

VEHICLE MODEL CLASSIFICATION USING DEEP LEARNING

Submitted in partial fulfillment of the requirements
for the award of
Bachelor of Engineering degree in Computer Science and Engineering

By

YADHU NANDHA KRISHNA K (37110857)
JAYAKUMAR S (37110287)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

March - 2021



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with “A” grade by NAAC | 12B Status by UGC | Approved by
AICTE

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119
www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide for the Project work done as a team JAYAKUMAR S (37110287), YADHU NANDHA KRISHNA K (37110857) who carried out the project entitled **“VEHICLE MODEL CLASSIFICATION USING DEEP LEARNING”** under my supervision from **November 2020 to March 2021**.

Internal Guide
Dr. Ajitha P , M.E., Ph.D.,

Head of the Department
Dr. S.Vigneshwari M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I, **YADHU NANDHA KRISHNA K (REG.NO. 37110857)** hereby declare that the project report entitled “**VEHICLE MODEL CLASSIFICATION USING DEEP LEARNING**” done by us under the guidance of **Dr. P.Ajitha M.E., Ph.D.**, at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to the Board of management of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to to **Dr. SASIKALA. T, M.E., Ph.D., Dean, School of Computing** and **Dr. S. VIGNESHWARI M.E., Ph.D., and Dr.L.Lakshmanan M.E., Ph.D., Heads of the Department,** Department of Computer Science and Engineering for providing me the necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. P. Ajitha M.E., PhD.,** for her valuable guidance, suggestions which paved way for the successful completion of my project work.

I wish to express my thanks to all teaching and non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of project.

ABSTRACT

These days, there are many traffic observation frameworks which are introduced in pretty much every city to record occasions and traffic. The reconnaissance framework is utilized for different goals, for example vehicles looking and continuous traffic observing, and so on. For the searching purpose, the framework can be utilized by police officer, for example, fugitive's vehicle ID in wrongdoing. Regularly, the officials physically distinguish the vehicle in recorded video as per its appearances. In spite of the fact that the exactness of this methodology is acceptable, the time has come burning-through and slanted to shortcomings because of human weariness for long length recordings.

Besides, recruiting workers is difficult. As of late, there are a few AI techniques which can be applied to order vehicles, for example Fuzzy Logic, Decision Tree, Adaboost, Random Forest, Neural Network, etc. Convolutional Neural Network is additionally one of such techniques. CNN is a kind of Profound Realizing which is in the class of the neural network. The technique is very notable in image classification field at the present due to its performance. In the proposed vehicle classification, we are going to use the state of the art transfer learning techniques to classify the vehicle of more than 95 % accuracy.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	vi
	LIST OF ABBREVIATIONS	vii
	LIST OF FIGURES	vii
		i
1	INTRODUCTION	
	1.1 Introduction	2
	1.2 Problem Statement	3
	1.3 Objective	3
2	LITERATURE SURVEY	
	2.1 Literature Survey	8
3	AIM AND SCOPE OF INVESTIGATION	
	3.1 System Analysis	10
	3.1.1 Existing System	11
	3.1.2 Proposed System	11
4	EXPERIMENTAL MATERIALS AND METHODS USED	
	4.1 Overview	14
	4.2 System Requirement	15

	4.2.2 Hardware Requirement	15
	4.3 System Architecture	16
	4.3.1 Architecture Diagram	16
	4.4 System Design	17
	4.4.1 UML Diagram	17
5	RESULTS AND DISCUSSION	
	5.1 Screenshots	21
6	SUMMARY AND CONCLUSIONS	
	6.1 Conclusions	22
	6.2 Future Scopes	23
	REFERENCE	25
	APPENDIX	
	A. Source Code	34
	B. Plagarism Report	35
	C. Paper	43

LIST OF ABBREVIATIONS

ABBREVIATIONS

EXPANSIONS

CNN

Convolutional Neural Network

PY

Python

UML

Unified Modelling Language

LIST OF FIGURES

FIGURE NO.	NAME	PAGE NO.
4.1	Data Set(Car Model)	12
4.2	Flow Chart for Model	13
4.3	Flow Chart for Application	14
4.4	Architecture Diagram for Application	16
4.5	UML Diagram	17
5.1	Feature Extraction	18
5.2	Classifier Model Execution	19
5.3	Classified Result (Aston_Martin)	20
5.4	Classified Result (BMW)	21

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION:

The establishment of a comprehensive, efficient and accurate intelligent transportation system can effectively reduce traffic pressure and solve congestion problems. Vehicle type classification system is a very important link in the intelligent transportation system.

The image-based vehicle type classification system will comprehensively improve the level of information collection and safety management of highways and effectively improve road congestion. So far, there are two main methods for vehicle type classification.

The first is the classification based on physical patterns, such as inductive coil detection and infrared detection. These methods are expensive and not suitable for large-scale use. The second is to extract the features of the vehicle first, and then the classifier is used to classify the vehicle type.

Convolutional neural network is a kind of feed forward neural network with deep structure and convolution computation. Feature recognition and extraction of the input image data can be carried out through convolution operation, so as to achieve the purpose of image recognition.

It is generally composed of input layer, convolution layer, activation function, pooling layer and fully connected layer. It is particularly important to design a convolutional neural network with stronger learning ability.

In this paper, an improved network model architecture is proposed for vehicle type classification, which has achieved good results.

Images used in the experiment in this paper are the monitoring images of vehicles in the expressway from the BIT-Vehicle dataset, respectively by two cameras in different time and place. The size of the pictures in the dataset varies.

The two types of vehicles may exist simultaneously in one picture at the same time, and the vehicle body only accounts for a small proportion of one picture. Therefore, before inputting image data into convolutional neural network for operation, corresponding preprocessing is needed. The image of the sample set is cropped and the part of the vehicle body area is left. The cropped image is a square to avoid the distortion of the image after resizing. The processed images are placed in different folders according to different categories of vehicle types to facilitate feature extraction.

In this paper, an improved convolutional neural network vehicle type classification system is designed, and the effectiveness of the improved convolutional neural network model is verified by the experimental research on the TensorFlow platform. A series of improvement methods make the model show a good effect on the accuracy of vehicle type classification. In the future, we will add more types of vehicle type images to the dataset, and further adjust the network structure and parameters on the basis of improving the network model to obtain the optimal network for vehicle classification.

CNN is an artificial neural network with multiple hidden layers, and multilayer nonlinear structure makes it have powerful feature expression ability. CNN contains three kinds of basic structures, the convolutional layer, the pooling layer and the fully-connected layer. The deep convolutional neural network combines local connection, weight sharing and space down sampling strategies, which greatly reduces the training parameters of the network, improves the accuracy of classification and the generalization ability of the network.

1.2 PROBLEM STATEMENT:

In order to help for the initial investigation purpose of identification of vehicles. A large number of car images must be required as data set for training of data model. Here, CNN plays a crucial role in the extraction of features from the images. In this paper, a deep learning classification technique(CNN) is used to classify the vehicle features. As accuracy is the most significant factor in this issue, by taking a more prominent number of pictures for training the network and by increasing the number of iterations, the accuracy can be improved. Keras and numpy play a vital role with respect to deep learning packages in python.

1.3 OBJECTIVE:

Proving a effective Image Identifaction analysis by creating a Deep Learning Model using CNN in python environment.

CHAPTER 2

LITERATURE SURVEY

2.1 Literature survey

Intelligent Traffic Signal Management System Using Cloud Vision API and Machine Learning

Author: Shwetha R. J, Srinath M, Vibha S

Time and money has equal powers, to save time humans do not give a second thought for money and this goes vice-verse. This is why nearly every house has a vehicle in present situation. The effect of this has lead to the increasing traffic especially during peak hours. Research is still going on this field to have an efficient way of traffic management. This paper is on one such proposal where dealing with the traffic management is different from the existing technology. We are mainly focusing on two aspects which although nearly interrelated. Firstly on cut shot the timing of the less dense traffic (keeping equal priority on each lane) to achieve this we use cloud vision APIs which incorporates Machine Learning by using image taken by HD cameras as our input based on the congestion on the next lane the whole system is carried. The former being the traffic signal jump or breaking law enforcement this is done by using IOT device RFID which send the alert to the owner of the vehicle through cloud.

