YOLOV4'ün bilgisayara kurulması ve eğitim başlatma dökümanı

İçindekiler

Hazırlayan: Zarquon ~ Hüseyin Ayvacı

YoloV3 ‘ün Windows’a kurulması

YoloV3’ün bir model olduğundan bunu çalıştırmak için bir yapay sinir ağı ortamına ihtiyacımız var kullanacağımız yapay sinir ağı ortamı darknet.

AlexeyAB darknet ➔https://github.com/AlexeyAB/darknet#how-to-train-to-detect-your-custom-objects

Darknet‘in kurulumunda gerekenler:

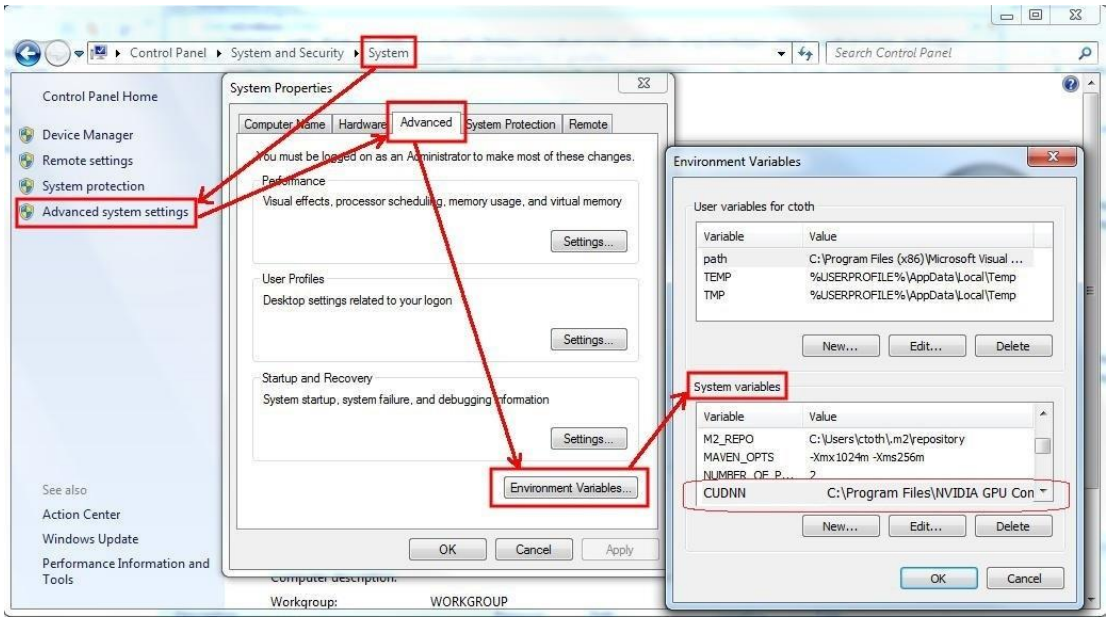
- Nvidia Cuda 10.0 ➔https://developer.nvidia.com/cuda-10.0-download-archive
- Nvidia CUDNN >=7.0 for CUDA 10.0➔https://developer.nvidia.com/rdp/cudnn-archive
- CUDNN’yi indirmek için bize bir Nvidia hesabı lazım sitesinden kaydolabilirisiniz.
- OpenCV >=2.4(önerilen 3.3)➔ https://sourceforge.net/projects/opencvlibrary/files/opencv-win/3.3.0/opencv-3.3.0-vc14.exe/download
- Tensorflow GPU 1.12.0 ➔cmd ekranını açıp “pip install tensorflow-gpu==1.12.0” yazıyoruz.
- Pyhton 3.6 ➔Anacondayı indirdiğimizde içinde python 3.6 ile birlikte geliyor bunu kullanmak daha iyi olacakatır. https://www.anaconda.com/distribution/
- Microsoft Visual Studio 2017➔ https://docs.microsoft.com/en-us/visualstudio/releasenotes/vs2017-relnotes

ÖNEMLİ NOT: Yükledikten sonra oluşan hataların çoğu CUDNN, CUDA, Tensorflow sürüm uyuşmazlığından gelmektedir.

Darknet’in Kurulması:

1-Darknet’i indirdiğimizde bize bir .rar formatında kurulacak içindeki dosyaları bir klasöre çıkartıyoruz.(Dikkat edin çıkarttığımız yerden bir daha başka bir konuma taşımak için tekrar kurmak gerekiyor bu yüzden kuracağımız yeri bir kere seçelim.)

