Computers and Artificial Intelligence; Vol.3, No.4; 2026
E-ISSN: 3006-3728 P-ISSN: 3006-371X
DOI: https://doi.org/10.70267/cai.v3n4.8298
Published by: Zeus Press

Traffic Signal Detection Algorithm Based on Dual-GAM Improved YOLO26 under Nighttime Conditions

Chongtao Hu*

College of Engineering, Huazhong Agricultural University, Wuhan, Hubei, 430070, China

**Corresponding author: Chongtao Hu.*

Abstract

The environmental recognition of autonomous vehicles depends much on real-time object detection, and identifying small traffic signals in low-light conditions during the night is a difficult problem. The size of traffic signals is small and vulnerable to the disruption caused by street lights and the lights of vehicles, which requires high-quality feature representation capability of detectors. To overcome such a challenge, this paper presents the Global Attention Mechanism (GAM) to YOLO26 and suggests a better algorithm, YOLO26-GAM. Initially, YOLO26 is chosen as the base by comparative experiments between various YOLO-series models. Next, Grad-CAM is applied to view the basis on which the model makes decisions, and GAM modules are added at the H12 and H15 locations in the neck network. Cross-experiments and ablation experiments are done to test the effectiveness of the attention mechanism and the insertion positions. Experimental findings on the curated TN-TLD nighttime traffic light dataset demonstrate that YOLO26-GAM performs the best of all the experiments with a Precision of 0.964, Recall of 0.947, mAP50 of 0.966, mAP50-95 of 0.657, and F1 of 0.955. It is the only configuration that is superior to the baseline, without the attention mechanism, in all evaluation metrics, confirming the efficiency of the proposed method in detecting nighttime traffic signals.

Keywords

nighttime traffic light detection, YOLO26, attention mechanism, global attention mechanism (GAM)

1. Introduction

With the rapid growth of the global population, there are serious problems in the traditional transportation systems, including traffic congestion and high frequency of traffic accidents. Based on the information provided by the World Health Organization, around 1.3 million people die every year in traffic accidents [1] and the overwhelming number of accidents is due to the inability to notice pedestrians, cyclists, or obstacles early enough, or to read the road signs incorrectly. Conversely, visual recognition, using multimodal perception (visible light, infrared, event cameras, etc.) It can function under conditions where human vision fails, such as darkness, backlight, and other conditions, and it does not suffer from fatigue or distraction [2]. It is thus anticipated that intelligent driving will have a huge part to play in enhancing traffic efficiency levels and guaranteeing the safety of night traffic. The high rate of development of intelligent transportation systems and autonomous driving technologies has made the issue of reliable and real-time nighttime traffic signal detection relevant in the sphere of intelligent driving [3].

When analyzing the technological development of this branch, visual detection techniques can be roughly classified into conventional and deep learning-based ones. Conventional visual detection methods are based on the classical image processing and pattern recognition theory, which mostly uses low-level features, including color, shape, and edge, to build detection models with the help of existing knowledge. Images were transformed by Charette et al., into HSV or Lab color space, and candidate regions were obtained by thresholding the important differences between red and green lights in certain channels[4]. The Hough transform was applied by Gomez-Moreno et al. [5] to identify the circular lamp bodies. Arrow shapes and sign edges were detected using template matching and contour analysis by A. De la Escalera et al. [6]. These methods may be effective in particular cases, but traditional methods are very dependent on manually crafted features and parameters and their generalization capabilities and detection performance usually deteriorate considerably in case of complex backgrounds, bad weather, or high intra-class variance.

Traffic signal detection has experienced a paradigm shift due to the emergence of deep learning. Convolutional neural network-based object detection models are able to learn hierarchical representations in the data automatically using large-scale data, which will increase the robustness and accuracy of the model. The fundamental concept behind two-stage frameworks is that they create a set of candidate regions first, followed by a classification and regression of each region. After Girshick et al. introduced R-CNN (Region-based Convolutional Neural Network), which was the first to apply CNNs to the problem of object detection [7], other two-stage detectors like Fast R-CNN and Faster R-CNN have been developed with higher accuracy and faster execution times. In the paper by Kim et al. [8], it was found that three deep learning network architectures were compared and it was found that RGB input with Faster R-CNN is more beneficial to detect traffic lights. The two-stage CNN methods are highly precise and recall, yet their inference rate is lower than that of the later single-stage networks.

In contrast to the two-stage paradigm, one-stage detection models such as You Only Look Once (YOLO) [9] and Single Shot MultiBox Detector (SSD) [10] remove the individual candidate region generation process and instead perform direct regression on the bounding box and class probabilities of the target on the feature maps, which is significantly faster at inference with comparable accuracy. The method proposed by Liu et al. [10] is a representative example of this category of methods. The prediction on feature maps of different scales enables the effective detection of targets of various sizes, and the network architecture is fully convolutional. Müller and Dietmayer [11] implemented the SSD framework directly on the detection and classification of traffic signals, and they employed multi-scale feature maps to address light sources of varying distances, which satisfies the low latency demands of autonomous vehicle perception systems. Nevertheless, SSD is based on shallower feature layers to identify objects, so it is hard to effectively tell apart signal-light-like interference in the background (e.g., taillights, and neon lights) and actual traffic signals [10], and it has poor results in detecting small objects.

YOLO is also a typical representative of the one-stage detection paradigm and was originally presented by Redmon et al. [9] in 2016. The main point is that it is possible to represent object detection as a regression task where bounding box positions and class probabilities are predicted directly through the output layer, hence the inference time of YOLO is much faster. Since YOLOv3, an FPN-like architecture has been used to make predictions at three scales, which enhanced the recall of far and low-resolution traffic signals by orders of magnitude [12]. Other variants like YOLOv4/v5 implemented CSPNet, PANet, the Mish activation function, and other training techniques and achieved more than 100 fps on a single GPU with a similar or even better mAP than two stage methods [13]. S. Du et al added SPD (Space-to-Depth), SK (Select Kernel), and WIoUv3 modules to YOLOv8s which can be effective in enhancing the performance of small traffic sign detection [14]. S. Chakrabarty noted that YOLO26 uses a new architecture and is faster and more accurate in terms of inference and detection than previous YOLO versions [15]. A systematic analysis by Jensen et al. on the Bosch small traffic signal dataset showed that despite the impressive speed of YOLOv2, its detection quality at small targets and changing lighting conditions is extremely low compared to Faster R-CNN and SSD [16]. Hence, this paper suggests an enhanced algorithm, YOLO26-GAM, to detect nighttime traffic signals through enhancement of YOLO26.

2. Related Work

2.1 Object Detection

The perception of the traffic signals is an essential process of object detection, and the creation of detection models has had a significant effect on the performance of traffic lights detection. Model accuracy and speed have been progressively improved since two-stage convolutional neural networks, followed by single-stage detectors, and more recently Transformer-based architectures.

The two-stage convolutional neural networks are described as R-CNN. The idea of R-CNN was initially developed by Girshick et al. [7]; then Fast R-CNN accelerated the process by sharing convolution and region-of-interest (RoI) pooling, and Faster R-CNN introduced the replacement of selective search with a Region Proposal Network (RPN) to be even more optimized [17, 18]. Faster R-CNN was employed by R. Gavrilescu et al. [19] to detect traffic lights and stop signs, confirming the practicability of Faster R-CNN in real-time surveillance for intelligent driving. In order to address the real-time bottleneck of two-stage approaches, single-stage detectors were developed.

