INTERNATIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT (IJSREM)
VOLUME: 06 ISSUE: 12 | DECEMBER - 2022 IMPACT FACTOR: 7.185 ISSN: 2582-3930

Car Parking Slot Detection using Convolutional Neural Networks

Prof. Mukund Kulkarni
Department of Electronics Engineering
Vishwakarma Institute of Technology
Pune, India
mukund.kulkarni@vit.edu

Komalesh Patil
Department of Computer Engineering
Vishwakarma Institute of Technology
Pune, India
komalesh.patil20@vit.edu

Chetan Patil
Department of Computer Engineering
Vishwakarma Institute of Technology
Pune, India
chetan.patil20@vit.edu

Tanish Modase
Department of Computer Engineering
Vishwakarma Institute of Technology
Pune, India
tanish.modase20@vit.edu

Tushar Muley
Department of Computer Engineering
Vishwakarma Institute of Technology
Pune, India
tushar.muley20@vit.edu

Aditya Pachore
Department of Computer Engineering
Vishwakarma Institute of Technology
Pune, India
aditya.pachore20@vit.edu

***Abstract* - Finding a parking spot has become a problem that must be addressed; it takes time and effort. Finding a parking place, especially in metropolitan locations, is a massive pain. This study tries to build one such parking system that decreases the inconveniences of parking in a variety of ways. The study describes a system that uses a Convolution Neural Network (CNN) machine learning model to evaluate parking slots in a parking spot as vacant or filled. Parking troubles are not restricted to creating annoyance to cars; they may also develop far larger and more widespread issues, affecting a considerably larger number of people and the environment. As a result, having a reliable parking infrastructure is essential. The technique presented in the research communicates parking information to a driver in advance, considerably decreasing the time spent waiting for a car.**

Keywords* - *Convolutional Neural Network (CNN), Machine Learning, Deep Learning, Image Processing, Computer Vision

I. INTRODUCTION

Nowadays, finding unoccupied or vacant parking spaces is difficult. This takes them roughly 6-8 minutes. This is the biggest cause of traffic congestion in major cities. In order to alleviate common societal concerns, smart cities require a smart parking system. The number of vehicles used to be greater than the available parking spaces. Violence increases as a consequence of overcrowding. Some of the issues that arise include distorted automobiles as a result of a space crisis and overcharging for parking. The majority of cities are suggesting adding parking lots to address this issue. Despite the limited space and resources, parking spaces are developed in parks, vacant lots, and high-rise buildings.[1] Smart Management enables traffic and time management to flow smoothly. Different parking space detection algorithms were developed by certain researchers, who used devices such as video cameras or sensors to identify a vacant place when it was needed. Automatic parking detection was developed when these algorithms were combined.

The technology is designed to be used in open spaces such as parking lots at retail malls or university parking lots. Implementing a sensor-based strategy to automated parking space recognition will result in extra paperwork, lost time, and inconvenience for users while such sensors are being installed on the ground. These parking spots, on the other hand, are monitored by video cameras for security reasons, as is the case in most shopping malls. Convolutional Neural Networks (CNN) have a structure that resembles that of a human neural network, which is made up of synapses and neurons. From this standpoint, complex activities might be taught through the network. This system uses CNN in conjunction with pre-existing architectures to detect the vacancy of a parking slot in real time.[2]

II. LITERATURE REVIEW

There have recently been a slew of studies based on neural networks and artificial intelligence. Many scholars are developing different kinds of neural networks to tackle a variety of problem scenarios.

