UDC 004.732.056 **doi:** https://doi.org/10.20998/2522-9052.2025.2.10

Svitlana Gavrylenko[1], Wang Zhihao[2]

[1] National Technical University "Kharkiv Polytechnic Institute", Kharkiv, Ukraine
[2] Chongqing Telecommunication Polytechnic College; Chongqing; China

PEDESTRIAN RED LIGHT TRAFFIC RECOGNITION MODEL BASED ON YOLOV8 ALGORITHM

Abstract. The object of the study is the process recognition of pedestrian red light traffic. **The subject of the study** are the methods of process recognition of pedestrian red light traffic. **The purpose of the paper** is to improve the efficiency of real-time pedestrian red light traffic recognition model. **The results obtained**. The pedestrian red light traffic recognition model based on real-time object detection architecture YOLOv8 was proposed. The architecture and characteristics of YOLOv8 model, including its improved network structure, multi-scale detection ability, and adaptive anchor adjustment were introduced in detail. To demonstrate the efficiency and benefits of applying the YOLOv8 model, its performance was evaluated in various scenarios. **Conclusions**. Experiments have confirmed the efficiency of the proposed method. The use of the developed method based on the YOLOv8 architecture allowed to increase precision up to 0.935. Overall, the average performance across all categories is 0.851, which means that the model has a relatively high detection accuracy. In addition, model has a high-speed index.

Keywords: deep learning network; YOLOv8 model; pedestrian red light traffic detection; road safety; intelligent transport system; object detection; performance evaluation.

Introduction

Traffic safety has always been a key public safety issue that needs to be solved urgently worldwide. Pedestrian running red lights is one of the main causes of traffic accidents in busy streets and intersections in cities. It not only threatens the safety of pedestrians, but also brings huge psychological and physical pressure to drivers, which increases the risk of traffic accidents. In addition, the behavior of pedestrians violating traffic signals can also lead to the chaos of traffic flow, which affects the overall efficiency and safety of urban traffic.

There are various reasons for pedestrians to run red lights, including pedestrians' ignorance of traffic rules, being in a hurry, or misunderstanding of traffic signals. In addition, imperfect traffic facilities and lack of effective supervision are also one of the important reasons for pedestrians running red lights. Therefore, it is of great significance to effectively identify and prevent pedestrian red light running behavior for improving traffic safety and optimizing traffic management.

Of course, the automatic recognition of pedestrian red light running behavior faces multiple challenges, including complex traffic environments, different lighting conditions, pedestrian occlusion problems, and diverse pedestrian behavior patterns [1, 2].

Early pedestrian recognition methods are mainly based on traditional image processing techniques and machine learning methods [3, 4]. These techniques usually involve steps such as preprocessing of images, feature extraction, and classifier design [5]. While these approaches have shown some success under certain conditions, they face a number of limitations when dealing with complex real-world scenarios:

- sensitive to environment: Early methods often depend on specific environmental conditions, such as illumination intensity, background complexity, etc. Environmental changes, such as dimming of light or cluttered background, can significantly reduce recognition accuracy;
- limited processing power: These methods are not effective in dealing with complex situations such as pedestrian occlusion, different pedestrian poses, and mutual occlusion between pedestrians. For example, when multiple pedestrians are close together or partially occluded in the image, it is difficult for traditional methods to accurately distinguish and identify each individual;
- limitations of feature selection and extraction: Early methods require manual design and selection of features, which is not only time-consuming and labor-intensive, but also feature selection is largely based on experience and may not be generalizable. In addition, manually extracted features may not adequately capture all important information about pedestrians, limiting the performance of the recognition system;
- real-time problem: Due to the high computational complexity, early methods are difficult to meet the requirements of real-time processing. In application scenarios such as traffic monitoring and safety management, real-time performance is an important indicator, which is of great significance for accident prevention and timely response;
- weak generalization: Early methods may perform well on the training set, but their performance tends to drop substantially when encountering new, unseen scenes or conditions. This is because hand-crafted features and rule-based methods lack sufficient generalization capabilities to cope with changeable real-world conditions.

Therefore, the improvement and development of new methods of recognizing pedestrians is a relevant task.

1. Traditional image processing methods

Traditional image processing method is the basis of pedestrian detection, which mainly includes image preprocessing, feature extraction and object detection [6]. Preprocessing aims to improve the image quality through filtering, denoising, brightness adjustment and other methods to provide a clearer image input for subsequent feature extraction and object detection [7, 8]. Feature extraction focuses on extracting effective features from preprocessed images, such as edges, corners, textures, etc., which are crucial for subsequent pedestrian recognition. According to the extracted features, object detection uses various algorithms (such as background subtraction, frame difference, etc.) to identify and locate the pedestrian in the image [9].

Feature extraction method is another important technology for early pedestrian recognition, which mainly includes Haar feature, Histogram of Oriented Gradients (HOG) feature [10] and SIFT (Scale-Invariant Feature Transform) [11] feature. These methods identify pedestrians by extracting and analyzing key features in images. For example, HOG features describe the shape information of pedestrians by analyzing the gradient direction distribution in an image, and are often used in combination with Support Vector Machine (SVM) [12, 13] classifier for pedestrian detection. Although feature extraction methods have made progress in improving the accuracy of pedestrian recognition, they still face problems such as slow processing speed and sensitivity to pedestrian pose variations and occlusions, which limit their application in real-time pedestrian recognition systems. Moreover, various real-time safety models of pedestrian red light running were developed based on the different combination of explanatory variables using the Bayesian Poisson-lognormal (PLN) model [14].