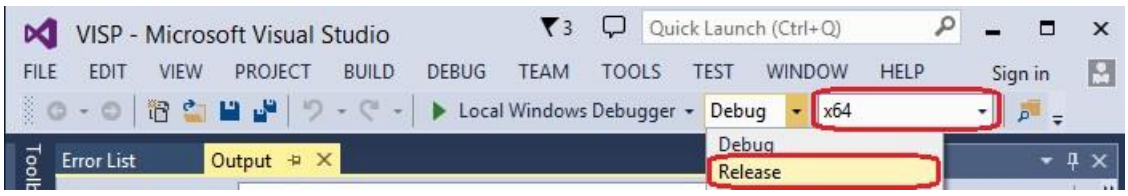
2-İndirdiğimiz CUDNN ….zip’inin içinde **bin, include, lib** klasörlerini göreceksiniz.Devamında CUDA Toolkit kurulumu ile birlikte gelen “C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v10.0” dizini içerisine; zip arşiv dosyası içerisinde yer alan **bin, include** ve **lib** klasörlerini kopyalayınız.Ayrıca “cudnn64_7.dll” dosyasını “darknet.exe” nin bulunduğu dizine de kopyalıyoruz.



3- Resim ne kadar karışık olsada aslında yapacağımız iş kolay. Windows tuşuna basıp "ortam" yazınca "Sistem Ortam Değişkenlerini Düzenleyin"e tılayın.Orada altta bulunan ortam değişkenleri butonuna tıklayın "Sistem Değişkenleri" sekmesindeki "Yeni" butonuna tıklayalım. Gelen pencereki "Değişken adı" kısmına "CUDNN" , "Değişken" kısmına da "C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v10.0" yazıyoruz.

4- "C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v10.0\extras\visual_studio_integration\MSBuildExtensions" dizinindeki dosyaları "C:\Program Files (x86)\Microsoft Visual Studio\2017\Community\Common7\IDE\VC\VCTargets\BuildCustomizations" dizinine kopyalayın.

5-Darknet klasörünün içinden "build\darknet\darknet.sln(eğer GPU değil de CPU'da çalıştıracaksanız darknet_no_gpu.sln) dosyasını VS 2017 ile açıyoruz.



Ardından yukarıda gösterildiği gibi üst kısımlardan Debug'u Relaese çeviriyoruz ve onun yanındaki kutuyu da x64 yapıyoruz.

Sonra yukarıdan "Derle" kısmından "Yapılandır darknet" e tıklıyoruz.

6- "C:\opencv_3.0\opencv\build\x64\vc14\bin" klasöründen opencv_world330.dll ve opencv_ffmpeg330_64.dll(bende opencv 3.3 yüklü olduğundan …330… sizde bu kısımlar farklı olabilir.) dosyalarını "darknet.exe" nin bulunduğu dizine kopyalıyoruz.

7-“darknet.exe” bulunduğu dizinde bir cmd ekranı açıp “`darknet.exe detector demo cfg/coco.data cfg/yolov3.cfg yolov3.weights -c 0`” yazdığımızda webcam açılıyorsa hatasız bir şekilde kurulum gerçekleşmiş demektir.

YoloV4 ’ün Ubuntu’ya kurulması

!!! Dikkat : Burada anlatılanlar Ubuntu FocalFossa(20.04 LTS) ‘da denenmiştir.

Darknet‘in kurulumunda gerekenler:

- Nvidia Cuda >= 10.2
- Nvidia CUDNN >=8.0.2
- OpenCV >=2.4
- Cmake >= 3.18
- GPU with CC>3.0

Compute Capability(CC) nedir?

GPU’nuzun bir nevi versiyon numarasıdır. Yeni sürümlerde daha yeni özellikler vardır. Yolo için CC’nin en az 3 olması gerekir.

https://en.wikipedia.org/wiki/CUDA#GPUs_supported

sitesinden CTRL+F ile ekran kartınızı aratarak GPU CC’nizi öğrenebilirsiniz.

Ekran kartı kurulumu

Ekran kartınızın Nvidia mimarisi olması gerekiyor aksi takdirde Yolo’yu GPU üzerinde çalıştıramazsınız buda acayip FPS’inizi düşürür. O yüzden testlerinizi iyi bir Nvidia ekran kartı olan bilgisayarda yapmanız daha iyi olur.

Terminale gelip ‘nvidia-smi’ yazarak ekran kartınızın güncelliğini kontrol ediniz. Terminalde sol üstte CUDA Version yazan yer bilgisayarınıza kuracağınız maksimum CUDA sürümüdür.