The first proposition of YOLO was by Redmon et al. [9]. YOLOv2 brought out the concept of anchor box and batch normalization and added multi-scale training using the Darknet-19 backbone to enhance localization accuracy and recall [20]. YOLOv3 came up with a deeper Darknet-53 and adopted concepts of feature pyramid networks to predict at three different scales to mitigate the issue of missing small objects effectively [12]. YOLOv4 was an organized combination of cross-stage partial networks, spatial pyramid pooling, and path aggregation networks along with Mosaic data augmentation and CIoU loss to bring detection accuracy to a new level in real-time inference conditions [13]. YOLOv5 was engineering-focused and offered multi-scale models and deployment tools that were easy to deploy. YOLOv6 was aimed at industrial use, and the EfficientRep backbone was developed with better re-parameterization efficiency and a separated detection head [21]. YOLOv7 re-optimized the network architecture and training strategies, such as the extended ELAN and model re-parameterization, achieving the best speed-accuracy frontier at that time [22]. YOLOv8 has an anchor-free detection head and more sophisticated loss functions, which simplified training further. Then, YOLOv9 presented programmable gradient information to overcome the information loss problem in deep supervision [23], and YOLOv10 was the pioneer of a stable dual assignment strategy to obtain end-to-end inference without maximum suppression, decreasing post-processing time [24]. YOLO26 is much faster in inference and has high accuracy due to the implementation of an NMS-free (Non-Maximum Suppression) end-to-end design and a set of training strategies [24].

Besides CNN-based detectors, DETR (Detection Transformer) based on Transformers totally eliminates the use of anchor boxes and NMS, converting the detection into a set prediction problem [25]. The further version, Deformable DETR, speeds up convergence by deformable attention and is better at small object performance [26].

2.2 Traffic Light Detection

The fundamental issues of traffic light detection are related to small objects, complex lighting, and very dynamic scenes. The currently available techniques, based on the historical development of technical paths and model architectures, can be divided into two groups: the CNN-based methods, the SSD series, the YOLO series, RT-DETR and other specialized models.

CNN techniques. DeepTLR by Weber et al. employs a single-stage convolutional network to predict traffic light states directly [27]. The group of Behrendt et al. developed a deep architecture that combines detection, tracking and classification and confirmed the benefit of temporal association in urban scenes [28]. To reliably detect and recognize traffic lights, R. Kulkarni et al. integrated a Faster R-CNN Inception V2 model with transfer learning [29]. To deal with very small traffic lights in dashcam images, C. Han et al. [30] minimized the RPN anchor size and deleted the pool4 layer of VGG-16 using the Faster R-CNN structure, which contributed greatly to the improvement of the detection mAP. Though these approaches have relatively high-precision, the computational complexity and the real-time behaviour remain bottlenecks.

Due to the performance jump of the YOLO series, there is a huge literature that has been written on how to apply these models to high-real-time traffic light detection. H. Zhang et al. [31] enhanced the ability of

YOLOv3 to detect small traffic signs by adding image mixing and a Multi-scale Spatial Pyramid Pooling (SSP) module. Q. Wang et al. [32] extended YOLOv4 with a shallow feature enhancement mechanism and a bounding box uncertainty prediction mechanism, which greatly enhanced localization and color discrimination of small traffic lights. C. Niu and K. Li [33] initially applied YOLOv5s to find traffic light regions, followed by AlexNet to recognize states, allowing detection of various types of traffic lights, including arrow lights and digital countdown lights. To overcome the issue of complicated backgrounds and tiny objects in traffic sign images, H. Zhang et al. [34] incorporated a high-resolution small-object detection layer into YOLOv8 and presented a bidirectional feature pyramid network (BiFPN), which performs bidirectional weighted feature fusion.

Pedestrian crossing traffic lights were recognized by Evan et al. [35] using a model based on SSD MobileNet V2. The SSD_Mobilenet_V1 model and hyperparameters connected with detection scale and aspect ratio were chosen by D. C. Burgos et al. [36], which is able to overcome the problem of detecting small objects at low resolution inputs.

Since the real-time Transformer detector RT-DETR was introduced, scholars have started considering the possibility of its application to detect traffic lights. Comparison by Y. Zhao et al. [37] showed that the accuracy of using RT-DETR directly on traffic light detection is as high as that of the YOLO series. The current traffic signal detection algorithm RT-DETR, which has a high computational cost and a slow speed, was optimized with the GELAN lightweight backbone network, ADown small-object-friendly downsampling, and DGSFM effective feature fusion by C. Tang et al. [38].

3. Methodology

3.1 YOLO26 Model Architecture

3.1.1 Overall Architecture

The unified real-time vision model series YOLO26, published by Ultralytics in 2026, can support object detection, instance segmentation, pose estimation, oriented object detection, and image classification at the same time [39]. Its general architecture is based on the traditional three-stage form of the “backbone-neck-head” in the YOLO series: the backbone is a cross-stage partial connection (CSP) style design, built purely out of standard convolutions, batch normalization, and SiLU activations, and so can be exported to inference backends, including TensorRT and ONNX, and its role is to extract image features on a per-layer basis; the neck is a bidirectional feature fusion structure with an upward path and a downward path, which is tasked with combining features across different backbone layers into pyramid features with the ability to represent features across multiple scales; the head is in a dual-head parallel design capable of selecting between a one-to-one head and a one-to-many head and both heads have shared backbone and neck features and execute object classification and bounding box regression in parallel on pyramid features at all levels.

The architectural differences between YOLO26 and its predecessor YOLO11 can be observed in the detection head and regression approach. Firstly, it uses a “one-to-one, one-to-many” dual-head model where the default one-to-one head facilitates NMS-free end-to-end inference. Secondly, the Distribution Focal Loss (DFL) module is entirely eliminated, and the regression head is simplified to a direct regression of four scalars. The three remedial measures on the training side to make up for the accuracy of the localization loss due to the elimination of DFL are introduced by YOLO26: the MuSGD optimizer, Progressive Loss, and Small Target-Aware Label Assignment (STAL).

3.1.2 End-to-End Path

In the case when a one-to-one head is chosen, as all objects have only one distinct prediction, inference only needs one filtering step using a confidence threshold and then direct decoding, no NMS is used at all in the whole process, which makes it easier to deploy and minimizes post-processing time. As post-processing of NMS is eliminated, the report states that YOLO26n can be up to about 43 percent faster than YOLO11n in ONNX inference on CPU [39]. The paper also states that the AP difference between the one-to-one head and one-to-many head (COCO) is 0.6-0.8, which is a fair compromise between accuracy and ease of deployment [39].

3.1.3 Direct Regression Without DFL

The bounding box regression in YOLOv8 and its later implementations takes the form of DFL: edges are represented as probability distributions over K discrete bins, and the expected value is decoded, i.e., the regression output at each spatial location grows to 4K logits [39]. The design presents two issues. Firstly, the head parameters and computation costs rise greatly, which is especially undesirable when using nano-scale models; this paper notes that eliminating DFL in YOLO11n leads to a reduction in the number of parameters and computational costs by 12 percent and 20 percent, respectively. Secondly, since the decoding range is constrained to (K-1) × stride pixels, DFL constrains the range of regression of large objects in high-resolution images and might cut off the actual size of the object.

YOLO26 does not use DFL but rather trains directly on four scalar values as bounding boxes and uses L1 loss as supervision. This modification allows the regression range to be unbounded by the number of discretized bins, which is very useful when it comes to localizing big objects given a high-resolution input. Experiments on controlled conditions in the paper demonstrate that the removal of DFL at 640 resolution leads to an increase of AP by 0.3 and at 1280 resolution-by 1.3 (the large-object metric AP [39] is particularly sensitive). Progressive Loss and STAL are used to mitigate localization accuracy loss in training.