Vehicle classification based on seismic signatures using convolutional neural network

Author: G. Jin, B. Ye, Y. Wu, and F. Qu

Seismic signals can be used for vehicle classification. However, this task becomes difficult as a result of various noises. Convolutional neural networks (CNNs) have been employed successfully in many fields as a result of its ability to learn low-/mid-/high-level features. This letter investigates the application of CNN to classify vehicles by means of the seismic trace that the geophone recorded. The study has two primary contributions. First, a deep CNN architecture for vehicle classification by seismic signal is proposed. Second, considering the similarities between speech recognition and vehicle classification based on seismic signal, log-scaled frequency cepstral coefficient (LFCC) matrix is proposed to extract features of seismic signals as the input of CNN. The data from DARPA's SensIt project, which contain seismic signals from two kinds of vehicles, are used to evaluate the method. By combining the proposed LFCC matrix and CNN architecture, the algorithm produces a state-of-the-art result compared with other methods.

A survey and comparison of low-cost sensing technologies for road traffic monitoring

Author: M. Bernas, B. Płaczek, W. Korski, P. Loska

This paper reviews low-cost vehicle and pedestrian detection methods and compares their accuracy. The main goal of this survey is to summarize the progress achieved to date and to help identify the sensing technologies that provide high detection accuracy and meet requirements related to cost and ease of installation. Special attention is paid to wireless battery-powered detectors of small dimensions that can be quickly and effortlessly installed alongside traffic lanes (on the side of a road or on a curb) without any additional supporting structures. The comparison of detection methods presented in this paper is based on results of experiments that were conducted with a variety of sensors in a wide range of configurations. During experiments various sensor sets were analyzed. It was shown that the detection accuracy can be significantly improved by fusing data from appropriately selected set of sensors. The experimental results reveal that accurate vehicle detection can be achieved by using sets of passive sensors. Application of active sensors was necessary to obtain satisfactory results in case of pedestrian detection.

Toward mitigating phantom jam using vehicle-to-vehicle communication

Author: M. Won, T. Park, and S. H. Son

Traffic jams often occur without any obvious reasons such as traffic accidents, roadwork, or closed lanes. Under moderate to high traffic density, minor perturbations to traffic flow (e.g., a strong braking motion) are easily amplified into a wave of stop-and-go traffic. This is known as a phantom jam. In this paper, we aim to mitigate phantom jams leveraging the three-phase traffic theory and vehicle-to-vehicle (V2V) communication. More specifically, an efficient phantom jam control protocol is proposed in which a fuzzy inference system is integrated with a V2V-based phantom jam detection algorithm to effectively capture the dynamics of traffic jams. Per-lane speed difference under traffic congestion is taken into account in the protocol design, so that a phantom jam is controlled separately for each lane, improving the performance of the proposed protocol. We implemented the protocol in the Jist/SWAN traffic simulator. Simulations with artificially generated traffic data and real-world traffic data collected from vehicle loop detectors on Interstate 880, California, USA, demonstrate that our approach has by up to 9% and 4.9% smaller average travel times (at penetration rates of 10%) compared with a state-of-the-art approach, respectively.

DeepWiTraffic: Low cost WiFi-based traffic monitoring system using deep learning

Author: M. Won, S. Sahu, and K.-J. Park

A traffic monitoring system (TMS) is an integral part of Intelligent Transportation Systems (ITS). It is an essential tool for traffic analysis and planning. One of the biggest challenges is, however, the high cost especially in covering the huge rural road network. In this paper, we propose to address the problem by developing a novel TMS called DeepWiTraffic. DeepWiTraffic is a low-cost, portable, and non-intrusive solution that is built only with two WiFi transceivers. It exploits the unique WiFi

Channel State Information (CSI) of passing vehicles to perform detection and classification of vehicles. Spatial and temporal correlations of CSI amplitude and phase data are identified and analyzed using a machine learning technique to classify vehicles into five different types: motorcycles, passenger vehicles, SUVs,

pickup trucks, and large trucks. A large amount of CSI data and ground-truth video data are collected over a month period from a real-world two-lane rural roadway to validate the effectiveness of DeepWiTraffic. The results validate that DeepWiTraffic is an effective TMS with the average detection accuracy of 99.4% and the average classification accuracy of 91.1% in comparison with state-of-the-art non-intrusive TMSs.

CHAPTER 3

AIM AND SCOPE OF THE PRESENT INVESTIGATION

3.1 SYSTEM ANALYSIS:

CNN:

CNN is an artificial neural network with multiple hidden layers, and multilayer nonlinear structure makes it have powerful feature expression ability. CNN contains three kinds of basic structures, the convolutional layer, the pooling layer and the fully-connected layer. The deep convolutional neural network combines local connection, weight sharing and space down sampling strategies, which greatly reduces the training parameters of the network, improves the accuracy of classification and the generalization ability of the network.

Pooling Layer:

Convolutional networks may include local or global pooling layers to streamline the underlying computation. Pooling layers reduce the dimensions of the data by combining the outputs of neuron clusters at one layer into a single neuron in the next layer. Local pooling combines small clusters, typically 2 x 2. Global pooling acts on all the neurons of the convolutional layer. There are two common types of pooling: max and average. *Max pooling* uses the maximum value of each cluster of neurons at the prior layer, while *average pooling* instead uses the average value.

Convolutional Layer:

In convolutional layer, some kernels are used to do convolution operation with the

feature maps obtained from the previous layer. After through a activation function, output the feature maps with the same number of kernels.

Fully Connected Layer:

Fully connected layers connect every neuron in one layer to every neuron in another layer. It is the same as a traditional multi-layer perceptron neural network (MLP). The flattened matrix goes through a fully connected layer to classify the images.

Weights:

Each neuron in a neural network computes an output value by applying a specific function to the input values coming from the receptive field in the previous layer. The function that is applied to the input values is determined by a vector of weights and a bias (typically real numbers). Learning consists of iteratively adjusting these biases and weights.

The vector of weights and the bias are called *filters* and represent particular features of the input (e.g., a particular shape). A distinguishing feature of CNNs is that many neurons can share the same filter. This reduces memory footprint because a single bias and a single vector of weights are used across all receptive fields sharing that filter, as opposed to each receptive field having its own bias and vector weighting.

3.1.1 EXISTING SYSTEM:

- As the population of the modern cities is increasing, the vehicular travel is also increasing leading to congestion on roads.
- The average number of vehicles in India is growing at the rate of 10.16% annually, over the last few years. Spending hours in traffic jam has become part and parcel of metropolitan life style.
- The traditional method for traffic control uses fixed traffic poles on the left side of the road at the traffic junctions that display the traffic light.
- The steady increase in the number of automobiles on the road has amplified the importance of managing traffic flow efficiently to optimize utilization of existing road capacity.
- Presently major cities are covered under the surveillance of CCTV camera so that the vehicles and humans are tracked.

3.1.2 PROPOSED SYSTEM:

- Proposed system is deployed in Python environment.
- System consist of 24 different vehicles as labels and is classified in accordance with the data set provided.
- The efficiency and accuracy of the model created is around 95% out of 100%.

CHAPTER 4

EXPERIMENTAL MATERIALS AND METHODS USED

4.1 OVERVIEW:

Modules:

- 1. Dataset Collection**
- 2. Creating a Shallow CNN**
- 3. Application Deployment**

1. Dataset Collection:

Curating the dataset for our need is a tough job. So by downloading a dataset (Stanford car dataset) we are going to process the dataset to maintain class balance and all the necessary processing will be handled to make sure there are no negative classes situated.

