

YTensorflow

- Copyright © ~ Yunus Emre AK

Döküman Renklendirme Yapısı

PDF Başlığı

Ana Başlıklar

Alt Başlıklar

İç Başlıklar

En İç Başlıklar

Tablo Başlığı

Bağlantılar

Değişmez ifadeler

Formüller

Önemli notlar

Terimsel ifadeler

Yorum satırları



İçerikler

- Temel Bilgileri
 - Hangi İşletim Sistemi Daha iyi
- Tensorflow Kurulumu
 - Anaconda Kurulumu
 - Tensorflow CPU veya GPU Kurulumu
 - Sanal Ortam Oluşturma ve Üzerine Kurma
 - Kurulumu Test Etme
- Tensorflow Algılama Modellerinin Kurulumu
 - Gerekli Paketlerin Kurulumları
 - Linux için OpenCv Kurulumu
 - Script Dosyaları için Gerekli Modüller
 - Tensorflow Models İndirilmesi
 - Models Klasörü Yapısı
 - Protobuflarların İşlenmesi
 - Obje Algılama Kütüphanelerinin Derlenmesi ve Yüklmesi
 - Gerekli Ortam Değişkenlerinin Tanımlanması
 - Anaconda Ortamı için Otomatik Tanımlama
 - Windows için Otomatik Tanımlama
 - Linux için Otomatik Tanımlama
 - Modellerin Kurulumunu Test Etme
- Labellmg Kurulumu
 - Labellmg Kaynak Kodlarını Derleme
 - Labellmg için Sanal Ortam Oluşturma
 - Labellmg Paketlerini Kurma ve Derleme
 - Labellmg Kurulumunu Test Etme
- Dizin Yapısını Oluşturma
 - Tensorflow Dizinizi Geçici Ortam Değişkenlerine Ekleme
 - Tensorflow Dizinizi Kalıcı Olarak Ortam Değişkenlerine Ekleme
 - Temel Klasörlerin Oluşturulması
 - Temel Dizin Yapısı
 - Çalışma Alanı Yapısı
 - Data Dizini Yapısı
 - Models Dizini Yapısı
- Özelleştirilmiş Tensorflow Obje Algılayıcısı Eğitme
 - Resim Etiketleme İşlemi
 - Derlenmiş Labellmg
 - Python ile Labellmg
 - Etiket Yollarını veya Adlarını Düzenleme
 - Etiket Haritası Oluşturma
 - Tensorflow Kayıtları Oluşturma
 - Resimlerdeki Hataları Bulma
 - Verileri Yeniden Adlandırma ve XML Hatalarını Düzeltme
 - Etiketlenmemiş Resimleri Bulma

- XML'i CSV'ye Çevirme
- CSV'lerden Resim Bilgilerini Analiz Etme
- CSV'yi Record'a Çevirme
- Bağlantıları (pipeline) Yapılandırma
 - Modellen İndirilmesi ve Gerekli Yere Taşınması
 - Modelin Yapılandırma Dosyaları
 - Modelin Yapılandırma Dosyasını Düzenleme
- Modeli Eğitme
 - Eğitim Scriptlerini Çalışma Alanına Kopyalama
 - Eğitimde Raporlanacak Seviyeyi Ayarlama (isteğe Bağlı)
 - Modeli train.py Dosyası ile Eğitime
 - Eğitime Başladığında Gelen Örnek Çıktı
 - Modeli model_main.py Dosyası ile Eğitme
 - Eğitim için Gereksinimlerin Kurulması
 - Windows için PyCocoTools Kurulumu
 - Linux için Cocotools
 - Eğitimi Hazırlama ve Başlatma
 - Eğitimi Etkileyen Faktörler
 - Eğitim İşlemini TensorBoard Kullanarak Takip Etme
 - Sonuç Grafiğini Dışarı Aktarma
- Hata Notları ve Açıklamaları
 - 'conda' is not recognized as an internal or external command
 - '...' is not recognized as an internal or external command
 - 'ImportError: No module named' Hataları
 - 'dict_keys' object does not support indexing
 - Object was never used (type <class 'tensorflow.python.framework.ops.Tensor'>)
 - 'unicodeescape' codec can't decode bytes in position
 - Allocation of X exceeds 10% of system memory
 - google.protobuf.text_format.ParseError, Expected string but found
 - Value Error: No Variable to Save
- Colab Üzerinden Tensorflow Modelini Eğitme
 - Colab Eğitimi için Gereken Dosyalar
 - Colab Üzerinden Eğitim Kodları
- Web Kamerası Kullanarak Obje Tespit Etme
- Harici Bağlantılar
 - Başlangıç için İdeal Olanlar
- Önemli Notlar
- Yapılacaklar
 - Sonra Yapılacaklar
 - Sonradan Eklenecek Scriptleştirme
 - Sonradan Derlenecek Bilgiler
 - TF Verilerini Alma
 - Recover the images from the TFRecord file
- Lisans ve Teferruatlar

Temel Bilgileri

- Python dili üzerinde makine öğrenimi gibi işlemler için Google tarafından sunulan kütüphanedir.
- Yabancı Kaynaklar: [1](#) [2](#)

Hangi İşletim Sistemi Daha iyi

Linux daha iyidir [1](#)

Kaynak için [buraya](#) bakabilirsiniz.

Tensorflow Kurulumu

- Tensorflow anaconda üzerinden daha sağlıklı, taşınabilir ve verimli çalışabilmekte
- Anacondanın sanal ortamları, paketlerin çakışmasını engelleyecektir
- Anaconda'nın tensorflowdaki avantajı için [buraya](#) göz atabilirsiniz.

Anaconda Kurulumu

- Anaconda kurulumu için [buraya](#) tıklayarak onun için hazırladığım dökümana erişebilirsiniz.