Eğer terminale yazdığınızda bir çıktı gelmiyor hata veriyorsa Nvidia sürücünüz yüklü değildir. Ubuntu APU sürücünüz üzerinden çalışıyordur.(CPU işlemcinizde entegre ekran kartı.) O yüzden ekran kartı sürücünüzü kurmanız gerekmektedir.

Diğer tüm işlemlerden önce ekran kartınızı kurmanız daha iyi olur çünkü ilerde göstereceğim ekran kartını kurma yöntemi ile kuramazsanız manuel ile kurmanız gerekmektedir. Bu kısımdada birkaç ubuntunuzu çökertebilirsiniz.(Ben çok yaşadım :/) O yüzden çöktüğünde tekrar ve tekrar diğer yazılımları kurmamak için bunu en baştan kurmanız daha iyi olur.

1-İlk olarak bilgisayarınızın bios menüsüne gelip “Secure Boot” seçeneğini “Disabled” yapın.

2- sudo apt-get purge nvidia* yazarak nvidia ile alakalı herşeyi siliyoruz.

3-sudo add-apt-repository ppa:graphics-drivers

4-sudo apt-get update komutlarını sırasyıla giriyoruz.

5- sudo apt-get install nvidia-xxx(xxx kısmı sizin sürücünüzün numarasıdır eğer numarasını bilmiyorsanız https://www.nvidia.com.tr/drivers sitesinden manuel arama kısmından sürücünüzü araştırıp numarasını bulun ve xxx kısmına yazın)

6- lsmod | grep nvidia

lsmod | grep nouveau

sudo apt-mark hold nvidia-xxx komutlarını sırasıyla giriyoruz.

7-En son işlemimiz bittiğinde sistemimizi yeniden başlatıp 'nvidia-smi' yazarak kontrol ediyoruz.

2. Yol(Bunu önermiyorum internette başka yöntemleri denerseniz daha iyi olur)

1-Nvidia driverinizi indirin.Bundan sonra yapacaklarınızda ubuntunun arayüzünü kullanmıcaksanız haberiniz olsun.

2- Ctrl-Alt-F1 tuşuna basarak arayüzü kapatıyoruz.Ve kullanıcı adı ve şifreyi girerek giriş yapıyoruz.

3- sudo service lightdm stop yada sudo lightdm stop komutlarından birini girerek arayüzü tamamen kapatıyoruz.

4- sudo init 3 diyerek yetkimizi arttırıyoruz.

5-İndirdiğimiz driverin dizinine gelerek sudo sh NVIDIA-Linux-x86_XX-XXX.XX.run yazarak çalıştırıyoruz.

6-Gelen seçeneğe "Continue İnstalling" diyerek devam ediyoruz.

7-"Sign in" içeren seçeneği seçerek devam ediyoruz.

8-İşlem bittiğinde sudo service lightdm restart komutunu girerek yeniden başlatıyoruz.

Cmake Yükleme

https://www.osetc.com/en/how-to-install-the-latest-version-of-cmake-on-ubuntu-16-04-18-04-linux.html sitesinden istediğiniz yolu seçerek kurabilirsiniz.Ben ilk denediğim ve işe yarayan yolu anlatacağım.

!!!! Dikkat aşağıda yapılacaklar eğer bilgisayarınızda ROS yüklüyse ROS'u bozacaktır. O yüzden ros yüklemeden önce kurunuz ya da Cmake versiyonunuz yetiyorsa güncellemeyin.

1-sudo apt-get update yazarak sistemimizdeki paketleri güncelleyelim.

2- sudo apt purge cmake yazarak cmake ile ilgili heşeyi siliyoruz.

3- https://cmake.org/download/ sitesinden cmakenin en son sürümünün .tar.gz uzantılı dosyasının linkini kopyalıyoruz.(!!2 tane .tar.gz uzantılı dosya var biz Linux-x86_64 olanı değil sadece cmake-x.xx.x.tar.gz dosyasının linkini kopyalıcaz.)

4-Yeni bir terminal ekranı açıp wget *kopyaladığımız_link* yazarak dosyayı indiriyoruz.

5- tar -zxvf cmake-x.xx.x.tar.gz(x yazan yerler sizin versiın sürümünüz) yazarak arşivden çıkarıyoruz.

Ve cd cmake-x.xx.x yazarak çıkarılan dosyanın konumuna gidiyoruz.

6-sudo ./bootstrap yazarak kurulama başlıyoruz işlem bittiğinde sudo make yazarak devam ediyoruz ve son olarak sudo make install yazarak işlemi sonlandırıyoruz.

Hatalar:

HATA:Could not find OpenSSL. Install an OpenSSL development package or configure CMake with -DCMAKE_USE_OPENSSL=OFF to build without OpenSSL.

