



Research Article

DEVELOPMENT OF CNN-BASED GUI FOR DETECTION OF NON-MOTORIZED VEHICLES

Authors: Sinan UĞUZ , Oğulcan ÇİFTÇİ 

To cite to this article: Uğuz, S. & Çiftçi, O. (2022). Development of CNN-based GUI for detection of non-motorized vehicles . International Journal of Engineering and Innovative Research ,4(3),p208-215 . DOI: 10.47933/ijeir.1178790

DOI: 10.47933/ijeir.1178790



To link to this article: <https://dergipark.org.tr/tr/pub/ijeir/archive>



International Journal of Engineering and Innovative Research

<http://dergipark.gov.tr/ijeir>

DEVELOPMENT OF CNN-BASED GUI FOR DETECTION OF NON-MOTORIZED VEHICLES

Sinan UĞUZ^{1*} , Oğulcan ÇİFTÇİ² 

¹Isparta University of Applied Sciences, Faculty of Technology, Computer Engineering Department, Isparta, Turkey.

²Isparta University of Applied Sciences, Faculty of Technology, Mechatronics Engineering Department, Isparta, Turkey.

*Corresponding Author: sinanuguz@isparta.edu.tr

<https://doi.org/10.47933/ijeir.1178790>

(Received: 22.08.2022; Accepted: 30.09.2022)

ABSTRACT: Today, several solutions are offered for traffic density. One of these proposals is to popularize the use of bicycles in the category of non-motorized vehicles. For this purpose, first of all, bicycle lanes must be built. The use of bicycle lanes or the percentage of bicycle use in normal traffic are important data. Deep learning techniques, which have become popular in recent years, can be used to obtain this data. The aim of this study is to present a model for detecting bicycles using different convolutional neural networks architectures. First, 962 open-source bicycle images obtained from the Internet are labeled. For this purpose, trainings were performed using YOLOv3, YOLOF, Faster R-CNN and Sparse R-CNN architectures. The model with the best performance is the Faster R-CNN with 0.92 mAP. At the end of the study, software was developed to recognize bicycles in real time.

Keywords: deep learning, bicycle detection, traffic, object detection.

1. INTRODUCTION

As population density in cities increases, the number of motor vehicles used also increases. People tend to move around the city using their private vehicles instead of public transportation. This causes more carbon emissions and pollutes the environment. If people use bicycles instead, the most common non-motorized mode of transportation in the city, both the damage to nature will decrease and people will be healthier. In addition, the widespread use of bicycles will reduce traffic density in cities. Therefore, it is important to encourage people to use bicycles. In addition, the authorities need to make cities safe for cycling and build bicycle lanes [1]. The first step to increase bicycle use is to build bicycle lanes in cities. Then, the use of these paths should be monitored. In this paper, a bicycle lane monitoring system is proposed to determine the usage rates of bicycle lanes. Nowadays, many municipalities build bicycle lanes. However, the time before and after must be properly analyzed. In advance, it is important to create the bike lane on the right route. This is because investing in an area where bicycle use may be low causes wasted time and money. The number of people using the road after the construction of the bike path is also important data for the authorities. The system developed in this paper serves as a basis for further studies.

The bicycle path monitoring system has been developed using convolutional neural networks (CNN), which have achieved significant success in object detection applications in recent years. It is seen that CNN-based computer vision systems for object detection produce outputs with very high accuracy rates and are successful in distinguishing different objects. Tzelepi and Tefas (2020) focused on the recognition of vehicles such as people, bicycles or motors with the camera placed on the drone. In this study, it is seen that the VGG-16 model is used and 90%-95% recognition is made on these objects [2]. Tao et al. (2021) experimented with different methods for detecting other vehicles in traffic with the camera positioned on the vehicle in order to provide autonomous driving [3]. Lin and Young (2017) proposed a bicycle detector for side-view image based on the observation that a bicycle consists of two wheels in the form of ellipse shapes and a frame in the form of two triangles [4]. Some researchers [5, 6] have tried to detect bicycles on the road with only wheel detection. Satyanarayana et al. (2022) conducted a study to identify different vehicles besides the bicycle. The detection accuracy of the proposed method on real-time videos is about 98.5%. The runtime performance of CNN model on NVIDIA Jetson Tx2 is 0.01 s. [7]. In another study, 95.05% mAP value was obtained with YOLOv3 for 6 different vehicles [8].