Deep learning methods have completely changed the development direction of pedestrian detection technology, especially the application of various network models including convolutional neural network (CNN) and recurrent neural network (RNN) in this field, which has greatly improved the accuracy and efficiency of detection [15, 16]. These methods can effectively identify and locate pedestrians in images by automatically learning and extracting complex features from image data, and maintain high accuracy even in complex environments [17, 18].

Deep learning models, especially convolutional neural networks (CNNS), have revolutionized the field of pedestrian detection and recognition. Its advantages are mainly shown in the following aspects:

1. Model extracts features automatically.
2. Model has demonstrated strong generalization ability by training on large datasets.
3. Efficient real-time processing.
4. Model can better deal with the mutual occlusion between pedestrians by learning a large number of pedestrian images with occlusion.
5. Model support end-to-end training, which means that the whole process from raw pixels to final detection results can be optimized through a unified framework.
6. Model can be designed to handle multiple related tasks at the same time (e.g., pedestrian detection and attribute recognition), which not only improves the efficiency of the model, but also further improves the ability of the model to handle complex scenes.

SSD (Single Shot MultiBox Detector) model uses a single convolutional neural network to simultaneously detect and classify objects. It's efficient for real-time applications but might have slightly lower accuracy compared to CNN models. An enhanced version of SSD is RetinaNet model [19]. It utilizes a pyramid network architecture and a new loss function, leading to more precise object localization [18, 19].

EfficientDet model prioritizes efficiency with a scalable depth and width architecture. It offers a good balance between speed and accuracy, making it suitable for mobile and cloud platforms [20].

CenterNet is a scale-invariant object detection model that directly predicts object centers and sizes. It's simple to implement and runs fast, but might have lower accuracy compared to some models [21].

DETR (DEtection and TRansformation) model uses a "point-to-point" approach for object detection. It achieves high accuracy and can perform additional tasks like object masking and tracking [22].

With the continuous development and optimization of deep learning technology, pedestrian detection models based on deep learning have more and more broad application prospects in traffic monitoring, intelligent transportation systems, and many other fields. The progress of these models is not only reflected in the improved recognition accuracy and real-time performance, but also in the ability to deal with complex scenes and diverse tasks.

2. YOLO model

Among the various object detection algorithms, YOLO (You Only Look Once) stands out for its remarkable balance of speed and accuracy, being able to quickly and reliably identify objects in an image [23, 24]. Since its inception, the YOLO family has gone through several iterations, each addressing limitations and improving performance over previous versions [25, 26].

The core principle of the YOLO model is to cast the object detection task as a single regression problem, which predicts the bounding boxes and class probabilities of the object directly from the image. Unlike traditional object detection methods, YOLO implements this process through a single deep learning model, which greatly improves the speed of detection while maintaining a high accuracy rate (Fig. 1).

As shown in Fig. 1, the basic principle of the YOLO model can be divided into the following key steps:

1. Image Segmentation: YOLO splits the input image into small cells (usually an S×S grid). Each cell is responsible for predicting the target whose center point falls within that cell.
2. Bounding Box Prediction: For each cell, YOLO predicts multiple bounding boxes and their corresponding confidence scores. The confidence reflects the probability that the bounding box contains the target and how well the predicted box matches the actual box.
3. Class Probability Prediction: Each cell is also responsible for predicting the probability that the object in the image belongs to each class. This probability is calculated on a global scale, independent of the number of bounding boxes.
4. Non-maximum Suppression (NMS): Since each cell predicts multiple bounding boxes, we end up with a large number of overlapping boxes. YOLO uses non-maximum suppression to merge overlapping boxes, ensuring that only one bounding box corresponds to each object.

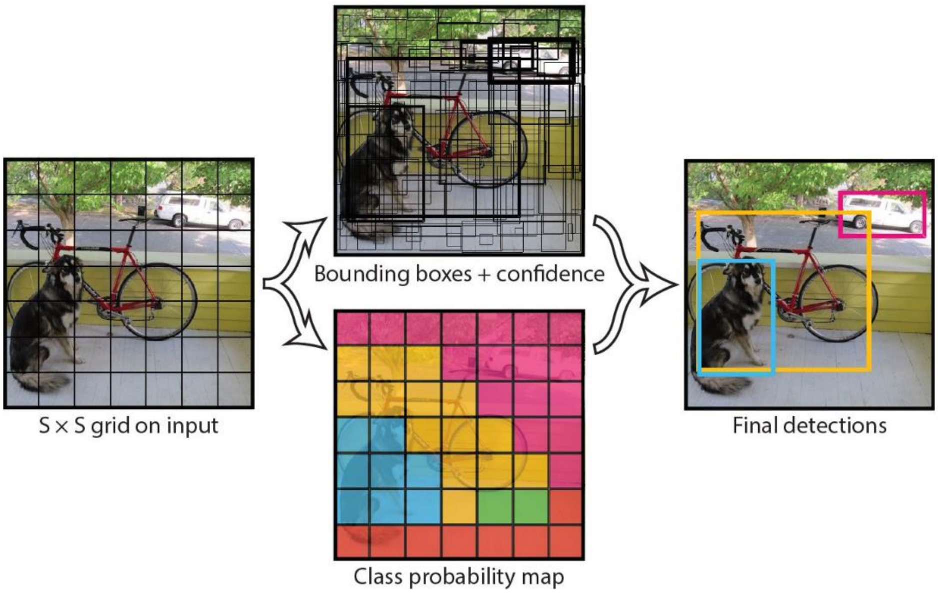

Fig. 1. YOLO model prediction diagram

5. Overall prediction: The YOLO model combines all the above predictions to output the location, class, and confidence score for each object in the image.