Hatanın sebebi OpenSSL kütüphanesinin yüklü olmadığından kaynaklanıyor

sudo apt-get install libssl-dev yazarak OpenSSL kütüphanesini kurunca sorun düzelecektir.

CUDA Kurma

1- https://www.nvidia.com.tr/drivers sitesinden ekran kartı özelliklerimizi girerek nvidia sürücümüzü buluyoruz. Eğer ekran kartınız yüklüyse terminale nvidia-smi yazarak da bilgisayarına yükleyebileceğiniz en yüksek cuda sürümünü bulabilirsiniz. Yada sürücü numaranıza göre aşağıdan kullanabileceğiniz maksimum CUDA sürümünü bulabilirsiniz.

2-

Table 3. CUDA Toolkit and Corresponding Driver Versions

CUDA Toolkit	Toolkit Driver Version	
	Linux x86_64 Driver Version	Windows x86_64 Driver Version
CUDA 11.5 GA	>=495.29.05	>=496.04
CUDA 11.4 Update 2	>=470.57.02	>=471.41
CUDA 11.4 Update 1	>=470.57.02	>=471.41

Table 3. CUDA Toolkit and Corresponding Driver Versions

CUDA Toolkit	Toolkit Driver Version	
	Linux x86_64 Driver Version	Windows x86_64 Driver Version
CUDA 11.4.0 GA	>=470.42.01	>=471.11
CUDA 11.3.1 Update 1	>=465.19.01	>=465.89
CUDA 11.3.0 GA	>=465.19.01	>=465.89
CUDA 11.2.2 Update 2	>=460.32.03	>=461.33
CUDA 11.2.1 Update 1	>=460.32.03	>=461.09
CUDA 11.2.0 GA	>=460.27.03	>=460.82
CUDA 11.1.1 Update 1	>=455.32	>=456.81
CUDA 11.1 GA	>=455.23	>=456.38
CUDA 11.0.3 Update 1	>= 450.51.06	>= 451.82
CUDA 11.0.2 GA	>= 450.51.05	>= 451.48
CUDA 11.0.1 RC	>= 450.36.06	>= 451.22
CUDA 10.2.89	>= 440.33	>= 441.22

Table 3. CUDA Toolkit and Corresponding Driver Versions		
CUDA Toolkit	**Toolkit Driver Version**	
	Linux x86_64 Driver Version	**Windows x86_64 Driver Version**
CUDA 10.1 (10.1.105 general release, and updates)	>= 418.39	>= 418.96
CUDA 10.0.130	>= 410.48	>= 411.31
CUDA 9.2 (9.2.148 Update 1)	>= 396.37	>= 398.26
CUDA 9.2 (9.2.88)	>= 396.26	>= 397.44
CUDA 9.1 (9.1.85)	>= 390.46	>= 391.29
CUDA 9.0 (9.0.76)	>= 384.81	>= 385.54
CUDA 8.0 (8.0.61 GA2)	>= 375.26	>= 376.51
CUDA 8.0 (8.0.44)	>= 367.48	>= 369.30
CUDA 7.5 (7.5.16)	>= 352.31	>= 353.66
CUDA 7.0 (7.0.28)	>= 346.46	>= 347.62

3-https://developer.nvidia.com/cuda-toolkit-archive sitesinden uygun cuda sürümünü seçiyoruz ve .run uzantılı dosyasını indiriyoruz.

4-sudo sh cuda_xx.x.x_xxx.xx.xx_linux.run –—silent yazarak CUDA 'yı kuruyoruz.

5-En son terminal ekranında yazanlar ekran kartınızın sürücüsünün kurulmadığı sadece CUDA' nın kurulduğundan bahsediyor zaten ekran kartınız kurulu ise sıkıntı yok.

6-Terminal ekranı açıp `sudo gedit ~/.profile` yazıp dosyamızı açıyoruz. Ve en alt kısma gelip

```
# set PATH for cuda installation
if [ -d "/usr/local/cuda/bin/" ]; then
export PATH=/usr/local/cuda/bin${PATH:+:${PATH}}
export
LD_LIBRARY_PATH=/usr/local/cuda/lib64${LD_LIBRARY_PATH:+:${LD_LIB
RARY_PATH}}
fi
```

Komutlarını yapıştırıyoruz bu bize CUDA'nın dizinin vericektir.

7-Bilgisayarı yeniden başlatıp terminal ekranına `nvcc –version` yazarak cuda versiyonunuzu öğrenebilirsiniz.

8-Eğer bir sıkıntı yaşarsanız ilk başta gidip yukarda eklediğimiz CUDA dizinini kontrol edin. O dizinde CUDA dosyanız 'cuda' isminde değilse. O dosyanın o konumda kısayolunu oluşturup ismini 'cuda' yapın.

