# 使用Mobilenet-SSD训练自己的数据

## 编译Caffe

首先，下载SSD源码：git clone <https://github.com/weiliu89/caffe/tree/ssd，下载后记得切换到ssd分支下。Caffe的编译安装过程参考官网。>

下载Mobilenet-SSD源码，放到caffe/examples下。

## 数据准备

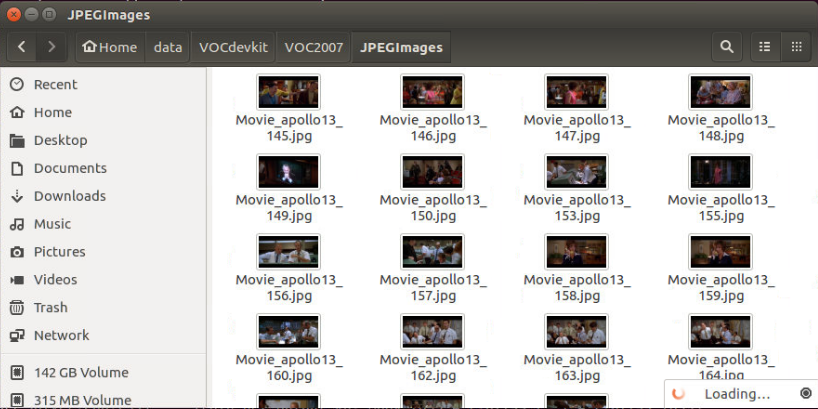
需要将自己的数据格式转成VOC2007的格式，在主目录下新建data文件夹，其下目录结构如下：