A. Car Parking Occupancy Detection Using Smart Camera Networks and Deep Learning

The paper explains a Convolutional Neural Network classifier is used in this study to classify images captured by a smart camera with limited resources. The detection methods utilized are reliable even when tests are done using photographs taken from a different viewpoint from the one used for training. It also displays its sturdiness when training and tests are conducted on diverse parking lots. Ground sensors are utilized in many parking lots to evaluate the state of the individual slots. This necessitates the installation and maintenance of sensors in each parking space, which can be expensive, particularly in parking lots with multiple vacant

spaces. Despite these noteworthy achievements, locating available parking places simply based on visual input remains a challenge. The proposed parking identification approach in this paper follows a recent trend of using Deep Learning techniques, specifically CNNs, to solve issues that require a high level of abstraction, such as vision, which requires a greater level of accuracy.[2]

B. Automated Parking Space Detection Using Convolutional Neural Networks

This paper shows how to use Caffe and the Nvidia DiGITS framework to create a real-time parking space categorization system based on Convolutional Neural Networks (CNN). DiGITS was used to train the model, and the result is a caffemodel that can be used to predict if a parking place is available or occupied. The system examines a defined area to see if a parking spot (bounding boxes defined at system startup) is occupied or not (occupied or vacant). Those bounding box coordinates are stored in JSON format from a frame of the parking lot camera, to be used subsequently by the system for sequential parking space prediction. The LeNet network with the Nesterov Accelerated Gradient and the AlexNet network with the Stochastic Gradient Descent as solver were used to train the system. On the validation set, both networks had a 99 percent accuracy rate. The accuracy was also 99 percent on a foreign dataset (PKLot).[3]

C. Real-time image-based parking occupancy detection using deep learning

The system presents an approach by using a CNN and binary SVM classifier. The features learned by the CNN from the PKLot and Barry street datasets were used to train and test the classifier. They evaluated the VGG model for trained classifiers and reported an accuracy of approximately 95%, while mAlexNet for binary classification reported an accuracy of 90.7%.[4]

D.Parking slot detecting system based on structure similarity index

The study talked about a model that was split into two parts: setup configuration and object detection. Canny Edge and Hough Line Transform are used to detect lines for parallel parking slot marking. The Structural Similarity Index Measurement (SSIM) evaluates the structure of the reference and target images to see if the region is empty or inhabited. The detection accuracy and precision of 92 percent and 89 percent, respectively, were obtained utilizing selected features from 50 sample photos of parking spots captured by surveillance camera.[5]

III. DATASETS

We have used a dataset comprising 545 parking slots images occupied and empty, each with different views and angles of view, as well as varying light conditions and occlusion patterns. This dataset was created using a three-step process that included image acquisition, labeling, and segmentation.[11] With 381 photos for training and 164 images for testing, the images were separated into two sets. And further into two classes occupied and empty.

IV. METHODOLOGY

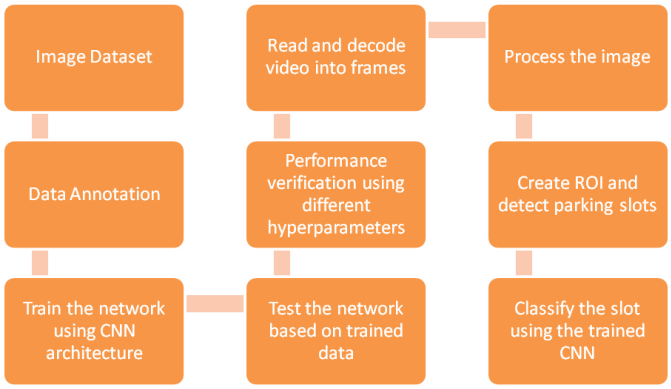

Fig. 1 Block Diagram of Proposed System

In this system, we used the VGG16 and mAlexNet CNN architectures. In addition to having five convolutional layers, AlexNet's design also includes five convolutional layers, two fully connected layers, and one softmax layer. While VGG16 has 16 layers total, 12 of them are convolutional, 3 are fully connected, and one is a softmax layer.

Following that, the parking detection model is built in two stages. The initial step is to locate all parking place s that are available. The main concept is to use canny edge detection and create a mask for any areas that aren't needed. Apply the Hough line transform to the edge picture after that. Finding out if a parking space is available is the next step. Once the location of each parking place is known, a CNN is used to look at each one and forecast whether it is filled or not.