The limited number of studies in the bicycle recognition literature are focused on solving the object detection problem. This study focused on solving an object detection problem centered on bicycle identification. For this purpose, a dataset of 962 bicycle images was obtained from the Internet. The training was performed using YOLOv3, Sparse R-CNN, Faster R-CNN and YoloF architecture. A program was developed to visualize the detection of cyclists through a graphical user interface (GUI).

1.1. Convolutional Neural Networks

As shown in Fig 1, elements of a typical CNN model can be understood in two stages: feature learning and classification. Convolution and the following pooling steps can be repeated in different numbers depending on the developed CNN architecture. Convolution process requires creating image in parts that can move vertically and horizontally on the image shaped square or rectangular. Stride values decides at what pixel rate these image parts can move in these directions. Another element for convolution process is the kernel (also called filter), which represents learnable set of weights corresponding to the size of image parts. Computing the inner products between an image part and kernel gives a digital value [9]. Repeating this process at each sliding position provides filtered image.

In a convolution process, if there are too much patterns specific to the picture, number of parameters increases, network gets more complicated and processing load increases [10]. With pooling operations, image output of convolution process is downscaled. After pooling is completed and before fully connected layer stage, obtained image is converted to one-dimensional data by flattening process. Lastly the fully connected layers (also called dense layer) takes the end result as input to reach a classification decision. And thus the classical neural network stages are completed and classification is performed.

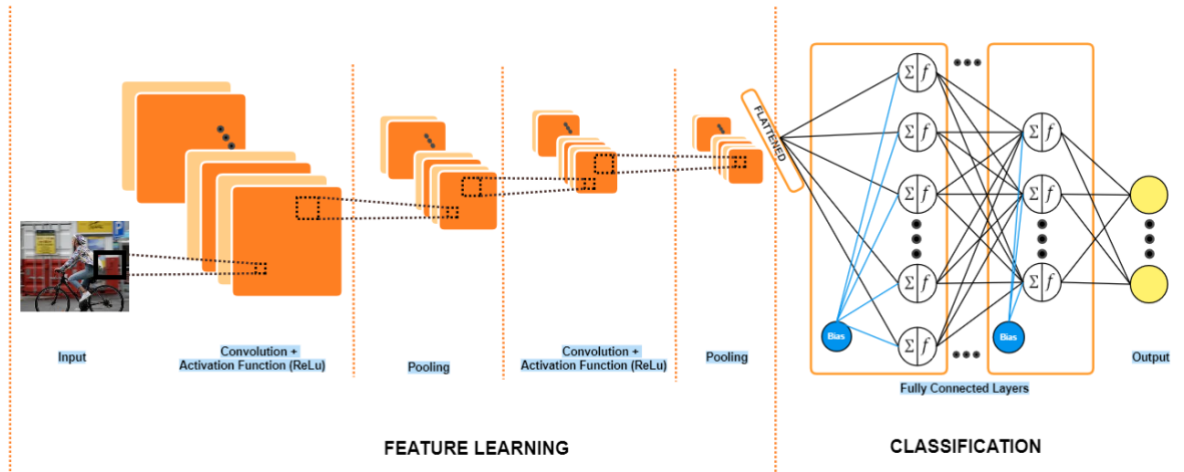


Figure 1. Classical CNN architecture.

CNN architectures first came into picture in 1998 with LeNET-5 [11], then came the AlexNet in 2012 [12], and later various other CNN architectures were developed which introduced successful solutions to object classification and object detection problems. Through these architectures numerous convolutional operations, pooling operations and various types of activation functions were tried. With the ImageNet Large Scale Visual Recognition Challenge competition (ILSVRC) (also known as ImageNet Challenge), millions of training images of various class labels were classified. VGG net architecture (which has 2 versions, 16 and 19) [13] created by Visual Geometry Group from Oxford University, has won the 2014 ImageNet Challenge.

