IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 25, Issue 1, Ser. I (Jan. – Feb. 2023), PP 29-36
www.iosrjournals.org

Car parking slot identification using SSD Resnet and Tensor Flow Object Detection API

[1] Rithik B,[2]Raghav G,[3]Kruthik A R,[4]Rahul Patwadi,[5]Aravind H S
[1][2][3][4][5] Electronics and Communication Engineering Department JSSATEB Bangalore,Karnataka,India.
Corresponding Author Email:[1]rithikbalaram77@gmail.com, [2]ragavgsm21@gmail.com,[3] arkruthik@gmail.com[4]rpatwadi@gmail.com,[5]aravindhs1@gmail.com

Abstract - *In recent years, parking of automobiles has become a tedious task owing to the growing number of vehicles. A crucial and still unsolved difficulty for such systems is how to effectively and efficiently detect and localize parking spots indicated by regular line segments around the vehicle. In actuality, a variety of negative circumstances, such as the variety of ground textures, changing lighting conditions, and unpredictable shadows cast by neighboring trees, make vision-based parking-slot recognition far more difficult than it appears. Hence, obtaining data on parking slots is a requirement for the development of parking slot detectors. We suggested a deep learning-based parking-slot-marking detection solution in this research. The detecting procedure entails creating a mask of the marking-points with the help of Single shot detector Resnet. Around 4000 surround-view photos were used from regular parking lots to create this collection.*

Index Terms *–Tensor Flow object detection API, SSD Resnet, Deep Learning, COCO Dataset, Parking Slot.*

Date of Submission: 12-01-2023 Date of Acceptance: 28-01-2023

I. INTRODUCTION

It is safe to state that practically everyone will possess a manual or automatic vehicle in the near future in the world we currently live in because more people are utilizing vehicles every day thanks to the development of automatic vehicles and the emerging technologies surrounding them. The challenge of modernizing existing solutions to accommodate impending or potential future changes arises with an increase in innovation level. The parking management system is one such remedy. Time is also important for everyone, in addition to everything else. Therefore, it is not satisfactory for practically anybody who has a vehicle to spend a lot of time figuring out the parking spot before moving on to the next duty.

Computer vision has become a widely used method in order to detect parking spaces. It is a visual based technique and hence does not require any sensors as used in the non-visual based approaches. One drawback of using vision-based approaches is that if a vehicle is blocking a parking space behind it, that space goes undetected. Surveillance cameras are the most commonly used vision-based parking space detectors. In our approach we used images taken from surveillance cameras.

The technique of using TensorFlow Object Detection API to develop a custom model for parking lot detection primarily targets open spaces like parking lots in malls and universities. It is a video-based detection method that utilizes cameras that are already in place for security reasons, making it a more efficient and cost-effective solution compared to sensor-based strategies. The architecture of convolutional neural networks (CNN) is used in this technique, which is similar to the network of synapses and neurons in the human brain. This allows the network to learn complex tasks, such as identifying vacant parking spaces quickly and accurately. This approach eliminates the need for paperwork, reduces time loss, and makes it easier for customers to find parking spots.

II. ARCHITECTURE AND LIBRARIES

DEEP LEARNING

This section provides an overview of the development of Deep Convolutional Neural Networks (DCNN), as the proposed parking space detection technique, called DeepPS, is based on this technology. DCNN is a leading method in the field of computer vision, which allows machines to automatically identify features in unstructured data for classification or recognition tasks. In recent years, DCNN has outperformed other methods by utilizing large labeled datasets, improved models, and training methods, as well as the power of GPU processing to analyze larger and deeper models. Since 2012, several popular DCNN architectures have been proposed, such as AlexNet, GoogLeNet, VGG, and ResNet. DCNN-based approaches have significantly advanced the state of the art in various areas, including image classification, object detection, face detection, image restoration, and more. The specific approach used in this study is SSD ResNet for object recognition.

This is a multi-stage detection system that uses an object proposal algorithm to find bounding boxes that contain objects with high probability in a given input image.