YOLO uses a convolutional network to extract features, and then uses a fully connected layer to obtain the prediction value. The network structure refers to the GoogLeNet model and contains 24 convolutional layers and 2 fully connected layers, as shown in Fig. 2. For convolutional layers, 1x1 convolutions are mainly used for channel reduction, followed by 3x3 convolutions.

For convolutional and fully connected layers, Leaky ReLU activation function: max (x, 0, 1) is used.

However, the last layer uses a linear activation function.

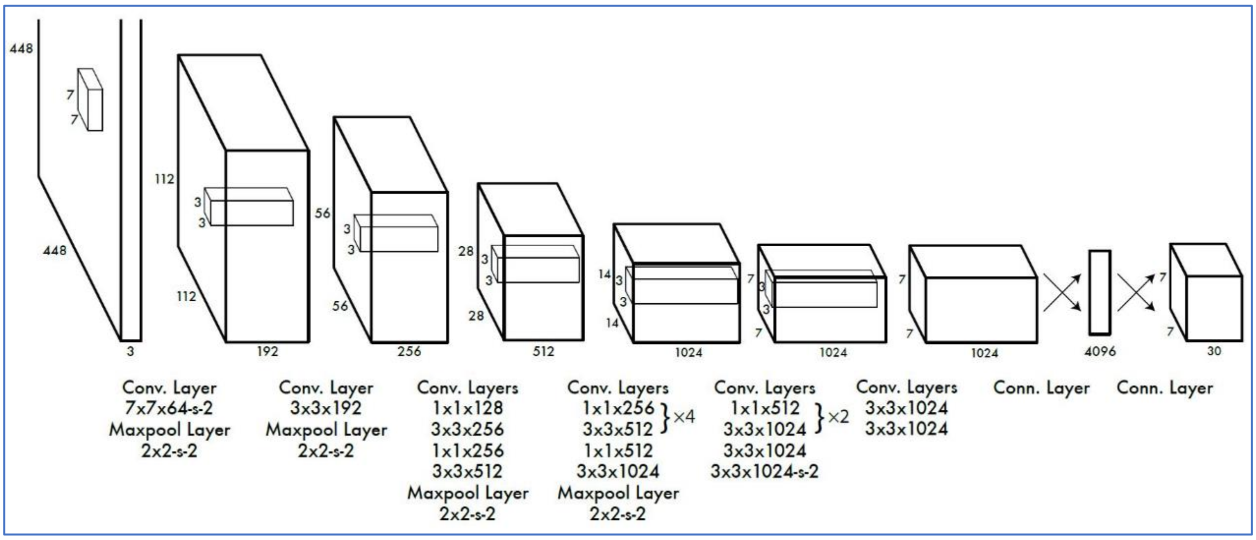

Fig. 2. Schematic diagram of the YOLO network structure

There have been eight updates for this computer vision model since its release; YOLOv1 – YOLOv8.

3. YOLOv8 model

YOLOv8 is a cutting-edge, state-of-the-art (SOTA) model that builds upon the success of previous YOLO versions and introduces new features and improvements to further boost performance and flexibility. YOLOv8 is designed to be fast, accurate, and easy to use, making it an excellent choice for a wide range of object detection and tracking, instance segmentation, image classification and pose estimation tasks [27, 28].

The key features of YOLOv8 are:

– improved network architecture: YOLOv8 adopts a new network architecture design, adds more depth and width, and introduces an attention mechanism and a more efficient feature extraction layer to improve the model's perception of target features;

– multi-scale prediction: Inherits the multi-scale prediction feature of YOLO series, YOLOv8 can detect objects at different scales, which helps to improve the detection accuracy of the model for small objects;

– adaptive anchor matching: YOLOv8 automatically adjusts the size and proportion of anchors through an improved anchor matching mechanism to better adapt to targets of different shapes and sizes and reduce the need for manually setting anchors;

– advanced Data Augmentation: We introduce more advanced data augmentation techniques such as AutoAugment to enhance model generalization and robustness;

– optimized loss function: YOLOv8 uses a new loss function to balance the classification error, positioning error and confidence error more finely to improve the overall performance of the model.

YOLOv8 has shown excellent performance on multiple public object detection datasets, especially in dealing with complex scenes and small object detection. Its accuracy is significantly improved compared with the previous generation model. Although the network structure is more complex, YOLOv8 still maintains good real-time processing ability and meets the needs of real-time object detection. On modern Gpus, YOLOv8 can achieve high frame rate processing speed. Through testing on diverse datasets, YOLOv8 proves its excellent generalization ability and is able to adapt to different application scenarios and environmental conditions

Compared with the previous version, YOLOv8 not only maintains a high detection accuracy, but also improves the running speed. For practical applications, YOLOv8 is an attractive choice, especially in scenarios that require high accuracy and efficiency, such as pedestrian red light recognition and analysis scenarios.

4. Realization method of pedestrian running red light recognition

4.1. Data preparation and preprocessing. Data collection is the first step in the establishment of a pedestrian red light recognition system, which is crucial for training an efficient and accurate deep learning model [29, 30].