2. METHODS

In this stud, firstly a dataset was created to solve bicycle detection problems. In the next step, the bike images in the dataset were labeled. Thereafter, the dataset trained with state-of-the-art models. In addition, the four different models were compared for the detection of bike images.

2.1. Image Dataset

The dataset in this study was obtained from the internet as open source. A total of 962 images of bicycles were obtained. The images taken at varying high resolution (4096 x 3456 pixel and 72 dpi), were rescaled to 800 x 600 pixel resolution in the dataset folder. Images are labeled as bikes. The coordinates of the bicycles in the images were individually labeled and after all the images were labeled, they were divided into 80% training and 20% testing sets. Figure 2 shows an example of the labeling process.



Figure 2. Labeled bike images.

2.2. Implementation Details of State-Of-The-Art CNN Models

The object detection models used in this study and the backbone networks used for these models are shown in Table 1. As seen in Table 1, YOLOv3, Sparse R-CNN, Faster R-CNN and YoloF architectures were used in this study. The backbone networks using these architectures are shown in parentheses. In the study, the models were trained with 100, 200 and 500 epoch iterations. The experiments were carried out with values of 2, 4, 8 for the batch size parameter. Stochastic gradient descent (SGD) was preferred as the optimizer for the trainings. Other hyper parameters were determined as learning rate 0.01, momentum 0.9, weight decay 0.001. The parameters used in the experiments gave good results in our previous studies [14,15]. Therefore, these parameters are preferred.

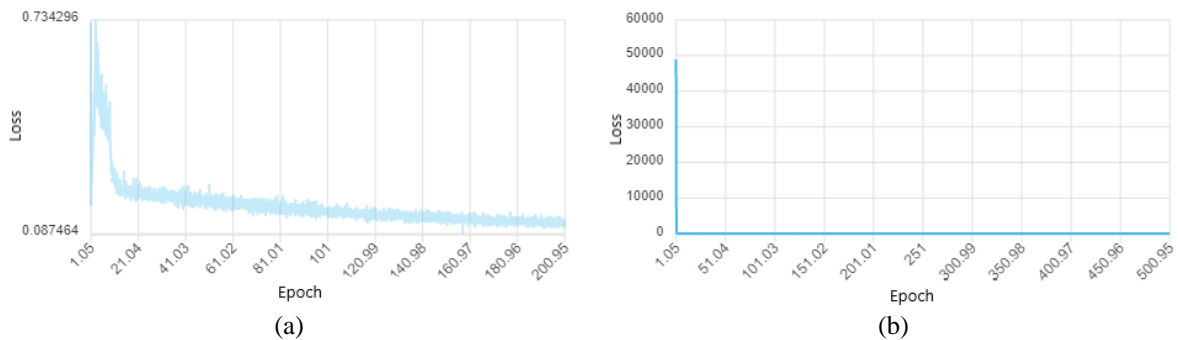
The developed system made use of the PyTorch framework and numerous Python packages. The experiments for bicycle detection in this study were carried out on a workstation with an Intel Xeon CPU, 16 GB Nvidia Quadro RTX5000 GPU, and 16GB RAM.

Table 1. The details of state-of-the-art CNN models

Models (Backbone)	Epoch	Batch Size
YoloF (yolof_r50_c5_8x8_1x_coco)	100	8
YoloF (yolof_r50_c5_8x8_1x_coco)	200	8
YOLOv3 (yolov3_d53_mstrain-416_273e_coco)	100	8
YOLOv3 (yolov3_d53_mstrain-416_273e_coco)	500	8
Sparse R-CNN (sparse_rcnn_r50_fpn_mstrain_480-800_3x_coco)	100	2
Sparse R-CNN (sparse_rcnn_r50_fpn_mstrain_480-800_3x_coco)	200	2
Faster R-CNN (faster_rcnn_r50_caffe_dc5_1x_coco)	100	4
Faster R-CNN (faster_rcnn_r50_caffe_dc5_1x_coco)	200	2

3. EXPERIMENTAL

For the performance evaluation of bicycle path monitoring system, AP was preferred, which are commonly used in the literature [16,17] for object detection problems. The evaluation metrics chosen in this work are similar to those of MS-COCO dataset [18], namely mean average precision (mAP). Figure 3 shows the loss curves of the best performing models. According to this, while the highest loss value with 0.087 belongs to YoloF, the lowest loss value with 0.0091 was obtained with Faster R-CNN model. As the number of training iterations continually increased, the loss values of all models decreased suddenly at first and then decreased slowly.



