Automated Vehicle speed Estimation and License Plate Detection for Smart Cities Development

Divya Sharma
Department of Computer Applications
Manipal University Jaipur
India
divya.sharma2414@gmail.com

Shilpa Sharma
Department of Computer Applications
Manipal University Jaipur
India
Shilpa.sharma@jaipur.manipal.edu.in

Vaibhav Bhatnagar
Department of Computer Applications
Manipal University Jaipur
India
Vaibhav.bhatnagar15@gmail.com

2022 IEEE World Conference on Applied Intelligence and Computing (AIC) | 978-1-6654-7988-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/AIC55036.2022.9848890

***Abstract*— Currently, the vehicle count is increasing progressively, subsequently, are road crimes, and accident cases escalating. Even though the government of smart cities has imposed certain laws and traffic rules to reduce the number of road accidents and deaths, the younger generations are still doing rash driving. Therefore, there is an urgent need to implement an automated system to keep an eye on the speedy vehicles and take further actions for maintaining the development of smart cities. The major aim of the paper is to perform the practical implementation of the system using the available Pytesseract, Haar Cascade and dlib library. This model initially performs the detection of vehicles, then estimates the vehicle speed, and finally recognizes the license plates of the speedy vehicles. The paper provides a comparison using four different video datasets to analyze the performance of the implemented system. On the basis of the observations, the implemented model acquires a recall of 89.02% and a precision of 91.9%.**

***Keywords*—** Speed Estimation, License Plate Recognition, Tracking, openCV, Haar Cascade

1. INTRODUCTION

As smart cities are growing rapidly, the number of vehicles on the road is also increasing exponentially, which promotes the overall development of the nation [1]. Each and every vehicle is identified with the help of an identity number that is known as a "License Plate Number" [2]. France invented and first used the license plate in 1983 [3]. As the vehicles are tremendously increasing which in turn increases the need for the betterment of management of traffic system [4]. And with this rapid growth of vehicles, the resultant traffic mismanagement creates some serious problems like road accidents, criminal activities, etc. To improve and implement of smart development of cities, Intelligent Transport Systems (ITS) started focusing on smart management of traffic and various different applications like highway security surveillance, smart parking lots, smart toll plazas, and improvised traffic rules [5]. But regrettably, all these tasks are taking place manually by indulging manpower [6]. ITS prioritizes the task of license plate recognition using various computer vision and deep learning techniques. Thus, with the help of ALPR, one can achieve and acquire details of vehicles and also the person with whom the vehicle is registered. But the license plate recognition approach faces some issues while recognizing and detecting the license plates like difference in angle of license plates, recognizing multiple vehicles license plates at the same time, light variations, difference in plate fonts, difficulty in recognizing the characters if the plate is crack or smashed and if the speed of vehicle is very fast. The paper focuses on one of the most complex smart cities traffic issues: detecting and identifying speedy vehicles. Then, to identify the fast and speedy vehicles, the traditional manual system government using is not enough to control and manage the rash driving vehicles. And even the mobile speed detection van standing on roads would not be able to keep an eye on each and every vehicle. Therefore, various automated systems using different methods and approaches have been proposed to track and estimate the speed of vehicles, and finally, recognize the license plate characters of the detected vehicle. The license plate recognition basically comprises of three major task that is initially it detect the vehicles license plates from the inputted image or video then second task is to segment the characters of license plates and finally recognize the segmented characters. The paper initially showcases the research work done till date under section II. Section III lists the detailed datasets description used for comparative analysis. The complete description has been elaborated in section IV under Implemented System that highlights the working of the system and explains the system flow using the flow diagram in Figure 1. In section V, the results and analysis of the implemented system based on the 4 different datasets are showcased. Finally, section VI covers the conclusion of the paper.