SSD ResNet

An SSD (Single Shot MultiBox Detector) is a deep learning model that uses a single convolutional network to detect objects in an image. It consists of two parts: a backbone model and an SSD head. The backbone model is typically a pretrained image classification network that serves as a feature extractor. The most common backbone model used is ResNet-like networks trained on the ImageNet dataset. These networks are modified by removing the last fully-linked classification layer, leaving a deep neural network that can extract semantic meaning from the input image while preserving the spatial structure of the image at lower resolutions. The ResNet34 backbone generates 256 7x7 feature maps for the input image. The SSD head is a series of one or more convolutional layers added to the backbone, which reads the output as bounding boxes and classifies objects at the spatial position of the last subcaste's activation.

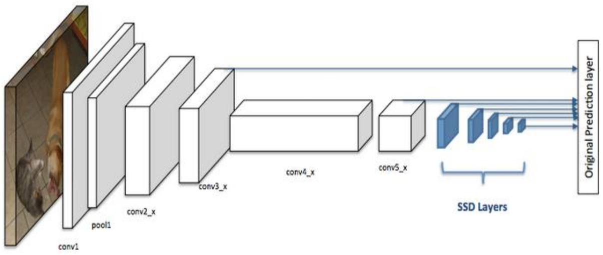


Fig1: Cross section view of the SSD Resnet Architecture

The Single Shot MultiBox Detector (SSD) method for object detection divides an image into grid cells, with each cell responsible for identifying objects within that area. The system predicts the class and location of objects, but assumes a background class and ignores position if no object is present. Each grid cell generates output for the position and shape of objects. SSD uses multiple pre-defined "anchor" boxes for each cell, which determine the size and shape of the cell. During training, the system matches these anchor boxes with the actual bounding boxes of objects in the image. The anchor box with the highest degree of overlap is used to determine the class and location of the object. Once trained, the network uses this information to predict the presence and location of objects in new images. Additionally, different aspect ratios and zoom positions can be applied to the anchor boxes to account for non-square shaped objects.

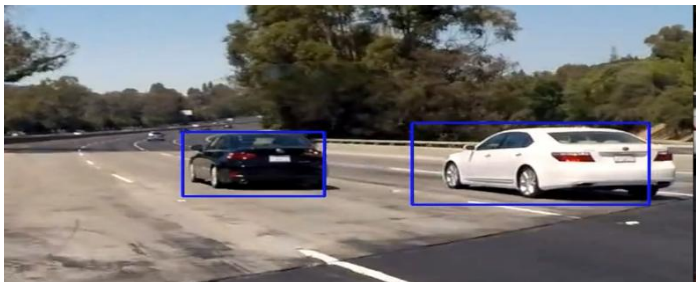


Fig 2: For example, the bounding box for the object on the right is wider than the bounding box of the object on the left.

SSD LOSS FUNCTION

$$L(x,c,l,g) = \frac{1}{N}(L_{conf}(x,c) + \alpha L_{loc}(x,l,g))$$

The loss function consists of two terms: Lconf and Lloc where N is the matched default boxes. Matched default boxes

$$L_{loc}(x,l,g) = \sum_{i\in Pos}^{N} \sum_{m\in\{cx,cy,w,h\}} x_{ij}^{k}\,\mathrm{smooth_{L1}}(l_i^m - \hat{g}_j^m)$$

$$\hat{g}_j^{cx} = (g_j^{cx} - d_i^{cx})/d_i^w \qquad \hat{g}_j^{cy} = (g_j^{cy} - d_i^{cy})/d_i^h$$

$$\hat{g}_j^{w} = \log\left(\frac{g_j^w}{d_i^w}\right) \qquad \hat{g}_j^{h} = \log\left(\frac{g_j^h}{d_i^h}\right)$$

LOCALIZATION LOSS

The Lloc loss is a measure of the accuracy of the predicted bounding box (l) in relation to the actual bounding box (g) in object detection. It is calculated using a smooth L1 loss function, which compares the coordinates for the center point (cx, cy), width (w), and height (h) of the bounding box. This loss function is similar to the one used in the Faster R-CNN method.

$$L_{conf}(x,c) = -\sum_{i\in Pos}^{N} x_{ij}^{p} log(\hat{c}_i^p) - \sum_{i\in Neg} log(\hat{c}_i^0) \quad \text{where} \quad \hat{c}_i^p = \frac{\exp(c_i^p)}{\sum_p \exp(c_i^p)}$$

CONFIDENCE LOSS
The Lconf loss is a measure of the confidence of the predicted object class (c) in object detection. It is calculated using a softmax loss function, with a weight of 1 for cross-validation. xijp is an indicator for matching the i-th predicted bounding box to the j-th actual bounding box of class p.

