www.revistageintec.net
ISSN: 2237-0722
revista GEINTEC Gestão, Inovação e Tecnologias

Automated Checkout for Stores: A Computer Vision Approach

Namitha James[1]; Nikhitha Theresa Antony[2]; Sara Philo Shaji[3]; Sherin Baby[4]; Jyotsna Annakutty[5]

[1,2,3,4]Student, Department of Computer Science and Engineering, Rajagiri School of Engineering and Technology, Kerala, India.

[1]namithajames17@gmail.com, [2]nikhithaantony99@gmail.com, [3]saraphiloshaji@gmail.com, [4]sherinbaby8299@gmail.com

[5]Assistant Professor, Department of Computer Science and Engineering, Rajagiri School of Engineering and Technology, Kerala, India.

[5]jyotsnaa@rajagiritech.edu.in

Abstract

One of the most common current advances made by companies that use cutting-edge technology to remain competitive in the market is the automation of processes. Using Computer Vision techniques is one way to get into the automation processes without spending a lot of money. This strategy hence results in increased productivity in the short term as well as increased earnings in the long run. In reality, it aimed to improve an algorithm that had already been developed, but working on it has been more difficult than anticipated due to the emergency caused by the Covid-19 pandemic. In this study, we proposed an automated checkout for stores based on the YOLOv4 algorithm to address the challenge. The dataset taken consists of the object categories inside a large supermarket. The proposed method includes features such as detection of the objects, count the similar objects, generate the bill of the commodities identified and allows the user to make the payment online. The Automated Checkout for Stores demonstrated fast checkout, a user-friendly interface, customer satisfaction, and, in this pandemic period, it provides social distancing as well as a safe and comfortable shopping experience for the customers.

Key-words: Camera, Detection, MS COCO, Stripe, YOLOv4.

1. Introduction

People's needs and demands for commodities are increasing. The retail industry has become one of the most competitive and fast-paced sectors of the economy. The retail industry is evolving. Adaptive retailers will thrive. That means creating innovative, exciting consumer experiences, as well as finding new ways to capture the attention of customers. Long checkout lines proved to be the most

significant barrier to a smooth in-store shopping experience [1]. The majority of online shoppers refuse to shop in-store due to long checkout lines. Customers hate having to wait in long lines to pay for their purchases. Long queues can be a turnoff, and customers who have to wait too long will often leave. As a result, profits are lost, consumer satisfaction is decreased, and the customer experience suffers. For many years, self-checkout stores have grown in popularity, and since the outbreak of COVID, it has become the preferred method of payment for many consumers. It cuts down on time spent in line and eliminates communication with both workers and other customers.The two types of object detection approaches are Machine learning-based approach and Deep learning-based approach [2]. The machine learning approach has a requirement of defining features whereas defining features is not required in a deep learning-based approach as it perform end-to-end object detection. Convolutional neural networks (CNN's) are a form of deep neural network that is widely used to check visual imagery [3].Object detection algorithms based on deep learning include region-based CNN algorithms (RCNN, Fast R-CNN, Faster R-CNN), Single Shot MultiBox Detector (SSD), and YOLO [4].

In this paper, an Automated Checkout for Stores based on You Only Look Once (YOLOv4) [5] algorithm is introduced. The system enables both large supermarkets and small convenience stores to offer a seamless shopping experience to their customers without interrupting business operations to upgrade the checkout area. The system detects the objects on the conveyor belt at the checkout counter, counts the similar products, generate the bill of the identified products and finally allow the customers to make online payment.

The contents of this paper are : First, we have chosen Microsoft COCO dataset which contains 80 classes from that we have taken 34 classes which are needed for our system. The next phase was model creation using a pre-trained model. Third, the objects in the MS COCO dataset are detected using a camera module using the YOLOv4 algorithm. After detection, similar objects are counted and stored into an output file. In the next phase, the price of the objects are calculated and the bill is generated. Finally, customers can make their own payments using Stripe, an online payment network.

