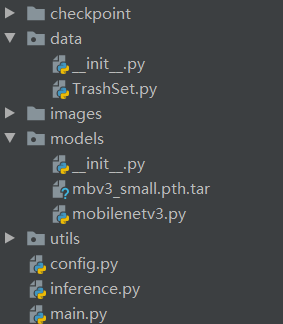
# 基于opencv的目标检测

detect.py为视频目标检测参考代码，利用opencv，包括knn,mog2,grabcut三种方式，可根据实际效果选择。

# 二、基于MobileNetV3的垃圾分类

## 程序结构



main.py: 模型训练代码

inference.py：前向传播

config.py: 程序运行时的相关参数

data/TrashSet.py：垃圾数据集类

images/文件夹：提供一些测试图片

models/mbv3\_small.pth.tar：mobilenetv3在imagenet预训练权重

models/mobilenetv3.py: 模型文件

utils/文件夹：提供一些关于imagenet的可用函数，此处用不到

## 模型训练结果

### 数据集

自己采集的垃圾分类数据集TrashBig, 其中有12个小类别

['bananapeel', 'battery', 'cardboard', 'clothes', 'drugs', 'glass', 'lightbulb', 'metal', 'paper', 'papercup', 'plastic', 'vegetable']

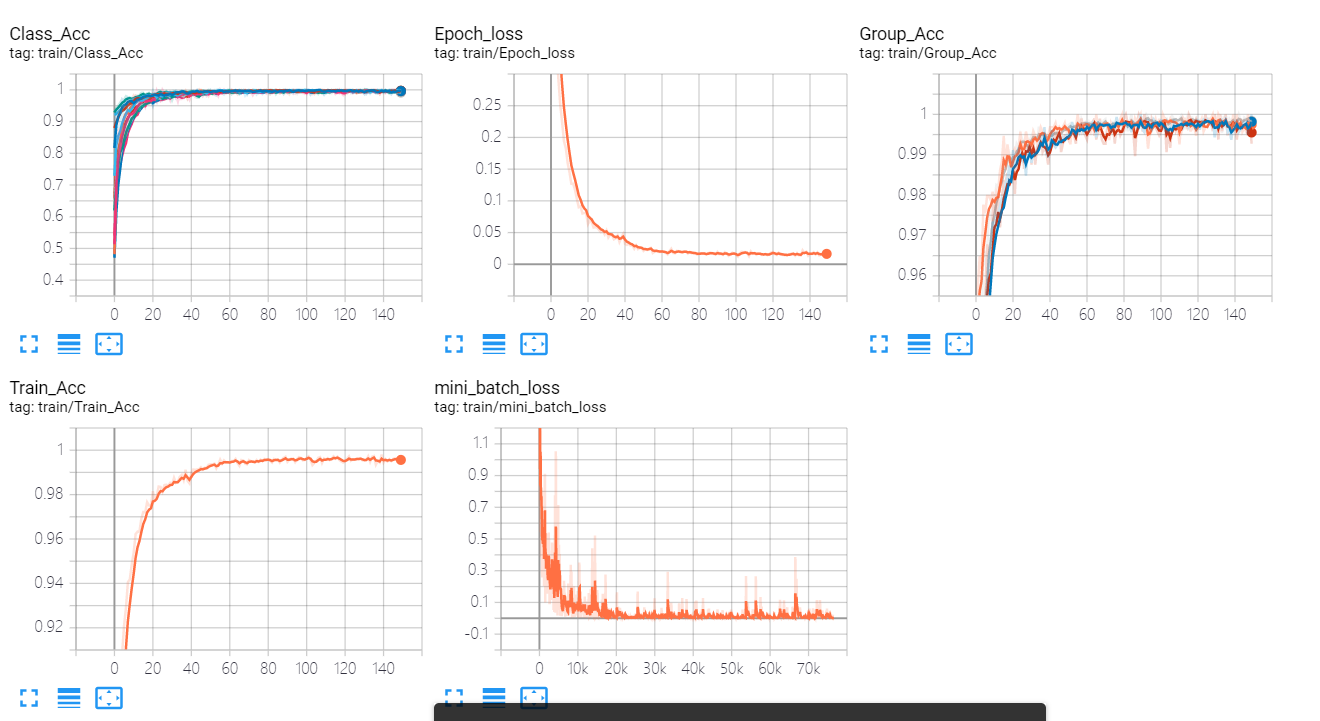
这12个小类别分别属于四个大类别

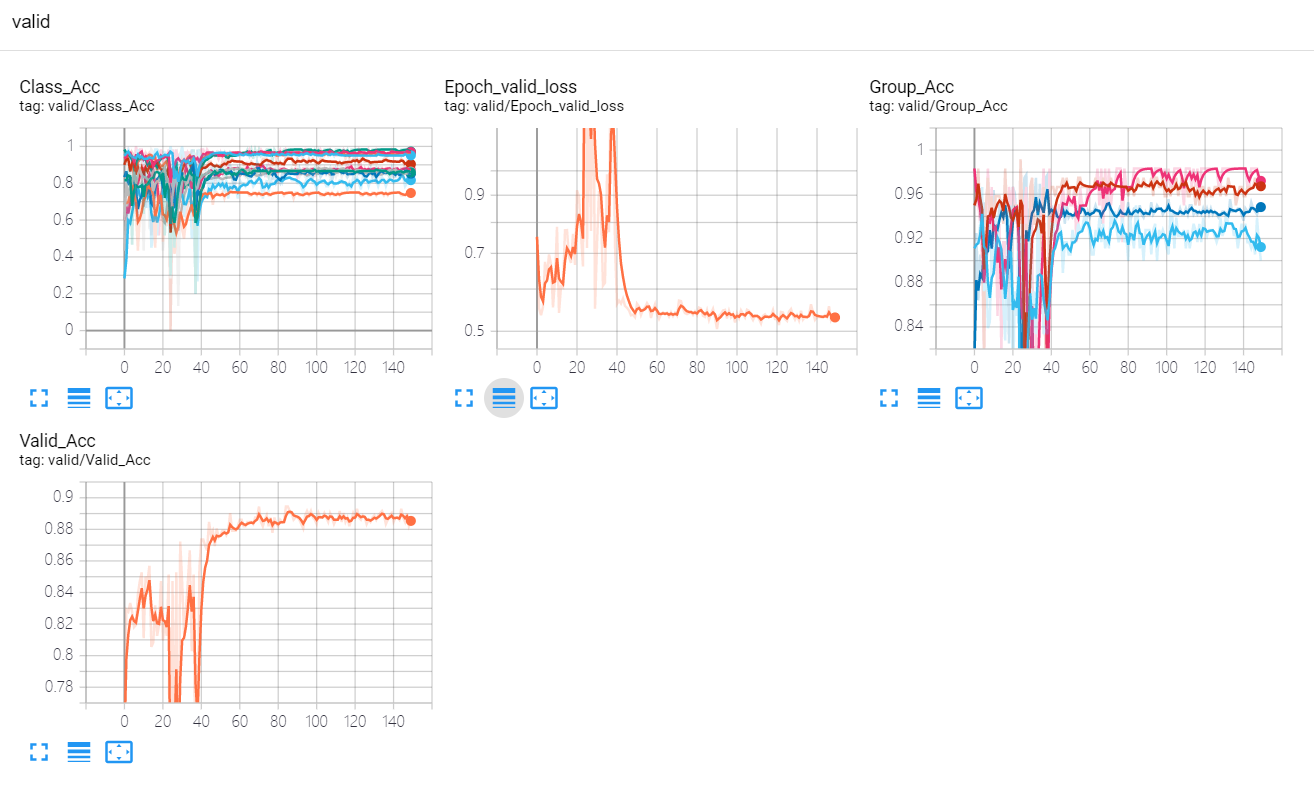
{'Recyclables': 2624, 'Kitchen waste': 939, 'Hazardous waste': 1581, 'Other': 963}

