DAYANANDA SAGAR UNIVERSITY

KUDLU GATE, BANGALORE – 560068

Bachelor of Technology
in
COMPUTER SCIENCE AND ENGINEERING

Special Topic- 2 Report

PEDESTRIAN DETECTION SYSTEM

By

HARSHA TIWARI - ENG20AM0029
HITESH K - ENG20AM0030
K S SRUJAN - ENG20AM0032
MOHAMMED FUZAIL - ENG20AM0040

Under the supervision of
Prof. ROSHNI M BALAKRISHNAN
ASSISTANT PROFESSOR, AI&ML

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING,
SCHOOL OF ENGINEERING
DAYANANDA SAGAR UNIVERSITY,

(2022-2023)

DAYANANDA SAGAR UNIVERSITY

School of Engineering
Department of Computer Science& Engineering
Kudlu Gate, Bangalore –560068
Karnataka, India

CERTIFICATE

This is to certify that the Special Topic 2 titled **"PEDESTRIAN DETECTION SYSTEM"** is carried out by **HARSHA TIWARI (ENG20AM0029), HITESH K (ENG20AM0030), K S SRUJAN (ENG20AM0032) and MOHAMMED FUZAIL (ENG20AM0040)**, bonafide students of Bachelor of Technology in Computer Science and Engineering at the School of Engineering, Dayananda Sagar University, Bangalore in partial fulfilment for the award of degree in Bachelor of Technology in Computer Science and Engineering, during the year **2022-2023**.

Prof. Roshni M Balakrishnan Assistant Professor Dept. of CSE, School of Engineering Dayananda Sagar University	**Dr. Girisha G S** Chairman, CSE School of Engineering Dayananda Sagar University	**Dr. Uday Kumar Reddy K R** Dean School of Engineering Dayananda Sagar University
Date:	Date:	Date:

Name of the Examiner **Signature of Examiner**

1.

2.

DECLARATION

We, **HARSHA TIWARI (ENG20AM0029), HITESH K (ENG20AM0030), K S SRUJAN (ENG20AM0032) and MOHAMMED FUZAIL (ENG20AM0040)** are students of the fifth semester B.Tech in **Computer Science and Engineering**, at School of Engineering, **Dayananda Sagar University**, hereby declare that the Special Topic 2 titled **"PEDESTRIAN DETECTION SYSTEM"** has been carried out by us and submitted in partial fulfilment for the award of degree in **Bachelor of Technology in Computer Science and Engineering** during the academic year **2022-2023**.

Student **Signature**

HARSHA TIWARI
USN: ENG20AM0029

HITESH K
USN: ENG20AM0030

K S SRUJAN
USN: ENG20AM0032

MOHAMMED FUZAIL
USN: ENG20AM0040

Place : Bangalore
Date :

ACKNOWLEDGEMENT

It is a great pleasure for us to acknowledge the assistance and support of many individuals who have been responsible for the successful completion of this Special Topic 2.

First, we take this opportunity to express our sincere gratitude to School of Engineering & Technology, Dayananda Sagar University for providing us with a great opportunity to pursue our Bachelor's degree in this institution.

We would like to thank **Dr. Uday Kumar Reddy K R, Dean**, **School of Engineering & Technology**, **Dayananda Sagar University** for his constant encouragement and expert advice. It is a matter of immense pleasure to express our sincere thanks to **Dr. Girisha G S, Chairman**, **Department of Computer Science, and Engineering**, **Dayananda Sagar University,** for providing the right academic guidance that made our task possible.

We would like to thank our guide **Prof. Roshni M Balakrishnan, Assistant Professor**, **Dept. of Computer Science and Engineering**, **Dayananda Sagar University**, for sparing her valuable time to extend help in every step of our Special Topic 2, which paved the way for smooth progress and the fruitful culmination of the project.

We would like to thank our Special Topic 2 Coordinators, Dr. Savitha Hiremath, Dr. Bondu Venkateswarlu, and all the staff members of Computer Science and Engineering for their support.