The data of pedestrian red light recognition systems are mainly derived from traffic surveillance cameras, which are usually installed at traffic intersections, crosswalks, or other important traffic nodes. To ensure that the model generalizes well, the following diversification conditions should be considered in data collection. Firstly, it’s weather conditions which directly affect the illumination, contrast and visibility of the image, and thus have a large impact on the model. Secondly, the different light conditions in different time periods, especially the low light conditions at night, are a challenge to the detection ability of the model. Finally, there is traffic flow. Peak hours and off-peak hours. Different traffic flows affect the density of pedestrians and vehicles, which in turn affects the ability of the model to deal with pedestrian occlusion situations. LabelMe software was used for data annotation, which supports tasks such as object detection, semantic segmentation, and instance segmentation

Data annotation includes: person localization, zebra crossing localization, Traffic light location. The purpose of Person localization step is to learn how to recognize pedestrians from complex traffic scenes. Location of the zebra crossing is annotated to help the model understand the relationship between the zebra crossing and the pedestrian location, so as to determine whether the pedestrian is within the legal crossing area. The location and state of traffic lights (red or green) are clearly defined, which is crucial to determine whether a pedestrian has run a red light.

Data augmentation is the next step: In order to improve the robustness of the model, the dataset is usually augmented, including random cropping, rotation, flipping, changing brightness and contrast

Before feeding into the model, the data needs to be preprocessed, such as resizing the image to fit the model input requirements, normalizing the pixel values, etc.

4.2. Dataset partitioning. The 320 labeled image data were divided into training set and validation set in proportion, in which the training set accounted for 80% and the validation set accounted for 20%. The data set was automatically randomly divided by writing a python program, and the partition results are shown in Table 3.2.

The training set is shown in Fig. 3 with a total of 256 images, and the validation set has a total of 64 images.

Fig. 3. Training set

4.3. Model training. The prepared YOLOv8 project is uploaded to Collaboratory, and the train.py code is written to efficiently complete the model training of the pedestrian running red light dataset. We use the model.train method with multiple parameters:

—data='pedestrian.yaml': This specifies a YAML configuration file containing information about the dataset. This file defines the paths for the training and validation datasets, class information, etc.

—epochs=200: This sets the number of times the training process will go through the dataset to 200.

—imgsz=640: Sets the size of the input image to 640 pixels. This is the image resolution accepted by the input layer of the model.

— device=0: This specifies the device used for training, where 0 is usually the first GPU

—batch=32: This sets the batch size to 32. This means that each training iteration will process 32 images simultaneously.

Here, we use the pre-trained yolov8n.pt model file, initialize the YOLO model, set the task to object detection, and then perform the detection on the specified bus.jpg image (Fig. 4).

Fig.4. Object detection

4.4. Model evaluation and optimization strategies. A summary of the results after model training is completed is shown in Fig. 5.

```
200 epochs completed in 0.159 hours.
Optimizer stripped from runs/detect/train2/weights/last.pt, 6.3MB
Optimizer stripped from runs/detect/train2/weights/best.pt, 6.3MB

Validating runs/detect/train2/weights/best.pt...
Ultralytics YOLOv8.1.0 🚀 Python-3.10.12 torch-2.1.0+cu121 CUDA:0 (Tesla V100-SXM2-16GB, 16151MiB)
YOLOv8n-pedestrian summary (fused): 168 layers, 3006428 parameters, 0 gradients
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1
[00:00<00:00,  2.92it/s]
                   all         64        178      0.841      0.814      0.851      0.606
                   red         64         55      0.935       0.78      0.842      0.553
                 green         64         22      0.985      0.818      0.914      0.543
            pedestrian         64         49      0.631      0.735      0.723      0.528
             crosswalk         64         52      0.814      0.924      0.927      0.802
Speed: 0.1ms preprocess, 0.6ms inference, 0.0ms loss, 0.8ms postprocess per image
```

Fig. 5. Summary of model results

4.5. Summary of YOLOv8 model. YOLOv8n-pedestrian summary (fused): 168 layers, 3006428 parameters, 0 gradients: Shows the structural summary of the model, including the number of layers, the total number of parameters, and the number of gradients.

4.5.1. Performance index. For all classes in the dataset (all), and for individual classes (red, green, pedestrian, crosswalk), the number of Images (Images), the number of Instances (Instances), and the following performance metrics are listed:

– Box(P): Precision of the bounding box;
– R: Recall;
– mAP50: Mean Average Precision at IoU=0.5;
– mAP50-95: Mean average precision over IoU ranging from 0.5 to 0.95.

For example, for the red class, with 64 images containing 55 instances:

– the precision is 0.935,
– the recall is 0.78,
– the mAP50 is 0.842, and the MAP50-95 is 0.553.

4.5.2. Index of speed. Speed: The preprocessing time (0.1ms), inference time (0.6ms), loss calculation time (0.0ms), and post-processing time (0.8ms) are given for each image.

Overall, this summary shows the training and validation process of the model, providing detailed information on comprehensive performance evaluation, which is very useful for understanding how the model performs on a specific task.

In Fig. 6, you can see that both training loss (top row) and validation loss (bottom row) show a decreasing trend, while the performance metrics (precision, recall, and mAP) show an increasing trend as training progresses.

This is often a sign of improved model performance during training. However, it should be noted that the validation loss seems to have a slight upward trend in the later stages of training, which is an indication of overfitting.

In this case, further regularization is required, or the training is terminated early to prevent overfitting. Furthermore, precision and recall tend to be stable, which means that the model is close to its performance limit on the current architecture and dataset.