3.1.4 Training Methods

MuSGD optimizer is a combination of the Muon optimizer and the usual momentum SGD. In case of multi-dimensional weight tensors, Muon updates are used, i.e., first, momentum updates are made, then multiple Newton-Schulz updates are done, which slightly orthogonalize the update direction, thus enhancing both the conditioning of the update and the speed of convergence. One-dimensional parameters are optimized using a pure SGD to preserve the scaling and translational parameters' stability. The experiments in the paper demonstrate that, under the same conditions, MuSGD may reach the same accuracy in 500 epochs as SGD does in 600 epochs (47.4 vs. 47.0 mAP) [39].

Dual-head training has intrinsic optimization asymmetry: the one-to-many head has more intense supervision at the beginning stage and is simpler to optimize, whereas the one-to-one head defines the ultimate end-to-end inference performance but is optimized slowly since there are tighter constraints on matching. Progressive Loss uses a weight coefficient that decreases linearly based on the number of training epochs

$$\alpha(t) = \max\left(1 - \frac{t}{\max(E-1,1)},\ 0\right) \cdot \left(\alpha_{init} - \alpha_{final}\right) + \alpha_{final} \tag{1}$$

to express the total loss as

$$L_{total} = \alpha(t)\, L_{one2many} + \left(1 - \alpha(t)\right) L_{one2one} \tag{2}$$

where t stands for the training epoch that is currently running, E represents the overall number of training epochs, α_{init} is the starting weight of the one-to-many head during the initial training phase, and is the ending weight. This transition will change the emphasis in the supervision of the one-to-many head in the earlier stages of training to the one-to-one head in the later stages of training, and thus it will be consistent with the end-to-end path during deployment. The official implementation has α(t) decreasing linearly between 0.8 and 0.1 (i.e., corresponding to the one-to-one weight increasing between 0.2 and 0.9) [39].

During the candidate screening, TAL must place the anchor center within the ground-truth box. Nevertheless, very tiny objects could have no viable anchor points due to downsampling discretization, which leads to the absence of positive examples in training. The method of STAL is only used on a temporary basis to clamp the side length of ground-truth boxes less than the smallest stride of 8 pixels to a reference size of 16 pixels in order to provide small objects with enough positive sample coverage. Matching and regression continue to involve the original ground-truth boxes without altering the localization target, and hence they do not affect the monitoring of normal objects.

3.2 Improved Model Architecture of YOLO26

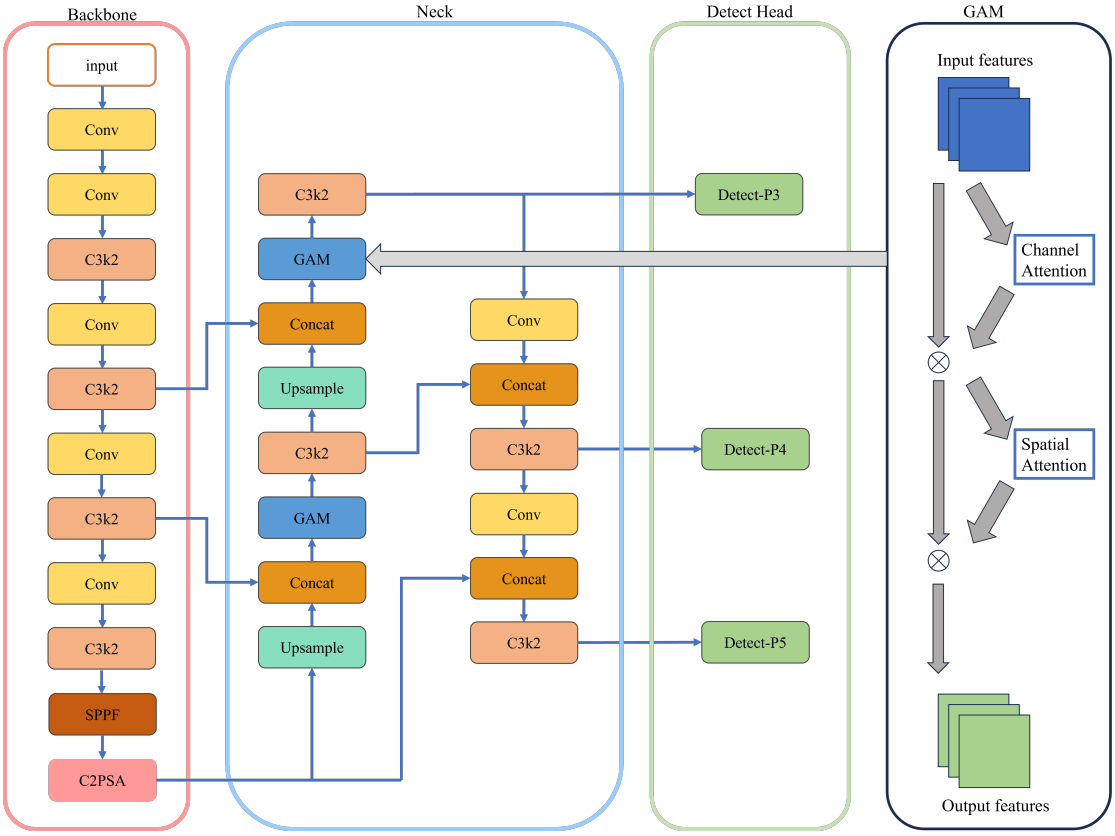
This paper uses YOLO26 as the base model and introduces the Global Attention Mechanism (GAM) at two key feature fusion locations in the neck network to build the enhanced nighttime traffic light detector model, which is referred to as YOLO26-GAM. YOLO26 neck uses an FPN+PAN architecture. Deep features are

upsampled and then concatenated and fused with backbone features at the same scale in the FPN top-down path: the Concat layer at layer 12 is used to fuse P5 and P4 features, and the Concat layer at layer 15 is used to fuse P4 and P3 features.

GAM modules are placed between layer 12 (Concat, P5, P4 fusion) and layer 13 (C3k2), and between layer 15 (Concat, P4, P3 fusion) and layer 16 (C3k2, P3 small-object branch) in this paper. To facilitate comparison with the official structure, all the layer indices used in this paper have been numbered according to the original YOLO26 official structure numbering. Figure 1 shows the main structure of the improved network.

The combined features resulting from the Concat operation include, at the same time, information about various semantic levels and spatial resolutions. Directly feeding them into the next convolutional step to fuse them makes the network have to do both "cross-scale information selection" and "spatial position focusing" implicitly within the convolutional weights, which raises the optimization effort. The introduction of an attention mechanism between the fusion and the convolution process of the fusion is analogous to rescaling the fused features through learnable, global parameters, so that later convolutions can use the rescaled features more directly, and decrease the cost of optimizing feature fusion. Following fusion at layer 15, the features are subjected to processing by the C3k2 block at layer 16 and are directly provided to P3/8 small-object detection branch. Small objects in driving images are normally only a few pixels wide, and nighttime traffic signals are often among them. Re-calibration of features prior to entering the small-object detection branch enables later branches to base their predictions on features more favorable to small-object discrimination, which directly matches the fundamental challenge underlying the task.

Figure 1: Key structure of the YOLO26-GAM network



4. Results and Discussion

4.1 Experimental Dataset and Experimental Setup

The experimental dataset that was used is TN-TLD (Taiwan Nighttime Traffic Light Dataset). This dataset has been gathered and released openly by Munir et al. during the development of the nighttime traffic light detection model LNT-YOLO and is accessible on Kaggle. The images are based on real night driving

conditions in the Taiwan area of China and have manual annotations of two types of traffic signals: circular and arrow lights [40]. TN-TLD is explicitly intended to handle nighttime low-light situations, and typical samples may include complicated situations, including vehicle lights and street lights interfering with the signal, like light sources and halo, occlusion, and partial occlusion. There are 26,976 images in total in the dataset, with 33,115 instances annotated, divided into 9,679 training images, 2,447 validation images, and 14,850 test images. The distribution of categories is given below: 0: red 14,736 1: yellow 1,446, 2: green 8,506, 3: left 215, 4: straight 4,580, 5: right 3,632.