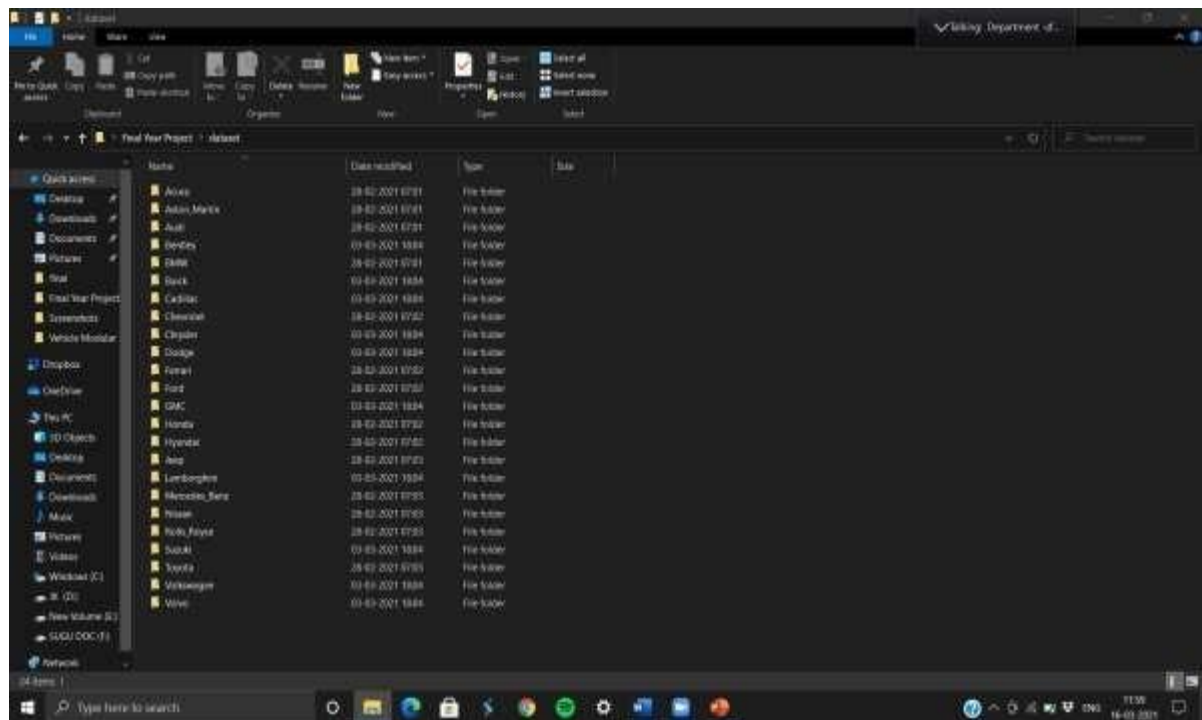


Fig: 4.1 Data Set (Car Model)

2. Creating Classifier using Shallow CNN:

By using Shallow CNN architecture, we will be training the feature maps so effectively so that the dense layer will do the necessary learnings to classify the car. The classifier consist of 4 x 4 Layers of CNN network. In each layer the image is cropped in a matrix for and the respective features are classified and trained as a data model.

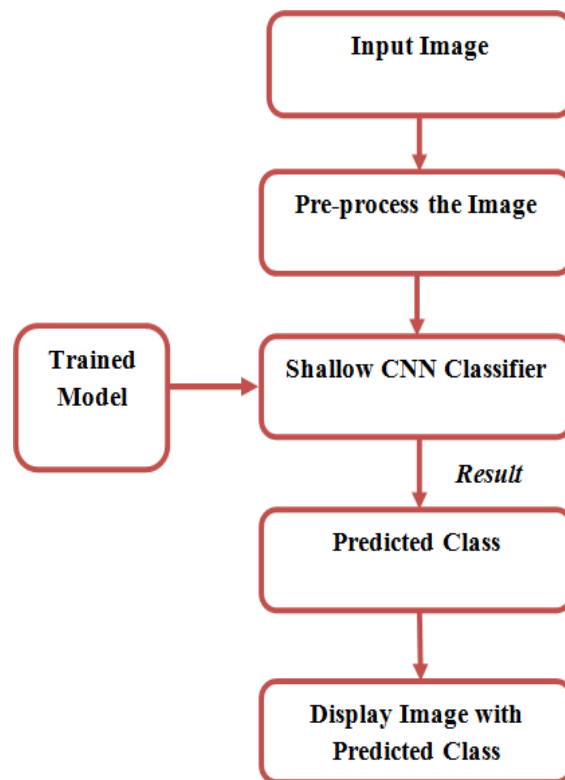


Fig: 4.2Flowchart for model

3. Application deployment

Once the training is done, the model will be dumped in a application.py file so that the model can be loaded anytime and once the input query image is given, the image will be preprocessed and will be passed to the classifier to say the label for the particular car present in the image.

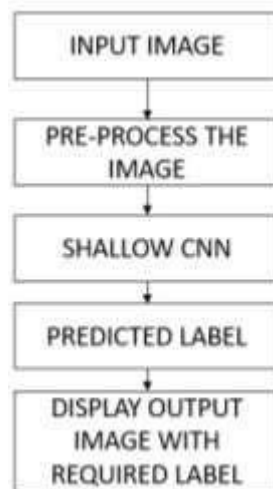


Fig:4.3 Flow Chart for Application

4.2 SYSTEM REQUIREMENTS:

4.2.1 SOFTWARE REQUIREMENTS:

- Operating system : Windows 7 / Windows 10 – 32Bit
- Language : Python 3.5
- Front end : asp.net
- Back end : Python anaconda script

4.2.2 HARDWARE REQUIREMENTS:

Computer Configuration:

- Processor : > 4 GHz
- Hard Disk : > 500GB
- RAM : > 8GB
- Keyboard : Standard Keyboard
- Monitor : 15 inch VGA colour