III. DATASET

To attain high efficiency and extra ordinary classification results, the main requirement for training any Machine Learning/Deep Learning model is data and by data we mean huge amount of data. Practically capturing such huge number of data(images/videos) is practically impossible by a group of few people. We decided to use 4000 images of the dataset available on the internet consist totally of 12,000 images captured from different surveillance and CCTV cameras. The Dataset is called Pklot database and is licensed under a Creative Commons Attribution 4.0 licence. These images were captured on various weather conditions i.e., Sunny,Rainy,and cloudy days. Images are also captured at different angles and the resolution of each image is 640x640 pixels. Furthermore, the images have different lighting conditions as well. On how the images captured,labelled and segmented are discussed in the paper (Almeida, P., Oliveira, L. S., Silva Jr, E., Britto Jr, A., Koerich, A., PKLot – A robust dataset for parking lot classification, Expert Systems with Applications, 42(11):4937-4949, 2015). Annotations is the process of categorizing and labelling data for AI Applications. This is a very important step as it tells what exactly the model has to learn. The Pklot dataset was originally annotated Out of the 12,000 images only 4000 images were taken to train the model as there was hardware restriction with the graphics memory and the CPU'sefficiency. The dataset was further divided into Train and Test set. While the Training set consisted of 3200 images, the Test set consisted of 800 images and there was no validation set created.

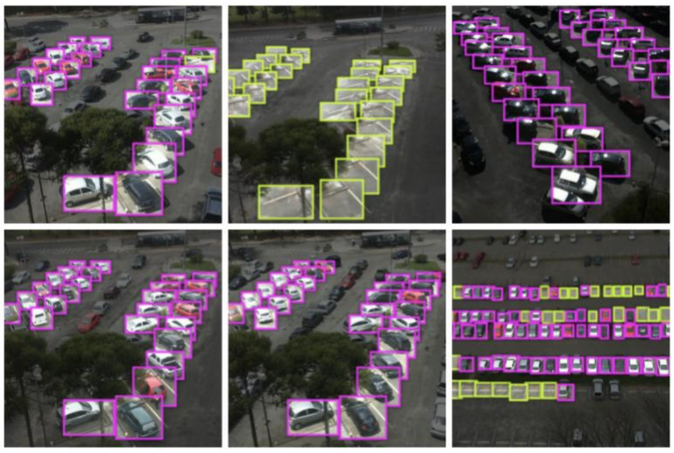


Fig 3 : Annotated images captured from a surveillance camera over different weather conditions.

IV. RELATED WORK

[1] The system for finding parking spots employs a combination of Support Vector Machines (SVM) and Markov Random Field (MRF) techniques. A camera is used to distinguish between occupied and unoccupied parking spaces, and a machine learning model is trained to detect the vacant spaces. The process is broken down into four steps: pre-processing, feature extraction, SVM recognition, and MRF correction. The color histogram is used to extract features, and after dimension reduction, 50 features are selected. SVM is used to classify the occupancy of a parking spot based on manually defined sections of the parking lot, and the addition of MRF improves the performance of SVM. Without MRF, SVM has an accuracy of around 84.35 and 85.57 for single space detection and three spots respectively. When MRF is used, accuracy increases to 93.53.

[2] A detector-based approach was used to detect a moving vehicle entering the parking lot. Based on the size of the vehicle and the time it takes to pass through the gate where the detector is located, it is determined whether a spot has been occupied by a vehicle. However, detectors can be costly to install and maintain, and their range of operation may be limited. Additionally, the lack of a roof for outdoor detector installation can present another major issue...

[3] This method describes the use of deep learning algorithms to locate parking spaces. The classifier was trained using two datasets: one from the National Research Council in Pisa, and another from PKLot. The network configurations used were mAlexNet and mLeNet, which are smaller versions of AlexNet and LeNet, and these smaller versions were used to speed up the classification process. The accuracy obtained is over 90% for both datasets..