The initial TN-TLD test set had some images that did not have any valid annotations. Manual verification showed that there are no traffic light instances in those images and they were part of pure background negative samples. The post-processing of the dataset has eliminated 6,659 images of this kind and left a total of 20,317 images. To provide reasonable distribution and quantity of categories across the training, validation, and test subsets, the cleaned dataset was randomly re-divided into the training, validation, and test subsets at the ratio of 7:2:1, with 14,159, 4,106, and 2,052 images, respectively.

The experiments were performed on a machine operating Windows 11, having an NVIDIA GeForce RTX 3060 laptop graphics card and 16 GB of RAM. The programming language that was used is Python 3.11, and the training was done using the deep learning framework PyTorch 2.9.1 (CUDA 13.0). The Ultralytics framework's auto-optimizer selection strategy was used in training.

Figure 2: Sample images from the generated traffic light dataset

4.2 Evaluation Metrics

The result of an object detection model is the class, confidence, and coordinates of every predicted box. At the evaluation stage, it is decided if a predicted box is right through the Intersection over Union (IoU) of the predicted box with the ground-truth box, and all predictions are divided into four groups depending on the confidence threshold and the IoU threshold. The predicted box that has an IoU higher than the threshold with a ground-truth box and has a uniform class is considered as a True Positive (TP); a predicted box which fails to correspond to any ground-truth box, also called a false detection, is considered as a False Positive (FP); a ground-truth box not matched by any prediction, or a missed detection, is considered as a False Negative (FN); True Negatives (TN) are typically not calculated in explicit form in detection problems.

Precision (P) is the ratio of correctly predicted samples to all samples that are classified as positive by the model as it measures the level of false prediction. Recall (R) is the fraction of positively identified objects out of the total number of ground-truth objects, which indicates the extent of missing detection.

$$P = \frac{TP}{TP + FP} \tag{3}$$

$$R = \frac{TP}{TP + FN} \tag{4}$$

The Precision-Recall (P-R) curve can be drawn in such a way that recall is the horizontal axis and precision is the vertical axis by passing through the entire set of confidence levels. Average Precision (AP) is the area beneath the curve.

$$AP = \int_0^1 P\,(R)\,dR \tag{5}$$

The arithmetic mean of AP of all C classes in the dataset is used to get the mean Average Precision (mAP) after calculating AP of each class. The mAP50 is the mAP obtained when an IoU threshold of 0.5 is applied to decide positive and negative samples, which is in line with the PASCAL VOC standard. Such a standard is less stringent in terms of localization accuracy and largely depicts the ability of the model to detect objects. The mAP50-95 is the average of the mAP values computed individually at 10 different IoU thresholds between 0.5 and 0.95 with a 0.05 step size, which is based on the COCO standard. Such a standard is more demanding in terms of the bounding box localization accuracy and reflects the localization quality of the model more thoroughly.

$$mAP = \frac{1}{C}\sum_{i=1}^{C} A\,P_i \tag{6}$$

In order to balance precision and recall at a given confidence level, the F1 score is defined as the harmonic mean of both.

$$F_1 = \frac{2PR}{P+R} \tag{7}$$

Besides detection accuracy, the effectiveness of a real-time detection model can be measured based on the amount of computational overhead and the rate of inference. The computational cost is given in FLOPs in the present work, which refers to the count of floating-point operations needed by the model to process one image, as measured by the Ultralytics framework, in units of 10^9. This is a measure of the computational complexity of the model.

4.3 Baseline Model Selection

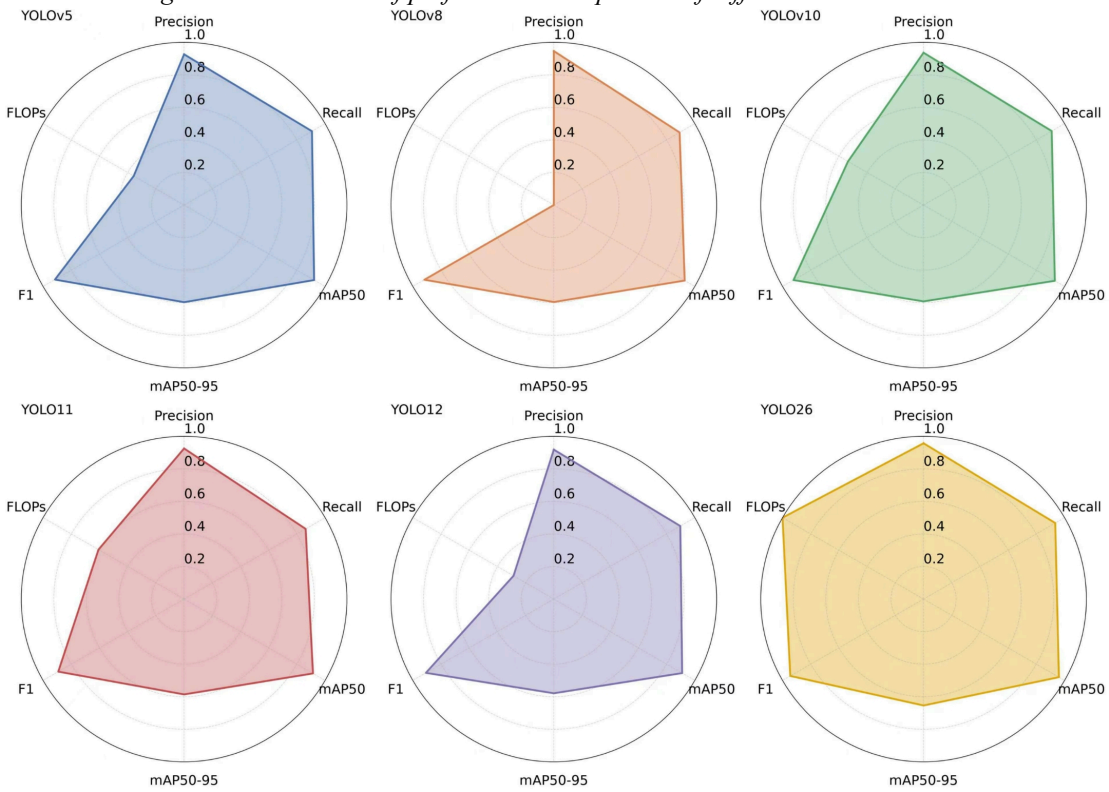
In order to properly assess the influence of the YOLO series on night-time traffic light recognition and establish a sound basis for further choosing the main model, six representative versions of YOLO were chosen to be compared experimentally, including YOLOv5, YOLOv8, YOLOv10, YOLO11, YOLOv12, and YOLO26. Released since then, YOLOv5 and YOLOv8 have become very popular in the industrial world, including in the context of video surveillance, industrial quality control, and autonomous vehicle assistance, because of their predictable training pipelines and complete toolchains of the ecosystem, as well as their extensive range of deployment options. The performance in terms of detection and robustness, and ease of use has been verified by extended testing in a wide range of practical applications, and they represent two of the most frequently used baseline models in industry [41-43]. YOLO10 is the earliest model to reach end-to-end inference without NMS in the YOLO structure using a uniform dual assignment system. Recent versions of YOLO include YOLO11, YOLOv12, and YOLO26. By adding those three newest models to the comparison, you can test if the latest architectural changes can be translated into real performance increases in the demanding case of small object detection in low-light conditions at night.