4.3 SYSTEM ARCHITECTURE:

4.3.1 ARCHITECTURE DIAGRAM:

ARCHITECTURE DIAGRAM

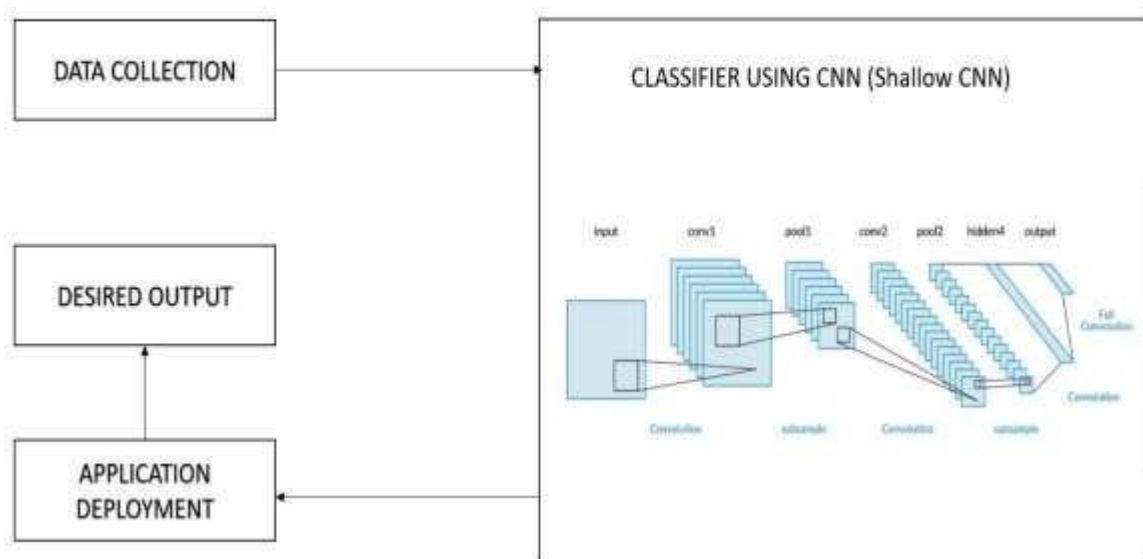


FIG:4.4 Architecture Diagram of Application

4.4 SYSTEM DESIGN

4.4.1 UML DIAGRAM:

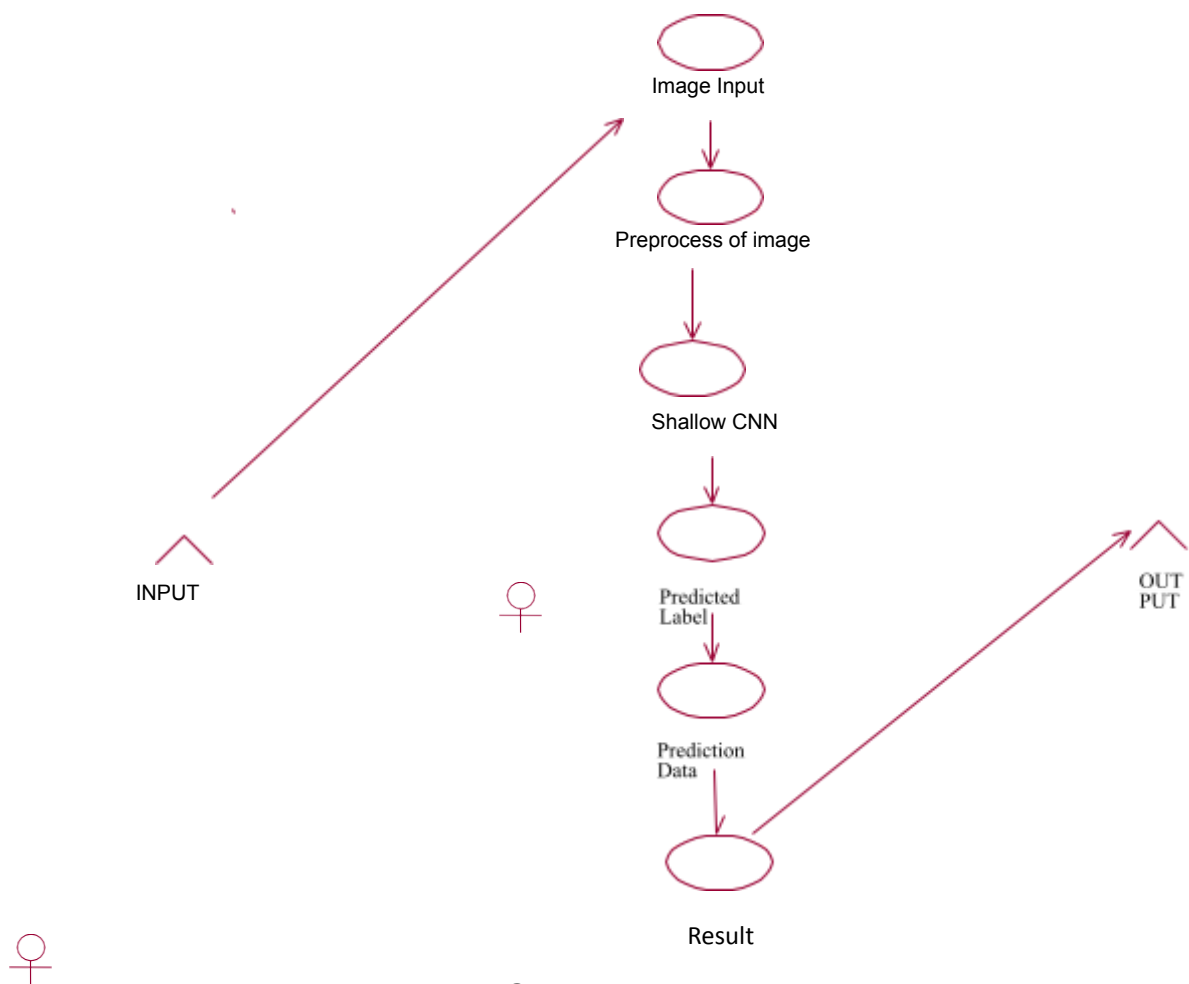


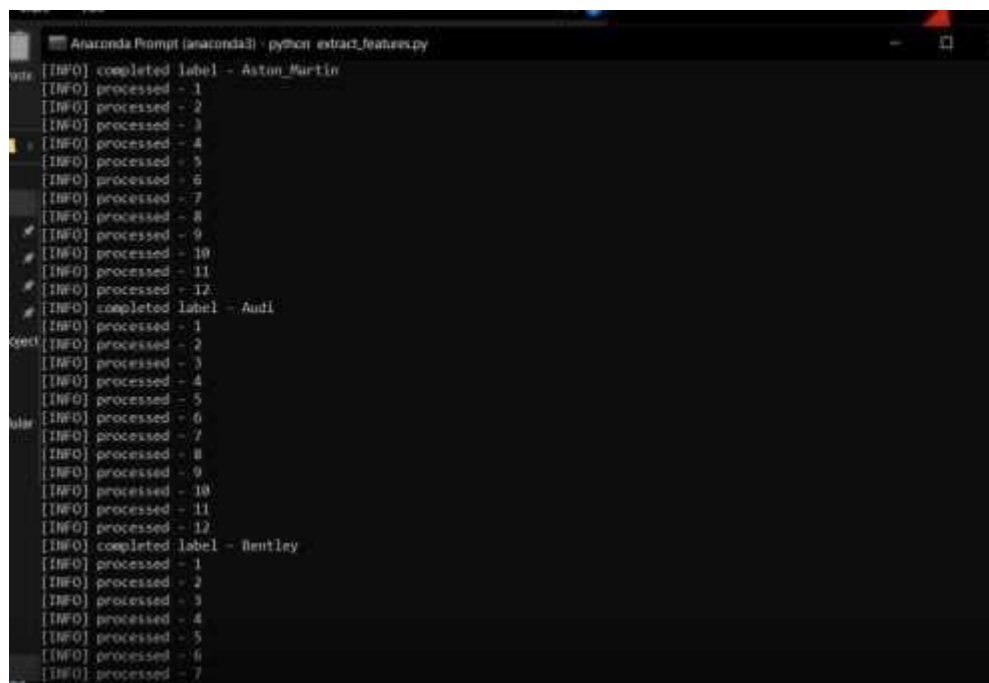
FIG 4.5 UML Diagram

CHAPTER 5

5.RESULTS AND DISCUSSION

5.1 SCREENSHOTS:

SHOT 1: FEATURE EXTRACTION



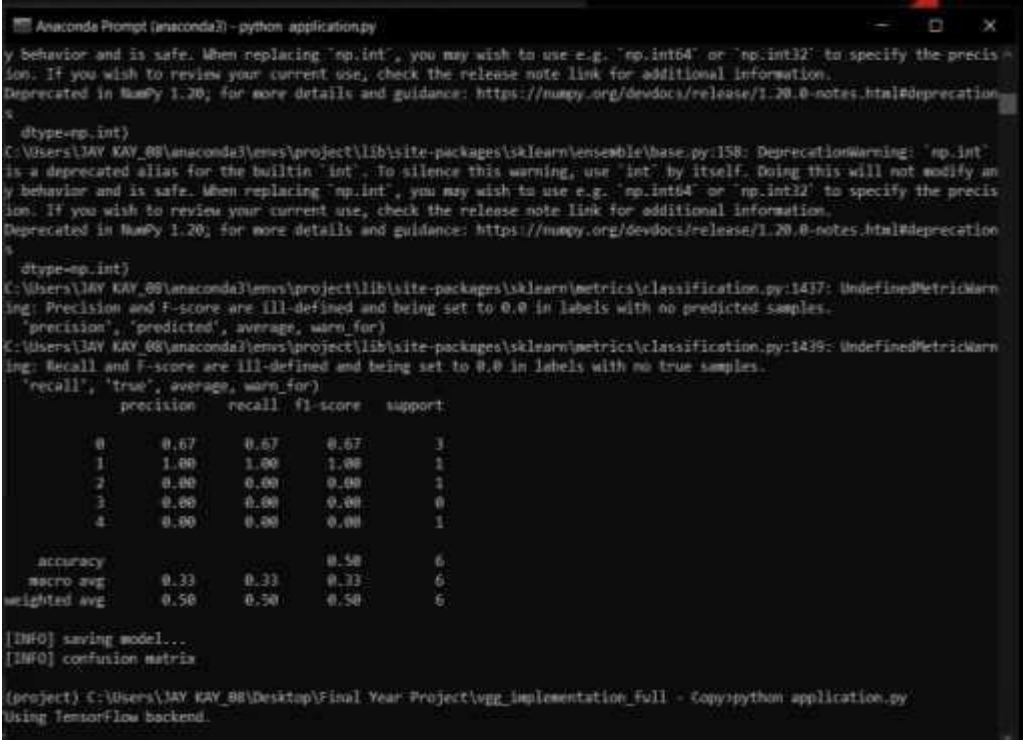
```
Anaconda Prompt (anaconda3) - python extract_features.py
[INFO] completed label - Aston Martin
[INFO] processed - 1
[INFO] processed - 2
[INFO] processed - 3
[INFO] processed - 4
[INFO] processed - 5
[INFO] processed - 6
[INFO] processed - 7
[INFO] processed - 8
[INFO] processed - 9
[INFO] processed - 10
[INFO] processed - 11
[INFO] processed - 12
[INFO] completed label - Audi
[INFO] processed - 1
[INFO] processed - 2
[INFO] processed - 3
[INFO] processed - 4
[INFO] processed - 5
[INFO] processed - 6
[INFO] processed - 7
[INFO] processed - 8
[INFO] processed - 9
[INFO] processed - 10
[INFO] processed - 11
[INFO] processed - 12
[INFO] completed label - Bentley
[INFO] processed - 1
[INFO] processed - 2
[INFO] processed - 3
[INFO] processed - 4
[INFO] processed - 5
[INFO] processed - 6
[INFO] processed - 7
```

Fig: 5.1 Feature extraction from given data set

The above Fig:5.1 shows the feature extraction process . Here, the process is carried out

in anaconda environment.