Tensorflow CPU veya GPU Kurulumu

- Bu kurulum CPU kurulumu olarak da geçmekte
- GPU kurulumu CPU'ya nazaran oldukça hızlı eğitim seçeneği sağlar
- GPU kurulumu için gereksinimleri sağlıyorsanız GPU kurulumu (tensorflow-gpu) yapmanız tavsiye edilir

Sanal Ortam Oluşturma ve Üzerine Kurma

```
conda create -n tensorflow tensorflow # CPU kurulumu
conda create -n tensorflow tensorflow-gpu # GPU kurulumu
```

Kurulumu Test Etme

Altaki komnut ile 'Hello, TensorFlow!' çıktısını almanız gerekmektedir.

```
python -c
>>> import tensorflow as tf
>>> hello = tf.constant('Hello, TensorFlow!')
>>> sess = tf.Session()
>>> print(sess.run(hello))
```

Tensorflow Algılama Modellerinin Kurulumu

- Algılama modelleri tabloma erişmek için [buraya](#) tıklayabilirsiniz
 - Resmi sitesi için [buraya](#) bakabilirsiniz
- Video üzerinden açıklama için [buraya](#) bakabilirsiniz

Resmi açıklamalar [models/research/object_detection/g3doc](#) dizinindedir.

Gerekli Paketlerin Kurulumları

Tensorflow modellerini kullanabilmek için alttaki kurulumlara da ihtiyaç olabilmekte:

```
conda install opencv pillow matplotlib pandas jupyter
```

Modül bulunamaması gibi durumlarda [lxml](#), [protobuf](#) paketlerini yüklemeyi deneyebilirsiniz.

Linux için OpenCv Kurulumu

GTK ve FFMPEG hatasını engellemek için pip ile kurulum yapın

```
pip install opencv-contrib-python
```

Script Dosyaları için Gerekli Modüller

```
pip install pynput # detect_from_desktop
```

Tensorflow Models İndirilmesi

Altaki talimatlar ve komutlar yardımıyla tensorflow modellerini kurun:

- Modelleri indirmek için [buraya](#) tıklayabilirsiniz
- İstersen [buraya](#) tıklayarak github linkine erişebilirsiniz
- İndirdiğiniz dosyanın içindekileri [models](#) dizinine koymanız gerekmektedir.

Bu adından sonrası [models/research/](#) dizininde gerçekleştirilmelidir.

```
powershell.exe Expand-Archive models-master.zip .  
ren models-master models  
move models %TENSORFLOW%  
cd %TENSORFLOW%\models\research\
```

Models Klasörü Yapısı

```
+ models
  + offical
  + research
  + sample
  ...
```

Protobufların İşlenmesi

Protobuf dosyaları (`.proto` uzantılı olan dosyalar) python kodları oluşturmak için kullanılan dosyalardır.

`TensorFlow/models/research/` dizininde

Windows:

```
for /f %i in ('dir /b object_detection\protos\*.proto') do protoc
object_detection\protos\%i --python_out=.
```

Linux:

```
protoc object_detection/protos/*.proto --python_out=.
```

Protobuflarların işlenmesiyle `.py` uzantılı dosyalar oluşacaktır

Obje Algılama Kütüphanelerinin Derlenmesi ve Yüklenmesi

```
# TensorFlow/models/research/ dizininde
python setup.py build
python setup.py install
```

Gerekli Ortam Değişkenlerinin Tanımlanması

Eğer daha önceden tanımlı `PYTHONPATH` ortam değişkeniniz **yoksa ilk olan, varsa ikinci olan** komutu kullanın.

Bu ortam değişkenlerinin **terminalin her açılışında yazılması** gerekmektedir.

```
set
PYTHONPATH=%TENSORFLOW%\models\research;%TENSORFLOW%\models\research\slim;%TENSORFLOW%\models\research\object_detection
```

```
set
PYTHONPATH=%PYTHONPATH%;%TENSORFLOW%\models\research;%TENSORFLOW%\models\research\slim;%TENSORFLOW%\models\research\object_detection
```

Anaconda Ortamı için Otomatik Tanımlama

- Her `conda activate <ortam_ismi>` komutu yazıldığında ortamlar dahil edilir
- Her `conda deactivate` yazıldığında ortamlar kaldırılır

Windows için Otomatik Tanımlama

```
cd <conda_ortamı_yolu>
mkdir .\etc\conda\activate.d
echo set
PYTHONPATH=%TENSORFLOW%\models\research;%TENSORFLOW%\models\research\slim;%TENSORFLOW%\models\research\object_detection >
.\etc\conda\activate.d\env_vars.bat
```

Linux için Otomatik Tanımlama

Resmi kaynak için [buraya](#) bakabilirsiniz.

```
cd <conda_ortamı_yolu>
mkdir -p ./etc/conda/activate.d
mkdir -p ./etc/conda/deactivate.d
echo export
PYTHONPATH=${PYTHONPATH}:${TENSORFLOW}/models/research:${TENSORFLOW}/models/research/slim:${TENSORFLOW}/models/research/object_detection >
etc/conda/activate.d/env_vars.sh
echo unset PYTHONPATH > etc/conda/deactivate.d/env_vars.sh
```

- `<conda_ortamı_yolu>` Conda ortamının kurulduğu yol
 - Örn: `%USERPROFILE%\Anaconda3\envs\tensorflow-cpu`

Modellerin Kurulumunu Test Etme

Jupyter notebook ile API'ları test etmemizi gerekmektedir.

```
cd object_detection
jupyter notebook
```

Jupyter notebook hakkında bilgi sahibi değilsen [buraya](#) tıklayarak ne yapman gerektiğini öğrenebilirsin.

Labellmg Kurulumu

- Labellmg tensorflow modelleri için etiketleme amaçlı kullanılmaktadır
- Derlenmiş sürümünü indirmek için [buraya](#) tıklayabilirsin

İndirilen dosyayı %TENSORFLOW%\addons dizinine atmanız daha verimli bir çalışma sağlayacaktır.