2. LITERATURE REVIEW

Tracking the speedy vehicle is a basic need for smart traffic management to control and manage the traffic on roads. Various different approaches have been implemented by different researchers using computer vision and machine learning approaches. Deep Belief network can be used for detecting and tracking the speed of the running vehicles. The models use a combination of Restricted Boltzman Machine and Back propagation Neural Network for predicting the speed of a vehicle [7]. A new approach has evolved which introduces a paradigm which follows a detect

978-1-6654-7988-2/22/$31.00 ©2022 IEEE
DOI: 10.1109/aic.2022.63

then track approach to estimate the vehicle speeds and includes the optical flow-based data driven approach for estimating the speed of the vehicles [8]. The system estimated the vehicles' speed using the videos from the NVIDIA AI City Challenge 2018. Another approach to estimates the vehicle speed is introduced in [9] which uses the concept of regression algorithms and enables the important algorithms to estimate the speed, such as Support Vector Machine, Random Forest, Artificial Neural Network, and GBDT. The concept of CNN has been used for the past few years, which is a learning method of first order. License plate recognition using CNN is a complex task as it requires a large number of samples, so the accuracy is also relatively high [10]. While training the network, the model uses the sequence expansion approach to extend the training set size.
An innovative approach used the concept of Support Vector Machine and CNN for Chinese character recognition for license plate characters [11]. Various types of datasets available for performing the license plate recognition and detection purpose like AOLP (Application-oriented License

Table .1 Dataset Video Tracks

Track No.	Frame Per Seconds	Resolutions	Time Duration
1	40	1920*1080	15 Minutes
2	40	1920*1080	25 Minutes
3	40	1920*1080	35 Minutes
4	40	1920*1080	15 Minutes

Plate) [12].The concept of vehicle logo recognition and classification is used in the paper and states that the classification of the vehicle logo is covered under multi-integrated layers with the help of the data presented. Another approach uses the concept of Deep Hybrid Architecture to recognize the license plate in various complex weather conditions [13]. It uses different types of networks, like the basic non-complex network; another is the regression network; then the multi scale network; and finally, the classification network. The model initially uses the basic network for extracting the base features from the image. The multi scale network extracts the multi scale feature for occupying and gathering vehicle number plates of various sizes and colors. Then the network identifies the characters available in the image by employing the regression network. The model proposed is trained using the back propagation model in an end-to-end manner.
Recently, an innovative technique has emerged to detect the speed of vehicles using a frame difference approach that relies on the bounding of the adjoining framses in the moving video sequences. The method extracts the vehicle features with the help of three unique frame approaches and the background difference approach, then the model locates the position of the moving vehicles using the centroid feature methods, and lastly, the model uses the concept of relationship mapping between the actual and pixel distance [14]. The model first acquires the video sequence from the CCTV cameras, and the camera position is constant. This model used the concept of pixel-based time variation for the two or three adjoining frames and the mean of thresholding for removing the static objects and separates the moving vehicles in the video sequence [10].

$$T_{k-1}(x,y) = |frame_k(x,y) - frame_{k-1}(x,y)| \quad (1)$$

Here,

$$frame_k(x,y) = k_{th}\ frame\ of\ video$$

$$frame_{k-1}(x,y) = k-1\ frame\ of\ video$$

$$T_{k-1}(x,y) = difference\ of\ image\ vedio\ sequence$$

Numerical representation of thresholding for separating the video sequence to binary images

$$S_{k-1}(x,y) = f(x) = \begin{cases} 0, \wedge\ T_{k-1}(x,y) \geq M \\ 1, \wedge\ T_{k-1}(x,y) < M \end{cases} \quad (2)$$

Here,

$$S_{k-1}(x,y) = Binary\ Images$$

$$M = Threshold\ Value$$

Here the use of background difference approach is to remove the real time image sequences and extract the static background image.

3. Dataset Description

The datasets used for performing the vehicle speed detection and license plate recognition has been captured from stationary cameras on roads randomly.

The Tools and Algorithm used is described below:

1. Opencv: Opencv is the most valuable python library for performing various types of computer vision tasks including real time functions with images or videos. The opencv library is available to their users under BSD license [15]. The opencv library is written and developed under C programming language with excellent portability feature. One can use this library over different operating systems like Linux, mac OS, Windows so called as cross platform.
2. Dlib: Dlib is a C++ based library which can be used f.0eqeEor various different real time applications like tracking vehicles on roads, recognizing facial expressions or features, detecting landmarks etc. it contains a pipeline of HOG and SVM specifically for detection purpose [16]. Dlib helps to detect and track the vehicles in this implemented model.
3. Pytesseract: It is also called as python tesseract; it is a most popular optical Character Recognition tool available for python [17]. This is used for extracting

and reading the text or character from an image. Tesseract is an openly available text reader under Apache 2.0. it contains neural network system for text reader. Here, pytesseract is used for recognizing the characters from the license plate images.