SHOT 2: CLASSIFIER MODEL EXECUTION



```
Anaconda Prompt (anaconda3) - python application.py
y Behavior and is safe. When replacing 'np.int', you may wish to use e.g. 'np.int64' or 'np.int32' to specify the precision. If you wish to review your current use, check the release note link for additional information.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecation
dtype=np.int)
C:\Users\JAY KAY_08\anaconda3\envs\project\lib\site-packages\sklearn\ensemble\base.py:158: DeprecationWarning: 'np.int'
is a deprecated alias for the builtin 'int'. To silence this warning, use 'int' by itself. Doing this will not modify any
behavior and is safe. When replacing 'np.int', you may wish to use e.g. 'np.int64' or 'np.int32' to specify the precision. If you wish to review your current use, check the release note link for additional information.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecation
dtype=np.int)
C:\Users\JAY KAY_08\anaconda3\envs\project\lib\site-packages\sklearn\metrics\classification.py:1437: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples.
'precision', 'predicted', average, warn_for)
C:\Users\JAY KAY_08\anaconda3\envs\project\lib\site-packages\sklearn\metrics\classification.py:1439: UndefinedMetricWarning: Recall and F-score are ill-defined and being set to 0.0 in labels with no true samples.
'recall', 'true', average, warn_for)
precision    recall  f1-score   support

0           0.67       0.67       0.67         3
1           1.00       1.00       1.00         1
2           0.00       0.00       0.00         1
3           0.00       0.00       0.00         0
4           0.00       0.00       0.00         1

accuracy          0.50         6
macro avg       0.33       0.33       0.33         6
weighted avg    0.50       0.50       0.50         6

[INFO] saving model...
[INFO] confusion matrix

(project) C:\Users\JAY KAY_08\Desktop\Final Year Project\vgg_implementation_Full - Copy>python application.py
Using TensorFlow backend.
```

Fig: 5.2 Classifier Model Execution

The above Fig: 5.2 shows the classifier model execution. Here, the screenshot shows the execution of classifier model(shallow CNN).

SHOT 3: CLASSIFIED RESULT(ASTON_MARTIN)



Fig: 5.3 Classified result(Aston_Martin)

The above Fig:5.3 shows the Expected output of classifying the car model(label) based on the data set given (Aston_Martin).

SHOT 4: CLASSIFIED RESULT(BMW)

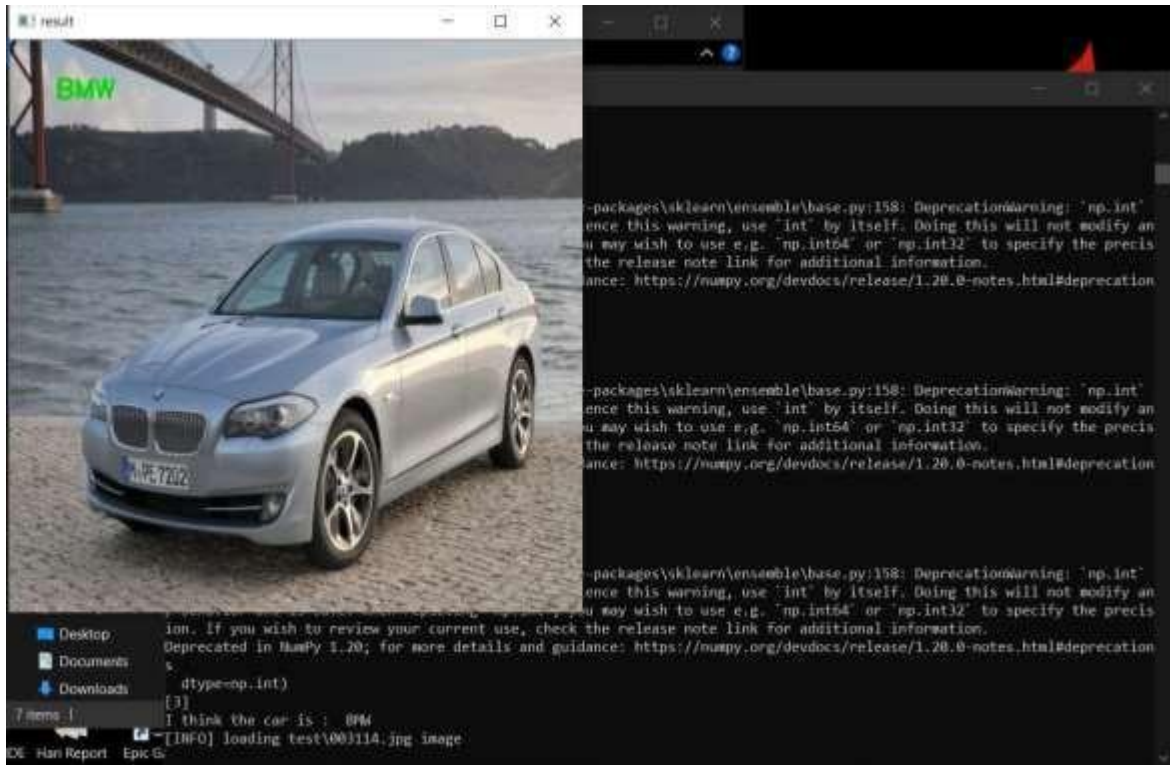


Fig:5.4Classified result(BMW)

The above Fig: 5.4 shows the Expected output of classifying the car model(label) based on the data set given (BMW).

CHAPTER 6 6.SUMMARY

AND CONCLUSIONS

6.1 CONCLUSIONS:

We presented a review of traffic monitoring systems focusing on the key functionality of vehicle classification. By categorizing the vehicle classification systems according to how sensors are installed into three types, i.e., in-roadway, overroadway, and side-roadway based systems, we discussed various research issues, methodologies, hardware design, and limitations. We also discussed a number of research challenges and future research directions. We expect that the rich contents about virtually all vehicle classification systems developed in the past decade will be useful resources for academia, industry, and government agencies in selecting appropriate vehicle classification solutions for their traffic monitoring applications.

6.2 FUTURE SCOPES:

Further enhancements can be done to the prototype by testing it with longer range RFID readers. Also GPS can be placed into the stolen vehicle detection module, so that the exact location of stolen vehicle is known. Currently, we have implemented system by considering one road of the traffic junction. It can be improved by extending to all the roads in a multi-road junction. Though it is well modulated system, it has been limited to certain restrictions. By understanding trends in technology we can sometimes make accurate predictions about what will happen in the future. This Project can be enhanced by Adding the GPS Technology to the existing system. Designated Check Posts can be equipped with cameras, and static scanners for more

reliability and security. Speed violations can also be tracked by the integration of the Interceptor to the existing system.

Unit tracking ability with Google Maps, both road and satellite. Detailed replay of all activities in graphical and tabular reports.

REFERENCES

- [1] M. Won, T. Park, and S. H. Son, "Toward mitigating phantom jam using vehicle-to-vehicle communication," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 5, pp. 1313–1324, 2016.
- [2] FHWA, "The 2016 traffic monitoring guide," https://www.fhwa.dot.gov/policyinformation/tmguidetmg_fhwa_pl_17_003.pdf, accessed: 2019- 04-12.
- [3] M. Won, S. Sahu, and K.-J. Park, "DeepWiTraffic: Low cost WiFi-based traffic monitoring system using deep learning," *arXiv preprint*, 2018.
- [4] H. Lee and B. Coifman, "Using LIDAR to validate the performance of vehicle classification stations," *Journal of Intelligent Transportation Systems*, vol. 19, no. 4, pp. 355–369, 2015.
- [5] FHWA, "The 2016 traffic detector handbook," <https://www.fhwa.dot.gov/publications/research/operations/its/06108/06108.pdf>, accessed: 2019- 04-12.
- [6] R. Tyburski, "A review of road sensor technology for monitoring vehicle traffic," *Institute of Transportation Engineers Journal*, vol. 59, no. 8, 1988. [7] S. R. E. Datondji, Y. Dupuis, P. Subirats, and P. Vasseur, "A survey of vision-based traffic monitoring of road intersections," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 10, pp. 2681–2698, 2016.
- [8] B. Tian, Q. Yao, Y. Gu, K. Wang, and Y. Li, "Video processing techniques for traffic flow monitoring: A survey," in *International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2011, pp. 1103–1108.
- [9] R. M. Inigo, "Traffic monitoring and control using machine vision: a survey," *IEEE Transactions on Industrial Electronics*, no. 3, pp. 177–185, 1985.