CUDNN Kurma

1- https://developer.nvidia.com/rdp/cudnn-download sitesinden cuda sürümünüze uygun CUDNN'nin son sürümünü seçiyoruz.

2-Bizden üyelik isteyecektir giriş yaptıktan sonra

cuDNN Runtime Library for Ubuntu20.04 x86_64 (Deb) ,

cuDNN Developer Library for Ubuntu20.04 x86_64 (Deb)

dosyalarını indiriyoruz.

3-İndirdikten sonra

sudo dpkg -i libcudnn8_8.2.0.53-1+cuda11.3_amd64.deb
sudo dpkg -i libcudnn8-dev_8.2.0.53-1+cuda11.3_amd64.deb
sudo dpkg -i libcudnn8-samples_8.2.0.53-1+cuda11.3_amd64.deb

terminale girip bu komutlar ile yazılımları yarıyoruz.

4- Cudnn 'yş test etmek için

cp -r /usr/src/cudnn_samples_v8/ $HOME

cd $HOME/cudnn_samples_v8/mnistCUDNN/
make
./mnistCUDNN

Hatalar

```
test.c:1:10: fatal error: FreeImage.h: No such file or directory
    1 | #include "FreeImage.h"
      |          ^~~~~~~~~~~~~
compilation terminated.
```

Sorun aşağıdaki kütüphaneler yüklü olamamasından kaynaklanıyor.

sudo apt-get install libfreeimage3 libfreeimage-dev

yazıp eksik kütüphaneleri yüklüyoruz.

OpenCV Yükleme

1-sudo apt-get update

sudo apt-get upgrade komutları ile bilgisayarımızdaki paketleri güncelliyoruz.

2- sudo apt-get install build-essential cmake git libgtk2.0-dev pkg-config libavcodec-dev libavformat-dev libswscale-dev

sudo apt install build-essential cmake git pkg-config libgtk-3-dev \ libavcodec-dev libavformat-dev libswscale-dev libv4l-dev \ libxvidcore-dev libx264-dev libjpeg-dev libpng-dev libtiff-dev \ gfortran openexr libatlas-base-dev python3-dev python3-numpy \ libtbb2 libtbb-dev libdc1394-22-dev libopenexr-dev \ libgstreamer-plugins-base1.0-dev libgstreamer1.0-dev

Komutları ile gerekli paketleri yüklüyoruz.

3- mkdir ~/opencv_build && cd ~/opencv_build

4- git clone https://github.com/opencv/opencv.git

git clone https://github.com/opencv/opencv_contrib.git

Yazarak gerekli dosyaları gitten çekiyoruz.

5-cd ~/opencv_build/opencv

mkdir -p build && cd build

6- `cmake -D CMAKE_BUILD_TYPE=RELEASE \ -D CMAKE_INSTALL_PREFIX=/usr/local \ -D INSTALL_C_EXAMPLES=ON \ -D INSTALL_PYTHON_EXAMPLES=ON \ -D OPENCV_GENERATE_PKGCONFIG=ON \ -D OPENCV_EXTRA_MODULES_PATH=~/opencv_build/opencv_contrib/modules \ -D BUILD_EXAMPLES=ON ..`

7-sudo make –j8

sudo make install

pkg-config --modversion opencv komutunu yazarak opencv sürümünüzü kontrol edebilirsiniz.Eğer kurulmadıysa .sh dosyası ile yüklemeyi deneyebilirsiniz.

Ben https://github.com/jayrambhia/Install-OpenCV dosyası ile yüklemiştim kullanımda gayet basit paketi indirip arşivden çıkarttıktan sonra paketteki ubuntu klasörüne gelip terminal açıyoruz ve chmod +x * yazarak dizindeki tüm dosyalara izin veriyoruz.Ardından sudo ./opencv_latest.sh yazarak opencvnin son sürümünü yüklüyoruz.

Eğer bunlar yardımcı olmadıysa https://www.youtube.com/watch?v=0vjC2UHptU4 adresindeki videoyu izleyerek deneyebilirsiniz.

GCC Yükleme

GCC:GNU C Compilerin açılımıdır.

1- sudo apt install build-essential

sudo apt-get install manpages-dev komutları ile gerekli paketleri ve GCC'yi yüklüyoruz.

2- gcc –version komutu ile GCC'nin cersiyonunu kontrol ediyoruz.

AlexAB Darknet'in kurulması;

Terminale git clone https://github.com/AlexeyAB/darknet.git yazarak ya da direk https://github.com/AlexeyAB/darknet sitesinden el ile çekerek darknet dosyasını indirin.

