PROJECT REPORT

Defaulters Detection at SMRTs and Buses

BY:

TEAM 8

Abhineet Mishra

Srikar Namburu

1. Executive Summary

Singapore has one of most convenient and helping public transport system around the world. It supports multiple people from all age groups and demographics across the country.

It is also known for its maintained hygiene and cleanliness in all the MRT stations, MRT, bus stations and buses.

To support such system the authorities has enforced a golden rule which forbids the passengers to eat or drink while commuting in any of these places to avoid any spilling which can cause discomfort to fellow passengers.

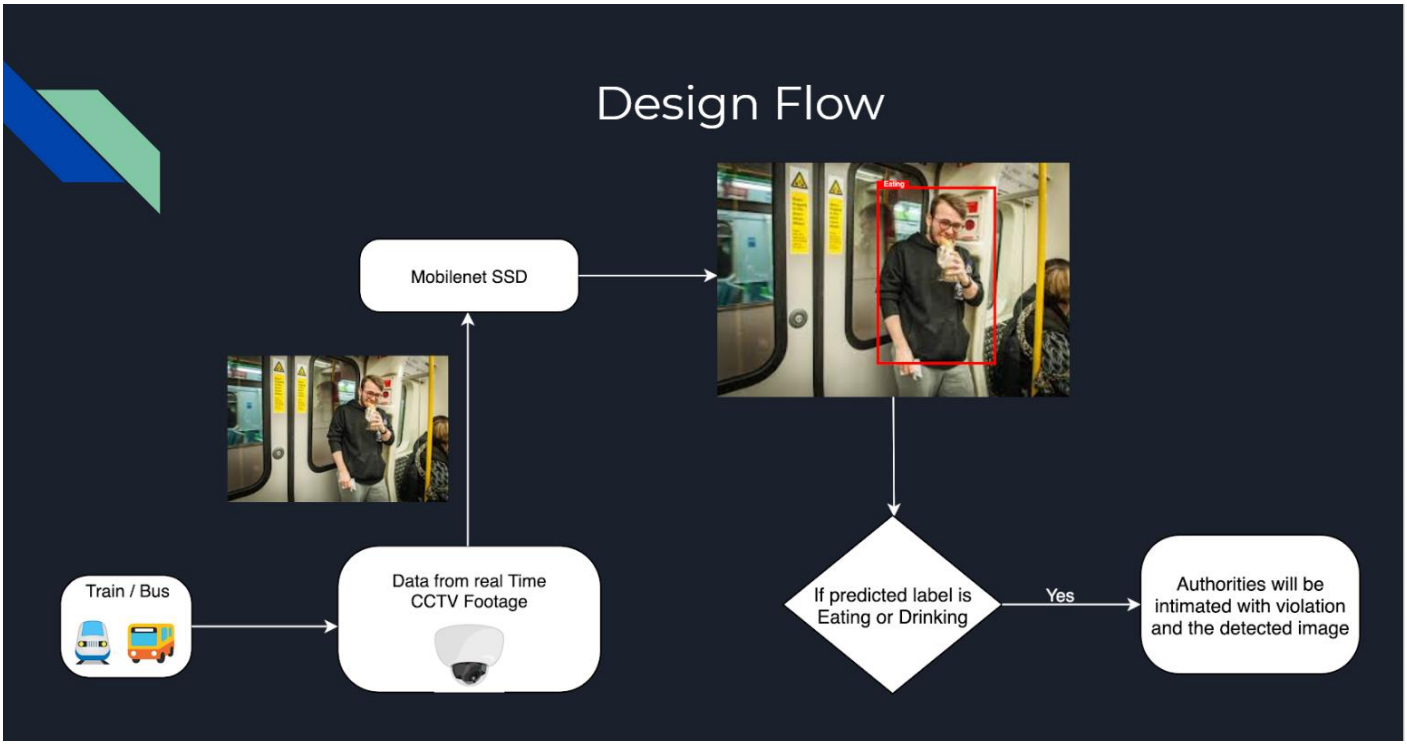
But some people still break this law and cause discomfort to all the fellow passengers as well as the transport organization.