[10] K. Yousaf, A. Iftikhar, and A. Javed, "Comparative analysis of automatic vehicle classification techniques: a survey," *International Journal of Image, Graphics and Signal Processing*, vol. 4, no. 9, p. 52, 2012.

- [11] S. Kul, S. Eken, and A. Sayar, "A concise review on vehicle detection and classification," in International Conference on Engineering and Technology (ICET), 2017, pp. 1–4.
- [12] N. K. Jain, R. Saini, and P. Mittal, "A review on traffic monitoring system techniques," in Soft Computing: Theories and Applications. Springer, 2019, pp. 569–577.
- [13] A. Puri, "A survey of unmanned aerial vehicles (UAV) for traffic surveillance," Department of computer science and engineering, University of South Florida, pp. 1–29, 2005.
- [14] K. Kanistras, G. Martins, M. J. Rutherford, and K. P. Valavanis, "Survey of unmanned aerial vehicles (UAVs) for traffic monitoring," Handbook of unmanned aerial vehicles, pp. 2643–2666, 2015.
- [15] M. Bernas, B. Płaczek, W. Korski, P. Loska, J. Smyła, and P. Szymała, "A survey and comparison of low-cost sensing technologies for road traffic monitoring," Sensors, vol. 18, no. 10, p. 3243, 2018.

APPENDIX

A.SOURCE CODE:

Application File(Application.py)

```
from keras.models import Sequential
from keras.layers import Conv2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense
import matplotlib.image as mpimg
import numpy as np
import matplotlib.pyplot as plt
import cv2

from keras.preprocessing import image
from keras.preprocessing.image import ImageDataGenerator
from keras.models import model_from_json
import h5py
from imutils import paths
import imutils

classnames=['Acura','Aston_Martin','Audi','Bentley','BMW','Buick','Cadillac','
Chevrolet','Chrysler','Dodge','Ferrari','Ford','GMC','Honda','Hyundai','Jeep','L
amborghini','Mercedes_Benz','Nissan','Rolls_Royce','Suzuki','Toyota','Volks
wagen','Volvo']

# load json and create model json_file =
```

```
open('model2.json', 'r')  
loaded_model_json = json_file.read()  
json_file.close()  
loaded_model = model_from_json(loaded_model_json)
```

```

# load weights into new model
loaded_model.load_weights("model2.h5") print("Loaded
model from disk")

image_paths = 'Test'

for path in sorted(list(paths.list_images(image_paths))):
    img_pred = image.load_img(path, target_size = (100, 100))
    img_pred = image.img_to_array(img_pred)
    img_pred = np.expand_dims(img_pred, axis = 0)
    rslt = loaded_model.predict(img_pred)
    #print (path)
    A = np.array(rslt[0])
    maximum_indices = np.where(A==max(rslt[0]))
    #print (maximum_indices[0])
    prediction = str(classnames[int(maximum_indices[0][0]]))
    print (prediction)
    image_output = cv2.imread(path)
    image_output = imutils.resize(image_output, width=800)
    cv2.putText(image_output, prediction, (10, 25),
cv2.FONT_HERSHEY_SIMPLEX,1, (255,255,0), 2)
    cv2.imshow("result",image_output)
cv2.waitKey(0)

```

Model.py

```
from keras.models import Sequential
from keras.layers import Conv2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense
import matplotlib.image as mpimg
import numpy as np
import matplotlib.pyplot as plt
import cv2
from keras.preprocessing import image
from keras.preprocessing.image import ImageDataGenerator
from keras.models import model_from_json
import h5py
import matplotlib.pyplot as plt

print ("[started]")
classnames=['Acura','Aston_Martin','Audi','Bentley','BMW','Buick','Cadillac','
Chevrolet','Chrysler','Dodge','Ferrari','Ford','GMC','Honda','Hyundai','Jeep','L
amborghini','Mercedes_Benz','Nissan','Rolls_Royce','Suzuki','Toyota','Volks
wagen','Volvo']
# Initialising the CNN
classifier = Sequential()

# Convolution
classifier.add(Conv2D(32, (5, 5), input_shape = (100, 100, 3), activation = 'relu'))
```

```
# Pooling classifier.add(MaxPooling2D(pool_size =  
(2, 2)))
```

```
# Adding a second convolutional layer
```

```
classifier.add(Conv2D(32, (3, 3), activation = 'relu'))
```

```
classifier.add(MaxPooling2D(pool_size = (2, 2)))
```

```
# Flattening
```

```
classifier.add(Flatten())
```

```
# Full connection
```

```
classifier.add(Dense(units = 128, activation = 'relu'))
```

```
classifier.add(Dense(units = 24, activation = 'softmax'))
```

```
# Compiling the CNN
```

```
classifier.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics =  
['accuracy'])
```

```
# Fitting the CNN to the images
```

```
train_datagen = ImageDataGenerator(rescale = 1./255,  
                                   shear_range = 0.2,  
                                   zoom_range = 0.2, horizontal_flip  
                                   = True)
```

```
#test_datagen = ImageDataGenerator(rescale = 1./255)
```

```
training_set = train_datagen.flow_from_directory('dataset',
```

```
target_size = (100, 100),  
batch_size = 32,  
class_mode =  
'categorical', classes=classnames)
```

```
history = classifier.fit_generator(training_set,
                                  steps_per_epoch = 200,
                                  epochs = 500)

# serialize model to JSON
model_json = classifier.to_json()
with open("model2.json", "w") as json_file:
    json_file.write(model_json)
# serialize weights to HDF5
classifier.save_weights("model2.h5")
print("Saved model to disk")

print(history.history.keys())

# summarize history for accuracy
plt.plot(history.history['acc'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.show()
```

Feature extraction:

```
# filter warnings
import warnings
warnings.simplefilter(action="ignore", category=FutureWarning)

# keras imports
from keras.applications.vgg16 import VGG16, preprocess_input
from keras.preprocessing import image
from keras.models import Model
from keras.models import model_from_json
from keras.layers import Input

# other imports
from sklearn.preprocessing import LabelEncoder
import numpy as np
import glob
import cv2
import h5py
import os
import json
import datetime
import time

# load the user configs
with open('conf/conf.json') as f:
    config = json.load(f)
```

```
# config variables
```

```
model_name          = config["model"]
```

```
weights             = config["weights"]
```

```

include_top = config["include_top"] train_path
                = config["train_path"]
features_path = config["features_path"]
labels_path = config["labels_path"] test_size
                = config["test_size"]
results = config["results"]
model_path = config["model_path"]

# start time
print ("[STATUS] start time -
{}".format(datetime.datetime.now().strftime("%Y-%m-%d %H:%M")))
start = time.time()

# create the pretrained models
# check for pretrained weight usage or not #
check for top layers to be included or not

base_model = VGG16(weights=weights)
model = Model(input=base_model.input,
output=base_model.get_layer('fc1').output)
image_size = (224, 224)
print ("[INFO] successfully loaded base model and model...") #
path to training dataset
train_labels = os.listdir(train_path)

# encode the labels
print ("[INFO] encoding labels...") le

```

```
= LabelEncoder()
```

```
le.fit([tl for tl in train_labels])
```

```
# variables to hold features and labels
```

```
features = []
```

```
labels = []
```

```
# loop over all the labels in the folder
```

```
count = 1
```

```
for i, label in enumerate(train_labels):
```

```
    cur_path = train_path + "/" + label
```

```
    count = 1
```

```
    for image_path in glob.glob(cur_path + "/*.jpg"):
```

```
        img = image.load_img(image_path, target_size=image_size) x
```

```
        = image.img_to_array(img)
```

```
        x = np.expand_dims(x, axis=0) x
```

```
        = preprocess_input(x) feature =
```

```
        model.predict(x)
```

```
        flat = feature.flatten()
```

```
        features.append(flat)
```

```
        labels.append(label)
```

```
        print ("[INFO] processed - " + str(count))
```

```
        count += 1
```

```
    print ("[INFO] completed label - " + label)
```

```
# encode the labels using LabelEncoder le
```

```
= LabelEncoder()
```

```
le_labels = le.fit_transform(labels)
```

```
# get the shape of training labels
```

```
print ("[STATUS] training labels: {}".format(le_labels))
```

```
print ("[STATUS] training labels shape: {}".format(le_labels.shape))

# save features and labels
h5f_data = h5py.File(features_path, 'w')
h5f_data.create_dataset('dataset_1', data=np.array(features))

h5f_label = h5py.File(labels_path, 'w')
h5f_label.create_dataset('dataset_1', data=np.array(le_labels))

h5f_data.close()
h5f_label.close()

# save model and weights
model_json = model.to_json()
with open(model_path + str(test_size) + ".json", "w") as json_file:
    json_file.write(model_json)

# save weights
model.save_weights(model_path + str(test_size) + ".h5")
print("[STATUS] saved model and weights to disk..")

print ("[STATUS] features and labels saved..")

# end time
end = time.time()
print ("[STATUS] end time - {}".format(datetime.datetime.now().strftime("%Y-%m-
```

