2581-4575

International Journal For Recent Developments in Science & Technology

Crossref

A Peer Reviewed Research Journal

MCS-YOLO A MULTISCALE OBJECT DETECTION METHOD FOR AUTONOMOUS DRIVING ROAD ENVIRONMENT RECOGNITION

[1]M GOWTHAM, [2]K. TULASI KRISHNA KUMAR

[1]MTECH II YEAR(CST), Sankethika Vidya Parishad Engineering College, Visakhapatnam, India.

[2]ASSISTANT PROFESSOR & PLACEMENT OFFICER, Sankethika Vidya Parishad Engineering College, Visakhapatnam, India.

Abstract: This research addresses critical challenges in autonomous driving technology, focusing on the improvement of object detection algorithms' accuracy and speed. Introducing the MCS-YOLO algorithm, our approach incorporates a coordinate attention module into the backbone, enhancing the aggregation of spatial coordinate and cross-channel information in feature maps. Additionally, a multiscale small object detection structure is designed to heighten sensitivity to dense small objects, complemented by the integration of the Swin Transformer structure for CNNs to prioritize contextual spatial information. Through extensive evaluation on the BDD100K autonomous driving dataset, the MCS-YOLO algorithm outperforms the YOLOv5s counterpart in mean average precision and recall rates. Remarkably, our algorithm achieves a real-time detection speed of 55 frames per second in actual driving scenarios. Further exploration with YoloV5x6 demonstrates promising results, showcasing a potential improvement in mean average precision to 0.798%. This research offers a robust and efficient solution for advancing object detection capabilities in autonomous driving, contributing to the continual evolution of intelligent transportation systems.

Index terms – "Coordinate attention mechanisms, autonomous driving, road environmental object detection, swin transformer, YOLOv5".

1. INTRODUCTION

In the 21st century, the escalating prevalence of automobiles as a fundamental mode of transportation globally has led to a surge in new vehicle registrations and licensed drivers. However, this rapid increase in motor vehicles has brought about challenges such as traffic accidents, congestion, and environmental concerns. Addressing these issues, autonomous driving technology emerges as a pivotal solution, contributing significantly to safety enhancement and informed decision-making in route planning during vehicular travel [1], [2].

The cornerstone of autonomous driving lies in the environmental perception system, tasked with precise and rapid identification of objects within the road environment. This identified information is crucial for informing decision systems to optimize route planning [3]. Early in the development of autonomous driving, expensive single or multi-sensor fusion methods were employed, requiring manual adjustment of vehicle parameters and extensive human involvement. However, with advancements in deep learning, sensing, and hardware technologies, computer vision (CV) and natural language processing (NLP) have flourished, offering more efficient solutions.

Girshick et al's. R-CNN model better acknowledgment [5]. Later progressions like He et al's. Spatial Pyramid Pooling (SPP) [6], Fast R-CNN [7], and Faster R-CNN [8] with a "Region Proposal Network" (RPN] improved detection accuracy and handling productivity. Excellent detection and division are added utilizing Mask R-CNN [9]. These advances exhibit the groundbreaking capability of deep learning-based object recognizable proof calculations for continuous, precise, and conservative independent vehicle natural detecting.

The "You Only Look Once" (YOLO) and "Single Shot MultiBox Detector" (SSD) calculations use regression approaches for object arrangement and jumping box prediction. The YOLO calculation inputs the total picture and relapses bounding box area and class in the result layer. Industry involves YOLO and SSD calculations for faster constant detection than R-CNN. A Transformer-based convolutional neural network for thick vision applications was utilized by Liu et al.

The Swin Transformer [20], [21] shows its power for arrangement, detection, and segmentation. ConvNext [22] trains CNNs utilizing Swin Transformer's streamlining approach. ConvNext beats Swin Transformer in surmising and accuracy with similar Lemon. Chen et al. [23] concocted a DW-YOLO strategy that increments network profundity and broadness to perceive vehicle objects. Zhou et al. [24] proposed a lightweight MobileYOLO procedure that limits boundaries and rates identification. Wang et al. [25] involved MobileNet on YOLOv4 for driving circumstances and acquired 35 FPS detection. A superior SA-YOLOv3 finder by Tian et al. [26] balances speed and exactness. Gupta et al. [27] utilized location and division to further develop self-driving vehicle versatile way of behaving by distinguishing street climate objects. Wang et al. [28] present an independent driving discovery network for hazy circumstances that upgrades object distinguishing proof precision and speed. Li et al. [29] fostered a Res-YOLO network model that limited missed discoveries and upgraded vehicle object detection accuracy.