We are also grateful to our family and friends who provided us with every requirement throughout the course. We would like to thank one and all who directly or indirectly helped us in the Special Topic 2.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

YOLO	You Only Look Once
COCO	Common Objects in Contact
IoU	Intersection over Union
FPS	Frames Per Second
FLOPS	Floating point Operations Per Second
mAP	Mean Average Precision
RPN	Region Proposal Network

LIST OF FIGURES

ABSTRACT

Pedestrian detection is a specific application of object detection. Compared with general object detection, it shows similarities and unique characteristics. In addition, it has important application value in the fields of intelligent driving and security monitoring. In recent years, with the rapid development of deep learning, pedestrian detection technology has also made great progress. However, there still exists a huge gap between it and human perception. Meanwhile, there are still a lot of problems, and there remains a lot of room for research. Regarding the application of pedestrian detection in intelligent driving technology, it is of necessity to ensure its real-time performance. Additionally, it is necessary to lighten the model while ensuring detection accuracy.

In this paper, we show how YOLO is used to train the pedestrian classifier and perform the pedestrian detection. Experimental results show that our method has good detection results under the complicated conditions of detecting small-scale pedestrians and pedestrian occlusion.

Chapter 1 INTRODUCTION

Pedestrians can be defined as any person who is walking, especially in a place where vehicles are moving. The pedestrians might be close to moving vehicles and are at a risk of being hit intentionally or unintentionally. Accidents can take place at any time, it happens in the blink of an eye. You might be driving and take your eyes off the road to reach for your coffee cup or turn around to tell your kids to quiet down, and when you look ahead, a pedestrian is crossing the road right in front of you. You hit the brakes—but it may be too late. Unfortunately, this scenario is all too common. One out of three vehicle-pedestrian crashes involve a vehicle going straight as a pedestrian crosses the road. And fatalities involving vulnerable road users, such as pedestrians, bicyclists, and motorcyclists, have increased over the past decade. To prevent these crashes, automakers now offer a "pedestrian detection" system in some models. If the system detects that a pedestrian could be in the vehicle's travel path, it alerts the driver or employs automatic emergency braking, preventing what could be a fatal crash. Pedestrian detection, as special object detection, has gradually become a research hotspot in the field of computer vision, and is also an important technology to promote the development of smart city. It detects some specific pedestrian targets by extracting some features of the image, and provides the necessary technical basis for the higher-level tasks such as behaviour recognition and analysis, pedestrian attitude analysis and research.

The system we designed is a YOLO model that detects pedestrians. The YOLO algorithm creatively predicts n boxes in each area on the area where the feature map is finally output. These boxes have different sizes

and positions, covering almost all possible target positions. And predict the offset of the preselection box and the real object, rather than the absolute position coordinates. We use the deep learning framework PyTorch to build the overall target detection network structure of YOLO. Firstly, we used Roboflow to manually label the pedestrian data set. Roboflow provides a free and interactive environment to annotate images by including our teammates to assign images to individuals and create an annotated dataset on the platform. Secondly, through the built-in library functions provided by PyTorch, analyse the three types of network structure information cfg files and build a deep learning model. Finally, we use the pre-training weights to obtain a new weight file, and use the interface DNN to analyse the deep learning model, load the trained model.

CHAPTER 2 – PROBLEM DEFINITION

The problem of pedestrian detection has been considered as a vital one in the domain of Computer Vision and Pattern Recognition. It is formulated as the problem of automatically identifying and locating the pedestrians in images or videos. In the past many techniques have been proposed to solve this problem accurately and efficiently. However, the variations in images such as body attire and pose, occlusion, different illumination parameters in different scenarios and clutter present in the background poses challenges in attaining high accuracy.

We will address this problem by creating an object detection algorithm which will identify pedestrians on the road that are dangerously close to the moving vehicles.
This is implemented by using an object detection algorithm, YOLO to detect pedestrians by training the algorithm on manually defined labels.