4. Haar Cascade Classifier:[16] The Haar cascade classifier was initially applied for facial feature detection, it is basically optimal choice for object detection. This is one of the types of cascade-based classifier which contains tons of training images. Haar cascade classifier extract numerical numbers with the help of the available features. As working with opencv it contains pre trained feature set in the form of XML files. The model used haar cascade classifier for detecting the vehicles; the classifier is trained with positive and negative images.

4. Vehicle Speed Estimation and License Plate Detection Model

The model uses the most useful python computer vision libraries that are opencv, numpy, pytesseract, dlib, time and is sub divided broadly into two tasks. Initially the model detects a high-speed vehicle by estimating the speed of vehicles then, models next task is to recognize the license number plate of the speedy vehicle. Figure1 represent the models flow chart.

A. Input Video and Image Estimation

The video track input has been taken from the publically available videos of the CCTV cameras on roads. The initial stage of the model is to acquire the sequence of images from the input video. The length of the video is long and will be computationally expensive and infeasible to focus the whole video at the same time. So, the model fixes a length of screen from where the model will start tracking and detecting the vehicles. The selected width and height is Height: 1300; Width: 710.

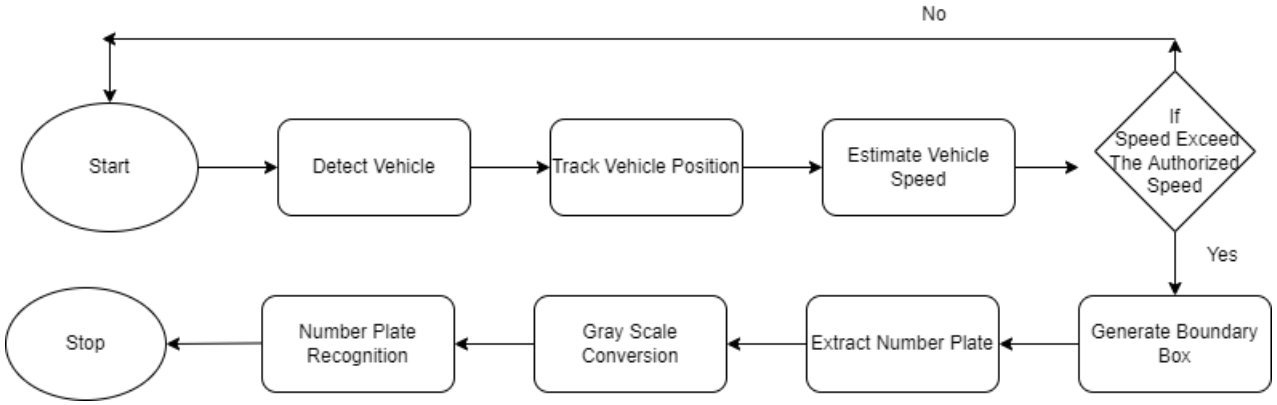

Fig .1 Flow Diagram

B. Vehicle Detection and Tracking

As the video is inserted and segmented into sequences of images, the model will first detect the vehicle as an object in the video. For detecting the vehicle, the system used the dlib library and designed a rectangle box to locate the vehicle from the particular window frame as shown in figure 2. In the tracking phase, the model will check the quality of the tracked vehicle. If the vehicle is coming within the window frame of the assigned location, then it will be considered, otherwise it will be eliminated. Now the model will track the vehicle height and width to create the rectangular boundary box over the detected vehicle as shown in Figure 2.The system will now track the position of the vehicle using get_position() function.

```
TrackX = int (trackPos.left())
TrackY = int (trackPos.top())
TrackWID = int (trackPos.width())
TrackHEI = int (trackPos.height())
```

To perform the tracking and indicating the model to start tracking the vehicle, model used the tracker.start_track() function to locate and track the particular vehicle. The model used the get_position() function to focus and keep the rectangle boundary box on the tracked vehicle.