A. Importing Libraries

Since the project is written in Python, we imported the relevant libraries for analysis, training and validation of the proposed model. For data cleaning, pre-processing and statistical tools, we utilized python libraries such as numpy, matplotlib, ImageDataGenerator and cv2. For implementing deep learning, keras was used. In Fig. 2, a code snippet for importing relevant libraries is provided.

```
import numpy
import os
from tensorflow import keras
from keras.preprocessing.image import ImageDataGenerator
from keras import optimizers
from keras.models import Sequential, Model
from keras.layers import Dropout, Flatten, Dense, Activation, Dropout, Conv2D, MaxPooling2D
from keras.layers.normalization import batch_normalization
from keras import backend as k
from keras.callbacks import ModelCheckpoint, ReduceLROnPlateau, EarlyStopping
from __future__ import division
import numpy as np
import matplotlib.pyplot as plt
import cv2
import os, glob
from moviepy.editor import VideoFileClip

%matplotlib inline
%config InlineBackend.figure_format = 'retina'
```

Fig. 2. Importing Libraries

B. *Image Processing*

We must preprocess the photograph in order to improve its quality. The processes like gray scaling and noise reduction are applied as shown in Fig. 3.

Fig. 3. Gray scaled image

C. *Canny Edge Detection*

Canny edge detection, a technique that drastically reduces the quantity of data that needs to be evaluated in order to extract structural information from a range of visual objects. Many computer vision systems have used it in their designs.

Some common techniques for edge detection are as follows:

a. Edge detection with a low error rate, implying that the detection should capture as many of the image's edges as possible.
b. The operator's edge point has to be accurate to determine the center of edge.
c. The edge of an image should, whenever possible, only be marked once and false edges shouldn't be caused by picture noise.

Fig. 4. Edge Detected image

D. *Region of Interest*

When it comes to determining the computational difficulty of slot identification, ROI is crucial. Only the regions that have been chosen are detected and transmitted to the next stage of processing. These ROI images are then sent into a suggested parking space recognition system. The use of ROI shortens the time for the frames to be processed.

Fig. 5. Image after detecting region of interest

E. *Hough Transform*

To separate the features of distinct forms inside a picture, utilize the Hough Transform approach. This method is commonly used to identify arbitrary forms such as straight lines, circles, ellipses, and so on. Following the detection of canny edges, the Hough Transform is applied to images in order to recover the desired image pixels, as shown in Fig. 6. The Hough Transform is utilized in this system to find the lanes of parking slots from given image data. The Hough Transform technique is appealing because it allows for gaps in feature border descriptions and is unaffected by noise.

Fig. 6. Drawing rectangles at all the parking slots

F. Compiling the CNN model

VGG16 CNN was used for feature extraction, and the VGG16 feature extractor's vectored outputs were then passed to fully connected dense layers for binary classification. The following are the parameters used in the suggested neural network model. The first step is to decrease the size of the input picture to 48x48. It is then input into the classification model, which consists of twelve two dimensional convolution layers of 3x3 filters with 1-pixel stride and five 2D max-pooling layers having kernel size of 2x2 and distance of two pixels, which are organized as shown in Table I. The final output layer and a flattening layer, are sequentially added to provide binary outputs.[20] Fig. 7. represents the model's executive summary compiled.