[4] A video-based system approach was used to address the problem of determining parking spot occupancy. Color histograms and Difference-of-Gaussian (DoG) algorithms were used to extract features. Three different machine learning algorithms, K-Nearest Neighbor (k-NN), Linear Discriminant Analysis (LDA), and Support Vector Machines (SVM), were used to classify the spots as vacant or occupied. To reduce classification errors, the results were smoothed through temporal integration. The DoG and color histograms feature extractors were paired with the machine learning algorithms in various combinations to find the most effective combination. The system was tested and achieved a success rate of over 92% on an unseen parking lot.

[5] Kaempchen et al. developed a stereo-vision-based strategy that makes use of a feature-based stereo technique, an iterative closest neighbour algorithm, and a template matching approach on a depth picture. Inverse perspective transformation theory, restricted coulomb energy (RCE) colour separation, and the least squares method of contour extraction are used.
A parking place labelling system was suggested by Xu et al. . However, this strategy relies only on parking spot markings. Poor viewing conditions on the markers, such as spills, reflections, and partial occlusion from close vehicles, may impair it. As parking space sensing technologies evolve, more academics are investigating parking space recognition techniques that rely on semi-automatic and fully automatic approaches, fostering the field's further development and expansion. In this part, the techniques and technologies for locating parking spaces will be compiled and categorised. In order to assist future researchers in more effectively resolving the parking space issue, it will also examine and describe the characteristics of current parking space diagnostic approaches as well as the challenges that parking space detection technology faces in this field. Each piece of knowledge can be used as a resource or aid in the detection of issues.

The majority of computer vision-based technologies have changed to the machine learning paradigm as a result of machine learning's resurgence. Deep learning-based smart parking systems have not received much research attention. Much of the current research on parking detection systems is presented in this chapter. A smartphone-based personal assistant system was introduced by S. Nawaz et al. Their method is based on an advanced sensor that locates an open parking space on the road. To find open parking places, they used wireless internet beacons throughout the streets. By detecting the rate of changes in beacons, their proposed system determines if the driver has actually begun to move. They claimed that their system performed better than GPC and Internet-based parking options. This technology uses less energy when compared to a few other location-based smart parking systems. This app runs on a smartphone and needs Wi-Fi in order to work.

A system grounded on free space has a unnaturally different working principle than one that's grounded on vision. A vision- grounded fashion aims to honor and detect parking spaces indicated by painted parking line parts on the ground. In our approach the discovery of objects in the image is done with the help of bounding boxes. The parking- niche discovery module receives the compass- view image as input, locates the parking spaces, and also sends the decision module the physical positions of the spaces relative to the vehicle- centred match system.

V. METHODOLOGY

Google's TensorFlow Object Detection API is used to train our custom model on the pklot dataset. Google's TensorFlow Object Detection API is an open-source framework built on TensorFlow that makes it easy to construct and train custom object detection models. The API includes pre-trained models known as the Model ZOO, which are trained on various datasets such as the Microsoft COCO dataset, KITTI dataset, and Open Images Dataset. These datasets contain a large number of images with various objects, with COCO having 300,000 images for 90 different objects, KITTI containing 7481 images with 3D bounding boxes, and Open Images consisting of more than 9 million images. For a specific purpose, five models can be selected from the Model ZOO as shown in a table provided.

Model Name	COCO mAP	Outputs	Speed(ms)
SSD ResNet50 V1 FPN 640x640 (RetinaNet50)	34.3	Boxes	46
SSD ResNet50 V1 FPN 1024x1024 (RetinaNet50)	38.3	Boxes	87
SSD ResNet101 V1 FPN 640x640 (RetinaNet101)	35.6	Boxes	57
SSD ResNet101 V1 FPN 1024x1024 (RetinaNet101)	39.5	Boxes	104

Table 1: SSD Models in GTOD API

In the proposed system we decided to go with the SSD ResNet50 V1 FPN 640x640 because of its efficiency and lesser training times.

FLOWCHART

The process of developing a custom object detection model using TensorFlow Object Detection API involves several steps.In this subsectionthe flowchart illustrates the process, which starts with initial setup and configuration. The model is then trained using weights obtained from a pre-trained model in the TensorFlow zoo. This is known as transfer learning, where the pre-trained weights are used to guide the model on what it needs to identify. This allows the model to quickly learn the new task without having to start from scratch.