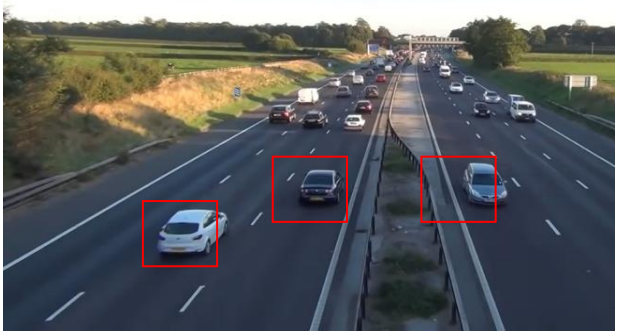

Fig.2 Boundary boxing on vehicle detection

C. Speed Estimation

After detecting and tracking the moving vehicle, the model needs to estimate the speed of the running vehicles. So, to estimate the speed of the vehicles a user defined function is created and as the length of the video is of large size, the model gives two fixed location that is location 1 and location 2. With the help of two distinct locations the model can measure the speed. For measuring the speed, the model used the following equations:

$$Pixel = math. \sqrt{(math.pow(locat2[0] - locat1[0].2) + math.pow(locat2[1] - locat2[1].2))} \quad (3)$$

$$pixelsperminute = 8.9 \quad (4)$$

$$dir.meters = pixels/pixelsperminute \quad (5)$$

$$Framepersecond = 40 \quad (6)$$

$$speed = dir.meters * framePerSecond * 3. \quad (7)$$

A boundary boxing is generated over the extracted vehicle image to observe it in a better and convenient way. Finally, the system will display the annotated text box over the tracked vehicle using the cv2.putText () function. Then the estimated speed of the boundary boxed vehicle is displayed on the vehicle as shown in figure 3.

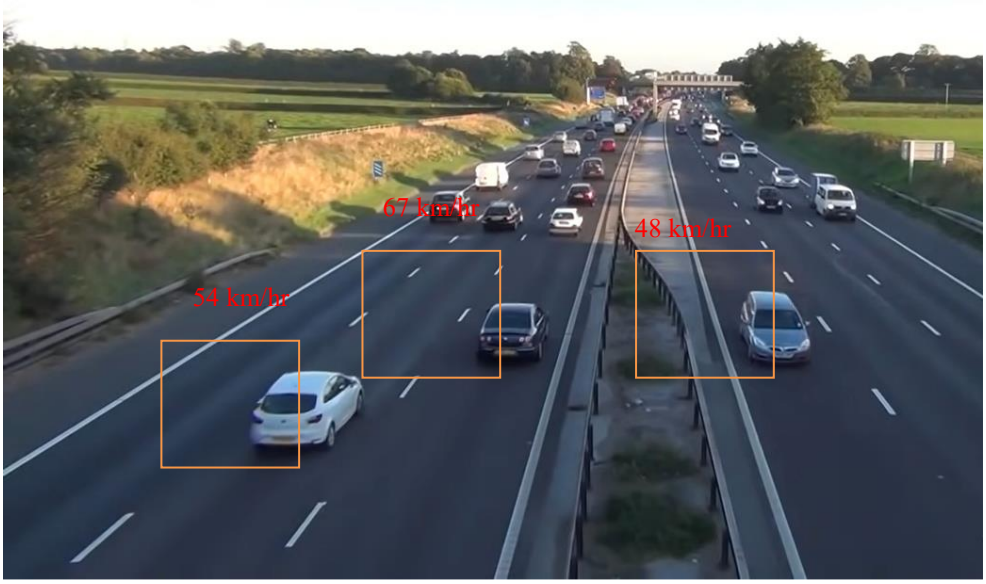

Fig.3. Display Calculated Speed of Vehicle

D. License Plate Recognition for Speedy Vehicle

As the system has calculated the estimated speed of the vehicles, the next task is to read and recognize the license plates character of the speedy vehicle. The system will recognize the license plate characters using the pytesseract engine. The publically available Haar cascade number plate xml file has been uploaded for recognizing the license plates characters.

Then the system used the pytessaract.image_to_string () function to extract and recognize the character of license plates from the image extracted. Finally, the recognized characters of the license plates are recorded for future action for the speedy vehicle owners.

5. Result Observations and Analysis

The implemented model has been evaluated using recall, precision, f-score and accuracy. Initially the model is Evaluated based on recall which evaluates the model by identifying the accurately recognized speedy vehicles license plates divided by the sum of ground truth. Another evaluation parameter is precision, it is based on the correctly recognized speedy vehicles license plates divided by the mean value of all the total detected vehicles. Then, with the help of recall and precisions value F-Score is evaluated:

$$F - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (8)$$

Figure 4 demonstrates the observed outcome for the license plate recognition performed using implemented model. The comparison is done based on four different approaches for

recognizing the license plate characters that is VGG16, Cascade Deep learning, ResNet101 and ResNet 160.