我们要将各文件夹放入相应的数据文件。

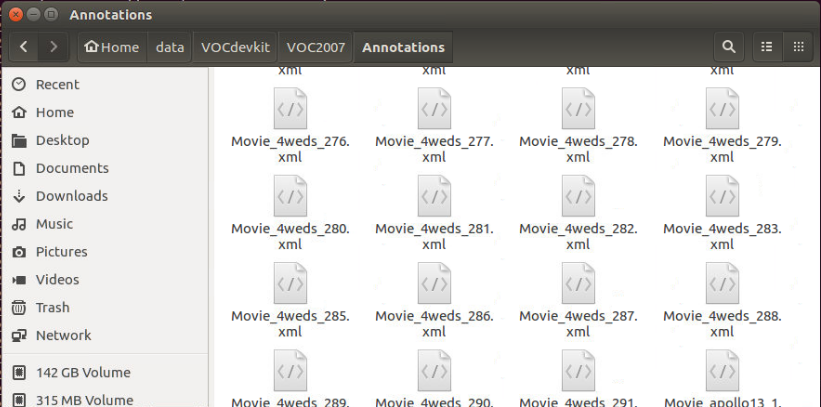
### JPEGImages文件夹

这里放置训练的图像文件，格式为jpg文件。



### Annotations文件夹

这里放置图像文件对应的标柱文件。



标注文件为xml格式，文件名一对应的图像文件名相同，格式类似如下：



由于我下载的数据集不是VOC格式的，而是matlab的mat文件格式，需要转换。为了方便以后数据转换方便，我先将数据转成如下的文本文件：



第一行为图像的高、宽、通道数量，下面每一行分别为目标的类型编号（从1开始）目标左上角坐标及右下角坐标，放置在SRC\_LABEL目录下。

我通过如下的python代码转成xml格式。

|  |
| --- |
| import os, sys  import numpy as np  class Object:  def \_\_init\_\_(self):  self.name = ''  self.minX = 0  self.minY = 0  self.maxX = 0  self.maxY = 0  class Annotation:  def \_\_init\_\_(self):  self.filename = ''  self.width = 0  self.height = 0  self.depth = 0  self.objs = []  self.classmap = ['background', 'hand']  def read(self, filepath):  if os.path.isfile(filepath):  self.filename = os.path.basename(filepath).split(".")[0] + ".jpg"  with open(filepath, 'r') as file:  file\_info = file.readline().split(' ')  self.height = int(file\_info[0])  self.width = int(file\_info[1])  self.depth = int(file\_info[2])  while(True):  obj\_info = file.readline().split(' ')  if not obj\_info or len(obj\_info) != 5:  break  obj = Object()  class\_idx = int(obj\_info[0])  obj.name = self.classmap[class\_idx]  obj.minX = int(obj\_info[1])  obj.minY = int(obj\_info[2])  obj.maxX = int(obj\_info[3])  obj.maxY = int(obj\_info[4])  self.objs.append(obj)  root\_dir = './VOCdevkit'  annotations\_dir = root\_dir + '/Annotations'  imagesets\_dir = root\_dir + '/ImageSets'  jpegimages\_dir = root\_dir + '/JPEGImages'  src\_dir = root\_dir + '/SRC\_LABEL'  dct = ['hand']  label\_list = os.listdir(src\_dir)  file\_count = 0  for i in range(0, len(label\_list)):  path = os.path.join(src\_dir, label\_list[i])  if os.path.isfile(path):  #read label  ano = Annotation()  ano.read(filepath = path)  image\_name = label\_list[i].split(".")[0]  annotaiton\_name = annotations\_dir + "/" + image\_name + ".xml"  with open(annotaiton\_name, 'w') as annotation:  annotation.write("<annotation>\n")  #folder  annotation.write("<folder>")  annotation.write("VOC2007")  annotation.write("</folder>\n")  #filename  annotation.write("<filename>")  annotation.write(ano.filename)  annotation.write("</filename>\n")  #source  annotation.write("<source>\n")  annotation.write("<database>")  annotation.write("Unknown")  annotation.write("</database>\n")  annotation.write("</source>\n")  #size  annotation.write("<size>\n")  annotation.write("<width>")  annotation.write(str(ano.width))  annotation.write("</width>\n")  annotation.write("<height>")  annotation.write(str(ano.height))  annotation.write("</height>\n")  annotation.write("<depth>")  annotation.write(str(ano.depth))  annotation.write("</depth>\n")  annotation.write("</size>\n")  #segmented  annotation.write("<segmented>")  annotation.write(str(0))  annotation.write("</segmented>\n")  #object  for i in range(len(ano.objs)):  annotation.write("<object>")  annotation.write("<name>")  annotation.write(ano.objs[i].name)  annotation.write("</name>\n")  annotation.write("<pose>")  annotation.write("Unspecified")  annotation.write("</pose>\n")  annotation.write("<truncated>")  annotation.write(str(0))  annotation.write("</truncated>\n")  annotation.write("<difficult>")  annotation.write(str(0))  annotation.write("</difficult>\n")  annotation.write("<bndbox>\n")  annotation.write("<xmin>")  annotation.write(str(ano.objs[i].minX))  annotation.write("</xmin>\n")  annotation.write("<ymin>")  annotation.write(str(ano.objs[i].minY))  annotation.write("</ymin>\n")  annotation.write("<xmax>")  annotation.write(str(ano.objs[i].maxX))  annotation.write("</xmax>\n")  annotation.write("<ymax>")  annotation.write(str(ano.objs[i].maxY))  annotation.write("</ymax>\n")  annotation.write("</bndbox>\n")  annotation.write("</object>\n")  annotation.write("</annotation>")  print file\_count, "file finish!"  file\_count = file\_count + 1 |

### ImageSets文件夹

ImageSets下有一个Main文件夹，放置训练、测试的图像文件名，可以由以下python代码生成：

|  |
| --- |
| import os  import random    train\_percent = 0.70  xmlfilepath = 'Annotations'  txtsavepath = 'ImageSets\Main'  total\_xml = os.listdir(xmlfilepath)    num=len(total\_xml)  list=range(num)  tr=int(num\*train\_percent)  train=random.sample(list,tr)  test = random.sample(list, int(num \* 0.1))    ftrain = open('ImageSets/Main/train.txt', 'w')  fval = open('ImageSets/Main/val.txt', 'w')  ftrainval = open('ImageSets/Main/trainval.txt', 'w')  ftest = open('ImageSets/Main/test.txt', 'w')    for i in list:  name=total\_xml[i][:-4]+'\n'  if i in train:  ftrain.write(name)  else:  fval.write(name)  ftrainval.write(name)  if i in test:  ftest.write(name)      ftrain.close()  fval.close()  print ("finished") |

### **生成lmdb数据**

### 训练

## 测试