Public Transport Access Detection

Dragoș-Mihail Comănac

Contents

Application and original contribution

Classification

AMS Mathematical Subject Classification - 68T45 Machine vision and scene understanding

ACM Computing Reviews Categories and Subject Descriptors - I.4 IMAGE PROCESSING AND COMPUTER VISION - I.4.6. Segmentation - Edge and feature detection

Application lifecycle

1. Application is opened
2. User chooses auditive or visualization mode using a voice command
3. In auditive mode, the live object detection automatically starts, and information is provided to the user in an auditive manner
4. In visualization mode the user can load an image and perform detection on it or can start object detection on the live feed from the camera
5. Application is closed by voice command or through a button

Functionalities

- choose auditive mode or visualization mode using a voice command
- auditive mode: perform live object detection on cars, busses, and license plates and based on the results of detection, provide auditory clues that guide the user towards the detected car or bus. If the detected object is a bus, then provide information about the location of the doors and the line of the bus (if available). If the detected object is a car, then provide information about the location of the doors and the license plate (if available)

- visualization mode: display the results of object detection on an image or on the live feed from the camera
- close the app using a voice command

Original contribution

The application aims to help the visually impaired persons by using computer vision and line detection, on a mobile device, to provide spatial information about public transport such as busses or ride sharing access. An object detector model will extract information about the vehicle and a line detector will analyze the part of the image containing the vehicle to extract information about the doors of the vehicle. This should eliminate the need for a dataset that has specific bounding box annotations for the doors.

1. Introduction

1.1 Context and motivation

1.2 Objectives

1.3 Paper structure

2. State of the art

- discussion on the current methods used in object detection and line detection

3. Theoretical foundations

- description of the basic concepts that are used

4. Design and implementation

- description of the proposed solution

5. Experimental results

5.1 Dataset

For training the object detection system we use the Open Images Dataset V4 [15], available at [13]. In total, the dataset contains 9.2 million images, including 14.6 million bounding boxes across 600 classes on 1.74 million images. We will use only a subset of classes: bus, car and license plate.

The tool used to download the Bus, Car and License plate classes and the corresponding bounding boxes is OIDv4 ToolKit [24].

	Number of images	Number of bounding boxes			
		Total	Bus	Car	License plate
Train	30939	80295	11927	60516	7852
Validation	4967	9985	92	9381	512
Test	14894	30660	353	28737	1570

Table 5.1: Statistics related to the subset of Open Images

Each image is resized so that it's dimension is 416x416 and the bounding boxes are modified accordingly with the following formulas:

$$xmin = 416/width * xmin$$
$$ymin = 416/height * ymin$$
$$xmax = 416/width * xmin$$
$$ymax = 416/height * ymin$$

Where width and height correspond to the image and xmin, ymin, xmax, ymax correspond to the lower left corner and upper right corner of the bounding box. The resizing helps the object detection task, as explained in [21], because this way we can split the image in a grid of 13x13 cells of size 32x32 so that there is a cell in the center that can detect the larger objects that are centered in the middle, rather than have 4 cells in the middle that try to detect the same object.

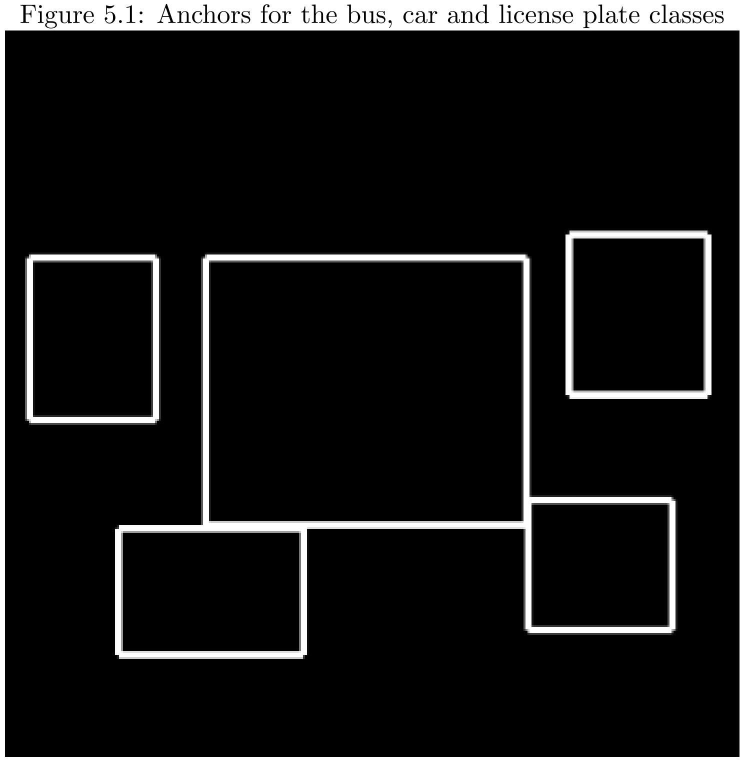
Each image is associated with multiple bounding boxes and each cell is responsible with detecting multiple bounding boxes through the use of anchors as explained in [21]. We use 5 anchors per cell so that each cell can detect various shapes and sizes. The anchor boxes are chosen using K-Means over the whole dataset.