2. LITERATURE SURVEY

This paper surveys the creators' momentum research on protected and tough independent driving in metropolitan settings with startling traffic. The review incorporates continuous innovations for climate detecting, restriction, arranging, and control to construct a completely practical vehicle stage. [1] The creators' work on Junior, Stanford's 2007 DARPA Metropolitan Test section, is extended to cover more practical driving conditions. The creators portray three unaided techniques that consequently adjust a 64-shaft turning LIDAR with preferable precision over hand perceptions. Online confinement with centimeter-level accuracy requires high-goal ecological guides.

International Journal For Recent Developments in Science & Technology
2581-4575
Crossref
A Peer Reviewed Research Journal

Deterrent following, bicycle, walker, and vehicle discovery, and traffic signal recognition are conceivable with further developed insight and ID calculations [6,29,39]. In light of approaching information, a progressive arranging framework makes many potential directions each second to enhance the vehicle's course. An upgraded regulator streamlines choke, brake, and guiding to boost solace and limit direction mistake. These calculations function admirably in each climate, day or night. Junior has driven independently for many kilometers in different certifiable settings on account of these advancements.

This study talked about the fast advances in AI, computer vision, ML, and independent vehicles [2]. The creators give an itemized outline to assist experienced specialists and fledglings with staying aware of this quick extending subject. This book gives an extensive presentation of independent vehicle PC vision challenges, datasets, and approaches, in contrast to earlier examinations. The review covers verifiable writing and current advances in independent driving fields such distinguishing proof, recreation, movement gauges, following, scene translation, and start to finish learning. The creators use benchmarking datasets including KITTI, Maxim, and Cityscapes to assess algorithmic execution. Open worries and proceeding with research difficulties make the review applicable to independent vehicle advances [2,4,27]. The creators give a committed site to smooth route among subjects and approaches, giving setting and data to further develop openness and correct missing references. This careful evaluation helps scholastics, experts, and novices fathom the advancing climate of PC vision in independent vehicles.

The impending send off of independent vehicles and the need to give wellbeing, trustworthiness, and an agreeable client experience for general acknowledgment. As client solace in driving styles goes from energetic to quiet, the creators propose a gaining from exhibit methodology to tailor independent vehicle conduct. [4] Clients may physically drive the vehicle to represent their ideal driving style, staying away from the difficult and mistake inclined errand of physically tweaking speed increase profiles, distances to different vehicles, and path change speed. An expense capability models the driving style, and component based converse support learning tracks down the model boundaries that best fit it. The model really processes vehicle directions in independent mode subsequent to getting the hang of, permitting it to replicate and adjust to various driving styles. It can learn cost works and mirror driving ways of behaving utilizing genuine driver information, demonstrating its value. This client driven methodology works on the independent vehicle's responsiveness to individual inclinations and incorporates independent innovation into fluctuated client encounters.

The halt in object acknowledgment execution on the PASCAL VOC dataset and proposes a novel, basic, and versatile discovery approach that significantly works on mean normal precision. The technique [5] accomplishes a dazzling 53.3% Guide, surpassing the past high by practically 30%. Utilizing high-limit Convolutional Neural Networks (CNNs) to handle base up area proposition permits exact item confinement and division, and administered pre-preparing for a helper task followed by space explicit tweaking functions admirably, particularly in situations with restricted marked preparing information. R-CNN (Regions with CNN features) utilizes these experiences to support execution. R-CNN beats OverFeat, a sliding-window identifier in view of a comparable CNN design, on the 200-class ILSVRC2013 location dataset [5,7,8,17,18]. R-CNN's outcome demonstrates the way that area suggestions can expand CNN execution, defeat past cutoff points, and further develop object location.

Existing profound convolutional brain organizations (CNNs) that need fixed-size input pictures lose acknowledgment exactness for pictures or sub-pictures of various sizes. The strategy utilizes "spatial pyramid pooling" in SPP-net, another organization structure [6]. This plan produces a fixed-length portrayal free of picture size or scale, making it more versatile. CNN-based picture order is improved by SPP-net's item distortion opposition. The article shows that SPP-net upgrades CNN engineering exactness on ImageNet 2012. SPP-net produces cutting edge characterization scores on Pascal VOC 2007 and Caltech101 datasets utilizing a solitary full-picture portrayal and no tweaking. In object discovery, SPP-net velocities highlight map handling and beats R-CNN [39]. In the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2014, SPP-net methodologies set #2 in object acknowledgment and #3 in picture arrangement among 38 groups. The book portrays significant cutthroat upgrades that exhibit SPP-net's versatility and proficiency in visual recognizable proof assignments.