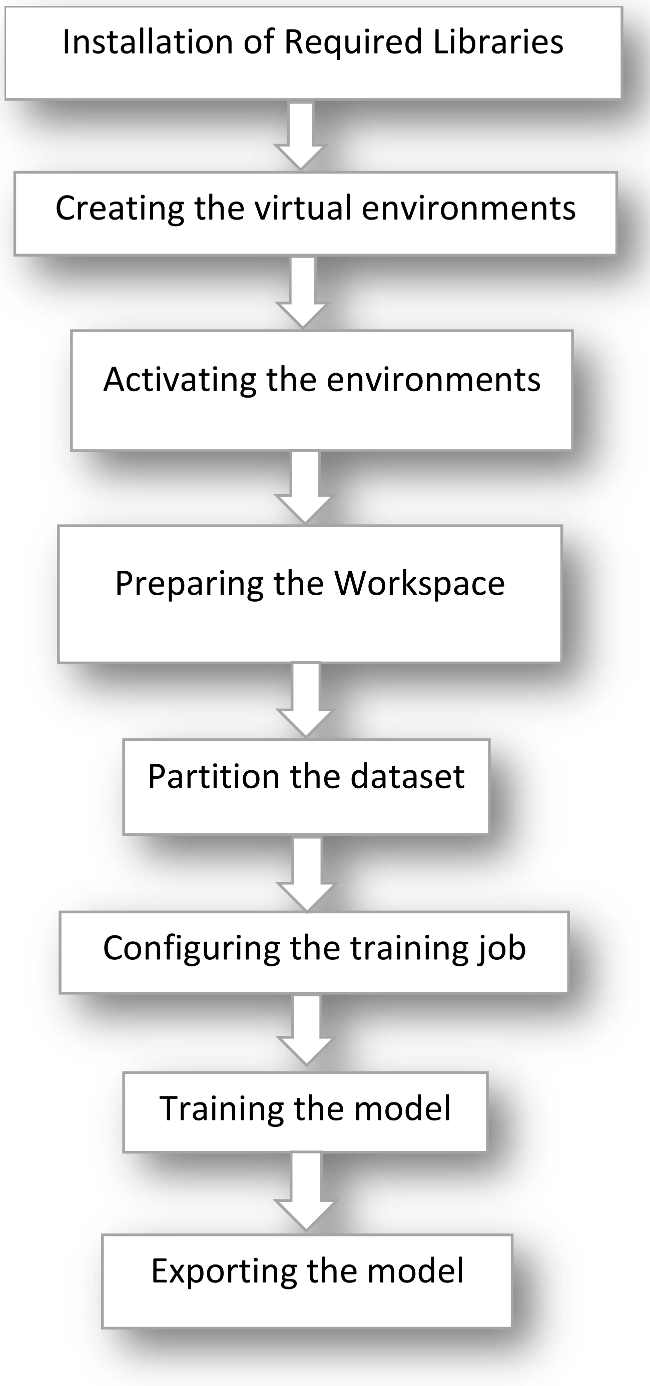

Installation of Required Libraries
Creating the virtual environments
Activating the environments
Preparing the Workspace
Partition the dataset
Configuring the training job
Training the model
Exporting the model

Fig 4: The Flowchart for Training a Custom Object Detection Model

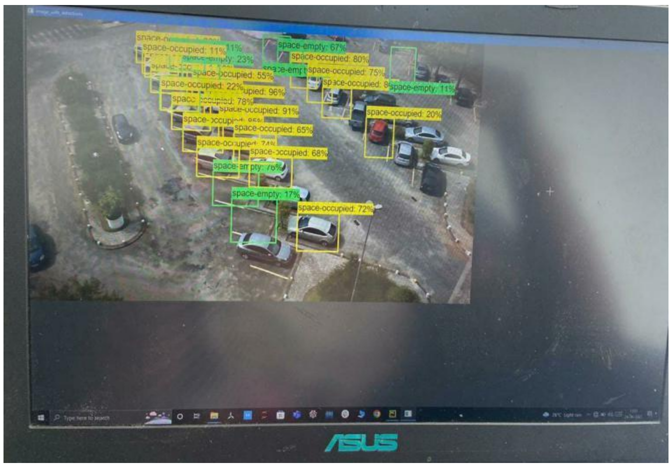
INFERENCE

To check the model's accuracy there needs to be inferencing done. In this subsection let us discuss how to the inferencing was done for the custom model we created.TFOD provides a sample inference code which can be modified according to the needs of the people. After the model was trained the model needed to be tested so we created a custom inference code in python by using the TFOD inference example.Using this .py code live inferencing was achieved using a webcam attached to the laptop.