For computing the distance between the centroid and bounding box the following formula is used, as in [21]:

$$distance(centroid, box) = 1 - IOU(centroid, box)$$

Where IOU represents intersection over union and is calculated as the area of the intersection of the 2 boxes over the area of union of the 2 boxes.

Figure 5.1: Anchors for the bus, car and license plate classes



This shows that there are large boxes centered in the middle of the image and multiple smaller ones around.

Each image needs to be associated with a ground truth. We represent the ground truth as a CxCxBx8 tensor. For each cell in the CxC grid we associate B anchor boxes. Each anchor box has the following parameters: b_x and b_y represent the center of the box, b_w and b_h represent the width and height, c is the probability that the anchor predicts an object and c_1, c_2, c_3 represent the class probabilities. In our case C is 13 and B is 5.

The following formulas are used to compute the center, width and height from the output of the model:

$$b_x = \sigma(t_x) + c_x$$
$$b_y = \sigma(t_y) + c_y$$
$$b_w = p_w \cdot e^{t_w}$$
$$b_h = p_h \cdot e^{t_h}$$

Where t_x, t_y, t_w, t_h represent the raw predictions to which the sigmoid function is applied, c_x, c_y represents the coordinates of the upper left corner of the cell that predicts the box and p_w, p_h represent the width and height of the anchor that predicts the box.

After the bounding boxes are interpreted, usually Non-maximum Suppression (NMS) is used to filter the boxes with the best score and overlap with the ground truth and then mean average precision is used to see how good are the predictions.

5.2 Perfomance evaluation

To measure an object detection system performance, usually frames per second (FPS) and mean average precision (mAP) are used. Precision measures the percentage of correct predictions for a given class.

$$Precision = \frac{TP}{TP + FP}$$

A prediction is considered true positive if the IOU is greater than a set threshold. Recall is another metric that measures the percentage of found positives (also known as true positive rate).

$$Recall = \frac{TP}{TP + FN}$$

Both precision and recall depend on the given threshold for true positives. This means that by plotting different values for precision and recall for different thresholds we get a curve. The area under the curve is called average precision. And the mean over the area under the precision-recall curve for all classes is the mean average precision.

Over time, the mean average precision became the standard way of evaluating the performance of an object detection system in order to compare it with other solutions.

During training, the following sum-squared error loss function is optimized to achieve better performance:

$$
\begin{aligned}
\lambda_{coord} \sum_{i=0}^{C^2} \sum_{j=0}^{B} 1_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] \\
+\lambda_{coord} \sum_{i=0}^{C^2} \sum_{j=0}^{B} 1_{ij}^{obj} [(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2] \\
+ \sum_{i=0}^{C^2} \sum_{j=0}^{B} 1_{ij}^{obj} (C_i - \hat{C}_i)^2 \\
+\lambda_{noobj} \sum_{i=0}^{C^2} \sum_{j=0}^{B} 1_{ij}^{noobj} (C_i - \hat{C}_i)^2 \\
+ \sum_{i=0}^{C^2} \sum_{j=0}^{B} 1_{ij}^{obj} \sum_{c \in classes} (p_{ij}(c) - \hat{p}_{ij}(c))^2
\end{aligned}
$$

Most grid cells do not contain any boxes. Therefore in order to balance the confidence scores λ_{coord} and λ_{noobj} are added in order to increase the loss from bounding box predictions and decrease the loss from confidence predictions. Also, the square root of the width and height is used to remedy the problem that error in small and large boxes is weighted the same.

1_{ij}^{obj} is 1 for the j'th bounding box in the i'th cell.

5.3 Related work

Since the hardware became sufficiently advanced and accessible, the interest in object detection system increased, thus the need for a standardized dataset for comparing solutions appeared. Pascal VOC is one of the first datasets that proposes a variety of tasks such as object detection. It started as a challenge in 2007 [5] and it continued until 2012 [6]. Since then, mostly the combined datasets from 2007 and 2012 are used for training and validation and the test dataset from 2007 is used for testing and they are used as a common ground to compare performance across all kinds of computer vision tasks, such as object detection.