Ardından arşivden çıkardığımız dosyanın içindeki "MakeFile" dosyasını açıyoruz ve şu kısımları değiştiriyoruz.

- GPU=1 (darknet'i GPU 'da çalıştırmak için.)
- CUDNN=1 (CUDNN kullanarak daha hızlı bir train ve test için)
- OPENCV=1 (kamera üzerinden test yapmak için)
- DEBUG=1 (YOLO nun debug versiyonu için)
- OPENMP=1 (Birden fazla işlemcide yoloyu kalibre etmek için)
- LIBSO=1 (darkneti bir kütüphane gibi kullanmak için)

Ardından dizinde bir terminal ekranı açarak

```
mkdir build-release
cd build-release
cmake ..
make
make install
```

```
```

komutunu çalıştırıyoruz.

Eğitim Başlatma

1-“darknet-master\build\darknet\x64\cfg” dizinindeki “yolov3.cfg” dosyasını darknet.exe ‘nin yanına kopyalayıp “yolov3-obj.cfg” isminde kaydediyoruz.

2-.cfg dosyamızı not defteri yardımıyla açıyoruz ve aşağıdaki değişiklikleri uyguluyoruz.

- Testing kısmının altındaki batch=64
- Testing kısmının altındaki subdivisions = 16
- max_batches = 2000*(sınıf sayısı)
- steps = %80*max_batches,%90*max_batches
- width=416
- height=416
- classes=sınıf sayısı➔2 tane classes var 2’ünüde değiştiriyoruz (yolov3 için 3 tane classes var)
- filters=255 olanları (sınıf sayısı + 5)*3 olarak değiştiriyoruz.(Dikkat edelim sadece filters=255 olanları değiştiriyoruz diğerlerine dokunmuyoruz)

3-“build\darknet\x64\data\” dizininde “obj.names” adlı dosya oluşturuyoruz ve her satırına sınıf isimlerini yazıyoruz.(Etiketlemede yapılan sırayla yazın.)

4--“build\darknet\x64\data\” dizinide “obj.data” adlı dosya oluşturuyoruz ve içine

```
classes= sınıf sayısı
train  = data/train.txt
valid  = data/test.txt
names = data/obj.names
backup = backup/
```

classes kısmına sınıf sayısını yazıp kopyalıyoruz.

5-“build\darknet\x64\data\obj\” dizinin eğitimde kullanıcağımız datasetimizi koyuyoruz.

6-“build\darknet\x64” dizininde python dosyası oluşturuyoruz ve içine

```
import os
path='data/obj/'
imgList =os.listdir("data/obj/")
print(imgList)
textFile = open("train.txt","w")

for img in imgList:
  if ".jpg" in img:
    imgPath = path + img +"\n"
    textFile.write(imgPath)
```

kodunu kopyalıyoruz bu kod bize -“build\darknet\x64\data\obj\” dizinindeki dosyaların isimlerini “train.txt” adlı bir dosyaya yazdıracak.Ardından bu “train.txt” dosyasını açıp en sonundaki boş satırı siliyoruz ve data klasörünün içine taşıyoruz.

7- Train klasöründeki yolov3-tiny.conv.11 yolov3-tiny.conv.15 dosyasını build\darknet\x64" dizinine kopyalıyoruz.(Bu dosya ylov3-tiny içindir yolov3 için http://pjreddie.com/media/files/darknet53.conv.74 linkindeki dosyayı kullanıp kullanınız)

8-"build\darknet\x64" dizininde bir cmd ekranı açıp

"darknet.exe detector train data/obj.data yolov3-tiny-obj.cfg yolov3-tiny.conv.15" yapıştırıp eğitime başlatıyoruz.