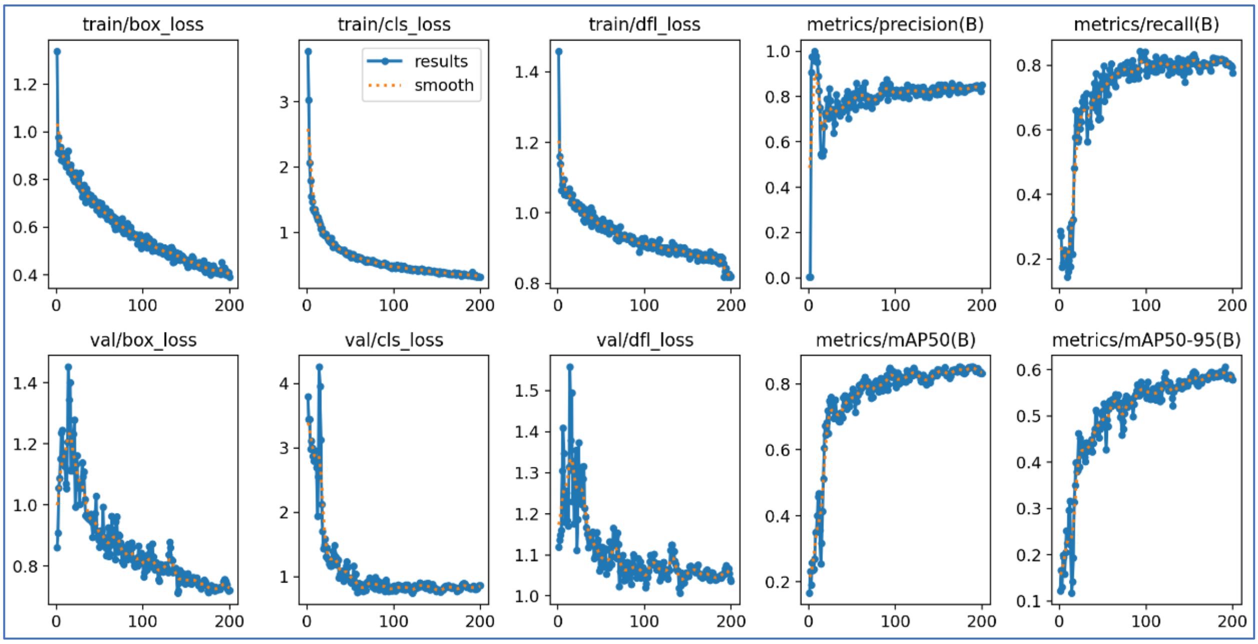

Fig. 6. Performance index

4.5.3. PR curve. In Fig. 7, crosswalk performs best with its PR curve closest to the top right corner, while pedestrian performs relatively poorly with its PR curve farther away from the top right corner.

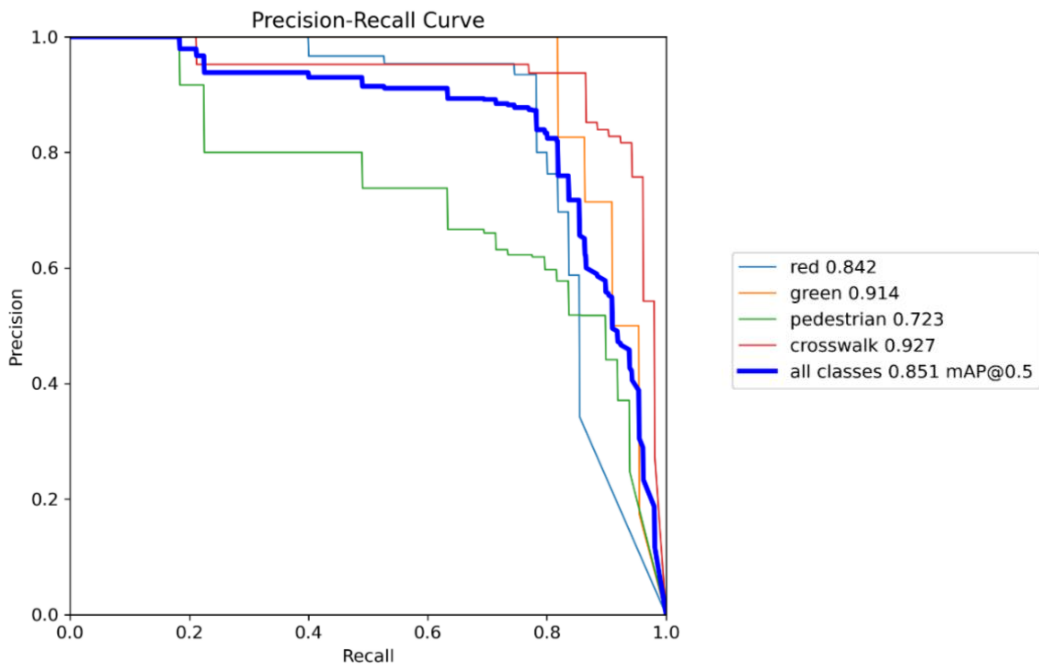

Fig 7. PR_curve

Overall, the average performance across all categories is 0.851, meaning that the model has a relatively high detection accuracy. When evaluating a model, in addition to looking at the performance of each category individually, it is important to consider the average performance across all categories, as this provides a view of the overall performance of the model.

Combined with the above results, here are some optimization strategies to improve the performance of YOLOv8 model on pedestrian traffic light dataset:

– Loss balancing: Since the validation loss fluctuates or rises slightly late in training, more meticulous learning rate scheduling or early stopping may be required to avoid overfitting.

– More data or data augmentation: If the decrease in the model's loss levels off, it probably means that the model has learned as much information as possible from the available data. Consider adding more training data or using more data augmentation tricks.

– Regularization: Increase model regularization (e.g. L1/L2 regularization or Dropout), especially if there is a large difference between training and validation loss.