The detection performance of the six models on the test set is depicted in Table 1 under unified experimental conditions. The experimental findings reveal that YOLO26 performed the best in the aspect of detection accuracy, which has an F1 score of 0.945 and an mAP50 of 0.960. Also, YOLO26 has no default NMS-free end-to-end inference path which removes the post-processing stage and offers a benefit of inference latency. Considering the overall detection accuracy along with the efficiency of deployment, YOLO26 was finally chosen as the primary model and further optimization experiments and comparative studies were made using it.

Table 1: Performance comparison of different YOLO versions

Model	Precision	Recall	mAP50	mAP50-95	F1	FLOPs(B)
YOLOv5	0.926	0.907	0.922	0.597	0.916	7.1
YOLOv8	0.946	0.892	0.928	0.596	0.918	8.1
YOLOv10	0.936	0.909	0.931	0.592	0.922	6.6
YOLO11	0.925	0.862	0.914	0.586	0.892	6.4
YOLO12	0.918	0.897	0.910	0.579	0.907	7.3
YOLO26	0.957	0.934	0.960	0.654	0.945	5.3

Figure 3: Radar chart of performance comparison of different YOLO versions



Note: The FLOPs axis is reverse-normalized (smaller is better), while the other axes are original accuracy values.

4.4 Heatmap Experiments Based on Grad-CAM

Though deep convolutional neural networks have demonstrated impressive results on such tasks as object recognition, their decision-making behavior has a "black-box" property that makes it hard to intuitively understand what areas of an image the model uses to make its decisions. In order to investigate the attention given to the chosen YOLO26 model in the context of nighttime traffic light detection and determine whether it actually pays attention to traffic signals or non-target interfering light sources, Gradient-weighted Class Activation Mapping (Grad-CAM) is used in this section to visualize and explore the decision basis of the model.

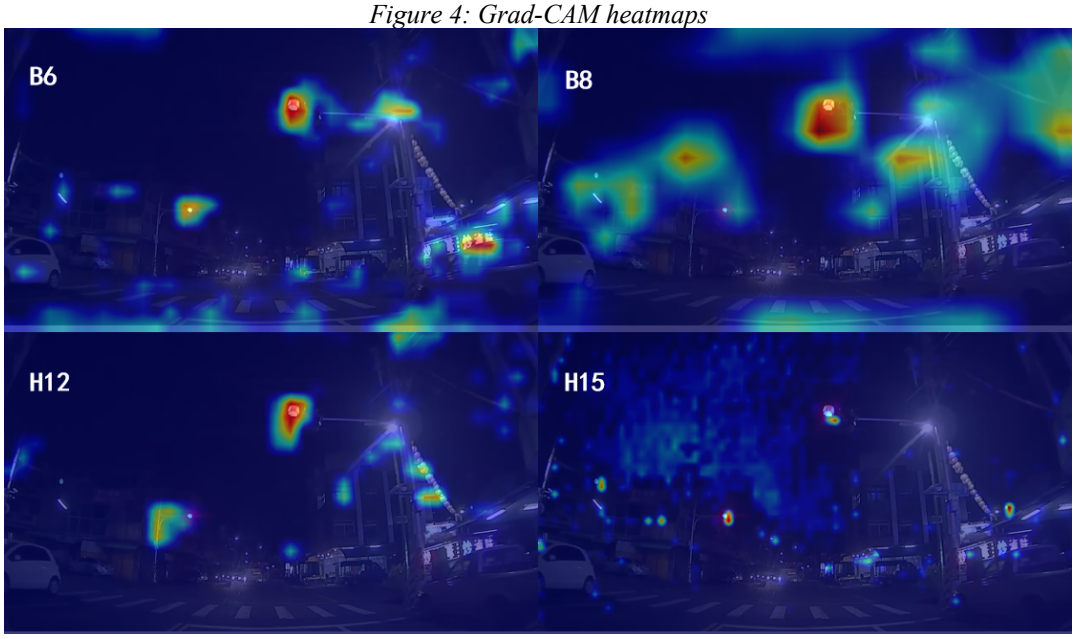
Selvaraju et al. proposed Grad-CAM as a gradient-based post-hoc interpretability approach. The main concept is to evaluate the influence of every feature channel on the target class choice by calculating the gradient of the target class score based on the feature maps of the final convolutional layer to produce a saliency heatmap indicating the areas of interest of the model [44].

In order to explore the attention evolution of the model at various network depths and feature scales, Grad-CAM heatmaps are created on important convolutional layers of the YOLO26 backbone and neck network, and four representative levels are chosen based on the feature pyramid scales: layer 6 (C3K2, backbone mid-scale features at 16× down-sampling, B6), layer 8 (C3K2, backbone deepest semantic features at 32× down-sampling, B6), layer 12 (Concat output of the first stage fusion in the neck, which is an upsampled deep P5 concatenated with backbone P4 features, H12), and layer 15 (Concat output of the second stage fusion in the

neck, which is an upsampled mid-level fused features concatenated with backbone P3 features, H15). All four layers will encompass both the backbone feature extraction and the neck cross-scale fusion processes and thus give an organized analysis of the attention shift between the backbone and the detection head. To make the comparison valid, the heatmaps of all four layers are produced using the same input image.

Based on Fig. 2, the red zones highlighted in the layer 6 heatmap focus on both the traffic signals and the ground region with significant contributions from streetlights and the lights of nearby shops. In the heatmap of layer 8, the response is also focused on the traffic lights and a portion of the sky background; it is, however, noted that the response is rather strongly impacted by the sky background. Being a sub-branch of small-object detection, the heatmap of layer 12 retains a unique response at the traffic lights. The heatmap of layer 15 continues to have a specific response to remote small traffic lights and the response to interfering lighting sources like streetlights and sign lights in the night time scene is significantly reduced, which means that the model has developed the capacity to identify traffic lights over interfering traffic-light-like sources during the neck fusion stage. Hence, the concept of attention mechanism is introduced at layers 12 and 15 in this experimental setting.

Figure 4: Grad-CAM heatmaps



4.5 Ablation Experiments Based on GAM

GAM (Global Attention Mechanism) was introduced by Liu et al. in 2021 to improve the representational power of the deep networks by decreasing the information reduction and promoting the interaction between the channel and spatial dimensions [45]. In order to examine the value of the GAM attention module to the effectiveness of the model presented in this paper, ablation experiments are prepared in this section. All other network structures, training configurations and data partitions remaining the same, the below variants are trained to compare them: YOLO26 without any attention module (baseline model), YOLO26 with the GAM attention module added only to layer 12, YOLO26 with the GAM attention module added only to layer 15, and YOLO26 with the GAM attention module added to both layer 12 and layer 15.

Table 2 shows the results of the ablation experiment. Adding the GAM module at layer 15 alone raised Precision by 0.957 - 0.967, but reduced Recall by 0.934 - 0.925, and mAP50 and F1 were almost constant. There was no improvement in introducing the GAM module at layer 12. In case of simultaneous introduction of the GAM module into both positions, all the metrics of the model achieved the highest values of the four groups, with Precision of 0.957 increased to 0.964, Recall of 0.934 increased to 0.947, mAP50 of 0.960 increased to 0.966, and F1 of 0.945 increased to 0.955.

The given outcomes show that adding an attention layer at any one location is not significantly useful in nighttime traffic light detection performance but might cause feature interference. On the other hand, the combination attention set-up at the two locations (H12 and H15) resulted in synergism, whereby the model

was able to achieve better accuracy and memory at the same time. Consequently, the next series of cross-experiments on attention mechanisms chose the H12 and H15 positions as the sites of insertion.