Labellmg Kaynak Kodlarını Derleme

Labellmg için Sanal Ortam Oluşturma

Tensorflow ortamının alt paketlerini etkilememesi için ek bir sanal ortamda kurulum sağlamalıyız.

```
conda create -n labelImg pyqt # QT grafik kütüphanesi
conda activate labelImg
conda install -c anaconda lxml
```

Labellmg Paketlerini Kurma ve Derleme

Paketlerin kurulumu için alttaki talimatları sırayla uygulayın:

- Labellmg dosyalarını indirmek için [buraya](#) tıklayın
- Diğer işlemler için indirdiğiniz dosya dizininde cmd açıp alttaki komutları yazın

```
# labelImg-master.zip dizininde
powershell.exe Expand-Archive labelImg-master.zip .
ren labelImg-master labelImg
mkdir %TENSORFLOW%\addons
move labelImg %TENSORFLOW%\addons
cd %TENSORFLOW%\addons\labelImg
pyrcc5 -o resources.py resources.qrc # QT grafiklerinin oluşturulması
```

'pyrcc5' is not recognized as an internal or external command hatası gelirse, yüklediğiniz [pyqt](#) sürümüne göre komutu kullanın (`pyrcc<pyqt_sürümü_ilk_basamağı>`)

LabelImg Kurulumunu Test Etme

```
conda activate tensorflow-cpu
cd %TENSORFLOW%\addons\labelImg
python labelImg.py
# python labelImg.py [IMAGE_PATH] [PRE-DEFINED CLASS FILE]
```

Dizin Yapısını Oluşturma

Tensorflow Dizininizi Geçici Ortam Değişkenlerine Ekleme

Altteki komut yardımıyla açık olan cmd ekranına ortam değişkeni tanımlayabilirsiniz.

```
set TENSORFLOW=<dizin_yolu>
```

- **<dizin_yolu>** Tensorflow'u kurmak istediğiniz dizin
 - Örn: "C:\Tensorflow"

Tensorflow Dizininizi Kalıcı Olarak Ortam Değişkenlerine Ekleme

- Bilgisayarına sağ tıklayın **Ayarlar** kısmına girin
- Sol alanda **Gelişmiş Sistem Ayarları**'na tıklayın
- Açılan ekranda **Ortam Değişkenleri** butonuna tıklayın
- Üst kısımdaki kullanıcı değişkenleri alanında **Yeni** butonuna tıklayın
- Değişken ismine: **Tensorflow** Değerine: 'dizin yolunuzu' yazın

Temel Klasörlerin Oluşturulması

İlerideki yapı için bu dizinin yolu **%TENSORFLOW%** olarak ifade edilecektir.

Proje yapısı tavsiye edilen dizin yapısına örnek olacak şekilde oluşturulmuştur.

Düzgün ve verimli çalışmak için buradaki yapıyı kullanmanız önerilir.

```
mkdir %TENSORFLOW%\workspace\example_detection
mkdir %TENSORFLOW%\workspace\example_detection\data
mkdir %TENSORFLOW%\workspace\example_detection\images\train
mkdir %TENSORFLOW%\workspace\example_detection\images\test
mkdir %TENSORFLOW%\workspace\example_detection\models
```

Temel Dizin Yapısı

```
+ addons
+ docs
+ models
+ scripts
+ workspace
  + example_detection
    + data
    + models
      + <model_ismi>
        + eval
        + train
        - *.config
        ...
      + <model_ismi>
        + eval
        + train
        - *.config
        ...
      ...
    ...
  + example2_detection
    + data
    + models
      + <model_ismi>
        + eval
        + train
        - *.config
        ...
      + <model_ismi>
        + eval
        + train
        - *.config
        ...
      ...
    ...
  ...
...
```

Dizin	Açıklama
addons	Labelling vs.
docs	Dökümanlar
models	Tensorflow Models dosyası
scripts	Kullanacağınız ortak kod parçaları
workspace	Çalışma Alanı

Çalışma Alanı Yapısı

```
+ workspace
  + example_detection
    + data
    + models
    ...
  + example2_detection
    + data
    + models
    ...
  ...
```

Dizin Açıklama

data	Eğitime katılacak verileri (<i>test.record</i> , <i>train.record</i> , <i>label_map</i>) içerir.
model	Eğitilecek modellerin dosyalarını içerir.

Data Dizini Yapısı

```
+ example_detection
  + data
    - label_map.pbtxt
    - test.record
    - train.record
  + models
  ...
...
```

Dosya Açıklama

<i>label_map.pbtxt</i>	Etiket haritası dosyası
<i>test.record</i>	Test için kullanılacak tensorflow kayıtları (TF record)
<i>train.record</i>	Eğitim için kullanılacak tensorflow kayıtları (TF record)

Models Dizini Yapısı

```
+ example_detection
  + data
  + models
    + <model_ismi>
      + eval
      + train
      - *.config
      ...
    + <model_ismi>
      + eval
      + train
      - *.config
      ...
    ...
  ...
...
```

Her bir model için ayrı dizinler oluşturulur.

İsim	Tipi	Açıklama
eval	Dizin	Test sonuçları burada tutulur.
train	Dizin	Eğitim çıktıları burada tutulur
.config	Dosya	Yapılandırma dosyası

Özelleştirilmiş Tensorflow Obje Algılayıcısı Eğitim

Özelleştirilmiş model eğitmek için alttakilerin yapılmış olması gerekmektedir:

- [Tensorflow CPU veya GPU kurulumu](#)
- [Tensorflow modellerinin kurulumu](#)
- [Labellmg kurulumu](#)

Resim Etiketleme İşlemi

Etiketleme işlemini **labellmg** üzerinden yapmaktayız.

Derlenmiş Labellmg

İndirdiğiniz dizindeki **labellmg.exe** dosyasını çalıştırmanız yeterlidir.

Python ile Labellmg

İşlemleri **Anconda Prompt** ile işlemler yapmalıyız.

```
conda activate labelImg
cd %TENSORFLOW%\addons\labelImg
python labelImg.py ../../workspace/example_detection\images # çıktıları
hedefleme
```

Labellmg kullanımı için [bu videoya](#) bakabilirsiniz.

Etiket Yollarını veya Adlarını Düzenleme

XML ve resim dosyalarını başka bir yolda oluşturduysanız alttaki script yardımıyla düzeltebilirsiniz

- Script dosyasını [buraya](#) tıklayarak indirmeli ve gerekli dizine alttaki komutla koymalıyız
- Komutları **Anaconda Prompt** üzerinden **tensorflow** ortamını aktif ederek uygulamayı unutmayın.

```
# Train verilerini yeniden adlandırma ve düzeltme
python xml_path_regulator.py -i
%TENSORFLOW%\workspace\example_detection\images\train -p train

# Test verilerini yeniden adlandırma ve düzeltme
python xml_path_regulator.py -i
%TENSORFLOW%\workspace\example_detection\images\test -p test
```