%d %H:%M"))))

B.PLAGRISM REPORT:

paper presentation.docx

ORIGINALITY REPORT

16%

SIMILARITY INDEX

4%

INTERNET SOURCES

14%

PUBLICATIONS

2%

STUDENT PAPERS

PRIMARY SOURCES



Watcharin Maungmai, Chaiwat Nuthong.
"Vehicle Classification with Deep Learning",
2019 IEEE 4th International Conference on
Computer and Communication Systems
(ICCCS), 2019

Publication

14%



carconsumption.com

Internet Source

2%

Exclude quotes On

Exclude matches Off

Exclude bibliography On

C. PAPER:

VEHICLE MODEL CLASSIFICATION USING DEEP LEARNING

Jayakumar S,

School of Computing,

Sathyabama Institute of Science and Technology,

Chennai, India

Yadhu Nandha Krishna K,

School of Computing,

Sathyabama Institute of Science and Technology,

Chennai , India

Dr.Ajitha.P

Associate Professor,

School of Computing,

Sathyabama Institute of Science and Technology,

Chennai, India

ABSTRACT:

One of the most significant issues in modern road safety and intelligent transportation systems is the automation of vehicle detection and identification. Many challenges have been solved in the advancement of image processing, pattern recognition, and deep learning technology in order to accomplish this goal. Vehicle Type Classification is a difficult task since the dataset has a large class imbalance, and several view-points for different cars can be identical. The proposed framework employs a shallow Convolutional Neural Networks (CNN) architecture to prevent overfitting and ensure that the correct features are learned, and we use an augmentation technique to produce synthetic images using the image data generation model in Keras due to class imbalance. The shallow CNN is used to extract features from the generated images, and then Softmax activation is used to classify them. Finally, the proposed system will achieve the classification of vehicle type i.e. classify the different car models with efficiently by novel methodology. The findings of the experiments

demonstrate that shallow CNN can do well in real-world situations.

Keywords: Deep Learning, Shallow Convolutional Neural Networks (CNN), Car Models Classification, Vehicle Classification, Image Augmentation.

1. INTRODUCTION

Surveillance cameras can be seen nearly anywhere in cities. The key goals of security systems installation are real-time tracking and incident searching. The authors of this paper concentrate solely on activity finding. Police officers may use the camera technology to accomplish their search goal i.e. to locate a particular car. In general, officers need details about the vehicle's features, like the color and model of the vehicle, as an initial clue for vehicle identification. Officers also waste a significant amount of time alone watching captured recordings. Typically, the time spent searching exceeds the length of the video, and they must repeat the search many times.

In order to solve above problems, there have been numerous vehicle classification systems

built. Recent advancements in sensing, machine learning, and wireless networking technology, in particular, have spawned a slew of novel vehicle classification systems.

Various machine learning models are presented in recently for vehicle type classification. For instance, [1] built a system that can monitor vehicles in surveillance videos. They suggested Motionlet as a new classifier. The classifier is an Adaboost learning-based detector [2]. Motionlet's key objective was to categorize twelve separate vehicle directions. As a result, they were able to reach an accuracy rate of 87 percent.

In [3], artificial neural network was used as a classifier to divide vehicles into two different categories: large and small. The accuracy of their analyses was greater than 90%.

In [4], contrasted the classification efficiency of different classifiers, such as regression tree, bagging, and random forest. In their trial, both approaches produced findings that were comparable. Random tree, on the other hand, was selected for two reasons: it took less time to learn than other techniques and had no instruction.

In recent times, a new method known as Deep Learning [7] has emerged that can also be used in classification tasks. A neural network having more than two hidden layers is known as deep learning. The following are some of the stages of neural network evolution:

1. There are more neurons in this network than in previous networks.
2. In neural networks, there are more complex ways to link layers/neurons.
3. A significant increase in the amount of computing resources required for training.

4. Feature extraction that is automated.

Unsupervised and pretrained networks, convolution neural networks (CNN), recurrent neural networks, recursive neural networks, and other forms of deep learning exist. Convolutional neural networks (CNNs) are the most widely used deep learning neural network. Convolution is the key function of these networks, which is programmed to learn rich features from the data set. The networks have been well to image object recognition and it regularly rank first in image classification competitions.

This paper presents the vehicle type classification (car models) using novel methodology of deep learning shallow convolutional neural networks. By using shallow CNN, overfitting problem will be overcome and class imbalance problem solve by image augmentation.

II. RELATED WORKS

Several studies [9]-[11] have used CNN as classifier in vehicle color recognition. Chen et al. [12] suggested using feature background to identify the vehicle's colour. They used their dataset, which included 15,601 vehicle images divided into eight color groups, to perform the classification. They were able to classify with a precision of 90.68 percent.

The authors used CNN with Chen's dataset [12] in [9], [11]. Their accuracy rates were 94.47 percent and 94.6 percent, respectively. In [10], they suggested Colomnet, a novel CNN structure that achieved 95.74 percent accuracy in their vehicle color classification trial. Alexnet [8] and GoogleNet [13] were outperformed by the framework.

Zhou et al. [14] suggested deep neural network or deep learning methods for vehicle recognition and classification. They used the YOLO [15] detector paradigm for detection. As classification approaches, Alexnet [8] was used. There are four types of classification with respect to classification modules: passenger vs other, cars vs vans, sedans vs taxis, and sedans vs vans vs taxis. Following the application of both structures, they were fine-tuned to work with the public dataset given in [16]. The accuracy of the experimental findings was greater than 90%.

The proposed method's key goal in this paper is to increase the accuracy of the previously described vehicle model and vehicle classification. In this study, a convolutional neural network with two convolution layers was chosen as the classifier. The CNN was selected due to its image recognition capabilities. The vehicle images from the dataset

Table 1: Output Classes in Car Model Classification

No of Classes	Output Classes
1	Acura
2	Aston Martin
3	Audi
4	Bentley
5	BMW
6	Buick
7	Cadillac
8	Chevrolet
9	Chrysler
10	Dodge
11	Ferrari
12	Ford
13	GMC
14	Honda
15	Hyundai
16	Jeep
17	Lamborghini

18	Mercedes Benz
19	Nissan
20	Rolls Royce
21	Suzuki
22	Toyota
23	Volkswagen
24	Volvo

are fed into shallow CNN. In the following part, we'll go over it in more depth.

III. PROPOSED METHODOLOGY

This system consists of shallow CNN structures that are used as classifiers in car model classification. The proposed method uses an input image, which is a car image, fed into the system. The classifications' output is identical to those listed in Table 1. The proposed system mainly consists of two process one is image augmentation technique which is used to expand the dataset by generating of new images which is implemented by image data generator model in keras; another one is shallow CNN for extract the rich features for accurate classification which done by following layers such as,

- Convolution layer with 5x5 Kernal
- Max pooling layer
- Convolution layer with 3x3 Kernal
- Max pooling layer
- Fully connected layer
- SoftMax classification



Fig.1 Training Process of Proposed Method

A. TRAINING PROCESS

In training process (Fig.1), first load the dataset then do the preprocessing which includes the image resizing process. Finally preprocessed dataset images are trained in the form of feature maps based on shallow CNN.

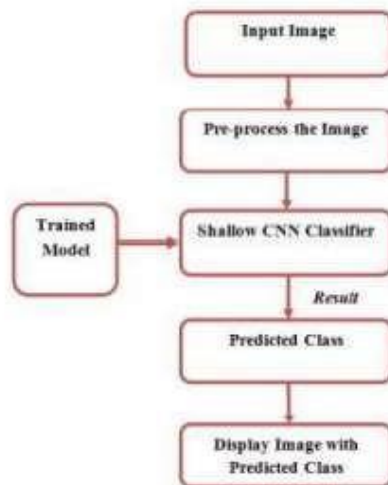


Fig.2 Testing Process of Proposed Method

B. TESTING PROCESS

Once the training is done, the model will be loaded and then input query image is given, the image will be preprocessed and will be passed to the classifier to predict the class for the particular car present in the image. Testing process is illustrated in Fig.2.