2. Literature Survey

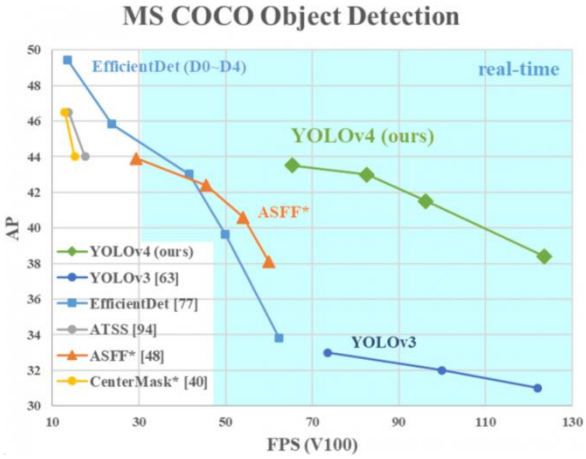
A. Computer Vision Algorithms

There are many object detection algorithms like R-CNN, Fast R-CNN[25], YOLO[4] etc.. Extracting region of interest (ROI) from an input image is the initial step of R-CNN. Selective Search

is the approach used for this. The boundary of an object is represented by a rectangle indicating each region of interest in the image. The neural network runs on the whole image one time, althougth the initial R-CNN features of the neural network were computed independently on each of as many as two thousand ROI.A novel approach called ROI Pooling is used at the network's end to carve out each ROI from the output tensor, reshape it, and classify it.

One of the fastest object tracking algorithms available is YOLO. While R-CNN models are more reliable in general, the YOLO family of models is much faster, achieving real-time object detection. During training and testing, YOLO[4] sees the whole picture, so it indirectly encodes qualitative details about classes as well as their presentation. Since it can't see the wider picture, Fast R-CNN[25], a top detection tool, misidentifies background patches in an image as artefacts. As opposed to Fast R-CNN[25], YOLO[4] makes half as many context mistakes.

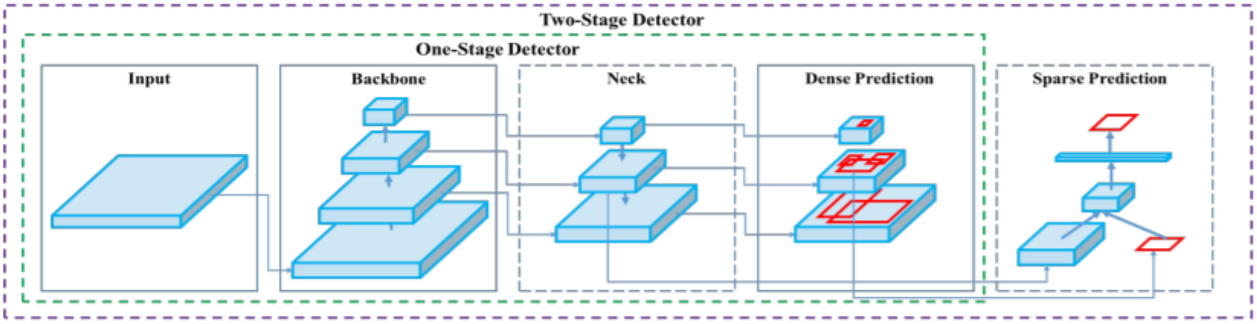
Fig. 1 - Comparison between YOLOv4 and state-of-art object detectors[5]



YOLOv3 [10], which was released in April 2018, made minor enhancements to previous models. The darknet architecture has been extended to 53 convolution layers in this update, but it only considers the object score and class scores during object detection and does not consider the bboxscore beacause of the lack of clarity in the information regarding score of the coordinates of bbox[11]. As a result, in 2020 developed the fourth version of YOLO popularly known as YOLOv4[5] was developed by Chien-Yao Wang, Alexey Bochkovskiyand Hong-Yuan Mark Liao. It greatly outperforms current methods in terms of detection and speed. YOLOv4 [5] needs only a single GPU and a smaller mini-batch size than most modern accurate version that need multiple GPUS and larger mini-batch size. This allows a single 1080 Ti or 2080 Ti GPU to train a fast and reliable object detector. On the MS COCO[6] dataset,YOLOv4[5] produces state-of-the-art

performance in real time with 43.5 percent AP running at 65 FPS on a Tesla V100 as shown in Fig 1. Hence, YOLOv4 was chosen as the basic algorithm of our project.