Etiket Haritası Oluşturma

- Alttaki komutla açılan dizinde **.pbtxt** uzantılı etiket haritası dosyası oluşturun
- Örnek dosya yapısı komutların altındadır.

```
cd %TENSORFLOW%\workspace\example_detection\annotations
start .
```

```
item {
  id: 1
  name: 'cat'
}
item {
  id: 2
  name: 'dog'
}
```

- **cat** ve **dog** etiket isimleridir

Tensorflow Kayıtları Oluşturma

- **Resim** verileri toplanır veya çekilir.
- Toplanan resimler **labelimg** yardımıyla etiketlenir ve **.xml** uzantılı dosyaları oluşturulur.
- **images** dizinine **resimler** ve onlara ait **xml** dosyaları %80'i test %20'i test olacak şekilde klasörlere ayrılarak yerleştirilir.
- **scripts/preprocessing** dizindeki **xml_path_regulator.py** scripti aracılığıyla xml ve resimlerde yol sorunları düzeltilir, veriler yeniden adlandırılır.
- **scripts/preprocessing** dizindeki **xml_to_csv.py** scripti aracılığıyla veriler **.csv** uzantılı tablosal bir dosyaya dönüştürülür.
- Oluşturulan **csv** dosyasında resimlerin etiketlerine göre sayıları **tablo** olarak gösterilir. (Excel yardımıyla)
- Verilerde denge durumunun (her veriden yaklaşık olarak aynı sayıda varsa) kontrolü yapılır.
- Her çeşit veri için bir **id** belirtilecek şekilde **label_map.pbtxt** adlı etiket haritası oluşturulur
- Oluşturulan **csv**, **etiket haritası** ve **resim** verileri **scripts/preprocessing** dizindeki **generate_tfrecord.py** scripti aracılığıyla veriler **.record** uzantılı kayıtlara dönüştürülür.
- Seçilen modele özgü yapılandırma dosyası indirilir.
- Yapılandırma dosyası olan ***.config** dosyasındaki **PATH_TO_CONFIGURED** olarak işaretlenen alanlar, **num_classes**, **num_examples** ve **batch_size** değerleri güncellenir.
 - **num_examples** test dizindeki resim sayısıdır (toplam class sayısı değil)

Resimlerdeki Hataları Bulma

Resimlerde hata olduğu zaman eğitim aşamasında tensorflow modeli çalışma hatası vermektedir. Resimleri kontrol etmek için [buradaki](#) scripti alttaki komutlarla kullanabilirsiniz.

```
python scripts\preprocessing\check_images.py -i
workspace\example_detection\images\train

python scripts\preprocessing\check_images.py -i
workspace\example_detection\images\test
```

Verileri Yeniden Adlandırma ve XML Hatalarını Düzeltme

Labellmg ile etiketlediğiniz resimleri farklı bir dizine taşımanız durumunda XML dosyalarındaki yollar uyuşmayacaktır. XML dosya yollarını düzeltmek, etiketsiz resimleri görüntülemek için [buradaki](#) script dosyamı alttaki komutlar ile kullanabilirsiniz.

```
python scripts\preprocessing\xml_path_regulator.py -i
%TENSORFLOW%\workspace\example_detection\images\train -p train

python scripts\preprocessing\xml_path_regulator.py -i
%TENSORFLOW%\workspace\example_detection\images\test -p test
```

Etiketlenmemiş Resimleri Bulma

Etiketlenmemiş resimleri [buradaki](#) script dosyası ile alttaki komutlar ile kullanabilirsiniz.

Eğer XML scriptini kullandıysanız bu kontrolü yapmanıza **gerek yoktur**, XML scripti bunu zaten yapmaktadır.

```
python scripts\preprocessing\find_unlabeled_imgs.py -i
%TENSORFLOW%\workspace\example_detection\images\train

python scripts\preprocessing\find_unlabeled_imgs.py -i
%TENSORFLOW%\workspace\example_detection\images\test
```

XML'i CSV'ye Çevirme

XML dosyalarını CSV dosyasında toplamak için [buradaki](#) scripti alttaki komutlar ile kullanabilirsin.

Komutları **Anaconda Prompt** üzerinden **tensorflow** ortamını aktif ederek uygulamayı unutmayın.

```
# Create train data:
python scripts\preprocessing\xml_to_csv.py -i
%TENSORFLOW%\workspace\example_detection\images\train -o
%TENSORFLOW%\workspace\example_detection\images\train_labels.csv

# Create test data:
python scripts\preprocessing\xml_to_csv.py -i
%TENSORFLOW%\workspace\example_detection\images\test -o
%TENSORFLOW%\workspace\example_detection\images\test_labels.csv
```

CSV'lerden Resim Bilgilerini Analiz Etme

Her bir etiketten kaç tane olduğunu anlamak için csv dosyalarını açıp alttaki yöntemi uygulayın.

- **class** hücreisinin bir altındaki hücreyi seçin
- **ctrl + shift + aşağı ok** ile tüm sınıf verilerini seçin
- Sağ alttaki butona tıklayın
- **Tables** sekmesine gelin
- Açılan sekmede **Pivot Table** butonuna tıklayın
- Tablo'dan etiketlenen verileri kontrol edin
- Fazladan etiketlenmiş verilerin ismini bulup, filename, width vs. verilerin yazıldığı alanda **CTRL + F** komutu ile aratıp, uygun dosya ismini ve **xml** dosyasını silin

	A	B	C	D	E	F	G	H	I
1	filename	width	height	class	xmin	ymin	xmax	ymax	
2	dur (1).jpg	1680	1050	hiz_yirmi	1419	139	1662	371	
3	dur (1).jpg	1680	1050	dur	942	459	960	497	
4	dur (1).jpg	1680	1050	trafige_ka	920	472	946	508	
5	dur (10).jpg	1680	1050	dur	944	453	974	489	
6	dur (100).jpg	1680	1050	dur	1116	339	1190	461	
7	dur (100).jpg	1680	1050	trafige_ka	972	431	1038	492	
8	dur (101).jpg	1680	1050	dur	1131	341	1185	451	
9	dur (101).jpg	1680	1050	trafige_ka	973	432	1041	496	
10	dur (102).jpg	1680	1050	dur	1136	325	1205	448	
11	dur (102).jpg	1680	1050	trafige_ka	973	430	1032	488	
12	dur (103).jpg	1680	1050	dur	1128	323	1213	457	
13	dur (103).jpg	1680	1050	trafige_ka	973	434	1045	489	
14	dur (104).jpg	1680	1050	dur	1150	323	1218	451	
15	dur (104).jpg	1680	1050	trafige_ka	981	437	1036	486	
16	dur (105).jpg	1680	1050	dur	1155	316	1229	447	
17	dur (105).jpg	1680	1050	trafige_ka	982	433	1040	489	
18	dur (106).jpg	1680	1050	dur	1159	307	1232	446	
19	dur (106).jpg	1680	1050	trafige_ka	988	434	1041	492	
20	dur (107).jpg	1680	1050	dur	1171	312	1243	435	
21	dur (107).jpg	1680	1050	trafige_ka	991	433	1043	492	
22	dur (108).jpg	1680	1050	dur	1176	296	1258	447	
23	dur (108).jpg	1680	1050	trafige_ka	982	426	1053	493	
24	dur (109).jpg	1680	1050	dur	1186	298	1265	439	