CHAPTER 3 – LITERATURE SURVEY

[1] Pedestrian detection is an important problem of computer vision. Compared with general object detection, it has important research value in the field of intelligent driving. It has similarities and differences with general object detection. This review first introduces the content of general object detection, then analyses the development of pedestrian detection, and elaborates on the common datasets and main problems faced by pedestrian detection.

[2] Main contribution of this paper is to provide a general overview of pedestrian detection process that is viewed from different sides of the discussion. We divide the discussion into three stages: input, process and output. This paper does not make a selection or technique best method and optimal because the best technique depends on the needs, concerns and existing environment. However, this paper is useful for future researchers who want to know the current researches related to pedestrian detection.

[3] This paper shows how we are using Pytorch framework to train the model on our custom dataset in the google colab editor on a GPU for higher precision and recall rate. After going through numerous iterations of training we were able to achieve high accuracy. And then we used the weights of the trained model and OpenCV to make predictions on input images and real-time input from the user.

[4] This paper uses the PyTorch framework to compare horizontally and vertically in pedestrian detection with YOLO, YOLO-Tiny, and YOLO-

SPP. YOLO and YOLO-SPP networks can guarantee a high recall rate, the average confidence can reach more than 80%, using the CPU to process each picture is usually more than 1s, mainly because YOLO and YOLO-SPP use complete Compared with YOLO-Tiny's YOLO network structure, the amount of calculation is much larger, but under GPU acceleration, the number of frames can reach about 10 frames, which can barely meet the real-time performance, and the effect is improved significantly. Using YOLO-Tiny as the input network is slightly inferior to the other two networks in terms of recall and confidence. However, due to the lighter network structure of YOLO Tiny, the amount of calculation is small when performing detection tasks. The processing speed of pictures and videos is faster, and it is better than the other two networks in real-time.

[5] Most of the techniques are appropriate with images containing pedestrians in the upright standing position. For other poses such as sitting or having some other object the detection algorithm’s accuracy degrades. Similarly, part-based models work well against the deformation, but they require prior information of the number of the parts. The count of different parts varies with the number of poses and viewing angles. This challenge becomes more complicated in complex scenarios.

[6] Sensors such as thermophile and infrared sensors accuracy needs to be improved. Extraction of 2D and 3D images on the basis of foreground and background needs to be done. Infrared cameras with low cost. This increases efficiency and accuracy of the model.

[7] Pedestrians have large span and slight distortion in a railway station pedestrian dataset, so small-size pedestrians that have low resolution are often missed. The advantage lies in not only making full use of pixel channel information to optimize the feature extraction network in a cascading manner in order to build a stronger feature extraction network, but also using unsupervised learning algorithms to effectively improve the RPN search mechanism, all of which effectively alleviate the problems of pedestrian detection that are difficult to detect small-size pedestrians.

[8] In order to improve the network performance as well as to increase the recall levels while keeping precision, we modified the architecture of YOLOv3 detector by reducing the number of classes of the task.
The original YOLOv3 was trained on the generic dataset COCO, which includes up to 80 different classes. However, in our task, we are only interested in pedestrians. Therefore, we trimmed the last layer of the network so that it predicts bounding boxes of only 6 floating point values (2 for position, 2 for shape, 1 for bounding box confidence and 1 for the class confidence) in contrast with the original 85 values. As computation time is a critical parameter, different grid sizes have been evaluated, with some of them performing better at slightly higher computation times, getting an improvement in mAP from 72.04% to 74.12%.

CHAPTER 4 – PROJECT DESCRIPTION

The project focuses on identification of pedestrians on the road. The approach of YOLO is used in order to create bounding boxes for the pedestrians and detect them faster and with a good accuracy.
YOLO predicts the coordinates of bounding boxes directly using fully connected layers on top of the convolutional feature extractor. Predicting offsets instead of coordinates simplifies the problem and makes it easier for the network to learn.