Fig. 2 - Object detector [5]



Object detection models: A modern detectoras shown in Fig. 2 Object detector [5] typically consists of two parts: a backbone and a head, all of which have been pre-trained on ImageNet and are used to predict object classes and bounding boxes. Recently in detectors installed, between the head and the backbone some layers are usually inserted, and the main use of the layers is the collection of feature maps from different points. These layers are called to as the object's neck. YOLOv4 consists of CSPDarknet53 as Backbone, SPP and PAN as Neck and YOLOv3[10] as Head.

Bag of freebies: This refers to training techniques that allow the object detector to improve its accuracy without increasing the cost of inference. These approaches only alter the method of training, not the expense. This is analogous to Data Augmentation, which aims to improve the variability of the input images to make images from various environments more robust. The different methods are MixUp [12], Random erase [13], CutOut [14] etc.As a backbone, YOLOv4 employs Bag of Freebies(BoF) such as Class label smoothing, Mosaic data augmentation, DropBlock regularization and CutMix data augmentation. While it uses Optimal hyperparameters, DropBlock regularization, CIoU-loss, CmBN, Mosaic data augmentation, Eliminate grid sensitivity, Self-Adversarial Training, Using multiple anchors for a single ground truth, Mish activation, Random training shapes as BoF for detector.

Bag of special: This refers to plugin plugins and post-processing methods that significantly improve object detection accuracy while only raising the cost of inference by a small amount. The modules enhance features such as expanding the receptive field, incorporating a mechanism of focus, and enhancing the ability to incorporate functionality etc. Bag of Specials (BoS) for backbone uses Cosine annealing scheduler [15], Multiinput weighted residual connections (MiWRC), Mish

activation Cross-stage partial connections (CSP) and for detector includes SAM block, SPPblock, DIoU-NMS, PAN path-aggregation block.

YOLOv4 is on the Pareto optimality curve, and it outperforms the fastest and most reliable detectors in terms of precision and speed. Since different approaches use GPUs with various architectures, YOLOv4 contrasts algorithms for inference time verification to different state-of-the-art approaches using GPUs with the Maxwell, Pascal, and Volta architectures.

B. Datasets

Datasets have played an important role in computer vision research throughout history. They not only allow for the training and evaluation of algorithms, but they also push research in new and more difficult directions. Object recognition datasets can be divided into three categories: those that focus on object classification, object identification, and semantic scene labeling [6]. Binary labels indicating whether or not objects are present in an image are needed for object classification [6] is known as image classification. The technique of object detection Detecting an object involves stating the presence of an object of a certain class as well as locating it in the picture. In semantic scene labeling, each pixel of an image should be identified as belonging to a group, such as umbrella, apple, orange, spoon, and so on in order to label semantic items in a scene.

MNIST[7] is one of the most widely used deep learning datasets. It's a dataset of handwritten digits with 60,000 examples in the training set and 10,000 examples in the test set. It's a decent database for experimenting with deep learning methods and trends on real-world data while investing as little time and resources as possible on data preprocessing.

ImageNet[8] consists of a set of images arranged according to the WordNet hierarchical structure [9]. WordNet has about 100,000 phrases, and ImageNet has given about 1000 images to explain each phrase on average.

A real-world image dataset known as the Street View House Numbers [24] can be used to evaluate algorithms of object detection. This necessitates the least amount of data preprocessing. It's similar to the MNIST dataset described earlier, but it includes more labelled data (over 600,000 images). The information was gathered using Google Street View house numbers.

The Microsoft Common Objects in COntext (MS COCO) [6]is a large-scale dataset with a lot of potential for object detection, segmentation, and captioning. There are 91 general object types in the dataset, with 82 of them having over 5,000 named instances. The dataset on the whole has 328,000 images with 2,500,000 labelled instances. Compared to the famous ImageNetdataset[8], MS

COCO dataset has comparatively lesser number of categories but it has more instances per category. This dataset includes comparatively more items present in a supermarket. Since our project is based on supermarket checkouts, we chose MS COCO as our dataset.

3. Methodology

The proposed method detects the objects in the MS COCO dataset which includes items from hypermarket. Then it counts the number of similar items, generates the bill and payment details in a GUI interface. In the proposed model's training procedure, the data preparation step and the model training phase were separated. During the data preparation phase, data labeling and preprocessing were done to prepare the training data. During the training procedure, the model was trained using the prepared training data.