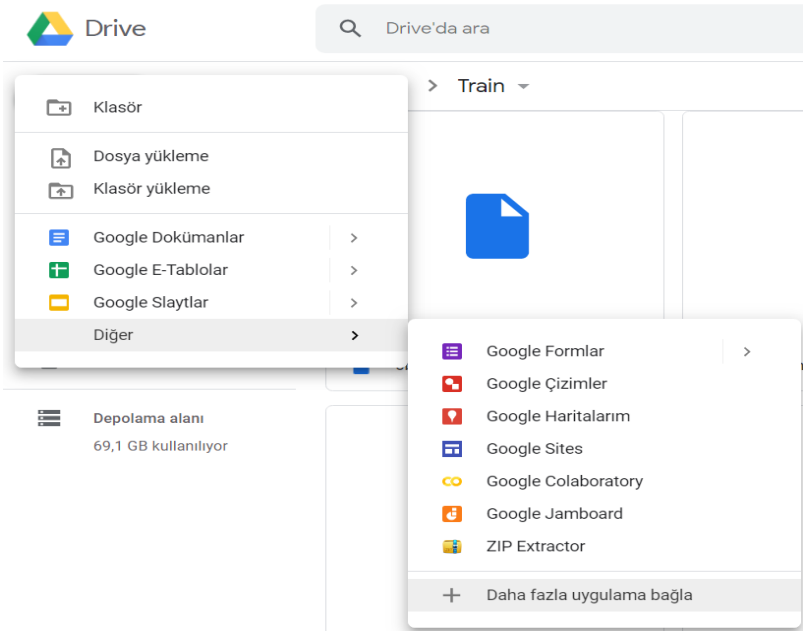
- -dont_show parametresi ile avg_loss değerini gösteren grafik açılmaz.
- -gpu 0.x parametresi ile ekran kartının yüzde kaçının kullanılacağının ayarlıyoruz eğer bu parametre kullanılmazsa eğitimi gpu nun tamamı kullanılarak yapılır buda ekran kartınızı yakabilir o yüzden 0.5 idealdir.

Not: Eğer Out of memory hatası alırsanız başta değiştirdiğmiz .cfg dosyasındaki subdivisions=16'yı 32 yada 64 olarak değiştirin.

2.Not: Bu eğitim yolov3-tiny içindir eğer yolov3 için eğitim başlatmak isterseniz 1. Ve 7. Adımda bahsedilen yerleri değiştirmeniz yeterlidir.

Google COLAB Üzerinde Eğitim başlatma

Drive'nıza ilk girdiğinizde Google COLAB yüklü halde gelmez ilk başta onu drivenıza kurmanız gerekir.



Drivenıza girdiğinizde "Yeni" kısmından Diğer ardından Daha fazla uygulma bağla sekmesine tıkladığımızda açılan pencereden "Colaboratory" diye arattığımızda çıkan uygulamyaı yüklüyoruz.

Ardından yine “Yeni” butonundan Diğer sekmesine gelip ordan “Google Colaboratory”’ye tıklıyoruz ve bize yeni bir notebook sayfası açıyor.

Sol üst kısımda bulunan “Untitled1.ipynb” dosyamızın ismi ona tıklayarak onun ismini değiştirebiliriz.

Yine sol üst kısımda bulunan “Edit” ten “Notebook Settings” ‘ e tıklayarak Hardware accelerator kısmını GPU yapıyoruz.

```
#Kimlik doğrulama
!apt-get install -y -qq software-properties-common python-software-
properties module-init-tools
!add-apt-repository -y ppa:alessandro-strada/ppa 2>&1 > /dev/null
!apt-get update -qq 2>&1 > /dev/null
!apt-get -y install -qq google-drive-ocamlfuse fuse
from google.colab import auth
auth.authenticate_user()
from oauth2client.client import GoogleCredentials
creds = GoogleCredentials.get_application_default()
import getpass
!google-drive-ocamlfuse -headless -id={creds.client_id} -
secret={creds.client_secret} < /dev/null 2>&1 | grep URL
vcode = getpass.getpass()
!echo {vcode} | google-drive-ocamlfuse -headless -
id={creds.client_id} -secret={creds.client_secret}
from google.colab import drive
drive.mount('/content/drive')
```

İlk yüklediğimiz kod bu oluyor. Bunu çalıştırdığımızda bizden 3 kere kimlik doğrulamamızı isteyecektir. Çıkan bağlantılara tıklayarak gelen kodu yapıştırıp işlemimize devam ediyoruz.

```
#Darknetin Kurulması

!git clone https://github.com/AlexeyAB/darknet

!apt-get update
!apt-get upgrade


!apt-get install build-essential
!apt-get install cmake git libgtk2.0-dev pkg-config libavcodec-
dev libavformat-dev libswscale-dev

!apt-get install libavcodec-dev libavformat-dev libswscale-d

!apt-get install libopencv-dev
```

```
%cd darknet

!ls
!sed -i 's/OPENCV=0/OPENCV=1/g' Makefile
!sed -i 's/GPU=0/GPU=1/g' Makefile




!ls
%cd ../
!ls

!apt install g++-5
!apt install gcc-5

!apt update
!apt upgrade

import tensorflow as tf
device_name = tf.test.gpu_device_name()
print(device_name)

print("'sup!'")

!/usr/local/cuda/bin/nvcc --version

%cd darknet
!wget https://pjreddie.com/media/files/yolov3.weights
!make

!./darknet detect cfg/yolov3.cfg yolov3.weights data/person.jpg

def imShow(path):
  import cv2
  import matplotlib.pyplot as plt
  %matplotlib inline

  image = cv2.imread(path)
  height, width = image.shape[:2]
  resized_image = cv2.resize(image,(3*width, 3*height), interpolation =
 cv2.INTER_CUBIC)

  fig = plt.gcf()
  fig.set_size_inches(18, 10)
  plt.axis("off")
  #plt.rcParams['figure.figsize'] = [10, 5]
```

```
  plt.imshow(cv2.cvtColor(resized_image, cv2.COLOR_BGR2RGB))
  plt.show()
imShow('predictions.jpg')
```