4.1 Proposed design

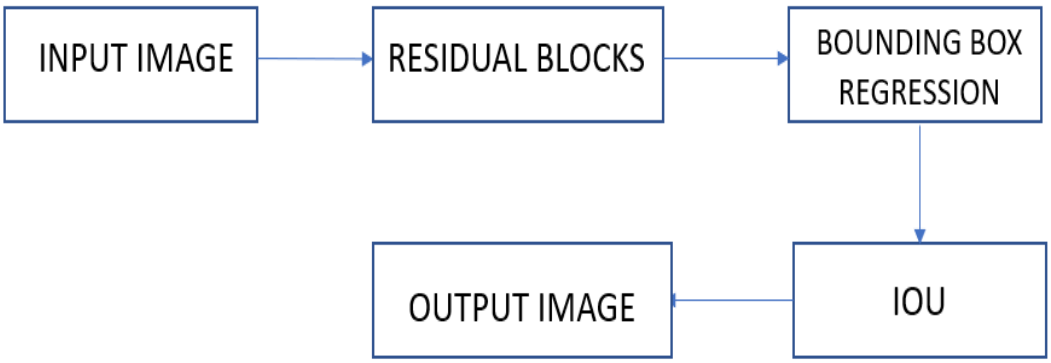

Figure 4.1

The image and annotation file are fed into the YOLO network, which passes it to the backbone convolutional network, which is passed to hidden layers to separate the most important features from the backbone and to improve the speed of the network. Final image is obtained with all bounding boxes over pedestrians.

4.2 Working of YOLO model

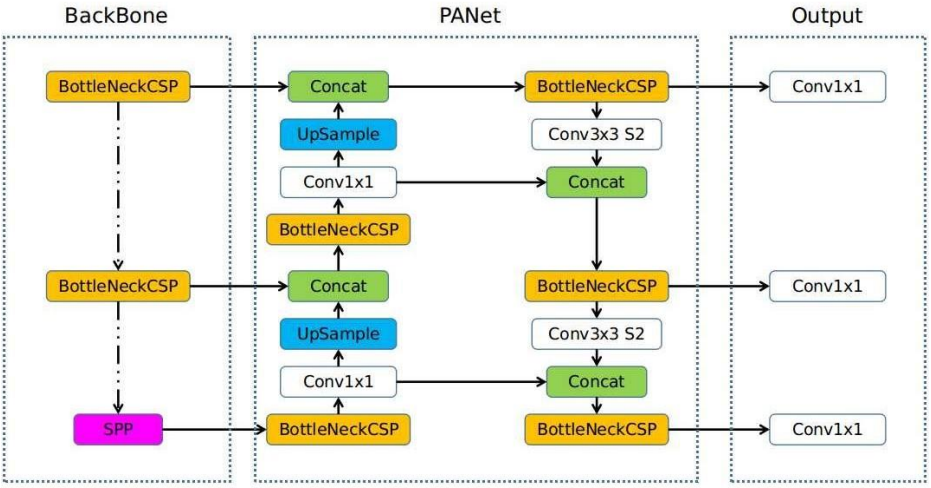

Figure 4.2

1. **Backbone**
   - A convolutional neural network that aggregates and forms image features at different granularities.
   - A stage of CSPDenseNet is composed of a partial dense block and a partial transition layer. In a partial dense block, the feature maps of the base layer in a stage are split into two parts
   - Applying this strategy comes with big advantages to YOLOv5, since it helps reducing the number of parameters and helps reducing an important amount of computation (less FLOPS) which lead to **increasing the inference speed** that is crucial parameter in real-time object detection models.

2. **Convolutional layer**
   - A series of layers to mix and combine image features to pass them forward to prediction.

- The Model Neck is mostly used to create feature pyramids. Feature pyramids aid models in generalizing successfully when it comes to object scaling. It aids in the identification of the same object in various sizes and scales.
- Feature pyramids are quite beneficial in assisting models to perform effectively on previously unseen data. Other models, such as FPN, BiFPN, and PANet, use various sorts of feature pyramid approaches.
- PANet is used as a neck in YOLO v5 to get feature pyramids.