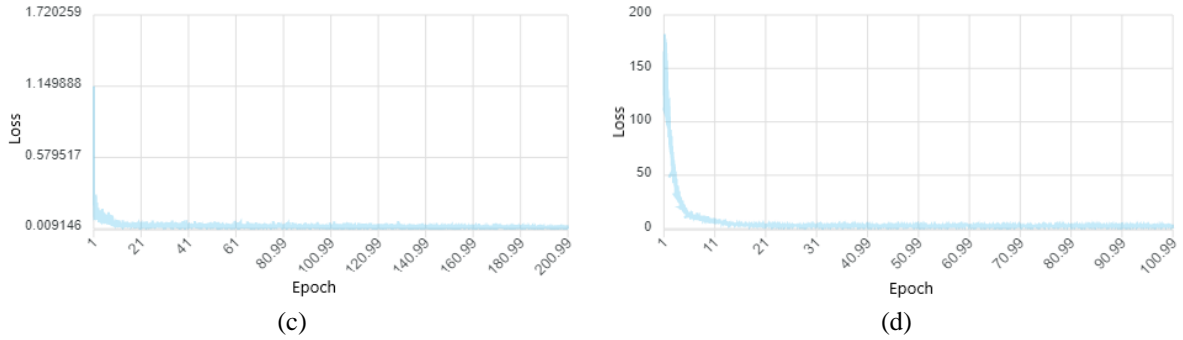


Figure 3. The loss value of the best state-of-the-art models (a) YoloF (200 epoch) (b) YoloV3 (500 epoch) (c) Faster R-CNN (200 epoch) (d) Sparse R-CNN (100 epoch).

Figure 4 shows the mAP values of the models. It is seen that mAP values of YoloV3, Faster RCNN and Sparse RCNN models are increasing gradually. However, the mAP value of the YoloF model, which provides lower performance performance, decreases after 100 epochs.