Table 2: Results of GAM ablation experiments

Group	H12	H15	Precision	Recall	mAP50	mAP50-95	F1
0	0	0	0.957	0.934	0.960	0.654	0.945
1	1	0	0.950	0.934	0.954	0.653	0.942
2	0	1	0.967	0.925	0.958	0.650	0.946
3	1	1	0.964	0.947	0.966	0.657	0.955

Note: 0 indicates that no attention mechanism is added, and 1 indicates that the attention mechanism is added.

4.6 Cross-Experiments Based on Attention Mechanisms

The SE (Squeeze-and-Excitation) [46] first reduces the feature maps by means of global average pooling to get a global descriptor of every channel.

$$z_c = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} u_c(i, j) \tag{8}$$

The feature map of the channel where u_c ; and H & W are the spatial dimensions.

After two consecutive fully connected layers are used to model the channel-wise dependencies, channel weights are produced by Sigmoid,

$$s = \sigma\big(W_2\, \delta(W_1 z)\big) \tag{9}$$

And finally, channel-wise recalibration is applied to the original features.

$$\tilde{x}_c = s_c \cdot u_c \tag{10}$$

where δ denotes ReLU, σ denotes *Sigmoid*, W_1 & W_2 are the weights of the fully connected layers.

The CA (Coordinate Attention) [47] overcomes the weakness of SE in loss of positional information by breaking down the global pooling into two one-dimensional pooling operations on the horizontal and vertical axes, which encode long-range dependencies on the row and column axes, respectively.

$$z_c^h(h) = \frac{1}{W} \sum_{i=1}^{W} x_c(h, i),\ z_c^w(w) = \frac{1}{H} \sum_{j=1}^{H} x_c(j, w) \tag{11}$$

The result of the combination of the features of both directions is subjected to a common 1x1 convolution and non-linear transformation. Features are then separated, and 1x1 convolutions with Sigmoid are performed in order to produce attention weights corresponding to two directions, respectively.

$$f = \delta\left(F_1\big([z^h,\ z^w]\big)\right) \tag{12}$$

$$g^h = \sigma\left(F_h\big(f^h\big)\right),\ g^w = \sigma\big(F_w(f^w)\big) \tag{13}$$

Finally, the input features are weighted.

$$y_c(i, j) = x_c(i, j) \times g_c^h(i) \times g_c^w(j) \tag{14}$$

The ECA (Efficient Channel Attention) [48] approach solves the problem of information loss due to dimensionality reduction in SE as it applies one-dimensional convolution to observe the local cross-channel interactions after the global average pooling is performed so that the dimensionality reduction does not occur.

$$\omega = \sigma\big(\mathrm{C1D}_k(y)\big) \tag{15}$$

In which the convolution kernel size k is adaptively determined by the number of channels.

$$k = \left|\frac{\log_2 C}{\gamma} + \frac{b}{\gamma}\right|_{odd} \tag{16}$$

where γ=2 and b=1 are default hyperparameters, and$|\cdot|_{odd}$ denotes taking the nearest odd integer.

Convolutional Block Attention Module (CBAM) [49] uses a serial channel-spatial dual attention structure. The channel attention both uses mean pooling and max-pooling, and it creates channel weights with the help of a shared multi-layer perceptron.

$$M_c(F) = \sigma\left(W_1\left(W_0\left(F_{avg}^c\right)\right) + W_1\left(W_0(F_{max}^c)\right)\right) \tag{17}$$

where F_{avg}^c & F_{max}^c are the channel descriptors derived by applying average pooling and max pooling to feature F, respectively. The spatial attention will do an average pooling and max pooling in the channel direction and concatenate them to produce spatial weights via a large-kernel convolution.

$$M_s(F) = \sigma\left(f^{7\times7}\left(\left[F_{avg}^s;\ F_{max}^s\right]\right)\right) \tag{18}$$

The two attention mechanisms are applied to the features sequentially.

$$F' = M_c(F) \otimes F\ ,\ F'' = M_s(F') \otimes F' \tag{19}$$

The GAM (Global Attention Mechanism) [45] aims to minimize information reduction and increase channel-spatial interaction. Its channel attention submodule does a three-dimensional permutation of the features, transposes one of the channel dimensions, models it with a multi-layer perceptron, and finally permutes it back to its original format, thus finishing channel recalibration without losing spatial detail.

$$M_c(F) = \sigma\left(\mathrm{MLP}\left(\mathrm{Permute}(F)\right)\right) \tag{20}$$

The submodule of spatial attention reduces the channel dimension by dilated convolution to produce a spatial attention map.

$$M_s(F) = \sigma\left(\mathrm{Conv}_{7\times7,\,d=4}(F)\right) \tag{21}$$

Two submodules are linked in series in a residual form, which contributes to the cross-dimensional interaction.

$$F_1 = M_c(F) \otimes F\ ,\ F_2 = M_s(F_1) \otimes F_1 \tag{22}$$

The main challenge in detecting traffic lights at night is the ability to differentiate between the traffic lights and other interfering light sources like streetlights and tail lights of vehicles in low-light environments that demand that the attention process must have both channel selection and spatial localization abilities. Both CBAM with sequential dual attention and GAM with cross-dimensional interaction include the channel and spatial dimensions, but ECA is mostly concerned with the channel dimension, and CA with spatial modeling uses directional decomposition, thus they do not align as well with the task requirements. Consequently, CBAM and GAM are finally chosen as the objects of the cross-experiment.

The cross-experiments are carried out with the assumption that the model structure, insertion locations, training setup, and data partitioning are exactly the same. The baseline is YOLO26 without any attention model, followed by the introduction of CBAM and GAM, respectively, at layer 12 and 15 of the neck network, which forms the experimental groups described in Table 3.

Table 3: Cross-experiments of attention mechanisms

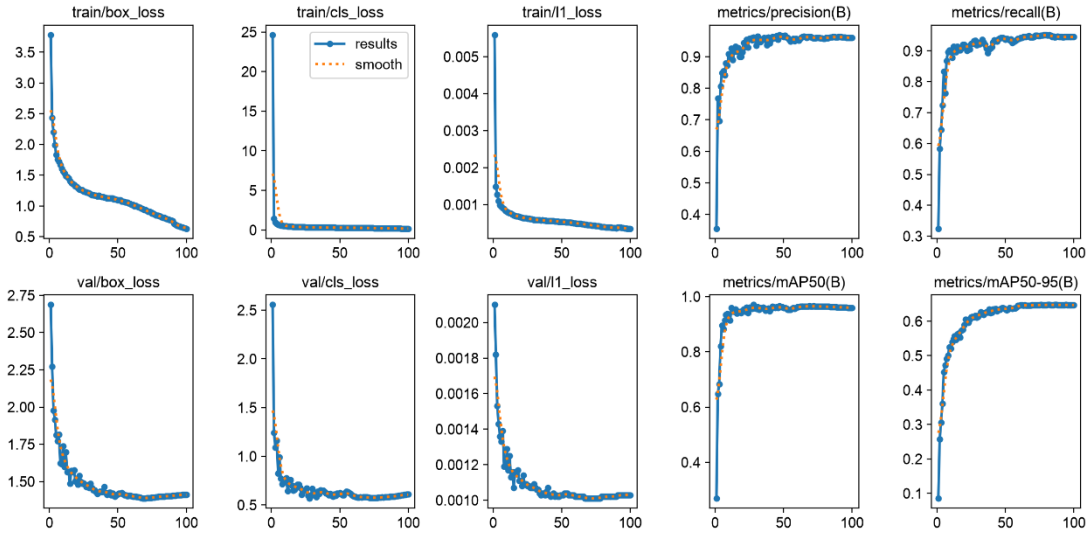
Group	H12	H15	Precision	Recall	mAP50	mAP50-95	F1
0	0	0	0.957	0.934	0.960	0.654	0.945
1	0	GAM	0.967	0.925	0.958	0.650	0.946
2	0	CBAM	0.965	0.939	0.963	0.646	0.952
3	CBAM	0	0.952	0.935	0.941	0.644	0.943
4	CBAM	GAM	0.943	0.930	0.958	0.649	0.936
5	CBAM	CBAM	0.959	0.908	0.951	0.65	0.933
6	GAM	0	0.950	0.934	0.954	0.653	0.942
7	GAM	GAM	0.964	0.947	0.966	0.657	0.955
8	GAM	CBAM	0.957	0.936	0.961	0.650	0.946

Note: 0 indicates that no attention mechanism is added.