A. Dataset Preparation

The dataset used is MS COCO dataset. COCO dataset, which stands for "Common Objects in Context," is a collection of challenging, high-quality datasets for computer vision, with the majority of the datasets being state-of-the-art neural networks. This term is frequently used to describe the file format in which particular datasets are saved. Object segmentation, recognition in context, Superpixel stuff segmentation, 330K photos (>200K labeled),1.5 million item instances, and 80 object categories are only a few of COCO's characteristics.

The 3 most popular tasks are: Object detection — the model should return a list of object classes and the coordinates of rectangles around them, items are distinct, independent objects, sometimes with pieces, such as apple and orange; the official dataset for this task often contains additional data for object segmentation. Segmentation of objects— the model should provide not only bounding boxes for objects, but also segmentation masks, which are the positions of polygons that are close to the item. Stuff segmentation — the model can perform object segmentation, but not on discrete objects, but on continuous patterns in the context. The items required for object detection were taken from the COCO dataset, which was fed into the network as an input.

B. Object Detection

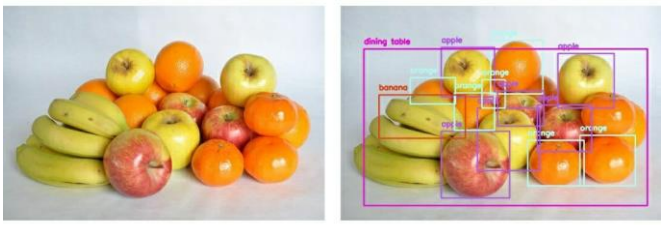
The labelled data of the categories and the pre-trained model of YOLOv4 were used in the training phase of our model. The Microsoft COCO dataset was used to train the pre-trained model given by the YOLOv4 authors.

The MSCOCO dataset was used to train these using the CSPDarkNet[18] code. Download the pre-trained model weights and save them as "yolov4.weights" in your current working directory. We use a single neural network to process the entire image. This network divides the image into areas by predicting bounding boxes and probability for each zone. The estimated probabilities are used to weight these bounding boxes.

Compared to classifier-based frameworks, our model has a number of advantages. At test time, it examines the entire picture, so its predictions are guided by the image's overall meaning. It also allows predictions with a single network evaluation, as opposed to R-CNN, which requires thousands of evaluations for a single picture. This makes it 1000 times faster than R-CNN and 100 times faster than Short R-CNN.

CSPDarknet displays the objects it found, their trust level, and the time it took to locate them. Since CSPDarknet was not built with OpenCV, it is unable to view detections directly. We take about 6-12 seconds per image because we're using CSPDarknet on the CPU. It will be significantly faster if we use the GPU version.

Fig. 3 - Object detection using YOLOV4



YOLO only shows items that have a trust of 0.50 or higher by default. The anticipated classes, as well as the image with bounding boxes drawn on top of it, will be displayed by YOLO. If you don't have a webcam that OpenCV can connect to, it won't operate, and the items spotted will be stored to a CSV file for counting.

C. Counting

Once the items has been detected and identified, the next phase of the automatic checkout system counts the number of similar items in the list. We aim to get labels of the items detected on an Excel sheet along with the count of each item present. Also a new CSV file[19] is created and the output after detection and identification is written to that file.