训练集图片共计6000张左右

验证数据集每类60张左右 ，共计720张左右。

### 训练结果







上图是验证集的每类精度

大类别的分类精度都在90%以上

### 使用自己的垃圾数据集训练模型

1、程序运行环境

Pytorch1.0 以上版本

opencv-python

numpy

tensorboard

2、数据集的组织要求

数据集目录下分为两个小目录：train和val

每个小目录下以类别名称命名的文件夹，存储对应类别的图片

3、config.py文件的修改

对config.py文件进行修改



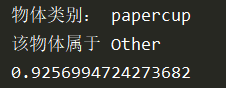
4、运行main.py

## 运行前向传播测试 ：inference.py

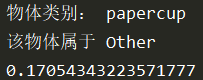
直接运行inference.py即可测试。

修改config.py 中DefaultConfig.InferWithGPU参数即可切换前向传播使用GPU还是CPU。

GPU (GTX-1060 6G) 条件下运行单张图片：



CPU (i5-8300h) 条件下运行单张图片：



结果：单张图片的运行时间GPU为0.925s，CPU下为0.17s。此处CPU运行速度明显快于GPU。这是因为数据传输会有很大的开销，而GPU处理数据传输要比CPU慢，而GPU的专长矩阵计算在小规模神经网络（此处为MobileNetV3）中无法明显体现出来。

**对inference.py文件稍加修改即可对自己的图片进行分类。**