Table I. The network structure of VGG16 CNN architecture

Layer		Size	Kernel Size	Stride	Activation
Input	Image	48 x 48 x 3	-	-	-
1	2 X Convolution	48 x 48 x 64	3x3	1	relu
	Max Pooling	24 x 24 x 64	3x3	2	relu
3	2 X Convolution	24 x 24 x 128	3x3	1	relu
	Max Pooling	12 x 12 x 128	3x3	2	relu
5	3 X Convolution	12 x 12 x 256	3x3	1	relu
	Max Pooling	6 x 6 x 256	3x3	2	relu
7	3 X Convolution	6 x 6 x 512	3x3	1	relu
	Max Pooling	3 x 3 x 512	3x3	2	relu
10	3 X Convolution	3 x 3 x 512	3x3	1	relu
	Max Pooling	1 x 1 x 512	3x3	2	relu
Output	FC	2	-	-	Softmax

```
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         [(None, 48, 48, 3)]       0
block1_conv1 (Conv2D)        (None, 48, 48, 64)        1792
block1_conv2 (Conv2D)        (None, 48, 48, 64)        36928
block1_pool (MaxPooling2D)   (None, 24, 24, 64)        0
block2_conv1 (Conv2D)        (None, 24, 24, 128)       73856
block2_conv2 (Conv2D)        (None, 24, 24, 128)       147584
block2_pool (MaxPooling2D)   (None, 12, 12, 128)       0
block3_conv1 (Conv2D)        (None, 12, 12, 256)       295168
block3_conv2 (Conv2D)        (None, 12, 12, 256)       590080
block3_conv3 (Conv2D)        (None, 12, 12, 256)       590080
block3_pool (MaxPooling2D)   (None, 6, 6, 256)         0
block4_conv1 (Conv2D)        (None, 6, 6, 512)         1180160
block4_conv2 (Conv2D)        (None, 6, 6, 512)         2359808
block4_conv3 (Conv2D)        (None, 6, 6, 512)         2359808
block4_pool (MaxPooling2D)   (None, 3, 3, 512)         0
block5_conv1 (Conv2D)        (None, 3, 3, 512)         2359808
block5_conv2 (Conv2D)        (None, 3, 3, 512)         2359808
block5_conv3 (Conv2D)        (None, 3, 3, 512)         2359808
block5_pool (MaxPooling2D)   (None, 1, 1, 512)         0
flatten_1 (Flatten)          (None, 512)               0
dense_1 (Dense)              (None, 2)                 1026
=================================================================
Total params: 14,715,714
Trainable params: 12,980,226
Non-trainable params: 1,735,488
_________________________________________________________________
```

Fig. 7. VGG16 model summary

As shown in Table II, Alexnet is a CNN pre-trained model having five convolutional layers, three max-pooling layers, three layers with full connectivity and one softmax layer. Convolutional filters are used in each convolutional layer, and the ReLU nonlinear activation function. Max pooling is done using the pooling layers. In most locations, the input size is specified as 224x224x3. We defined an activation layer of an activation function named 'ReLU' for each convolution layer. Fig. 8 represents the summary of the model compiled.

Table II. The network structure of AlexNet CNN architecture

Layer		Size	Kernel Size	Stride	Activation
Input	Image	224 x 224 x 3	-	-	-
1	Convolution	54 x 54 x 96	11x11	4	relu
	Max Pooling	26 x 26 x 96	3x3	2	-
2	Convolution	22 x 22 x 256	5x5	1	relu

	Max Pooling	10 x 10 x 256	3x3	2	-
3	Convolution	8 x 8 x 384	3x3	1	relu
4	Convolution	6 x 6 x 384	3x3	1	relu
5	Convolution	4 x 4 x 256	3x3	1	relu
	Max Pooling	1 x 1 x 256	3x3	2	-
6	FC	4096	-	-	relu
7	FC	4096	-	-	relu
8	FC	1000	-	-	relu
Output	FC	2	-	-	Softmax