– Improving detection performance for some categories: The mAP of pedestrian categories is relatively low, and more pedestrian samples or improved data augmentation strategies for pedestrians may be needed.

– Handling class imbalance: For classes with poor classification performance (such as pedestrians), resampling techniques can be tried to increase the frequency of these classes in the training data, or to give more weight to these classes in the loss function.

– Threshold Optimization: Adjusting the threshold used for classification can help find a better balance between precision and recall in some classes.

4.6. Comparison between YOLOv8 and Faster R-CNN. YOLOv8 and Faster R-CNN are both single-stage object detection algorithms, but their algorithm architectures are different. Faster R-CNN adopts a two-stage detection process, including a region proposal network (RPN) and an object classification network. RPN generates candidate regions, and then the object classification network classifies these candidate regions and regress bounding boxes.

In contrast, YOLOv8 employs a one-stage detection flow that directly maps the input image to bounding boxes and class probabilities. It uses a single network to perform object detection without generating candidate regions.

A comparison of algorithm architectures is shown in Tabl. 1.

The performance of the developed YOLOv8 model is compared with Faster R-CNN is shown Table 2 for details.

Table 1 – **Comparison of YOLOv8 and Faster R-CNN algorithm architectures**

Feature	**YOLOv8**	**Faster R-CNN**
Detection process	Single stage	Two-stage
Proposals	*Without*	RPN
Network structure	*Single network*	RPN+classification network

Table 2 – **Performance comparison of YOLOv8 and Faster R-CNN**

Quality indicators	**YOLOv8**	**Faster R-CNN**
Accuracy	0.851	0.87
Detection speed, ms	**1.5**	4.2
Memory, MB	**6.3**	14.5

The detection speed of YOLOv8 is faster because it uses a single-stage detection process without generating candidate regions. Faster R-CNN has slightly higher detection accuracy because it uses a two-stage detection flow, which can perform finer classification and regression of candidate regions.

5. Results analysis and discussion

With the increase of urban traffic density, pedestrian running red light has become a serious traffic safety problem. This paper proposes pedestrian red light traffic recognition model based on real-time object detection algorithmYOLOv8. YOLOv8 improves previous YOLO versions in several aspects, including faster detection speeds, higher accuracy, and better handling of overfitting issues.

This is achieved through a novel anchor-free approach, introducing a new "Mish" activation function, and incorporating spatial and channel attention mechanisms.

The architecture and characteristics of YOLOv8 model in detail, including its improved network structure, multi-scale detection ability, and adaptive anchor adjustment were introduced.

To demonstrate the effects and benefits of applying the YOLOv8 model, its performance was evaluated in various scenarios.

It was found that the use of the YOLOv8 model allows to increase precision up to 0.935. Overall, the average performance across all categories is 0.851, meaning that the model has a relatively high detection accuracy.

In addition, model has high Index of speed:

– the preprocessing time (0.1ms),

– inference time (0.6ms),

– loss calculation time (0.0ms),

– post-processing time (0.8ms for each image).

When a computer's GPU memory is low, the inference speed of the system may suffer, mainly because the GPU memory is not enough to hold large models and data, resulting in memory swapping and other additional computational overhead that slows down inference.

Here are some specific cases that can cause slow inference and how to solve them:

– memory consumption when the model is loaded. Solution: Use lightweight models, model compression techniques, distributed training, etc.;

– memory footprint of input data. Solution: Reduce the size of the input data or the batch size to reduce the memory footprint;

– memory consumption during model inference. Solutions: Optimize the model architecture, reduce the depth and width of the computation graph, use the memory optimization capabilities provided by deep learning frameworks, etc.

In the future, in the field of pedestrian red light traffic recognition based on YOLOv8 model, in-depth research can be carried out through the following aspects:

– further improve the real-time and accuracy of the model, and improve the performance of the model in complex scenes by optimizing the model structure, adjusting parameters, and using hardware acceleration.

– explore new methods such as multimodal information fusion, self-supervised learning, and reinforcement learning to improve the robustness and generality of pedestrian red light recognition systems and adapt to more diverse traffic environments.

– strengthen research on privacy protection and data security, and design a safe and reliable pedestrian red light recognition system to ensure the security and privacy of user information.

– carry out the collection and annotation of large-scale data sets combined with practical application scenarios, so as to provide more abundant and diverse data support for the further development of pedestrian red light recognition technology.

Through continuous research and efforts, the pedestrian red light recognition technology based on YOLOv8 model will play an increasingly important role in the field of traffic safety, and make a positive contribution to the realization of smart city traffic management and improve the level of traffic safety.