C. MODELS

The proposed system achieves the high accuracy by solving of two problems such as overfitting and class imbalance problems. Our light weight model of shallow CNN is used for overcome the overfitting problem and image augmentation technique is used for rectify the class imbalance problem.

a. Image Augmentation

Image augmentation is an effective technique for artificially creating differences in original images in order to extend the data set of an individual image. This generates new and exclusive images from an established image data set that contains a wide range of potential images. This is accomplished by using various transformation techniques such as zooming an actual image, rotating it by a few degrees, shearing or cropping an existing group of images, and so on. This is implemented by image data generator model in Keras. Deep learning Convolutional Neural Networks (CNN) require a large number of images to be efficiently equipped. This improves the model's efficiency by allowing it to generalize further and thereby reduce overfitting. Image augmentation used as a pre-processing phase before training the model or in real time.

Some Techniques for Image Augmentation

- Rotation: rotates the image by a given degree.

- Flipping: flips the image vertically or horizontally.
- Shearing: like a parallelogram, it moves one part of the graphic.
- Cropping: objects appear in various places in the scene in different proportions.
- Zooming in and out.
- Adjusting contrast

b. Shallow CNN

Convolutional Neural Networks are a form of feed forward artificial neural network that is somewhat identical to traditional neural networks. The weights and biases of the neurons in the network can be learned. Every neuron receives information and executes certain tasks. The convolution layer, pooling layer, and fully connected layer are the three main layers in CNN. The contribution of neurons bound to local regions within the input given will be calculated by the convolution layer, which computes the dot product between the weights and biases. To minimize the scale of the feature maps, a pooling layer is used. As a result, the parameter will be reduced, and the computation time will be reduced. In general, CNN employs max pooling. Each neuron in the fully

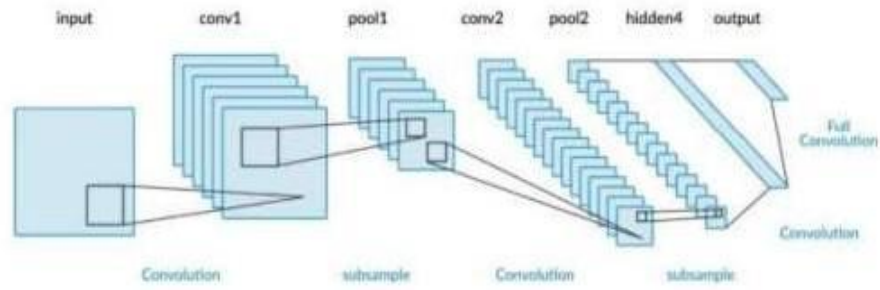


Fig.3 Shallow CNN Architecture

connected layer is linked to neurons in the previous CNN layer. The layers are fully connected, much as in a traditional neural network.

The convolutional and max-pooling layers derive high-level features from images, and the input images are classified using a fully connected layer. The network was trained using a cross-entropy cost function and the stochastic gradient descent algorithm.

IV. EXPERIMENTAL RESULTS

The proposed system uses the approach of deep learning shallow CNN model to classify the 24 car models which are given in Table 1. The proposed system uses the Stanford Car Dataset for car model classification. Sample images of car dataset are shown in Fig.4. The dataset is categorized into two parts: training and testing, which accounts for 75% and 25% of the total dataset, respectively.

Table 2: Accuracy Analysis

Models	Accuracy (%)
Decision Tree [5]	79.380
Random Forest [6]	79.821

Shallow CNN **93.341**



Fig.4 Dataset Sample Images

Table 2 is shown the accuracy analysis with different models. In that, our proposed method shallow CNN based car model classification achieve the accuracy is 93.341 % and other models are DT is 79.380 % and RF 79.821 % of accuracy.



Fig.5 Car Models Classification Results by Proposed Method

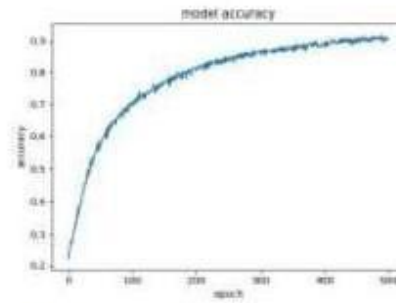


Fig.6 Accuracy Analysis by Proposed Method

V. CONCLUSION

The light weight shallow CNN model is proposed as a type of CNN and classifiers to classify car models in this paper. The proposed method effectively overcomes the two problems such as overfitting and class imbalance. Using of image data generator in Keras, image augmentation would implemented which used to expanding dataset to rectify the class imbalance problem; and overfitting solved by shallow CNN. We conducted several tests on our dataset, with the findings demonstrating that our classification scheme has a high accuracy rate.

REFERENCES

- [1] R. Feris, B. Siddiquic, Y. Zhai, J. Petterson, L. Brown, and S. Pankanti, "Attribute-based vehicle search in crowded surveillance videos," In Proceedings of the 1st ACM International Conference on Multimedia Retrival(ICMR '11), New York, 2011.
- [2] P. Viola and M. Jones, "Robust Real-time Object Detection," International Journal of Computer Vision, 2001.
- [3] S.D. Changalascity, A.S. Badawy, L.S. Thota, and W. Ghribi, "Moving vehicles classification in

- WEKA," 2015 International Conference on Advanced Computing and Communication Systems, Coimbatore, 2015, pp. 1-6 DOI: 10.1109/ICACCS.2015.7324081
- [4] P.O. Gislason, J.A. Benediktsson, and J.R. Sveinsson, "Random forests for land cover classification," *Pattern Recognition Letters*, vol. 27, no. 4, pp. 294300, Mar. 2006
- [5] K. Saripan, C. Deachakul, and W. Moungrmai, "Smart vehicle search in surveillance video," 2015 Information Technology Conference, Chongqing, 2015
- [6] K. Saripan and C. Nuthong, "Tree-based vehicle classification system," 2017 14th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), Phuket, 2017, pp. 439-442.
- [7] Patterson, J. and Gibson, A. (2017). *Deep learning: a practitioner's approach*. Sebastopol: O'Reilly Media.
- [8] A. Krizhevsky, I. Sutskever, G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks." In *Advances in neural information processing systems*, 2012, pp. 1097-1105.
- [9] R. F. Rachmadi, I. K. E. Purnama, M. H. Purnomo, "Vehicle Color Recognition using Convolutional Neural Network." *arXiv preprint arXiv:1510.07391*
- [10] B. Su, J. Shao, J. Zhou, X. Zhang and L. Mei, "Vehicle Color Recognition in The Surveillance with Deep Convolutional Neural Networks," 2015 Joint International Mechanical, Electronic and Information Technology Conference, Chongqing, 2015
- [11] C. Hu, X. Bai, L. Qi, P. Chen, G. Xue, and L. Mei, "Vehicle Color Recognition With Spatial Pyramid Deep Learning.", *IEEE Transactions on Intelligent Transportation Systems*, 2015
- [12] P. Chen, X. Bai, and W. Liu, "Vehicle Color Recognition on Urban Road by Feature Context.", *IEEE Transactions on Intelligent Transportation Systems*, 2014
- [13] C. Szegedy, W. Liu, Y. Jia, and A. Rabinovich, "Going deeper with convolutions.", In *arXiv:1409.4842*
- [14] Y. Zhou, H. Nejati, T. T. Do, N. M. Cheung and L. Cheah, "Image based vehicle analysis using deep neural network: A systematic study," 2016 IEEE International Conference on Digital Signal Processing (DSP), Beijing, 2016, pp. 276-280.
- [15] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *arXiv preprint arXiv:1506.02640*, 2015.
- [16] X. Ma and W.E.L. Grimson, "Edge-based rich representation for vehicle classification," in *Tenth IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2005, vol. 2, pp. 1185-1192
- [17] K. Ying, A. Ameri, A. Trivedi, D. Ravindra, D. Patel, and M. Mozumdar, "Decision tree-based machine learning algorithm for in-node vehicle classification," 2015 IEEE Green Energy and Systems Conference (IGESC), Long Beach, CA, 2015, pp.71-76. DOI:10.1109/IGESC.2015.7359454.