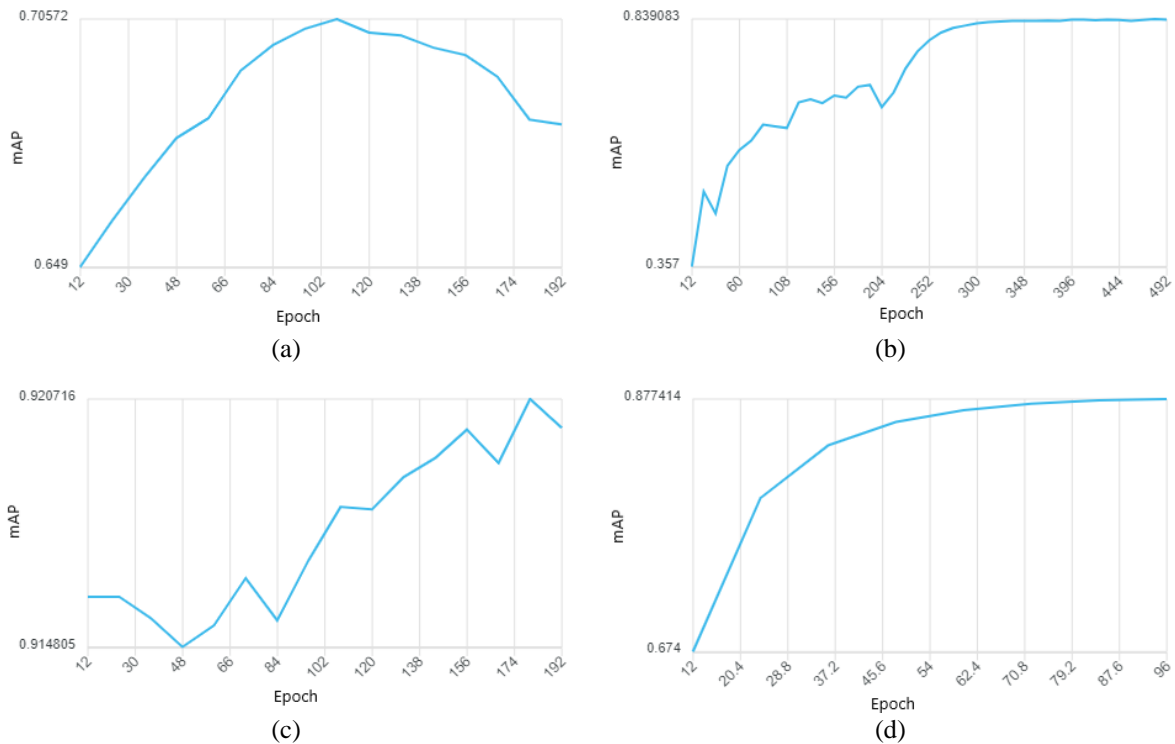


Figure 4. The mAP value of the best state-of-the-art models (a) YoloF (200 epoch) (b) YoloV3 (500 epoch) (c) Faster R-CNN (200 epoch) (d) Sparse R-CNN (100 epoch).

As can be seen from the inference time values in Table 2, Yolo models produce results in a shorter time than other models.

Table 2. The inference time values of state-of-the-art CNN models.

Models (Backbone)	Epoch	Inference time
YoloF (yolof_r50_c5_8x8_1x_coco)	100	0.42
YoloF (yolof_r50_c5_8x8_1x_coco)	200	0.46
YOLOv3 (yolov3_d53_mstrain-416_273e_coco)	100	0.72
YOLOv3 (yolov3_d53_mstrain-416_273e_coco)	500	0.77
Sparse R-CNN (sparse_rcnn_r50_fpn_mstrain_480-800_3x_coco)	100	0.95
Sparse R-CNN (sparse_rcnn_r50_fpn_mstrain_480-800_3x_coco)	200	0.98
Faster R-CNN (faster_rcnn_r50_caffe_dc5_1x_coco)	100	1.12
Faster R-CNN (faster_rcnn_r50_caffe_dc5_1x_coco)	200	1.16

4. RESULTS AND DISCUSSION

Subsequently, in this study, a GUI was developed for the state-of-the-art models based application using PyQt5 which is one of Python's libraries. In the GUI screen shown in Figure 5, the image is first selected from the file system with the "Select Image" button. The "Select .pth" button enables selection of the file containing the weights of the best state-of-the-art model. The "Select config" button is used to load the configuration file. The program also detects bicycles on video. For this, the "Select video" button must be selected. After all selection processes are completed, the results can be viewed in real time on the right side of the screen.

**Figure 5.** GUI software developed for bicycle detection.

At later stages, alternatives to turn the developed software into a web-based or a mobile application can be considered.

5. CONCLUSIONS

In this study, a dataset was first created to solve the bicycle detection problem. In the next step, the bicycle images in the dataset were labeled. Then, the dataset was trained with state-of-the-art models. In addition, the four different models for bicycle image recognition were compared. First, 962 open-source bicycle images from the Internet were labeled. For this purpose, trainings

were performed with YOLOv3, YOLOF, Faster R-CNN and Sparse R-CNN architectures. As a result of the trainings, a value of 0.92 mAP was obtained with Faster R-CNN.

In this study, it is possible to detect cyclists from photos and videos with bicycle images. The detection process can be followed through a GUI. In the later stages of the study, real-time web applications can be developed. For example, images obtained with cameras from bike paths can be detected instantly with cards with high graphics processors such as Nvidia Jetson. Another option is to send images over the Internet to a local or Web-based server. The results can be evaluated in real time by running the detection process on the server.

7. ACKNOWLEDGEMENTS

This research was funded by the Scientific and Technology Research Council of Turkey (TUBITAK), under project name: TUBITAK 2209, 1919B012102097. The authors gratefully acknowledge the financial support provided by the TUBITAK.