So far, the general idea in creating object detection datasets is to take an image and add bounding boxes around the objects of interest. But if we want to detect a specific feature of an object, for example if we want to detect the doors of a car, the current approach would be to take a dataset of images and add bounding boxes for the doors. This approach can take a long time and it requires further training of the model. Our approach should simplify this process, because we try to eliminate the need for additional bounding boxes and detect directly the required shapes. For example, if we need to detect bus doors we are looking for long vertical lines in the detected patch, that can be extracted by a line detector that eliminates the need for extra training.

6. Conclusions and future work

- summary of the solution - critical analysis of the solution - future improvements

7. Bibliography

[1] Cuneyt Akinlar and Cihan Topal. Edlines: A real-time line segment detector with a false detection control. *Pattern Recognition Letters*, 32:1633–1642, 10 2011.

[2] Md. Zahangir Alom, Tarek M. Taha, Christopher Yakopcic, Stefan Westberg, Paheding Sidike, Mst Shamima Nasrin, Brian C. Van Essen, Abdul A. S. Awwal, and Vijayan K. Asari. The history began from AlexNet: A comprehensive survey on deep learning approaches. *CoRR*, abs/1803.01164, 2018.

[3] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *CoRR*, abs/2004.10934, 2020.

[4] Lamya El Alamy, Sara Lhaddad, Soukaina Maalal, Yasmine Taybi, and Yassine Salih-Alj. Bus identification system for visually impaired person. In *2012 Sixth International Conference on Next Generation Mobile Applications, Services and Technologies*, pages 13–17, 2012.

[5] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[6] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[7] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*, abs/1311.2524, 2013.

[8] Lilit Hakobyan, Jo Lumsden, Dympna O'Sullivan, and Hannah Bartlett. Mobile assistive technologies for the visually impaired. *Survey of Ophthalmology*, 58(6):513–528, 2013.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *CoRR*, abs/1406.4729, 2014.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[11] Derek Hoiem, Yodsawalai Chodpathumwan, and Qieyun Dai. Diagnosing error in object detectors. *Computer Vision ECCV 2012*, pages 340–353, 2012.

[12] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, and Kevin Murphy. Speed/accuracy trade-offs for modern convolutional object detectors. *CoRR*, abs/1611.10012, 2016.

[13] Ivan Krasin, Tom Duerig, Neil Alldrin, Vittorio Ferrari, Sami Abu-El-Haija, Alina Kuznetsova, Hassan Rom, Jasper Uijlings, Stefan Popov, Shahab Kamali, Matteo Malloci, Jordi Pont-Tuset, Andreas Veit, Serge Belongie, Victor Gomes, Abhinav Gupta, Chen Sun, Gal Chechik, David Cai, Zheyun Feng, Dhyanesh Narayanan, and Kevin Murphy. Openimages: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from https://storage.googleapis.com/openimages/web/index.html*, 2017.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[15] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, and et al. The open images dataset v4. *International Journal of Computer Vision*, 128(7):1956–1981, Mar 2020.

[16] Dongjin Lee, Hosub Yoon, Chankyu Park, Jaehong Kim, and Cheong Hee Park. Automatic number recognition for bus route information aid for the visually-impaired. In *2013 10th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, pages 280–284, 2013.

[17] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. *Lecture Notes in Computer Science*, page 21–37, 2016.

[18] Wai-Ying Low, Mengqiu Cao, Jonas De Vos, and Robin Hickman. The journey experience of visually impaired people on public transport in london. *Transport Policy*, 97:137–148, 2020.

[19] Geneva: World Health Organization. World report on vision. page 77, 2019.

[20] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640, 2015.

[21] Joseph Redmon and Ali Farhadi. YOLO9000: better, faster, stronger. *CoRR*, abs/1612.08242, 2016.

[22] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *CoRR*, abs/1804.02767, 2018.

[23] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.

[24] Angelo Vittorio. Toolkit to download and visualize single or multiple classes from the huge open images v4 dataset. https://github.com/EscVM/OIDv4_ToolKit, 2018.

[25] Na Wang, Ruoyan Chen, and Kang Xu. A real-time object detection solution and its application in transportation. In *2021 International Conference on Communications, Information System and Computer Engineering (CISCE)*, pages 486–491, 2021.

[26] Junjie Yan, Zhen Lei, Longyin Wen, and Stan Z. Li. The fastest deformable part model for object detection. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2497–2504, 2014.