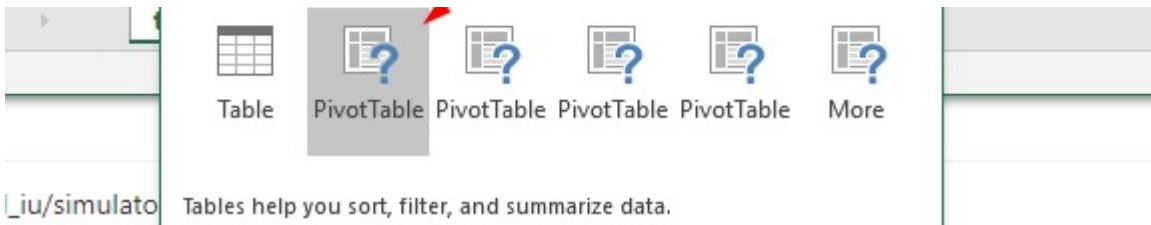
train_labels

A	B	C	D	E	F	G	H	I
yirmi_son	1680	1050	yirmi_son	937	402	1018	479	
yirmi_son	1680	1050	yirmi_son	942	396	1022	478	
yirmi_son	1680	1050	yirmi_son	948	400	1031	483	
yirmi_son	1680	1050	yirmi_son	950	392	1043	484	
yirmi_son	1680	1050	yirmi_son	953	399	1046	481	
yirmi_son	1680	1050	yirmi_son	956	397	1054	480	
yirmi_son	1680	1050	yirmi_son	968	391	1052	475	
yirmi_son	1680	1050	yirmi_son	967	389	1056	472	
yirmi_son	1680	1050	yirmi_son	972	388	1062	482	
yirmi_son	1680	1050	yirmi_son	962	443	1026	497	
yirmi_son	1680	1050	yirmi_son	979	393	1073	478	
yirmi_son	1680	1050	yirmi_son	974	383	1078	487	
yirmi_son	1680	1050	yirmi_son	984	396	1081	478	
yirmi_son	1680	1050	yirmi_son	982	388	1089	478	
yirmi_son	1680	1050	yirmi_son	991	374	1081	482	
yirmi_son	1680	1050	yirmi_son	994	389	1097	469	
yirmi_son	1680	1050	yirmi_son	1009	381	1108	479	
yirmi_son	1680	1050	yirmi_son	1015	375	1109	472	
yirmi_son	1680	1050	yirmi_son	1022	379	1117	468	
yirmi_son	1680	1050	yirmi_son	1020	373	1132	480	

2

3

Formatting Charts Totals Tables Sparklines



Row Labels	Count of class
dur	1075
durak	1185
gec	1154
giris_yasak	938
hiz_otuz	1037
hiz_yirmi	1153
park	1949
park_yasak	2405
sag_ileriden	1003
sag_yasak	1747
sol_ileriden	954
sol_yasak	1787
trafige_kapali	1447
yirmi_son	999
Grand Total	18833

The screenshot shows a Google Sheet with a dataset of image files. The 'filename' column (A) contains entries like 'dur(1).jpg', 'dur(10).jpg', etc. A red circle highlights the formula bar showing 'dur(1).jpg'. A red arrow points to the 'Find and Replace' dialog box, which is open and shows 'hiz' in the 'Find what' field. The dialog box has 'Find' and 'Replace' tabs, and a red arrow points to the 'Find what' field. The spreadsheet has columns A through Q and rows 1 through 24. The 'train_labels' tab is selected at the bottom.

CSV'yi Record'a Çevirme

CSV dosyalarını TF kayıtlarına çevirmek için [buradaki](#) scripti alttaki komutlar ile kullanabilirsiniz.

Komutları **Anaconda Prompt** üzerinden **tensorflow** ortamını aktif ederek uygulamayı unutmayın.

```
python generate_tfrecord.py --
label_map=%TENSORFLOW%\workspace\example_detection\data\label_map.pbtxt --
csv_input=%TENSORFLOW%\workspace\example_detection\images\train_labels.csv --
img_path=%TENSORFLOW%\workspace\example_detection\images\train --
output_path=%TENSORFLOW%\workspace\example_detection\data\train.record

python generate_tfrecord.py --
label_map=%TENSORFLOW%\workspace\example_detection\data\label_map.pbtxt --
csv_input=%TENSORFLOW%\workspace\example_detection\images\test_labels.csv --
img_path=%TENSORFLOW%\workspace\example_detection\images\test --
output_path=%TENSORFLOW%\workspace\example_detection\data\test.record
```

Bağlantıları (pipeline) Yapılandırma

- Tensorflow'un resmi açıklaması için [buraya](#) tıklayabilirsiniz

Modelin İndirilmesi ve Gerekli Yere Taşınması

- Tensorflow önceden eğitilmiş modelleri indirmek için [buraya](#) tıklayabilirsiniz
- .tar.gz** uzantılı olacağı için [winrar](#) ya da [7zip](#) gibi ek uygulamalarla [buraya](#) **çıkart** demen gerekmekte
 - Klasör'e** **çıkart** değil [buraya](#) **çıkart** diyeceksiniz.