3. METHODOLOGY

i) Proposed Work:

We provide an advanced MCS-YOLO approach for object identification and recognition in autonomous driving contexts, which integrates a coordinate attention module, a multiscale tiny object detection framework, and a Swin Transformer. This technique seeks to markedly improve precision and velocity in object detection. Our MCS-YOLO algorithm exhibited enhanced performance, achieving a mean average accuracy (mAP) of 53.6%, as evidenced by rigorous ablation tests and comparison trials conducted on the BDD100K dataset [41]. To enhance detecting capabilities, our suggested system investigates sophisticated methodologies by using Yolov5x6 and YoloV8. These supplementary approaches seek to elevate the mAP over 60%, guaranteeing robust and efficient object recognition. Each algorithm, such as Faster RCNN, AD-Faster RCNN, YoloV3, YoloV3-tiny, YoloV4, YoloV5s, YoloV5s Improved Version, Yolo V7 - small, Yolo V5x6, Yolo V8, and MCS YoloV5s, enhances the thorough assessment of detecting skills across diverse circumstances [12,13,14,15,23,24]. This multifaceted strategy seeks to enhance the environmental perception system for autonomous driving, guaranteeing improved precision and efficacy in recognizing objects essential for the safe and dependable navigation of autonomous vehicles.

International Journal For Recent Developments in Science & Technology
2581-4575
Crossref
A Peer Reviewed Research Journal

ii) System Architecture:

The system architecture is a carefully constructed framework that efficiently processes data, beginning with input and advancing through image processing, utilizing advanced data augmentation techniques. The foundation consists of model construction, utilizing a comprehensive array of sophisticated models, such as YoloV5s, the Enhanced YoloV5s, MCS YoloV5s, Yolo V5x6, YoloV4, YoloV3, YoloV3-tiny, Yolo V7, Yolo V8, Faster RCNN, and AD-Faster RCNN [12,13,14,15,23,24]. The models are subjected to comprehensive evaluation, measuring performance parameters like accuracy, recall, and mean average precision (mAP) [40]. The model deemed most successful, based on these measures, is chosen for object detection. This design guarantees an efficient procedure, enhancing autonomous driving through strong and precise perception in various road conditions. The use of a varied model set facilitates flexibility, allowing the system to perform optimally across a range of circumstances, hence enhancing the development of autonomous vehicle technology.

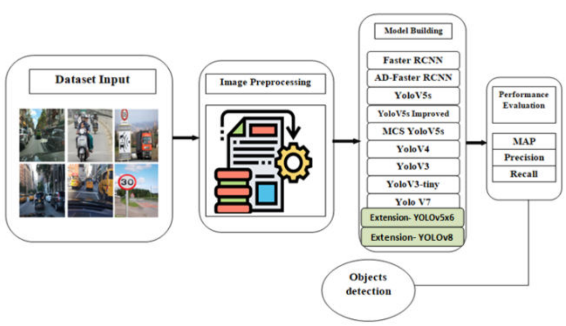


Fig 1 System Architecture

iii) Dataset collection:

The assessment of the MCS-YOLO algorithm in autonomous driving perception use the BDD100K dataset, recognized for its authenticity and comprehensiveness. This authoritative public dataset, gathered from real-life scenarios, includes various weather conditions, driving situations, and times of day, comprising 10 objective categories. The collection comprises 100,000 photos, encompassing six unique weather conditions: bright, overcast, rainy, snowy, and foggy. To improve model validation, 20,000 unlabeled photos were eliminated, and the remaining dataset was divided in an 8:1:1 ratio for training, validation, and testing sets, respectively. The training set contains 64,800 photos, the validation set comprises 7,200 images, and the test set consists of 8,000 images. Object center points primarily aggregate in the central region of the image, facilitating a uniform distribution of objects and a significant representation of small targets within the dataset, thereby establishing a solid basis for assessing the efficacy of the MCS-YOLO algorithm in autonomous driving perception.

Fig 2 Dataset images

iv) Image Processing:

Image processing is essential for object recognition in autonomous driving systems, involving numerous critical phases. The preliminary stage entails transforming the input image into a blob object, so enhancing it for further analysis and modification. Subsequently, the categories of objects to be recognized are established, specifying the precise classifications that the algorithm intends to recognize. Concurrently, bounding boxes are established, delineating the areas of interest within the picture where objects are anticipated to be situated. The analyzed data is subsequently transformed into a NumPy array, an essential procedure for effective numerical computation and analysis.

The next phase is loading a pre-trained model, utilizing established information from comprehensive datasets. This involves examining the network layers of the pre-trained model, which encompass learning characteristics and parameters essential for precise object identification. Furthermore, output layers are obtained, yielding conclusive predictions and facilitating efficient item identification and categorization.

Additionally, in the image processing pipeline, the picture and annotation file are combined, guaranteeing complete information for subsequent analysis. The color space is modified by converting from BGR to RGB, and a mask is generated to emphasize pertinent characteristics. The image is ultimately scaled to enhance its suitability for subsequent processing and analysis. This detailed image processing workflow provides a strong basis for reliable and precise object recognition in the evolving environment of autonomous driving systems, hence improving safety and decision-making on the road.

v) Data Augmentation:

Data augmentation is essential for developing diverse and strong training datasets for machine learning models, especially in image processing and computer vision. The original dataset is enhanced by randomizing, rotating, and warping the image.