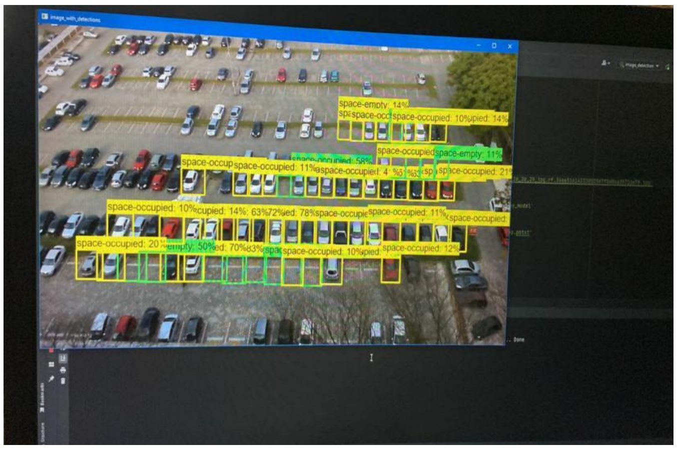
VI. Results

SSD using ResNet was the TensorFlow model to be trained. 50,000 steps were taken in the workout as a whole. Figure uses Anaconda prompt to show the overall loss versus the number of training steps. The losses ranged fromto , respectively. A decreasing number indicates that the model is learning during training and a minimum loss is sought. In the event that the loss does not continue to decline, training can be halted at any time. Every five minutes, it periodically saves a checkpoint during the training.

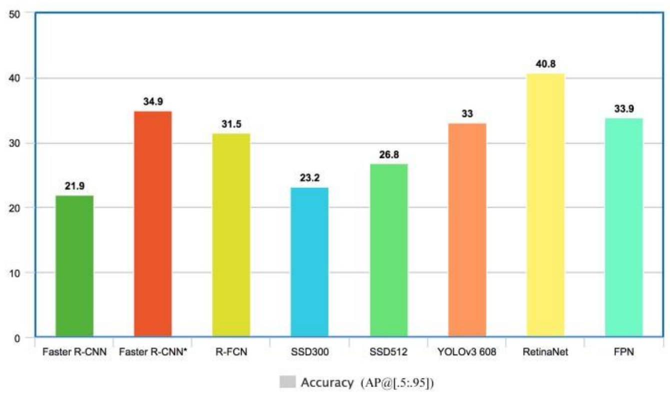
Utilizing an NVIDIA GeForce GTX 1050, training the model took 11 hours. Depending on the computer's GPU power, this will change.



Following the training, the inference graph was exported. The classifier used for object detection is contained in the inference graph. Figure depicts the sample detection with cars parked in a parking lot. Some cars and empty parking lots were not recognised by the SSD using ResNet. Even though this model has some loss, real-time applications can use its fast detection.



The yellow boxes indicate the occupancy of the parking spots and the green boxes indicate that the parking space is empty. The bounding boxes in the parking spots help the object detection algorithm to different between the other objects in the image. The percentage of occupancy of parking spots by the vehicles is displayed and shows us the space occupied in each bounding box.



The above figure shows us the accuracy of models on the COCO data set. The SSD ResNet can be used in real life approaches due to the fast-processing rate compared to the other approaches.

VII. Conclusion

Convolutional neural networks simplify the process of creating classifiers by automatically extracting and using features from your dataset. Computer vision could become a better parking spot detection tool in the future, and less worry about hardware costs and space requirements than other techniques.