Google Colabrotory'e AlexAB nin darknetini kuruyoruz.

```
!unzip -q /content/drive/'My Drive'/obj.zip
 -d /content/darknet/build/darknet/x64/data
```

Komutu ile zip halinde driveye yüklediğimiz datasetimizi zipten çıkarıyoruz.Siz burda -q yazan kısımdaki path'i datasetinizin drivedeki yerine göre değiştirebilirsiniz.

→trainwriter.py

```
import os
path='/content/darknet/build/darknet/x64/data/obj/'
imgList =os.listdir("/content/darknet/build/darknet/x64/data/obj/")
print(imgList)
textFile = open("/content/darknet/build/darknet/x64/data/train.txt","w")

for img in imgList:
  if ".jpg" in img:
    imgPath = path + img +"\n"
    textFile.write(imgPath)
```

Kodunu bir "trainwriter.py" komut satırı açıp kopyalıyoruz.(Bu kodun üstteki bilgisayarda eğitim yaparken "trainwriter.py" dosyası ile farklı olduğunu belirtmek isterim)

Eğitimde kullanıcağımız cfg dosyasını oluşturuyoruz.(Windowsta eğitim başlatma kısmında anlatılmıştır.)

"obj.names" ve "obj.data" dosyalarını oluşturuyoruz.(Windowsta eğitim başlatma kısmında anlatılmıştır.)

Google Colabrotory'e gelerek drivede "Train" adlı bir dosyanın içine hazırladığımız "trainwriter.py",.cfg dosyası .conv dosyası "obj.names", "obj.data" dosyalarını atıyoruz.

```
#Drivedeki Train dosyamızdaki data dosyalarının taşınması

!cp /content/drive/'My Drive'/Train/trainwriter.py /content/darknet/bui
ld/darknet/x64/
!cp /content/drive/'My Drive'/Train/yolov3-tiny-
obj.cfg /content/darknet/build/darknet/x64/
!cp /content/drive/'My Drive'/Train/yolov3-
tiny.conv.15 /content/darknet/build/darknet/x64/
!cp /content/drive/'My Drive'/Train/obj.names /content/darknet/build/da
rknet/x64/data/
```

```
!cp /content/drive/'My Drive'/Train/obj.data /content/darknet/build/dar
knet/x64/data/
```

Bu kodu çalıştırarak drivedeki dosyalarımızı Colab dosyalarının içine kopyalıyoruz.

```
!python3 /content/darknet/build/darknet/x64/trainwriter.py
```

Komutu ile trainwriter programını çalıştırıyoruz.

```
!./darknet/darknet detector train /content/darknet/build/darknet/x64/da
ta/obj.data /content/darknet/build/darknet/x64/yolov3-tiny-
obj.cfg /content/darknet/build/darknet/x64/yolov3-tiny.conv.15 -
dont_show
```

Komutu ile en baştan bir eğitim başlatıyoruz.

```
!cp /content/darknet/build/darknet/x64/backup/yolov3-tiny-
obj_(iterations).weights /content/drive/'My Drive'/Train/
```

Komutu ile biten eğitimi drivemize kaydediyoruz

```
!./darknet detector train /content/darknet/build/darknet/x64/data/obj.d
ata /content/darknet/build/darknet/x64/yolov3-tiny-
obj.cfg /content/darknet/build/darknet/x64/backup/yolov3-tiny-
obj_(iterations).weights -dont_show
```

Dökümana ilerde eklenicekler

*YOLO, OpenCV, DarkNet, CUDA, Cudnn gibi kullandığımız yazılımların tanıtımı yapılacak.

*Darkneti nasıl ROS ile çalıştırırız ve en sağlıklı şekilde ROS'u sistemimize nasıl kurarız.

*Alexey Darknetindeki çoğu komut eklenecek.

*Darknet'i kullanarak nasıl python kodu yazarız.

*Alexey'in dışındaki DarkNet'ler incelenecek.

*Güzel bir kapak hazırlanacak :))) Burası aşırı önemli