Figure 5 represent the recall, precision and F-Score for the four different video tracks used for implementing the model and analyzing the accuracy of the implemented model

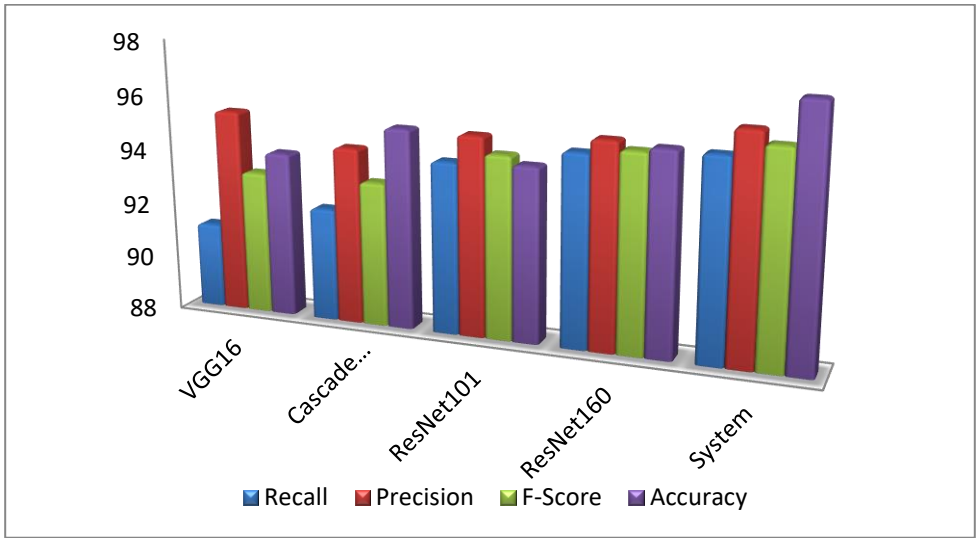

Fig.4. License Plates Recognition Analysis

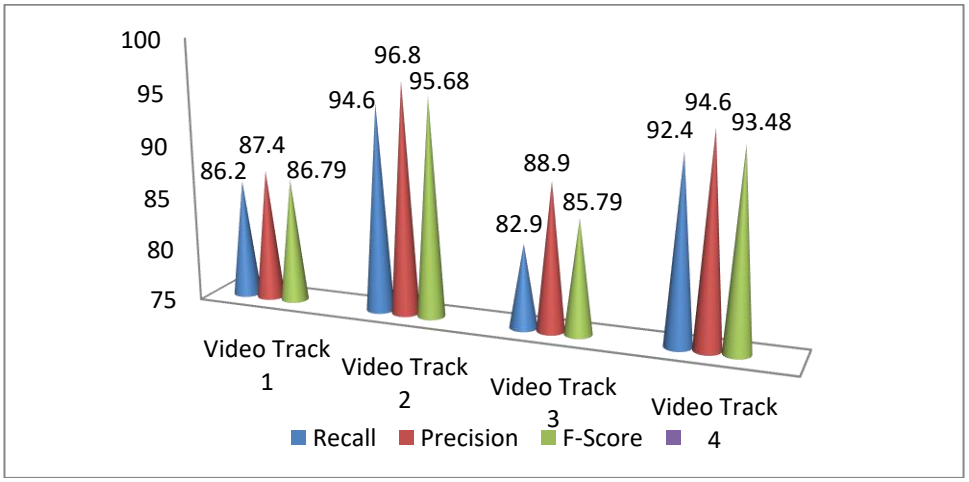

Fig.5. Evaluation based on Different Dataset

5. CONCLUSION

Due to an increase in the number of road accidents and rash driving vehicles, the need for detecting the speed of vehicles is a demanding factor in a smart city. The paper focuses on detecting the speedy vehicle and then performing licence plate recognition to take further action for the speedy vehicle owner. The model used the concept of Cascade Classifier, the pytesseract OCR tool, and some well known computer vision libraries like dlib, numpy, time, etc. to perform vehicle detection and licence plate recognition. The system uses the thrasholding technique for performing image processing tasks. The model helps transportation services for capturing all the speedy vehicles on road which can be a tricky task by manual exercises and keep a record for their license plates numbers for future action. For future enhancement, the model will improve the licence plate recognition system in various different lightning conditions and will try to capture the licence plate character even in dark night lights