References

1. Deng, J, Xuan, X, Zhao, W., Yao, L. H. and Wang, Z. (2020), “A review of research on object detection based on deep learning”, *J. Phys*: *Conf Ser*., vol. 1684, 012028, doi: https://doi.org/10.1088/1742-6596/1684/1/012028
2. Shigematsu, K., Konishi, Y., Mitsudome, R. and Tsubouchi, T. (2018), “Recognition of Pedestrian Traffic Light at Crosswalk for a Mobile Robot Using Deep Learning”, *J-STAGE*, vol. 54, is. 1, pp. 99–110, doi: https://doi.org/10.9746/sicetr.54.99
3. Guo, T., Huynh, C. P. and Solh, M. (2019), “Domain adaptive pedestrian detection in thermal images”, *2019 IEEE International Conference on Image Processing*, pp. 1660–1664, doi: https://doi.org/10.1109/ICIP.2019.8803104
4. Gavrylenko, S., Chelak, V. and Hornostal, O. (2022), “Construction Method of Fuzzy Decision Trees for Identification the Computer System State”, *32nd International Scientific Symposium Metrology and Metrology Assurance, MMA 2022*, doi: https://doi.org/10.1109/MMA55579.2022.9992878
5. Hornostal, O. and Gavrylenko, S. (2023), “Application of heterogeneous ensembles in problems of computer system state identification”, *Advanced Information Systems*, vol. 7, no. 4, pp. 5–12, doi: https://doi.org/10.20998/2522-9052.2023.4.01
6. Svyrydov, A., Kuchuk, H. and Tsiapa, O. (2019), “Improving efficienty of image recognition process: Approach and case study”, *Proceedings of 2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies*, DESSERT 2018, pp. 593–597, doi: https://doi.org/10.1109/DESSERT.2018.8409201
7. Gavrylenko, S. and Hornostal, O. (2023), “Study of methods for improving the meta-algorithm of the bagging classifier”, *2023 IEEE 4th KhPI Week on Advanced Technology, KhPI Week 2023 - Conference Proceedings*, doi: https://doi.org/10.1109/KhPIWeek61412.2023.10312977
8. Kuchuk, H. and Malokhvii, E. (2024), “Integration of IoT with Cloud, Fog, and Edge Computing: A Review”, *Advanced Information Systems*, vol. *8*, no. 2, pp. 65–78, doi: https://doi.org/10.20998/2522-9052.2024.2.08
9. Rasouli, A. and Tsotsos, J. K. (2020), “Autonomous Vehicles That Interact With Pedestrians: A Survey of Theory and Practice”, *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 900–918, doi: https://doi.org/10.1109/TITS.2019.2901817
10. Dalal, N. and Triggs, B. (2005), “Histograms of oriented gradients for human detection”, *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, San Diego, CA, USA, vol. 1, pp. 886-893, doi: https://doi.org/10.1109/CVPR.2005.177 .
11. Nguyen, M. T., Nguyen V. D. and Jeon, J. W. (2017), “Real-time pedestrian detection using a support vector machine and stixel information”, *2017 17th International Conference on Control, Automation and Systems*, ICCAS, Jeju, Korea (South), pp. 1350–1355, doi: https://doi.org/10.23919/ICCAS.2017.8204203 .
12. Abid, N., Ouni, T., Ammari, A.C. and Abid, M. (2022), “Efficient and high-performance pedestrian detection implementation for intelligent vehicles”, *Multimedia Systems*, vol. 28, pp. 69–84, doi: https://doi.org/10.1007/s00530-021-00799-1
13. Hunko, M., Tkachov, V., Kuchuk, H. and Kovalenko, A. (2023), “Advantages of Fog Computing: A Comparative Analysis with Cloud Computing for Enhanced Edge Computing Capabilities”, *2023 IEEE 4th KhPI Week on Advanced Technology, KhPI Week 2023 - Conference Proceedings*, code 194480, doi: https://doi.org/10.1109/KhPIWeek61412.2023.10312948
14. Wu, Y., Guo, Y. and Yin, W. (2021), “Real Time Safety Model for Pedestrian Red-Light Running at Signalized Intersections in China”, *Sustainability*, vol. 13(4), 1695, doi: https://doi.org/10.3390/su13041695
15. Srivastava, S., Divekar, A.V., Anilkumar, C., Naik, I., Kulkarni, V. and Pattabiraman, V. (2021), “Comparative analysis of deep learning image detection algorithms”. *J Big Data*, vol. 8, article 66, doi: https://doi.org/10.1186/s40537-021-00434-w
16. Kim, J., Park, S. and Ro, Y. (2022), “Uncertainty guided cross-modal learning for robust multispectral pedestrian detection”, *IEEE Trans. on Circuits and Systems for Video Techn.*, pp. 1510–1523, doi: https://doi.org/10.1109/TCSVT.2021.3076466 .