As a future work, we will first improve the camera position so that each car in the parking lot can be seen clearly, without a car blocking part of the car next to it, and accuracy of all parking spots in the video. We plan to improve the detection. Since parking space availability is derived from well-defined parking spaces, the system can use several artificial intelligence techniques to direct drivers to the next available parking space

References

[1]. Ohya, A. Kosaka, and A. Kak, "Vision-based navigationby a mobile robot with obstacle avoidance using single-cameravision and ultrasonic sensing,"IEEE Transactions on Roboticsand Automation, vol. 14,pp. 969–978, Dec 1998

[2]. S. Lee, D. Yoon, and A. Ghosh, "Intelligent parking lotapplication using wireless sensor networks," in CollaborativeTechnologies and Systems,2008. CTS 2008. InternationalSymposium on, pp. 48–57, May 2008.

[3]. Q. Wu, C. Huang, S. y. Wang, W. c. Chiu, and T. Chen,"Robust parking space detection considering inter-spacecorrelation," in2007IEEE International Conference onMultimedia and Expo, pp. 659–662,July 2007

[4]. M. Tschentscher, C. Koch, M. Knig, J. Salmen, and M.Schlipsing,"Scalable real-time parking lot classification: Anevaluation of image features and supervised learningalgorithms," in2015 International Joint Conference on NeuralNetworks (IJCNN), pp. 1–8, July 2015.

[5]. P. R. de Almeida, L. S. Oliveira, A. S. B. Jr., E. J. S. Jr.,and A. L.Koerich, "{PKLot}a robust dataset for parking lotclassification, "Expert Systems with Applications, vol. 42, no.11, pp. 4937 – 4949,2015.

[6]. Guilleum Budia Tirado, Sudhanshu Kumar Semwal Autonomous Parking Spot Detection System for Mobile Phones using Drones and Deep Learning, Computer Science Research Notes CSRN 3101 WSCG 2021 Proceedings.

[7]. Alvaro Fuentes, Sook Yoon, Sang Cheol Kim and Dong Sun Park,A Robust Deep-Learning-Based Detector for Real-Time Tomato Plant Diseases and Pests Recognition,Article, MDPI

[8]. EleniI.Vlahogianni,Javier Del Ser, Konstantinos Kepaptsoglou,Ibai Laña, Model Free Identification of Traffic Conditions Using Unmanned Aerial Vehicles and Deep Learning, Journal of Big Data Analytics in Transportation (2021) 3:1–13

[9]. Paulo R.L. de Almeida , Luiz S. Oliveira , Alceu S. Britto Jr., Eunelson J. Silva Jr, Alessandro L. Koerich,PKLot – A robust dataset for parking lot classification,Expert Systems with Applications 42 (2015) 4937–4949

[10]. Galib Ibne Haidar,Hasin Ishraq Reefat,Smart Parking System Using M obileNet and Single ShotDetection A lgorithm and IoT,2020 11th International Conference on Electrical and Computer Engineering (ICECE)

[11]. Cheng-hsiung hsieh, Dung-ching lin, Cheng-jia wang, Zong-ting chen,Jiun-jian liaw,Real-Time Car Detection and Driving Safety alarm system with Google Tensorflow Object Detection API

[12]. Bo Pang, Erik Nijkamp, Ying Nian Wu,Deep Learning With TensorFlow:A Review, Journal of Educational and Behavioral Statistics

[13]. Peter Goldsborough, A Tour of TensorFlow,Proseminar Data Mining

[14]. Hoo-Chang Shin, Holger R. Roth, Mingchen Gao, Le Lu,Ziyue Xu, Isabella Nogues, Jianhua Yao, Daniel Mollura, and Ronald M. Summers,Deep Convolutional Neural Networks forComputer-Aided Detection: CNN Architectures,Dataset Characteristics and Transfer Learning,IEEE TRANSACTIONS ON MEDICAL IMAGING, VOL. 35, NO. 5, MAY 2016

[15]. Ross Girshick,Fast R-CNN, Microsoft Research

[16]. Kaiming He, Georgia Gkioxari ,Piotr Dollar, Ross Girshick,Mask R-CNN,Facebook AI Research (FAIR)

[17]. Hassam Tahir,Muhammad Shahbaz Khan,Muhammad Owais Tariq, Performance Analysis and Comparison of Faster R-CNN, Mask R-CNN and ResNet50 for the Detection and Counting of Vehicles, 2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS) ISBN: 978

Rithik B, et. al. "Car parking slot identification using SSD Resnet and Tensor Flow Object Detection API." *IOSR Journal of Computer Engineering (IOSR-JCE)*, 25(1), 2023, pp. 29-36.