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 54, 54, 96)	34944
activation (Activation)	(None, 54, 54, 96)	0
max_pooling2d (MaxPooling2D)	(None, 26, 26, 96)	0
conv2d_1 (Conv2D)	(None, 22, 22, 256)	614656
activation_1 (Activation)	(None, 22, 22, 256)	0
max_pooling2d_1 (MaxPooling 2D)	(None, 10, 10, 256)	0
conv2d_2 (Conv2D)	(None, 8, 8, 384)	885120
activation_2 (Activation)	(None, 8, 8, 384)	0
conv2d_3 (Conv2D)	(None, 6, 6, 384)	1327488
activation_3 (Activation)	(None, 6, 6, 384)	0
conv2d_4 (Conv2D)	(None, 4, 4, 256)	884992
activation_4 (Activation)	(None, 4, 4, 256)	0
max_pooling2d_2 (MaxPooling 2D)	(None, 1, 1, 256)	0
flatten (Flatten)	(None, 256)	0
dense (Dense)	(None, 4096)	1052672
activation_5 (Activation)	(None, 4096)	0
dropout (Dropout)	(None, 4096)	0
dense_1 (Dense)	(None, 4096)	16781312
activation_6 (Activation)	(None, 4096)	0
dropout_1 (Dropout)	(None, 4096)	0
dense_2 (Dense)	(None, 1000)	4097000
activation_7 (Activation)	(None, 1000)	0
dense_3 (Dense)	(None, 2)	2002
activation_8 (Activation)	(None, 2)	0

Total params: 25,680,186
Trainable params: 25,680,186
Non-trainable params: 0

Fig. 8. AlexNet model summary

G. *Fitting the Model with Training Data*

Once the model is built, it is important to fit it with training and testing data generated earlier. To do that, we imported ModelCheckpoint, EarlyStopping and ReduceLROnPlateau from the keras.callbacks module. For training the model, we set the epochs as 15. It is possible to save the best model and stop the model's training if the accuracy does not improve with more epochs using early stopping. Models can also benefit from ReduceLROnPlateau's ability to reduce the learning rate by a factor of 0.2 as observed metrics grow or decrease.

H. *Capture Input Video*

As an input device, the algorithm uses a camera. The video can be initialized with the cv2.VideoCapture() function. Using OpenCV, the video streams are then used as the system's input data. OpenCV can split video data into frames as well.The CNN model is then loaded on the video frame and thus the algorithm is implemented.

V. RESULTS AND DISCUSSIONS

For the experiments, the comparison between AlexNet and VGG16 allowed us to run the categorization on a recorded video containing parking lots.

The VGG16 model was found to be quite accurate in terms of training data. At the end of epoch 13, the model produced with an accuracy of 98.01% and loss of 7.03% as depicted in Table III. The model produced validation accuracy of 94.38%.