3. Head

- The model Head is mostly responsible for the final detection step. It uses anchor boxes to construct final output vectors with class probabilities, objectness scores, and bounding boxes.
- It consumes features from the neck and takes box and class prediction steps.

CHAPTER 5 - REQUIREMENTS

Operating system	Windows 11/10/8 macOS Linux
RAM	512MB and above
Processor	Pentium 4 and above
Softwares	Python GitHub GitBash Google Colab Jupyter Notebook Anaconda
Storage	5GB and above

CHAPTER 6 - METHODOLOGY

YOLO is a single stage deep learning algorithm which uses convolution neural network for object detection. It is popular due to its speed and accuracy. There are various deep learning algorithms, but they are unable detect an object in a single run but YOLO, on the other hand, makes the detection in a single forward propagation through a neural network which makes it suitable for real-time application. This property has made YOLO algorithm popular among the other deep learning algorithms.

However, YOLOv5 is different from the previous releases. It utilizes PyTorch instead of Darknet. It utilizes CSPDarknet53 as backbone. This backbone solves the repetitive gradient information in large backbones and integrates gradient change into feature map that reduces the inference speed, increases accuracy, and reduces the model size by decreasing the parameters. It uses path aggregation network (PANet) as neck to boost the information flow. PANet adopts a new feature pyramid network (FPN) that includes several bottom ups and top-down layers. This improves the propagation of low-level features in the model. PANet improves the localization in lower layers, which enhances the localization accuracy of the object. In addition, the head in YOLOv5 is the same as YOLOv4 and YOLOv3 which generates three different output of feature maps to achieve multi scale prediction. It also helps to enhance the prediction of small to large objects efficiently in the model. The image is fed to CSPDarknet53 for feature extraction and again fed to PANet for feature fusion. Finally, the YOLO layer generates the results.

6.1. Methods of data collection and processing

We have downloaded the dataset from an open-source website, which allows us to use the images with proper consent. The dataset we have used is made by installing a camera on a car, and recording the footage in a video format. This video is then divided into frames to get individual images of pedestrians on the roads.

After downloading the dataset, a website called Roboflow is used to label and annotate the images. The dataset is uploaded to Roboflow website and its annotation tool was used to annotate the images. The website also allows us to create a file online which consists of the images, their respective annotations and the yaml file to configure our dataset.

In annotation of images, the pictures consisting of good quality and distinguished pedestrian images were given priority. This enables us to train an effective model. The model will recognize and distinguish between a pedestrian and other object with a better efficiency and accuracy.

6.2. Processing dataset

The final step in preparing our dataset to train our YOLO model is to combine all images and separate them into testing, training and validation folders. We separate images and labels as two different folders.
With the help of Roboflow, we optimize the dataset as to fit the PyTorch specifications and create yaml file accordingly.
This dataset can be downloaded as a zip file onto our system.

CHAPTER 7 - EXPREIMENTATION

The first step of implementation of our code is to link the dataset with the code. In order to link it, we need to prepare our dataset. This includes annotating all the images and storing the annotations in text files. Hence, each image has its corresponding annotation file. We have achieved this by using a platform called Roboflow. It allows us to upload our dataset and work on it with our team mates. We divide the images within the team mates and through roboflow, we do not have to worry about images being misplaced as it automates the process of adding the image and its annotated file to the dataset.

Once we have annotated all the images, we choose to create a dataset which is split into three parts, which are training, testing and validation. The data is kept in folders, and there are three folders each for test, train and validate. They have sub folders for images and labels. We have a YAML file which stores the classes and paths to our dataset.

The implementation of code was through cloning the official YOLOv5 code in our notebook so that we can make use of it to apply our custom dataset that we have created. The official YOLO version is available on GitHub which was made by Ultralytics. We have used YOLOv5s model in our implementation.