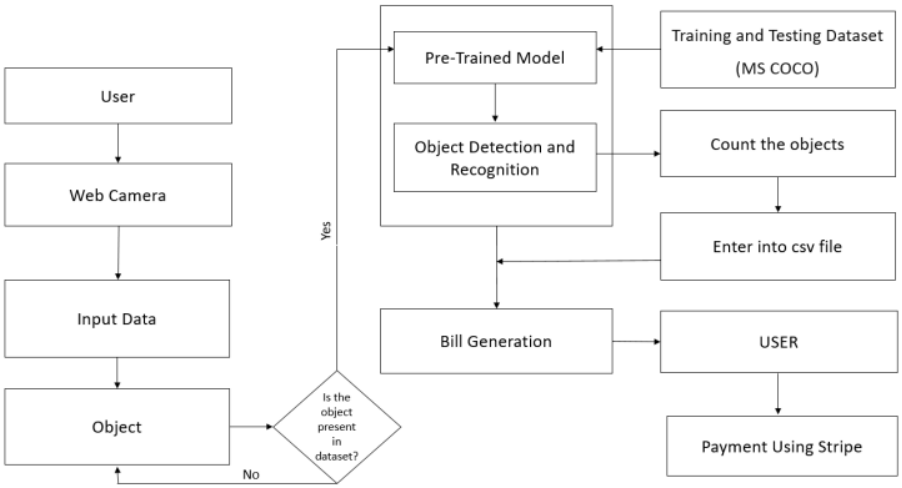
D. Bill Generation and Payment

In the bill generation part, there will be output CSV file which contains the list of detected objects which is present in the MS COCO dataset with their count and another file with the list of objects along with their price. With the help of these two files the price of the object is obtained and a bill is generated. The generated bill will be displayed on the screen for payment. The Python programming language is one of the most scalable, and it has special libraries for machine learning applications. Pay option in GUI leads us to the payment of the products purchased from the supermarket. Here we implement payment through a library in Python. Payment in the automatic checkout system is carried out through the library Stripe. Stripe is a web application programming interface(API) for payment processing. [20]. It contains a pre-defined collection of classes for API services that dynamically initialize themselves from API responses, making it compliant with a variety of Stripe API models. In the Flask application[21], we use Stripe Checkout to accept one-time payments.

E. GUI

A graphical user interface is included in a self-checkout framework for non-bar-coded objects. Flask framework is used to make the interface. Flask is a Python-based microweb platform. In the interface there is a button named shopping, when we click that the camera is turned on. Then the objects are scanned and identified. After shopping, the bill will be generated where the total amount is calculated and will be displayed on the screen. For bill payment there will be a button named pay. The payment is done through stripe which is a library used for payment where you can enter the card details and other details needed for payment.

Fig. 4 - Flowchart



4. Result and Discussion

The experimental environment was generated on a laptop with an 8th Gen Core i5 processor and 4 GB RAM using Python libraries and OpenCV which is a free and open-source software library for computer vision and machine learning.

For detecting and segmenting objects encountered in daily life, we used the MS COCO (Microsoft Common Objects in Context) dataset [6]. Choosing object groups is a difficult task. The categories must be descriptive of all categories, applicable to real-world applications, and occur frequently enough to allow for the collection of a broad dataset. Item definitions may have a wide range of specificity. Out of the 80 object categories we have chosen only 34 object categories that are available in a hypermarket. Finding non-iconic depictions of objects in natural settings and from different viewpoints was emphasized. According to the dataset statistics [22], the images offer a lot of contextual information, with multiple objects per picture.

The YOLOv3 was found to have a higher detection efficiency than R-CNN, Fast R-CNN, and Faster R-CNN. In the field of object detection, another analysis found that the YOLOv2[16] is faster than the Faster R-CNN. Combining the above claims and previous analysis, the YOLO algorithm was found to be more suitable for hypermarket object detection than the two-stage detection algorithms. With similar efficiency, YOLOv4 is two times as fast as EfficientDet.

Self-checkout has been growing in recognition for years, and because of the outbreak of Covid, it has turned out to be the desired way for lots of clients to pay for his or her purchases. It reduces the time spent equipped in line and minimizes contact with every workforce and exclusive

shoppers. A less equipped manner provides a better client experience. Increased purchaser pride will result in prolonged purchaser loyalty and prolonged earnings for retailers.

Detection results of yolo on 34 items in the MS COCO dataset were obtained. The objects are detected by their names and displayed on the screen. The confidence scores of the detected objects are calculated and compared. We take the score of those objects that are maximum and have a confidence value of more than 50%. The items are identified and counted and saved to a CSV file. The prices of the commodities are predetermined and listed in a file. We merge both the files to calculate the price of the total items. Users can interact through a user-friendly GUI interface [23]. The payment is shown as a demo.

5. Conclusion

We used a computer vision solution that uses a single camera mounted above the conveyor to detect and count items on the checkout counter in real-time. The pre-trained model was used as a baseline for generating a new prediction model based on the collected COCO dataset. The Average Precision obtained by using YOLOv4 is 43.5% by the MS COCO dataset. Flask web framework is used to create the GUI interface, also Stripe online payment processing platform is used for payment. After learning about the Automated Checkout Stores approach, we can expect to witness a rapid increase in the number of these types of stores in the near future.