Klasör içinde aynı isimde başka klasör olmasın

```
# Modelin çıkartıldığı dizinde
cd <model_ismi>
move * %TENSORFLOW%\workspace\example_detection\pre_trained_model
move saved_model %TENSORFLOW%\workspace\example_detection\pre_trained_model
cd %TENSORFLOW%\workspace\example_detection\pre_trained_model
```

- <model_ismi>** Seçip, indirdiğiniz **.tar.gz** uzantılı dosyanın adı
 - TAB** tuşu ile dizindeki dosya adlarını tamamlayabilirsiniz
 - *.tar.gz** uzantısı yazılmayacak
 - Örn: `ssd_inception_v2_coco_2018_01_28`
 - Örn: `ssd_mobilenet_v1_ppn_shared_box_predictor_300x300_coco14_sync_2018_07_03`

Modellin Yapılandırma Dosyaları

Seçtiğiniz modelin `*.config` dosyasını `example_detection/training` klasörü altına kopyalamanız gerekmektedir.

```
mkdir %TENSORFLOW%\workspace\example_detection\training

copy %TENSORFLOW%\models\research\object_detection\samples\configs\
<model_ismi>.config %TENSORFLOW%\workspace\example_detection\training
```

- `<model_ismi>` Seçip, indirdiğiniz `.tar.gz` uzantılı dosyanın adı
 - TAB tuşu ile dizindeki dosya adlarını tamamlayabilirsiniz
 - `*.tar.gz` uzantısı yazılmayacak
 - Tarih son ekini içermemeli
 - `*_2018_07_03.tar.gz` ise `*.tar.gz` olarak yazılmalı
 - Örn: `ssd_inception_v2`
 - Örn: `ssd_mobilenet_v1_ppn_shared_box_predictor_300x300_coco14_sync`

Modellin Yapılandırma Dosyasını Düzenleme

Yapılandırma örnek dosyası için [buraya](#) bakabilirsiniz.

Düzenlenecek Satır	Açıklama	Örnek
<code>num_classes</code>	Etiket türü sayısı	<code>2</code>
<code>batch_size</code>	Toplu işleme boyutu	<code>24</code>
<code>num_steps</code>	Adım sayısı	<code>2000</code>
<code>fine_tune_checkpoint</code>	Eğitilmiş modelin yolu	<code>"./pre_trained_model/model.ckpt"</code>
<code>label_map_path</code>	Etiket haritası yolu	<code>"./annotations/train.record"</code>
<code>input_path</code>	Train dosyası yolu	<code>"./annotations/train.record"</code>
<code>input_path</code>	Test dosyası yolu	<code>"./annotations/test.record"</code>

Modeli Eğitme

Modeli eğitmek için `train.py` script dosyasını kullanacağız.

Modeli önerilen dosya olan `model_main.py` ile eğitmek için [buraya](#) bakmalısın.

Eğitim Scriptlerini Çalışma Alanına Kopyalama

Çalışma ortamının düzgün ilerlemesi adına alttaki komut ile gerekli yere scripti kopyalayalım

```
copy %TENSORFLOW%\models\research\object_detection\legacy\train.py
%TENSORFLOW%\workspace\example_detection
copy %TENSORFLOW%\models\research\object_detection\model_main.py
```

Eğitimde Raporlanacak Seviyeyi Ayarlama (isteğe Bağlı)

Eğitimde uyarı ve bilgileri gizlemek için `TF_CPP_MIN_LOG_LEVEL` adlı ortam değişkeni oluşturup seviyesini tanımlıyoruz

```
set TF_CPP_MIN_LOG_LEVEL=2
```

Modeli train.py Dosyası ile Eğitime

TODO Daha düzgün ve detaylı linkli bir yazı ekle

Eskimiş olan bir eğitim kodudur, `model_main.py` kod dosyası tensorflow tarafından önerilir.

```
python train.py --logtostderr --train_dir=training/ --
pipeline_config_path=training\<yapılandırma_dosyası>
```

- `<yapılandırma_dosyası>` Modelimizin yapılandırma dosyasının tam adı
 - **training** klasörüne attığımız yapılandırma dosyaları
 - Örn: `ssd_inception_v2_coco.config`

Eğitime Başladığında Gelen Örnek Çıktı

```
INFO:tensorflow:depth of additional conv before box predictor: 0
INFO:tensorflow:depth of additional conv before box predictor: 0
INFO:tensorflow:depth of additional conv before box predictor: 0
INFO:tensorflow:depth of additional conv before box predictor: 0
INFO:tensorflow:depth of additional conv before box predictor: 0
INFO:tensorflow:depth of additional conv before box predictor: 0
INFO:tensorflow:Restoring parameters from
ssd_inception_v2_coco_2017_11_17/model.ckpt
INFO:tensorflow:Running local_init_op.
INFO:tensorflow:Done running local_init_op.
INFO:tensorflow:Starting Session.
INFO:tensorflow:Saving checkpoint to path training\model.ckpt
INFO:tensorflow:Starting Queues.
INFO:tensorflow:global_step/sec: 0
INFO:tensorflow:global step 1: loss = 13.8886 (12.339 sec/step)
INFO:tensorflow:global step 2: loss = 16.2202 (0.937 sec/step)
INFO:tensorflow:global step 3: loss = 13.7876 (0.904 sec/step)
INFO:tensorflow:global step 4: loss = 12.9230 (0.894 sec/step)
INFO:tensorflow:global step 5: loss = 12.7497 (0.922 sec/step)
INFO:tensorflow:global step 6: loss = 11.7563 (0.936 sec/step)
INFO:tensorflow:global step 7: loss = 11.7245 (0.910 sec/step)
INFO:tensorflow:global step 8: loss = 10.7993 (0.916 sec/step)
INFO:tensorflow:global step 9: loss = 9.1277 (0.890 sec/step)
INFO:tensorflow:global step 10: loss = 9.3972 (0.919 sec/step)
INFO:tensorflow:global step 11: loss = 9.9487 (0.897 sec/step)
INFO:tensorflow:global step 12: loss = 8.7954 (0.884 sec/step)
INFO:tensorflow:global step 13: loss = 7.4329 (0.906 sec/step)
INFO:tensorflow:global step 14: loss = 7.8270 (0.897 sec/step)
INFO:tensorflow:global step 15: loss = 6.4877 (0.894 sec/step)
```

Modeli model_main.py Dosyası ile Eğitme

Bu dosya ile eğitim önerilen eğitim şeklidir.