In the case when attention is presented in the location H12 only, irrespective of whether GAM (Group 6) or CBAM (Group 3) is used, the model has a lower mAP50 compared to the baseline (0.954 and 0.941 as compared to 0.960), which means that adding attention in the position H12 alone does not have a positive impact and even negatively affects CBAM. In the position where attention is presented in the H15 location only, GAM (Group 1) attains the maximum Precision in all the experiments (0.967), but it reduces Recall to 0.925, and F1 is more or less similar to the baseline. With respect to CBAM (Group 2), the mAP50 increases by a small amount to 0.963, whereas the mAP50-95 decreases to 0.646 (baseline: 0.654). These findings suggest that a single attention position is hard to achieve precision and recall together and can even damage another type of metric because of excessive suppression of features. Recall in the dual-CBAM setup (Group 5) is 0.908, which is the lowest value of all experiments, a decline of 2.6 percent, as compared to the baseline, which indicates that the cascaded attention of dual CBAM is too much suppressing features and leads to a significant loss in detection rate; its mAP50 and mAP50-95 are also lower than the baseline. Concerning the hybrid configurations, the mAP50s of Group 8 (H12=GAM, H15=CBAM) and Group 4 (H12=CBAM, H15=GAM) are 0.961 and 0.958, respectively, both lower than the value of the dual-GAM configuration. The best performance among all experiments is achieved by Group 7 (GAM introduced at H12 and H15): Precision equals 0.964, Recall equals 0.947, mAP50 equals 0.966, mAP50-95 equals 0.657, and F1 equals 0.955, and it is the only configuration that performs better than the baselines in all metrics.

Figure 5 illustrates the loss and accuracy curves of the dual-GAM configuration after being trained on the TN-TLD data set for 100 epochs. Looking at the loss, the training and validation sets have a total decreasing trend in box loss and cls loss. The initial phase of training sees the loss diminishing at a high rate, but the rate of decrease slows down steadily and ultimately seems to be leveling off, which indicates that the model is converging. By the end of training, the training and validation losses are almost identical, and the validation loss also drops approximately simultaneously with the training loss with insignificant bounce, so there is no clear evidence of overfitting.

Figure 5: Variation curves of loss and accuracy metrics



According to the obtained findings, the ultimate structure of the model is defined in the following way: The introduction of GAM attention mechanisms takes place after the fusion nodes at layer 12 and layer 15 (original official indices) in the YOLO26 neck network and prior to the associated C3k2 modules, which is named as YOLO26-GAM. In such a configuration, the best performance regarding the values of both mAP50 and mAP50-95 can be obtained, and all the metrics are superior to the baseline, which confirms the rationality of the chosen attention mechanism and its insertion locations.

5. Conclusion

In order to overcome the poor detection rate of small traffic lights during nighttime in autonomous cars, YOLO26 is used as the base model and the Global Attention Mechanism (GAM) is added to the two fusion

nodes H12 and H15 in the feature pyramid of the neck network, resulting in the proposed enhanced algorithm YOLO26-GAM. Regarding model selection, YOLO26 was chosen as the baseline based on identical experiments using six different models: YOLOv5, YOLOv8, YOLOv10, YOLO11, YOLOv12, and YOLO26. Positions of insertion and configuration of the attention mechanism were established by Grad-CAM visualization analysis, ablation experiments, and CBAM/GAM cross-experiments. The YOLO26-GAM configuration performed best compared to all other configurations and was the only configuration that outperformed the baseline on each of the measures on the curated TN-TLD dataset. It means that it is possible to improve the recognition ability of YOLO at night by incorporating the dual attention mechanism into YOLO26-GAM, which enhances the traffic light detection and recognition of autonomous vehicles in nighttime conditions. There are still some limitations to the work presented in this paper. The experiments were only tested on one dataset, TN-TLD, which is quite small in size, and the data were split in this paper, so they cannot be compared with previous literature reports based on the official partition. The suggested algorithm does not specifically consider harsh daytime situations, including sun glare, shadows, and very reflective backgrounds. To check the generalization behavior of the proposed approach, in the near future, these harsh conditions will be accounted for, and the algorithm will be validated with larger-scale and more diverse driving datasets.

References

[1] World Health Organization. (2023). Global status report on road safety 2023: Summary. https://www.who.int/publications/i/item/9789240086252

[2] Zhang, J., Yue, Y., & Zhu, X. (2025). Real-time fusion and object detection based on visible light and infrared images. In Proceedings of the 2025 9th International Conference on Control Engineering and Artificial Intelligence. https://doi.org/10.1145/3722150.3722160

[3] Jensen, M. B., Philipsen, M. P., Møgelmose, A., Moeslund, T. B., & Trivedi, M. M. (2016). Vision for looking at traffic lights: Issues, survey, and perspectives. IEEE Transactions on Intelligent Transportation Systems, 17(7), 1800–1815. https://doi.org/10.1109/TITS.2015.2509509

[4] De Charette, R., & Nashashibi, F. (2009). Real time visual traffic lights recognition based on spot light detection and adaptive traffic lights templates. In 2009 IEEE Intelligent Vehicles Symposium (pp. 358–363). IEEE. https://doi.org/10.1109/IVS.2009.5164304

[5] Gomez-Moreno, H., Maldonado-Bascon, S., Gil-Jimenez, P., & Lafuente-Arroyo, S. (2010). Goal evaluation of segmentation algorithms for traffic sign recognition. IEEE Transactions on Intelligent Transportation Systems, 11(4), 917–930. https://doi.org/10.1109/TITS.2010.2054084

[6] De la Escalera, A., Armingol, J. M., & Mata, M. (2003). Traffic sign recognition and analysis for intelligent vehicles. Image and Vision Computing, 21(3), 247–258. https://doi.org/10.1016/S0262-8856(02)00156-7

[7] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 580–587). IEEE.

[8] Kim, H.-K., Park, J. H., & Jung, H.-Y. (2018). An efficient color space for deep-learning based traffic light recognition. Journal of Advanced Transportation, 2018, Article 2365414. https://doi.org/10.1155/2018/2365414

[9] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 779–788). IEEE.