Currently these defaulters are spotted by manual manpower which exhausts the resources as well is inefficient at times.

We propose a solution which automatically detect such defaulters using machine learning.

2. Solution

We propose a system with following design:



3 **Data collection and data processing**

A) To train such model we need to have image data for people eating or drinking. We have used google crawler to collect such data from google images using multiple keywords. Total 205 images have been downloaded

The script can be found in the first cell of data-processing.ipynb notebook file

```
# -*- coding: utf-8 -*-
"""

from icrawler.builtin import GoogleImageCrawler


google_crawler = GoogleImageCrawler(storage={'root_dir': '\straw'})
google_crawler.crawl(keyword='people sipping from straw', max_num=5000)
```

B) Once the images are downloaded, we use the LabelMe tool to manually label the data its bounding boxes as well its ground truth label.

This tool creates a json files for each image containing the bounding box coordinates as well as the label.

C) Even though we have labelled the data this collection of data is not in proper format for training. To do so we separate the data into two folders namely training (for training) and val (for validation and testing). Then we create and use a script to convert data as per the COCO dataset format so that our data changes in a standard format for training

This script of conversion can be found in cell number 2 and 3 of data-processing.ipynb file. We use this script to separately create a single json file namely trainval.json

```
if __name__ == "__main__":



    args = "/data/val"

    labelme_json = glob.glob(os.path.join("/data/val", "*.json"))

    labelme2coco(labelme_json, "trainval2.json")
```

```
save coco json
trainval2.json
```

D)As per our requirement we have come across mobilenet v1 model which is trained on voc-pascal dataset. To use that structure we need to convert our dataset format from COCO to voc-pascal.This is done by script written in cell 4.

E) Finally, we need to create a text file containing names of all the filenames we will be using for the training purpose. This script is written in cell 6 of dataprocessing.ipynb

4) **Model Creation:**

A) We have decided to use mobilenet ssd due to its good accuracy compounded with working speed.

First thing we need to do for model creation is create a model structure which is same as that provided by the google.

We have created such structure using keras in the file models/mobilenet_v1.py

This model will be responsible for extracting the features from images and passing them to single shot detector layers to detect the objects

B) Once mobilenet_v1 is created we need to add the SSD layers which detects the object and generate the anchor boxes based on the aspect ratios and offsets provided by us.

These anchor boxes are used as reference bounding boxes whenever an object is detected.

Addition of ssd layers is done in models/ssd_mobilenet.py which adds layers for detection and generates anchor boxes using script from misc/keras_layers_AnchorBoxes.py

C) Now we use ssd_mobilenet.py to create model in model.ipynb file . We use the same hyper parameters as used in voc pascal dataset training such as aspect ratio, offsets and variances.

Once the model structure is created, we load the weights of mobilenet_v1 trained on voc pascal dataset to our model this helps in better convergence of model. mobilenet_1_0_224_tf.h5 are the model weights for mobilenet_v1.

D)Next we generate training dataset and validation dataset into the required batch format via script from misc/ssd_batch_generator.py

E) Next we define the way in which we encode the ground truth labels in the format required by ssd to compute the loss. This is done by misc/ssd_box_encode_decode_utils.py.

F) Lastly, we convert both our datasets (training and validation) into the ssd_encoder format so that they can be used in training now

5) **Model training:**

We have trained our model with a batch size of 32 images over 250 epochs. We have used a dynamic learning rate scheduler to have a low learning rate towards the end of model training.

6)**Performance Evaluation and testing:**

We have tested our model on 10 images and found out that it can create bounding boxes across people drinking or eating.

We have used the standard method of mAP to calculate the precision of the model which comes around 0.87

7)**Limitations:**

A) Model could only be trained on jpg images for now.

B) Model was able to detect the defaulter but was not able to classify properly hence we have converted it to single class model detecting the defaulter.