- `train.py` ile eğitime nazaran, kaldığı yerden devam eder
 - 1000 adım yapıldıysa, ikinci eğitimi 1200 yaptığınızda 200 adım eğitir
 - `train.py` eğitiminde modelin sonucunun ayrılıp, sonuç üzerinden eğitim yapılması gerekir

Eğitim dosyaları arasında **performans veya kalite farkı yoktur**, kaynak için [buraya](#) bakabilirsiniz.

Eğitim için Gereksinimlerin Kurulması

Eğitim için `pycocotools` kurulumu gereklidir

Windows için PyCocoTools Kurulumu

Windows desteğiyle kurulum yapmak için alttaki komutu koşturun

```
pip install  
git+https://github.com/philferriere/cocoapi.git#subdirectory=PythonAPI
```

Açıklama için [buraya](#) bakabilirsiniz.

Linux için Cocotools

```
git clone https://github.com/cocodataset/cocoapi.git  
cd cocoapi/PythonAPI  
make  
cp -r pycocotools /content/models/research/  
cd ../..  
rm -rf cocoapi
```

Eğitimi Hazırlama ve Başlatma

Resmi kaynağa ulaşmak için [buraya](#) bakabilirsiniz.

- `model_main.py` eğitim için önerilen dosyadır
- Varsayılan olarak ekrana raporlama yapmaz, yapmasını isterseniz [buraya](#) bakabilirsiniz

```
# From the tensorflow/models/research/ directory
PIPELINE_CONFIG_PATH={path to pipeline config file}
MODEL_DIR={path to model directory}
NUM_TRAIN_STEPS=50000
SAMPLE_1_OF_N_EVAL_EXAMPLES=1
python object_detection/model_main.py \
    --pipeline_config_path=${PIPELINE_CONFIG_PATH} \
    --model_dir=${MODEL_DIR} \
    --num_train_steps=${NUM_TRAIN_STEPS} \
    --sample_1_of_n_eval_examples=${SAMPLE_1_OF_N_EVAL_EXAMPLES} \
    --alsologtostderr
```

Eğitimi Etkileyen Faktörler

Training times can be affected by a number of factors such as:

- The computational power of your hardware (either CPU or GPU): Obviously, the more powerful your PC is, the faster the training process.
- Whether you are using the TensorFlow CPU or GPU variant: In general, even when compared to the best CPUs, almost any GPU graphics card will yield much faster training and detection speeds. As a matter of fact, when I first started I was running TensorFlow on my Intel i7-5930k (6/12 cores @ 4GHz, 32GB RAM) and was getting step times of around 12 sec/step, after which I installed TensorFlow GPU and training the very same model - using the same dataset and config files- on a EVGA GTX-770 (1536 CUDA-cores @ 1GHz, 2GB VRAM) I was down to 0.9 sec/step!!! A 12-fold increase in speed, using a “low/mid-end” graphics card, when compared to a “mid/high-end” CPU.
- How big the dataset is: The higher the number of images in your dataset, the longer it will take for the model to reach satisfactory levels of detection performance.
- The complexity of the objects you are trying to detect: Obviously, if your objective is to track a black ball over a white background, the model will converge to satisfactory levels of detection pretty quickly. If on the other hand, for example, you wish to detect ships in ports, using Pan-Tilt-Zoom cameras, then training will be a much more challenging and time-consuming process, due to the high variability of the shape and size of ships, combined with a highly dynamic background.
- And many, many, many, more. . . .

Eğitim İşlemini TensorBoard Kullanarak Takip Etme

Anaconda Prompt üzerinden alttaki komutlar uygulanır:

```
activate tensorflow_cpu # ya da gpu
tensorboard --logdir=training\
```

Alttaki gibi bir çıktı gelmesi gerekmektedir:

```
TensorBoard 1.6.0 at http://YOUR-PC:6006 (Press CTRL+C to quit)
```

Çıktıyı görüntülemek için verilen url'i tarayıcınıza kopyalaman yeterlidir.

Sonuç Grafiğini Dışarı Aktarma

Anaconda Prompt üzerinden alttaki komutlar uygulanır:

```
activate tensorflow_cpu # ya da gpu

copy %TENSORFLOW%\models\research\object_detection\export_inference_graph.py
%TENSORFLOW%\workspace\example_detection

cd %TENSORFLOW%\workspace\example_detection

python export_inference_graph.py --input_type image_tensor --
pipeline_config_path training/<yapılandırma_dosyası> --
trained_checkpoint_prefix training/model.ckpt-<checkpoint> --output_directory
trained-inference-graphs/output_inference_graph_v1.pb
```

- **<yapılandırma_dosyası>** Modelimizin yapılandırma dosyasının tam adı
 - **training** klasörüne attığımız yapılandırma dosyaları
 - Ör: `ssd_inception_v2_coco.config`
- **<checkpoint>** **example_detection/training** dizinindeki gösterilmek istenen adımın numarası
 - Ör: `13302`

Hata Notları ve Açıklamaları

'conda' is not recognized as an internal or external command

Anaconda Prompt üzerinden terminal işlemlerinize devam etmeniz durumunda sorun gidecektir.

'...' is not recognized as an internal or external command

Gerekli Paketlerin Kurulumları tamamlanmadığı için bu hata ile karşılaşılıyor olabilirsiniz.

'ImportError: No module named' Hataları

PythonPath ayarlanmadığı için bu hata ile karşılaşmaktasınız.

```
set
PYTHONPATH=%TENSORFLOW%\models\research;%TENSORFLOW%\models\research\slim;%TENSORFLOW%\models\research\object_detection
```

Dökümandaki ilgili alana yönelmek için [buraya](#) tıklayabilirsiniz.

'dict_keys' object does not support indexing

Açıklama linki için [buraya](#) bakabilirsiniz.

```
start
%TENSORFLOW%\models\research\object_detection\models\feature_map_generators.py
```

- Satır 518'deki yere alttaki kodu yapıştırın

```
image_features = image_features[list(image_features.keys())[0]]
```

Object was never used (type <class 'tensorflow.python.framework.ops.Tensor'>)

Yakında..

'unicodeescape' codec can't decode bytes in position

Modelinizin .config dosyanıza yazdığınız tam yol verilerinde \ yerine / veya \\ kullanmalısınız.