[10] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). SSD: Single shot multibox detector. In B. Leibe, J. Matas, N. Sebe, & M. Welling (Eds.), Computer Vision – ECCV 2016 (pp. 21–37). Springer. https://doi.org/10.1007/978-3-319-46448-0_2

[11] Müller, J., & Dietmayer, K. (2018). Detecting traffic lights by single shot detection. In 2018 21st International Conference on Intelligent Transportation Systems (ITSC) (pp. 266–273). IEEE. https://doi.org/10.1109/ITSC.2018.8569651

[12] Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement. arXiv. https://arxiv.org/abs/1804.02767

[13] Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. arXiv. https://arxiv.org/abs/2004.10934

[14] Du, S., et al. (2024). TSD-YOLO: Small traffic sign detection based on improved YOLO v8. IET Image Processing, 18(11), 2884–2898. https://doi.org/10.1049/ipr2.13166

[15] Chakrabarty, S. (2026). YOLO26: An analysis of NMS-free end-to-end framework for real-time object detection. arXiv. https://arxiv.org/abs/2601.12882

[16] Jensen, M. B., Nasrollahi, K., & Moeslund, T. B. (2017). Evaluating state-of-the-art object detector on challenging traffic light data. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (pp. 9–15). IEEE.

[17] Girshick, R. (2015). Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision (pp. 1440–1448). IEEE.

[18] Ren, S., He, K., Girshick, R., & Sun, J. (2016). Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(6), 1137–1149. https://doi.org/10.1109/TPAMI.2016.2577031

[19] Gavrilescu, R., Zet, C., Foșalău, C., Skoczylas, M., & Cotovanu, D. (2018). Faster R-CNN: An approach to real-time object detection. In 2018 International Conference and Exposition on Electrical and Power Engineering (EPE) (pp. 0165–0168). IEEE. https://doi.org/10.1109/ICEPE.2018.8559810

[20] Redmon, J., & Farhadi, A. (2017). YOLO9000: Better, faster, stronger. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 6517–6525). IEEE.

[21]Li, C., et al. (2022). YOLOv6: A single-stage object detection framework for industrial applications. arXiv. https://arxiv.org/abs/2209.02976

[22] Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2023). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 7464–7475). IEEE.

[23] Wang, C.-Y., Yeh, I.-H., & Liao, H.-Y. M. (2024). YOLOv9: Learning what you want to learn using programmable gradient information. In European Conference on Computer Vision (pp. 1–21). Springer.

[24] Wang, A., et al. (2024). YOLOv10: Real-time end-to-end object detection. Advances in Neural Information Processing Systems, 37, 107984–108011.

[25] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020). End-to-end object detection with transformers. In European Conference on Computer Vision (pp. 213–229). Springer.

[26] Zhu, X., Su, W., Lu, L., Li, B., Wang, X., & Dai, J. (2020). Deformable DETR: Deformable transformers for end-to-end object detection. arXiv. https://arxiv.org/abs/2010.04159

[27] Weber, M., Wolf, P., & Zöllner, J. M. (2016). DeepTLR: A single deep convolutional network for detection and classification of traffic lights. In 2016 IEEE Intelligent Vehicles Symposium (IV) (pp. 342–348). IEEE.

[28] Behrendt, K., Novak, L., & Botros, R. (2017). A deep learning approach to traffic lights: Detection, tracking, and classification. In 2017 IEEE International Conference on Robotics and Automation (ICRA) (pp. 1370–1377). IEEE. https://doi.org/10.1109/ICRA.2017.7989163

[29] Kulkarni, R., Dhavalikar, S., & Bangar, S. (2018). Traffic light detection and recognition for self driving cars using deep learning. In 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA) (pp. 1–4). IEEE. https://doi.org/10.1109/ICCUBEA.2018.8697819

[30] Han, C., Gao, G., & Zhang, Y. (2019). Real-time small traffic sign detection with revised Faster-RCNN. Multimedia Tools and Applications, 78(10), 13263–13278. https://doi.org/10.1007/s11042-019-7238-9

[31] Zhang, H., et al. (2020). Real-time detection method for small traffic signs based on YOLOv3. IEEE Access, 8, 64145–64156. https://doi.org/10.1109/ACCESS.2020.2984554

[32] Wang, Q., Zhang, Q., Liang, X., Wang, Y., Zhou, C., & Mikulovich, V. I. (2021). Traffic lights detection and recognition method based on the improved YOLOv4 algorithm. Sensors, 22(1), 200. https://doi.org/10.3390/s22010200

[33] Niu, C., & Li, K. (2022). Traffic light detection and recognition method based on YOLOv5s and AlexNet. Applied Sciences, 12(21), 10808. https://doi.org/10.3390/app122110808

[34] Zhang, H., Liang, M., & Wang, Y. (2025). YOLO-BS: A traffic sign detection algorithm based on YOLOv8. Scientific Reports, 15, 7558. https://doi.org/10.1038/s41598-025-88184-0

[35] Evan, Wulandari, M., & Syamsudin, E. (2020). Recognition of pedestrian traffic light using TensorFlow and SSD MobileNet V2. In IOP Conference Series: Materials Science and Engineering, 1007(1), 012022. https://doi.org/10.1088/1757-899X/1007/1/012022

[36] Burgos, D. C., Iztueta, E. J., Ormaechea, I. M., Martínez-Otzeta, J. M., & Mugica, A. C. (2024). Deep learning-based traffic light detection in a custom embedded hardware platform for ADAS applications. IEEE Access, 12, 127862–127878. https://doi.org/10.1109/ACCESS.2024.3456789

[37] Zhao, Y., et al. (2024). DETRs beat YOLOs on real-time object detection. In 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 16965–16974). IEEE.

[38] Tang, C., Li, Y., Wang, L., & Li, W. (2025). Real-time traffic light detection based on lightweight improved RT-DETR. Journal of Real-Time Image Processing, 22(2), 82. https://doi.org/10.1007/s11554-025-01621-4

[39] Jocher, G., Qiu, J., Liu, M., Lyu, S., Akyon, F. C., & Kalfaoglu, M. E. (2026). Ultralytics YOLO26: Unified real-time end-to-end vision models. arXiv. https://arxiv.org/abs/2606.03748

[40] Munir, S., & Lin, H.-Y. (2025). LNT-YOLO: A lightweight nighttime traffic light detection model. Smart Cities, 8(3), 95. https://doi.org/10.3390/smartcities8030095

[41] Bajpai, A. (2024). Attire-based anomaly detection in restricted areas using YOLOv8 for enhanced CCTV security. arXiv. https://arxiv.org/abs/2404.00645

[42] Gui, Z., & Geng, J. (2024). YOLO-ADS: An improved YOLOv8 algorithm for metal surface defect detection. Electronics, 13(16), 3129. https://doi.org/10.3390/electronics13163129

[43] Wibowo, A., Trilaksono, B. R., Hidayat, E. M. I., & Munir, R. (2023). Object detection in dense and mixed traffic for autonomous vehicles with modified YOLO. IEEE Access, 11, 134866–134877. https://doi.org/10.1109/ACCESS.2023.3334567

[44] Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). Grad-CAM: Visual explanations from deep networks via gradient-based localization. In Proceedings of the IEEE International Conference on Computer Vision (pp. 618–626). IEEE.

[45] Liu, Y., Shao, Z., & Hoffmann, N. (2021). Global attention mechanism: Retain information to enhance channel-spatial interactions. arXiv. https://arxiv.org/abs/2112.05561

[46] Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 7132–7141). IEEE.

[47] Hou, Q., Zhou, D., & Feng, J. (2021). Coordinate attention for efficient mobile network design. In 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 13708–13717). IEEE.

[48] Wang, Q., Wu, B., Zhu, P., Li, P., Zuo, W., & Hu, Q. (2020). ECA-Net: Efficient channel attention for deep convolutional neural networks. In 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 11531–11539). IEEE.

[49] Woo, S., Park, J., Lee, J.-Y., & Kweon, I. S. (2018). CBAM: Convolutional block attention module. In European Conference on Computer Vision (pp. 3–19). Springer.

Funding

This research received no external funding.

Conflicts of Interest

The authors declare no conflict of interest.

Acknowledgment

This paper is an output of the science project.