17. Yaloveha V., Hlavcheva D., and Podorozhniak A. (2021), "Spectral Indexes Evaluation for Satellite Images Classification using CNN", *Journal of Inf. and Organizational Sciences*, vol. 46, no. 2., pp. 95–113, doi: https://doi.org/10.31341/jios.45.2.5
18. Panigrahi, S. and Raju, U. S. N. (2021), "An improved Faster RCNN for Pedestrian Detection", *2021 Int. Conf. on Control, Automation, Power and Signal Processing*, Jabalpur, India, pp. 1–6, doi: https://doi.org/10.1109/CAPS52117.2021.9730492
19. Huang, L., Wang, Z. and Fu, X. (2024), "Pedestrian detection using RetinaNet with multi-branch structure and double pooling attention mechanism", *Multimed Tools Appl 83*, pp. 6051–6075, doi: https://doi.org/10.1007/s11042-023-15862-4
20. Zhang, C., Yu, M., Min, J., Fu, M. and Zheng, B. (2023), "Pedestrian detection based on improved Efficient Det", *PREPRINT* (Version 1) available at Research Square, doi: https://doi.org/10.21203/rs.3.rs-3611172/v1
21. Fu, X., Wang, Z., Huang, L., Wang, Y. and Zhang, C. (2022), "Pedestrian detection method based on feature fusion Centernet", *2022 IEEE 6th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, Beijing, China, pp. 1834–1838, doi: https://doi.org/10.1109/IAEAC54830.2022.9929508.
22. Gavrylenko, S., Poltoratskyi, V. and Nechyporenko, A. (2024), "Intrusion detection model based on improved transformer", *Advanced Information Systems*, vol. 8, no. 1, pp. 94–99, doi: https://doi.org/10.20998/2522-9052.2024.1.12
23. Ahmad, T., Ma, Y., Yahya, M., Ahmad, B. and Nazir, S. (2020), "Object detection through modified YOLO neural network", *Scientific Programming*, vol. 2020, is. 1, 8403262, doi: https://doi.org/10.1155/2020/8403262
24. Xu, M., Wang, Z., Liu, X., Ma, L. and Shehzad, A. (2022), "An efficient pedestrian detection for real time surveillance systems based on modified YOLOv3". *IEEE J. Radio Freq*., id. 6, pp. 972–976, doi: https://doi.org/10.1109/JRFID.2022.3212907
25. Redmon, J. and Ali, F (2018), "YOLOv3: an incremental improvement", *arXiv*, preprint arXiv:1804.02767, doi: https://doi.org/10.48550/arXiv.1804.02767
26. Zhou, J., Yang, D., Song, T., Ye, Y., Zhang, X. and Song, Y. (2024), "Improved YOLOv7 models based on modulated deformable convolution and swin transformer for object detection in fisheye images", *Image and Vision Computing*, vol. 144, April 2024, doi: https://doi.org/10.1016/j.imavis.2024.104966
27. Sohan, M., Sai Ram, T. and Rami Reddy, C.V. (2024), "A Review on YOLOv8 and Its Advancements", In: Jacob, I.J., Piramuthu, S., Falkowski-Gilski, P. (eds) Data Intelligence and Cognitive Informatics. ICDICI 2023, *Algorithms for Intelligent Systems*, Springer, Singapore, doi: https://doi.org/10.1007/978-981-99-7962-2_39
28. Bilous, N., Malko, V., Frohme, M. and Nechyporenko, A. (2024), "A. Comparison of CNN-Based Architectures for Detection of Different Object Classes", *AI*, vol. 5, pp. 2300–2320, doi: https://doi.org/10.3390/ai5040113
29. Petrovska, I., Kuchuk, H., Kuchuk, N., Mozhaiev, O., Pochebut, M. and Onishchenko, Yu. (2023), "Sequential Series-Based Prediction Model in Adaptive Cloud Resource Allocation for Data Processing and Security", *2023 13th International Conference on Dependable Systems, Services and Technologies*, DESSERT 2023, 13–15 October, Athens, Greece, code 197136, doi: https://doi.org/10.1109/DESSERT61349.2023.10416496
30. Gavrylenko, S., Chelak, V. and Hornostal, O. (2020), "Kesearch of Intelligent Data Analysis Methods for Identification of Computer System State", *30th International Scientific Symposium Metrology and Metrology Assurance*, MMA 2020, 9254252, doi: https://doi.org/10.1109/MMA49863.2020.9254252

Received (Надійшла) 11.12.2024
Accepted for publication (Прийнята до друку) 02.04.2025

ВІДОМОСТІ ПРО АВТОРІВ/ ABOUT THE AUTHORS

Гавриленко Світлана Юріївна – доктор технічних наук, професорка, професорка кафедри "Комп'ютерна інженерія та програмування", Національний технічний університет "Харківський політехнічний інститут", Харків, Україна;
Svitlana Gavrylenko – Doctor of Technical Science, Professor, Professor of Department of "Computer Engineering and Programming", National Technical University «Kharkiv Polytechnic Institute», Kharkiv, Ukraine;
e-mail: gavrilenko08@gmail.com ; ORCID Author ID: https://orcid.org/0000-0002-6919-0055;
Scopus ID: https://www.scopus.com/authid/detail.uri?authorId=57189042150.

Чжихао Ван – магістр комп'ютерної інженерії, викладач кафедри «Інженерія зв'язку та промислового Інтернету», Чунцинського політехнічного коледжу телекомунікацій; Чунцин; Китай;
Wang Zhihao – Master of Computer Engineering, Lecturer of Department of "Communication Engineering and Industrial Internet", Chongqing Telecommunication Polytechnic College; Chongqing; China;
e-mail: clayrin25@gmail.com; ORCID Author ID: https://orcid.org/0009-0006-4701-3003.

Модель розпізнавання руху пішоходів на червоне світло на основі алгоритму YOLOv8

С. Ю. Гавриленко, Ван Чжихао

Анотація. Об'єктом дослідження є процес розпізнавання руху пішоходів на червоне світло. **Предметом дослідження** є методи розпізнавання руху пішоходів на червоне світло. **Метою роботи** є підвищення ефективності моделі розпізнавання руху пішоходів на червоне світло в режимі реального часу. **Отримані результати.** Запропоновано модель розпізнавання руху пішоходів на червоне світло на основі архітектури виявлення об'єктів у реальному часі YOLOv8. Детально представлено архітектуру та характеристики моделі YOLOv8, включаючи покращену структуру її мережі, здатність до багато-масштабного виявлення та адаптивного налаштування зв'язку. Ефективність застосування моделі YOLOv8 було оцінено за допомогою різних налаштувань моделі. **Висновки**. Дослідження підтвердили ефективність запропонованого методу. Використання розробленого методу на основі архітектури YOLOv8 дозволило підвищити влучність до 0,935. Загалом середня продуктивність усіх категорій якості моделі становить 0,851, що означає, що модель має відносно високу точність виявлення. Крім того, модель є більш оперативною у порівнянні з іншими методами.

Ключові слова: мережа глибокого навчання; модель YOLOv8; виявлення руху пішоходів на червоне світло; безпека дорожнього руху; інтелектуальна транспортна система; виявлення об'єктів; оцінка продуктивності.