Allocation of X exceeds 10% of system memory

- Rast geldiğim bu [kaynağa](#) göre **ssd_mobilenet_v2_coco modeline** özgü bir hatadır.
- Hatanın çözüm kaynağı için [buraya](#) tıklayabilirsiniz

google.protobuf.text_format.ParseError, Expected string but found

Config dosyalarının text editör üzerinden düzenlemesi durumunda, türkçe karakterler için text editörü yapıyı değiştirmekte ve tensorflow bunu algılayamamaktadır. Sorunu çözmek için alttakiler yardımıyla **.config** dosyasını düzenleyin:

- VsCode
- Notepad++
- Sublime
- Atom

Harici kaynak için [buraya](#) bakabilirsiniz.

Value Error: No Variable to Save

Model eğitimi yapıldığı sırada gelen bir hatadır, çözümü için **.config** dosyanızı bu şekilde düzenleyin:

```
train_config: {  
  ...  
  fine_tune_checkpoint: "./pre_trained_model/model.ckpt"  
  fine_tune_checkpoint_type: "detection"  
  ...  
}
```

ssd_mobilenet_v1_quantized_300x300_coco14_sync modelinde test edilmiştir.

Colab Üzerinden Tensorflow Modelini Eğitme

Colab ücretsiz GPU sunduğu için çok hızlı bir eğitim imkanı sunar.

Colab Eğitimi için Gereken Dosyalar

- label_map.pbtxt
- test.record
- train.record
- *.config
- model_main.py (eskisi: train.py)
- export_inference_graph.py

Colab Üzeriinden Eğitim Kodları

Detayları öğrenmek için [buraya](#) tıklayarak colab notuma erişebilirsiniz.

Web Kamerası Kullanarak Obje Tespit Etme

Script dosyasına [buraya](#) tıklayarak erişebilirsiniz

Harici Bağlantılar

- [Traffic Light Detection Using the TensorFlow* Object Detection API](#)
- [Tensorflow in Anaconda](#)
- [Tensorflow create a tfrecords file from csv](#)
- [Tensorflow Object Detection, error while generating tfrecord \[TypeError: None has type NoneType, but expected one of: int, long\]](#)
- [Tensorflow Github Preparing Inputs](#)
- [TensorFlow Object Detection API in 5 clicks from Colaboratory](#)
 - [5steps_object_detection.ipynb](#)
 - [labels_analysis_object_detection.ipynb](#)
- [Custom training: walkthrough](#)
- [TensorBoard: Visualizing Learning](#)
- [TPU'yu Bu kadar Popüler Kılan Nedir?](#)

Başlangıç için İdeal Olanlar

- [Güncel Makaleler](#)
- [IBM Cloud üzerinden model](#)
- [Zero to Hero: Guide to Object Detection using Deep Learning: Faster R-CNN, YOLO, SSD](#)

Önemli Notlar

- `train.py` işlemi için `images` dizinindeki resimlere ihtiyaç yok, `tf_records`'lar yeterlidir.

Yapılacaklar

- ☐ **Tensorflow notları buraya taşınacak!**
- ☐ Her yeni eğitim için yapılacaklar için hızlı notlar oluştur
 - ☐ Num classes'lar değişecek (config ve detection_utils)
 - ☐ Yollar değişecek (config ve detection_utils)
 - ☐ Label_map.pbtxt değişecek
 - ☐ Test sayısı config'e girilecek
- ☐ Git sorununu düzelt, temiz çıktıya geç
 - ☐ Gitignore'a gereksiz herşeyi ekle (*, ** gibi karakterler ile)
 - ☐ Models klasörünü ele alma, dışarıdan indirme linkini ver
- ☐ VsCode için modül bulunamadı sorunu düzeltilecek
 - ☐ PythonPath ile ortam değişkeni ayarlamaya çalışılacak
 - ☐ VsCode, python proje çalıştırıcısı olarak ele alınacak

Sonra Yapılacaklar

- ☐ TF recordları oluşturma kısmı otomatikleştirecek ve dizinler bağımlı yollar halinde belirtilecek
 - ☐ .csv deki class'lardan label_map oluşturulacak.
 - ☐ generate.tfrecord.py içerisinde tüm diğer scriptler eklenecek ve FROM: ? ile xml veya csv bilgisi alacak
 - ☐ Benim yapıma uygun yapıya sahip olanların CLI parametresi vermesine gerek olmayacak
- ☐ El ile yapılan tüm işlemler otomatize edilecek
 - ☐ images içindeki test, train adlı dizinlerin ismi otomatik alınacak
 - ☐ grap_images.py script'i olacak ve resimleri gerekli dizinlere yerleştirmek için yol alacak (yerleştirileceklerin yolu)
 - ☐ Etiketli veriler hazır olduğunda tek bir script generate_tf_data ile direk eğitime hazır hale getirilecek
 - ☐ label_map csv'den alınacak
 - ☐ config yolu otomatik tanımlanacak
 - ☐ recordlar xml'den oluşturulacak
- ☐ pre_trained_model klasörü yeniden adlandırılacak ve modeller için alt klasörler olacak
 - ☐ base_model ile modelin sıfır hali tutulacak
- ☐ inference_graph klasörü yeniden adlandırılacak ve modeller için alt klasörler olacak
- ☐ Tf recordları farklı yöntemlerle elde etmeyi araştır
- ☐ TF recoderlardan resimleri elde etmeyi araştır

Sonradan Eklenecek Scriptleştirme

- ☐ Linux için sh script
 - ☐ Protobuf, tensorflow vs. her biri için
- ☐ Windows için bat executable

Sonradan Derlenecek Bilgiler

- ☐ Yaptığım teknikleri video'ya veya yazıya kayıt edeceğim
 - ☐ csv'lerden alanı seçip tablo formatına alarak class sayılarını görme vs.

TF Verilerini Alma

https://www.tensorflow.org/tutorials/load_data/tf_records

Recover the images from the TFRecord file

```
for image_features in parsed_image_dataset:  
    image_raw = image_features['image_raw'].numpy()  
    display.display(display.Image(data=image_raw))
```

Lisans ve Teferruatlar

Bu yazı **MIT** lisanslıdır. Lisanslar hakkında bilgi almak için [buraya](#) bakmanda fayda var.

- [Github](#)
- [Website](#)
- [LinkedIn](#)

Yardım veya destek için [iletişime](#) geçebilirsiniz ☐

~ Yunus Emre Ak