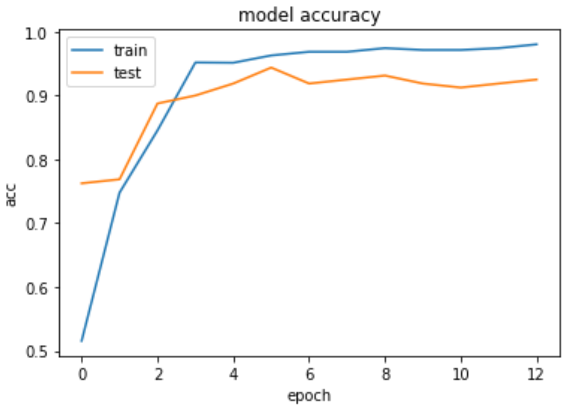

Fig. 9. VGG16 model accuracy

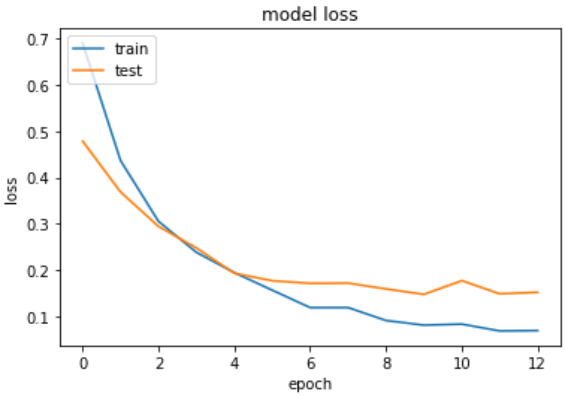

Fig. 10. VGG16 model loss

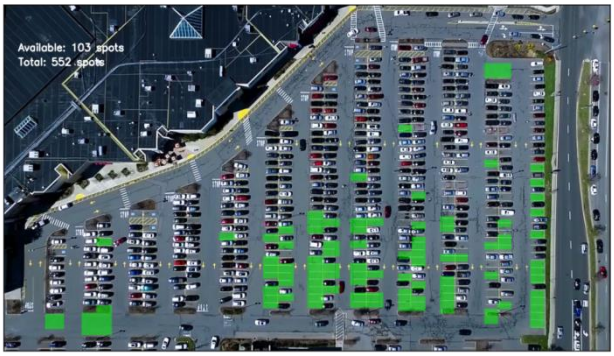

Fig. 11. depicts the empty car parking slots by using VGG16 CNN architecture

Table III. Model accuracy and loss for VGG16 and AlexNet architecture

CNN Architecture	Accuracy	Loss	Validation Accuracy	Validation Loss
VGG16	96.28%	15.74%	94.38%	17.76%
AlexNet	94.56%	15.55%	96.25%	33.13%

The AlexNet model was found accurate in terms of validation data. The model produced an validation accuracy of 96.25% and loss of 15.55% as depicted in Table III. Thus, the AlexNet model can accurately detect the empty parking slots.

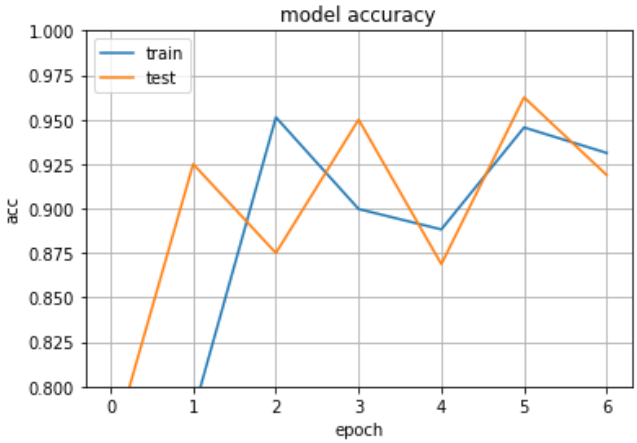

Fig. 12. AlexNet model accuracy

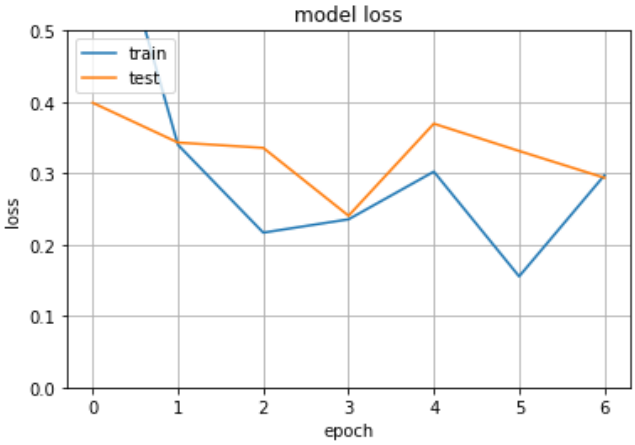

Fig. 13. AlexNet model loss

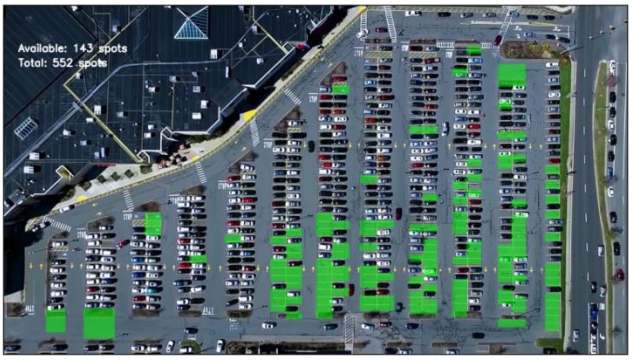

Fig. 14. depicts the empty car parking slots by using AlexNet CNN architecture

VI. FUTURE SCOPE

The proposed system can be integrated in a mobile application for users to find the vacant parking spot for a car.