REFERENCES

1. Lin, C. H., & Li, Y. (2019, August). A license plate recognition system for severe tilt angles using mask R-CNN. In 2019 International Conference on Advanced Mechatronic Systems (ICAMechS) (pp. 229-234). IEEE.
2. Kessentini, Y., Besbes, M. D., Ammar, S., &Chabbouh, A. (2019). A two-stage deep neural network for multi-norm license plate detection and recognition. Expert systems with applications, 136, 159-170.
3. Lazrus, A., & Choubey, S. (2011). A robust method of license plate recognition using ANN. Int. J. Comput. Sci. Inf. Technol, 2(4), 1494-1497.
4. Xie, L., Ahmad, T., Jin, L., Liu, Y., & Zhang, S. (2018). A new CNN-based method for multi-directional car license plate detection. IEEE Transactions on Intelligent Transportation Systems, 19(2), 507-517.
5. Dhar, P., Guha, S., Biswas, T., & Abedin, M. Z. (2018, February). A system design for license plate recognition by using edge detection and convolution neural network. 2018 International Conference on Computer, Communication, Chemical, Material and Electronic Engineering (IC4ME2) (pp. 1-4).
6. Pustokhina, I. V., Pustokhin, D. A., Rodrigues, J. J., Gupta, D., Khanna, A., Shankar, K., ... & Joshi, G. P. (2020). Automatic vehicle license plate recognition using optimal K-means with convolutional neural network for intelligent transportation systems. Ieee Access, 8, 92907-92917.
7. Jia, Y., Wu, J., & Du, Y. (2016, November). Traffic speed prediction using deep learning method. In 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC) (pp. 1217-1222). IEEE.
8. Hua, S., Kapoor, M., &Anastasiu, D. C. (2018). Vehicle tracking and speed estimation from traffic videos. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (pp. 153-160).
9. Zhang, Z., & Yang, X. (2020). Freeway traffic speed estimation by regression machine-learning techniques using probe vehicle and sensor detector data. Journal of transportation engineering, Part A: Systems, 146(12), 04020138.
10. Wang, J. X. (2016, July). Research of vehicle speed detection algorithm in video surveillance. In 2016 International Conference on Audio, Language and Image Processing (ICALIP) (pp. 349-352). IEEE.
11. Omar, N., Sengur, A., Al-Ali, S. G. S. (2020). Cascaded deep learning-based efficient approach for license plate detection and recognition. Expert Systems with Applications, 149, 113280
12. Yadav, U., Verma, S., Xaxa, D. K., Mahobiya, C. (2017, April). A deep learning based character recognition system from multimedia document. In 2017 innovations in power and advanced computing technologies (i-PACT) (pp. 1-7). IEEE
13. Kluwak, K., Segen, J., Kulbacki, M., Drabik, A., & Wojciechowski, K. (2016, March). ALPR-extension to traditional plate recognition methods. In Asian Conference on Intelligent Information and Database Systems (pp. 755-764). Springer, Berlin, Heidelberg.
14. Huang, L., & Xia, Y. (2020). Joint blur kernel estimation and CNN for blind image restoration. Neurocomputing, 396, 324-345.
15. S.S. Bedi, G.S. Tomar & Shekhar Verma, "Robust Watermarking of Image in the Transform Domain using Edge Detection", IEEE International Conference on simulation UKSIM 2009, pp.233-238, Mar 25-29, 2009.
16. Sharma, P. S., Roy, P. K., Ahmad, N., Ahuja, J., & Kumar, N. (2019, March). Localisation of License Plate and Character Recognition Using Haar Cascade. In 2019 6th International Conference on Computing for Sustainable Global Development (INDIACom) (pp. 971-974). IEEE.
17. Zamir, S. W., Arora, A., Khan, S., Hayat, M., Khan, F. S., Yang, M. H., & Shao, L. (2021). Multi-stage progressive image restoration. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 14821-14831)