REFERENCES

- [1] Kerr, J., Emond, J. A., Badland, H., Reis, R., Sarmiento, O., Carlson, J., ... & Natarajan, L. (2016). Perceived neighborhood environmental attributes associated with walking and cycling for transport among adult residents of 17 cities in 12 countries: the IPEN study. *Environmental health perspectives*, 124(3), 290-298. <https://doi.org/10.1289/ehp.1409466>
- [2] Tzelepi, M., & Tefas, A. (2017). Human crowd detection for drone flight safety using convolutional neural networks. In *2017 25th European Signal Processing Conference (EUSIPCO)* (pp. 743-747). IEEE. <https://doi.org/10.23919/EUSIPCO.2017.8081306>
- [3] Tao, C., Lu, K., & Cao, F. (2020). Multi-target Tracking with EmbedMask and LSTM Model Fusion. In *International Conference on Security, Privacy and Anonymity in Computation, Communication and Storage* (pp. 310-320). Springer, Cham. https://doi.org/10.1007/978-3-030-68884-4_26
- [4] Lin, Y. B., & Young, C. P. (2017). High-precision bicycle detection on single side-view image based on the geometric relationship. *Pattern Recognition*, 63, 334-354. <https://doi.org/10.1016/j.patcog.2016.10.012>
- [5] Leung, M. K., & Huang, T. S. (1990). Detecting wheels of vehicle in stereo images. In [1990] Proceedings. 10th International Conference on Pattern Recognition (Vol. 1, pp. 263-267). IEEE.
- [6] Beran, V., Herout, A., & Řezníček, I. (2009). Video-based bicycle detection in underground scenarios, in: *Proceedings of the 17th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, pp. 103–107.
- [7] Satyanarayana, G. S. R., Deshmukh, P., & Das, S. K. (2022). Vehicle detection and classification with spatio-temporal information obtained from CNN. *Displays*, 102294. <https://doi.org/10.1016/j.displa.2022.102294>
- [8] Taheri Tajar, A., Ramazani, A. & Mansoorizadeh, M. A lightweight Tiny-YOLOv3 vehicle detection approach. *J Real-Time Image Proc* 18, 2389–2401 (2021). <https://doi.org/10.1007/s11554-021-01131-w>
- [9] Vasilev, I., Slater, D., Spacagna, G., Roelants, P., & Zocca, V. (2019). *Python Deep Learning: Exploring deep learning techniques and neural network architectures with Pytorch, Keras, and TensorFlow*. Packt Publishing Ltd. Birmingham.
- [10] Mueller JP, Massaron L (2019) *Deep learning for dummies*. John Wiley & Sons, Canada.
- [11] LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86 (11):2278-2324
- [12] Krizhevsky A, Sutskever I, Hinton GE Imagenet classification with deep convolutional neural networks. In: *Advances in neural information processing systems*, 2012. pp 1097-1105
- [13] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
- [14] Çelik, A., & Uğuz, S. (2022). A deep learning based system for real-time detection and sorting of earthworm cocoons. *Turkish Journal of Electrical Engineering and Computer Sciences*, 30(5), 1980-1994. <https://doi.org/10.55730/1300-0632.3917>
- [15] Uğuz, S., & Uysal, N. (2021). Classification of olive leaf diseases using deep convolutional neural networks. *Neural Computing and Applications*, 33(9), 4133-4149. <https://doi.org/10.1007/s00521-020-05235-5>
- [16] Sharma, K., & Garg, V. K. (2018). Comparative analysis of vermicompost quality produced from rice straw and paper waste employing earthworm *Eisenia fetida* (Sav.). *Bioresource technology*, 250, 708-715.

- <https://doi.org/10.1016/j.biortech.2017.11.101>
- [17] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Fei-Fei, L. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3), 211-252. <https://doi.org/10.1007/s11263-015-0816-y>
- [18] de Aguiar, A. S. P., dos Santos, F. B. N., dos Santos, L. C. F., de Jesus Filipe, V. M., & de Sousa, A. J. M. (2020). Vineyard trunk detection using deep learning—An experimental device benchmark. *Computers and Electronics in Agriculture*, 175, 105535. <https://doi.org/10.1016/j.compag.2020.105535>