In case of heavy traffic, instead of using a mobile application, the system can be linked with a smart voice assistant to answer the empty parking slot.

We can use direct commuters to book vacant parking slots. For instance, QR code can be generated for empty slots and as the user scans the QR the nearest empty slot will be booked.

Along with detection of the slot, license plate recognition can be done, which can keep track of the parking slot and besides who has parked the car can also be tracked.

VII. CONCLUSION

In this paper, we show that utilizing Convolutional Neural Networks and a single camera, we can get a success rate that is comparable to that of older approaches. The VGG16 architecture-based classifier had a validation accuracy of 94.38%, whereas the AlexNet architecture had a validation accuracy of 96.25%.

Convolutional Neural Networks (CNNs) make developing a classifier easier because they automatically extract and utilize data from the dataset. Fig. 14 shows how the system can create an accurate solution based on the present status of the parking spot.

Finally, this example shows how car slot recognition can be used to track parking lot capacity over time without requiring expensive hardware such as GPUs or the time and effort of parking lot managers patrolling at regular intervals.

VIII. ACKNOWLEDGEMENT

We would like to express the special thanks of gratitude to our project guide Prof. Mukund Kulkarni who gave us the golden opportunity to do this wonderful project on the topic Car Parking Slot Detection using CNN, who also helped us in doing a lot of research. And, we came to know about so many new things and we are really thankful to him.

IX. REFERENCES

[1] Shubhankar Gautam, “Issues with Parking in Indian Metropolises”

[2] Giuseppe Amato, Fabio Carrara , Fabrizio Falchi , Claudio Gennaro “Car parking occupancy detection using smart camera networks and Deep Learning”

[3] Julien Nyambal, Richard Klein “Automated parking space detection using convolutional neural networks”

[4] Debaditya Acharya, Weilin Yan, Kourosh Khoshelham “Real-time image-based parking occupancy detection using deep learning”

[5] Ginanjar Suwasono Adi, Muhammad Yusuf Fadhlan, S Slameta, Griffani Megiyanto Rahmatullah “Parking slot detection system based on structural similarity index”

[6] Paulo Almeida, Luiz S. Oliveira, Alceu S. Britto Jr, Eunelson J. Silva Jr, Alessandro Koerich “PKLot – A Robust Dataset for Parking Lot Classification”

[7] Adesh Pawar, Ajay Pawar, Ashish Pawar, Ganesh Pawar, Anagha Chaudhari “An Elaborative Study of Smart Parking Systems” Vol 10 Issue 10, October – 2021

[8] Giuseppe Amato, Fabio Carrara, Fabrizio Falchi, Claudio Gennaro, Carlo Meghini, Claudio Vairo “Car Parking Occupancy Detection Using Smart Camera Networks and Deep Learning” June 2016

[9] Jordan Cazamias, Martina Marek “Parking Space Classification using Convolutional Neural Networks” Stanford University

[10] “Develop a Computer Vision Parking Lot Occupancy Application” Viso Suite Guide

[11] Janak Trivedi, Mandalapu Sarada Devi, Dave Dhara “Canny Edge Detection based real-time intelligent parking management system”

[12] S Rahman, M Ramli, F Arnia, R Muharar, M Luthfi, S Sundari “Analysis and Comparison of Hough Transform Algorithms and Feature Detection to Find Available Parking Spaces”

[13] Vignesh Dhuri, Afzal Khan, Yash Kamtekar, Dhananjay Paatel, Ichhanshu Jaiswal “Real-Time Parking Lot Occupancy Detection System with VGG16 Deep Neural Network using Decentralized Processing for Public, Private and Parking Facilities”