References

Schögel, Marcus, and Severin Dominic Lienhard. "Cashierless Stores–the New Way to the Customer?." *Marketing Review St. Gallen* (2020).

Mahto, Pooja, Priyamm Garg, Pranav Seth, and J. Panda. "Refining Yolov4 for Vehicle Detection." *International Journal of Advanced Research in Engineering and Technology (IJARET)* 11, (2020) no. 5.

Zhao, Zhong-Qiu, Peng Zheng, Shou-tao Xu, and Xindong Wu. "Object detection with deep learning: A review." *IEEE transactions on neural networks and learning systems* 30, no. 11 (2019): 3212-3232.

Kim, Jeong-ah, Ju-Yeong Sung, and Se-ho Park. "Comparison of Faster-RCNN, YOLO, and SSD for Real-Time Vehicle Type Recognition." In *2020 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia)*, 2020, pp. 1-4. IEEE.

Bochkovskiy, Alexey, Chien-Yao Wang, and Hong-Yuan Mark Liao. "Yolov4: Optimal speed and accuracy of object detection." *arXiv preprint arXiv:2004.10934* (2020).

Lin, Tsung-Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. "Microsoft coco: Common objects in context." In *European conference on computer vision*, Springer, Cham, (2014), pp. 740-755.

Le Cun, Yann. "The MNIST database of handwritten digits" *http://yann.lecun.com/exdb/mnist/* (1998).

Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "Imagenet: A large-scale hierarchical image database." In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248-255.Ieee, 2009

Miller, George A. *WordNet: An electronic lexical database*. MIT press, 1998.

Redmon, Joseph, and Ali Farhadi. "Yolov3: An incremental improvement." *arXiv preprint arXiv:1804.02767* (2018).

Choi, Jiwoong, Dayoung Chun, Hyun Kim, and Hyuk-Jae Lee. "Gaussian yolov3: An accurate and fast object detector using localization uncertainty for autonomous driving." In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 502-511. 2019.

Zhang, Hongyi, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. "mixup: Beyond empirical risk minimization." *arXiv preprint arXiv:1710.09412* (2017).

DeVries, Terrance, and Graham W. Taylor. "Improved regularization of convolutional neural networks with cutout." *arXiv preprint arXiv:1708.04552* (2017).

Zhong, Zhun, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. "Random erasing data augmentation." In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, (2020) pp. 13001-13008.

Loshchilov, Ilya, and Frank Hutter. "Decoupled weight decay regularization." *arXiv preprint arXiv:1711.05101* (2017).

E. Dong, Y. Zhu, Y. Ji and S. Du, "An Improved Convolution Neural Network for Object Detection Using YOLOv2," *2018 IEEE International Conference on Mechatronics and Automation (ICMA),* 2018.

J. D. Schaffer, D. Whitley and L. J. Eshelman, "Combinations of genetic algorithms and neural networks: a survey of the state of the art," *[Proceedings] COGANN-92: International Workshop on Combinations of Genetic Algorithms and Neural Networks,* 1992.

M. N. Chan and T. Tint, "A Review on Advanced Detection Methods in Vehicle Traffic Scenes," 2021 6th International Conference on Inventive Computation Technologies (ICICT), 2021.

C. Tapsai, "Information Processing and Retrieval from CSV File by Natural Language," *2018 IEEE 3rd International Conference on Communication and Information Systems (ICCIS),* 2018.

Horridge, Matthew and Bechhofer, Sean. 'The OWL API: A Java API for OWL Ontologies'. 1 Jan. 2011: 11 – 21.

P. Vogel, T. Klooster, V. Andrikopoulos and M. Lungu, "A Low-Effort Analytics Platform for Visualizing Evolving Flask-Based Python Web Services," *2017 IEEE Working Conference on Software Visualization (VISSOFT),* 2017.

Kuznetsova, Alina, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali et al. "The open images dataset v4." *International Journal of Computer Vision* (2020): 1-26.

Galitz, Wilbert O. *The essential guide to user interface design: an introduction to GUI design principles and techniques*. John Wiley & Sons, 2007.

Netzer, Yuval, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. "Reading digits in natural images with unsupervised feature learning." (2011).

Girshick, Ross. "Fast r-cnn."In *Proceedings of the IEEE international conference on computer vision*, pp. 1440-1448. 2015.