The implementation was done using Google Colab first, but due to having limited resources, the run time for 100 epochs to complete was very high and not feasible to be executed, so we used Jupyter Notebook with the help of Anaconda to train our model. We created an environment to run

our laptop's Nvidia graphic card using CUDA and CUDNN softwares. The specs used are an Intel i5-10300H CPU chipset and an Nvidia GeForce GTX 1650Ti graphics card set up. By using our own resources, we reduced the run time for 100 epochs drastically and hence optimizing the model.

Hence, we obtained a better precision and recall which in turn gives us a better mAP (mean Average Precision). mAP was calculated over different IoU thresholds in the range of 0.5 - 0.95. With the help of these metrics, we analyse our YOLOv5 model.

CHAPTER 8 – TESTING AND RESULTS

The model is divided in such a way that it needs to detect if an image contains pedestrians or not. In order to achieve this, we need to have both images and their respective annotation files that we have created in the data set.
To train the model, we use three sets of data, one being training, the second being testing and the third being validation. The validation phase is followed by testing phase. The training dataset is larger than the validation dataset.
We can use training dataset to evaluate the performance and progress of algorithms' training and adjust or optimize it for improved results.

Training data has two main criteria. It should:

- Represent the actual dataset
- Be large enough to generate meaningful predictions

We split our dataset into 70% for training, 20% for validation and 10% for testing. A validation dataset is a sample of data held back from training your model that is used to give an estimate of model skill when tuning the model's hyperparameters.

The validation dataset is different from the test dataset that is also held back from the training of the model, but is instead used to give an unbiased estimate of the skill of the final tuned model when comparing or selecting between final models.
An epoch in machine learning means one complete pass of the training dataset through the algorithm. This epochs number is an important

hyperparameter for the algorithm. It specifies the number of epochs or complete passes of the entire training dataset passing through the training or learning process of the YOLO algorithm. With each epoch, the dataset's internal model parameters are updated. Hence, a 1 batch epoch is called the batch gradient descent learning algorithm. Normally the batch size of an epoch is 1 or more and is always an integer value.
For our model, we are going to run it for 100 epochs to get a good accuracy.

The following graphs represent how our model has been trained for our dataset

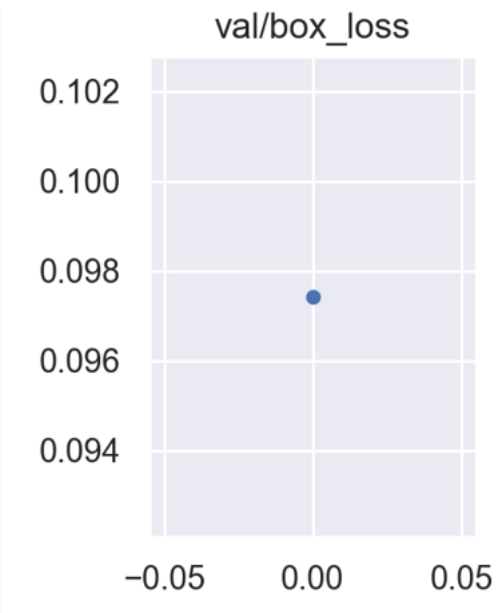

Figure 8.1

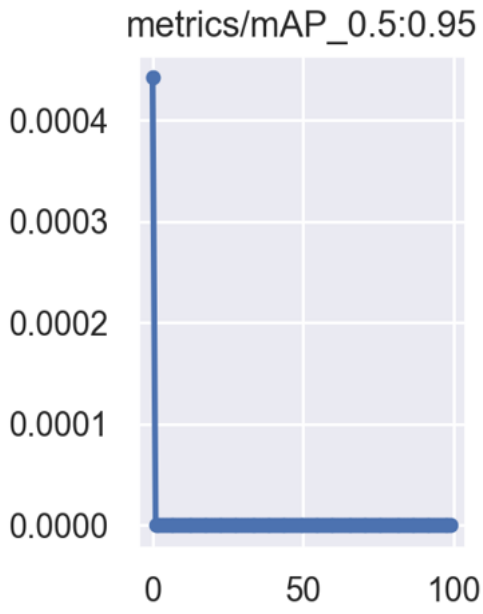

Figure 8.2

Final results can be observed as we compare the input image and the output we receive:

Figure 8.3

Figure 8.4

CHAPTER 9 - CONCLUSION

From the above stated data, we can conclude how a YOLO model can be used to implement this pedestrian detection system. In addition to increased accuracy in predictions and a better Intersection over Union in bounding boxes (compared to real-time object detectors), YOLO has the inherent advantage of speed. YOLO is considered because it gives results to larger datasets in a small amount of time. It is an open-source project and hence it allows us to utilize its full potential without any hindrances.
Upon successful training of our dataset, we can process images for object detection and detect pedestrians on the roads.

Hence, this project is very much useful in the real world as safety is a big concern in our everyday lives. Major automakers consider the safety of everyone seated in the car and, also for those in its surroundings. Development of this system will enable us to make a safer environment for everyone.

Chapter 10 – FUTURE ENHANCEMENTS

The project may be very useful in the real world in terms of detecting pedestrians on the road and acting accordingly. This will prevent many accidents from taking place. Our project currently trains a model to detect these pedestrians, but with further enhancements we can modify the project so that we can use a camera to provide a live feed of the roads and surroundings. This would be input as a video and the output will be instantaneous. It will greatly help drivers to detect pedestrians on the road. We can also set parameters such that the distance between the pedestrian and moving vehicle is used and whenever the safe distance is compromised, it may alert the user. We can also improve the overall working and accuracy of the model by using more advanced versions of YOLO such as YOLOv7 with image segmentation which will greatly increase accuracy.

Lastly, the project can further be developed to detect other obstacles on the road such as bicycles, because cyclists can sometimes be undetected by the driver. Hence, this project has immense scope for improvements and the social impact would help create a safe environment for pedestrians.

REFERENCES

[1] Di Tian, Yi Han, Biyao Wang, Tian Guan, Wei Wei, "A Review of Intelligent Driving Pedestrian Detection Based on Deep Learning", *Computational Intelligence and Neuroscience*, vol. 2021, Article ID 5410049

[2] Achmad Solichin, Agus Harjoko and Agfianto Eko Putra, "A Survey of Pedestrian Detection in Video", *International Journal of Advanced Computer Science and Applications (IJACSA)*, 5(10), 2014.

[3] Shayan Alam, Satyam, Prakhar Vashistha, Javed Miya, Ranjit Kumar, Suresh Kumar, "Pedestrian Detection Based on Deep Learning", *International Journal For Research in Applied Science & Engineering Research*, 2022

[4] Xun Zuo, Jiaojun Li, Jie Huang, Fan Yang, Tian Qiu and Yang Jiang, "Pedestrian detection based on one-stage YOLO algorithm", *Journal of Physics: Conference series*, J. Phys.: Conf Ser 1871 012131, 2021

[5] Bali, Shweta and Tyagi, S. S, "A Review of Vision-Based Pedestrian Detection Techniques", *International Journal of Advanced Studies of Scientific Research*, Volume 3, Issue 9, 2018.

[6] Z. Li, K. Wang, L. Li and F. -Y. Wang, "A Review on Vision-Based Pedestrian Detection for Intelligent Vehicles," *2006 IEEE International Conference on Vehicular Electronics and Safety*, 2006

[7] He, Jiaojiao & Liu, Ken & Zhang, Yongping & Yao, Tuozhong & Zhao, Zhongjie & Xiao, Jiangjian & Peng, Chengbin, "A Channel-Cascading Pedestrian Detection Network for Small-Size Pedestrians", *8th International Conference, IScIDE*, Lanzhou, China, 2018

[8] Ortiz Castelló V, del Tejo Catalá O, Salvador Igual I, Perez-Cortes J-C. “Real-time on-board pedestrian detection using generic single-stage algorithms and on-road databases”, *International Journal of Advanced Robotic Systems*. 2020