidcore-dev libx264-dev libxine2-dev

: 특정 코덱의 비디오 파일 읽어오거나 기록하기 위한 패키지

$  sudo apt-get install libv4l-dev v4l-utils

: 리눅스 실시간 비디오 캡처 지원하기 위한 디바이스 드라이버와 API

$    sudo apt-get install libgstreamer1.0-dev libgstreamer-plugins-base1.0-dev

: 비디오 스트리밍

$ sudo apt-get install libgtk2.0-dev mesa-utils libgl1-mesa-dri libgtkgl2.0-dev libgtkglext1-dev

: UI & openGL

$ sudo apt-get install libatlas-base-dev gfortran libeigen3-dev

: OpenCV 최적화

$  sudo apt-get install python2.7-dev python3-dev python-numpy python3-numpy

: Python

* 1. OpenCV 다운로드 및 설치

$ mkdir opencv

$  cd opencv

$ wget -O opencv.zip<https://github.com/opencv/opencv/archive/4.2.0.zip>

$ unzip opencv.zip

$ wget -O opencv\_contrib.zip [https://github.com/opencv/opencv\_contrib/archive/4.2.0.zip](%20https://github.com/opencv/opencv_contrib/archive/4.2.0.zip)

$ unzip opencv\_contrib.zip

$ cd opencv-4.2.0/

$ mkdir build

$  cd build

$ 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\_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=OFF \

-D WITH\_GTK=ON \

-D WITH\_OPENGL=ON \

-D OPENCV\_EXTRA\_MODULES\_PATH=../../opencv\_contrib-4.2.0/modules \

-D WITH\_V4L=ON \

-D WITH\_FFMPEG=ON \

-D WITH\_XINE=ON \

-D BUILD\_NEW\_PYTHON\_SUPPORT=ON \

-D OPENCV\_GENERATE\_PKGCONFIG=ON ../

$ make

$ make install

$ sudo ldconfig

* 설치 제대로 되었는지 확인

$ pkg-config --modversion opencv4



1. Darknet 빌드

$ cd darknet

$ make

* 빌드 완료 후, darknet 바이너리 생성됨. 생성된 바이너리 이용하여 학습 및 평가 가능

1. 데이터 학습
   1. Bound box RGB Training

$ ./darknet detector train ./bb\_rgb/bb\_rgb\_train.data ./bb\_rgb/bb\_rgb.cfg ./bb\_rgb/yolov4.conv.137 --dont\_show -map

* 학습 종료 후 ./bb\_rgb/backup 폴더에 Weights 파일 저장됨
  1. Bound box NIR Training

$ darknet detector train ./bb\_nir/bb\_nir\_train.data ./bb\_nir/bb\_nir.cfg ./bb\_nir/yolov4.conv.137 –gpus 0,1 –dont\_show

* 학습 종료 후, ./bb\_nir/backup 폴더에 weights 파일이 저장됨
  1. 참고사항
* bb\_rgb, bb\_nir 폴더 아래에 .data 파일 존재함

이 파일에서 학습해야 하는 이미지의 리스트 정보, weights가 저장되는 폴더 위치, class 이름이 정의된 파일 정보에 대한 내용이 포함되어 있음.

* 학습 전에, 학습 해야하는 이미지 리스트(train.txt, val.txt)의 파일 경로 확인하여 수정이 필요함
* 보다 자세한 기능에 대한 내용은 Darknet git hub 참고

<https://github.com/AlexeyAB/darknet>