Image variability is created by randomizing brightness, contrast, and color saturation. This stochastic technique improves model generalization to new data and various environments.

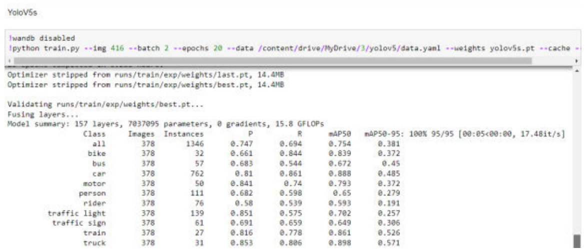
Changing the image's orientation by degrees is called rotation. This augmentation method teaches the model to detect objects from diverse angles, replicating real-world circumstances.

Scaling, shearing, and flipping change the picture. These distortions resemble real-world object look and orientation, enriching the dataset.

These data augmentation methods expand the training dataset, helping the model acquire robust features and patterns. This enhances the model's generalization and performance on different and difficult test conditions. Data augmentation helps reduce overfitting, improve model performance, and improve machine learning model dependability, notably in autonomous driving picture recognition.

vi) Algorithms:

YoloV5s: YoloV5 (You Only Look Once) detects objects quickly and accurately. It grids a picture and predicts bounding boxes and class probabilities for each cell. YoloV5s, the smaller variant, balances performance and efficiency.



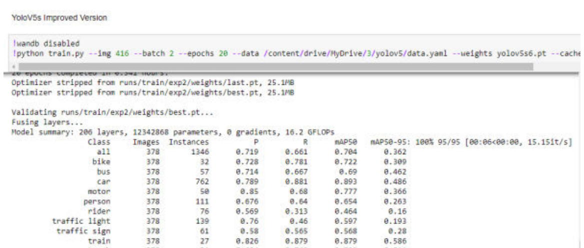
```
YoloV5s

!wandb disabled
!python train.py --img 416 --batch 2 --epochs 20 --data /content/drive/MyDrive/3/yolov5/data.yaml --weights yolov5s.pt --cache -
Optimizer stripped from runs/train/exp/weights/last.pt, 14.4MB
Optimizer stripped from runs/train/exp/weights/best.pt, 14.4MB

Validating runs/train/exp/weights/best.pt...
Fusing layers...
Model summary: 157 layers, 7037095 parameters, 0 gradients, 15.8 GFLOPs
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 95/95 [00:05<00:00, 17.48it/s]
                   all        378       1346      0.747      0.694      0.754      0.381
                  bike        378         32      0.661      0.844      0.839      0.372
                   bus        378         57      0.683      0.544      0.672       0.45
                   car        378        762       0.81      0.861      0.888      0.485
                 motor        378         50      0.841       0.74      0.793      0.372
                person        378        111      0.682      0.598       0.65      0.279
                 rider        378         76       0.58      0.539      0.593      0.191
         traffic light        378        139      0.851      0.575      0.702      0.257
          traffic sign        378         61      0.691      0.659      0.649      0.306
                 train        378         27      0.816      0.778      0.861      0.526
                 truck        378         31      0.853      0.806      0.898      0.571
```

Fig 3 YOLOV5s

YoloV5s Improved Version: This encompasses improvements beyond the fundamental YoloV5s, namely with architectural alterations, training methodologies, and hyperparameter optimization. Enhancements seek to augment precision and efficacy in object detecting endeavors.



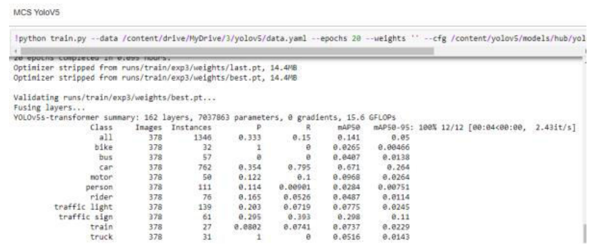
```
YoloV5s Improved Version

!wandb disabled
!python train.py --img 416 --batch 2 --epochs 20 --data /content/drive/MyDrive/3/yolov5/data.yaml --weights yolov5s6.pt --cache
Optimizer stripped from runs/train/exp2/weights/last.pt, 25.1MB
Optimizer stripped from runs/train/exp2/weights/best.pt, 25.1MB

Validating runs/train/exp2/weights/best.pt...
Fusing layers...
Model summary: 206 layers, 12342868 parameters, 0 gradients, 16.2 GFLOPs
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 95/95 [00:06<00:00, 15.15it/s]
                   all        378       1346      0.719      0.661      0.704      0.362
                  bike        378         32      0.728      0.781      0.722      0.309
                   bus        378         57      0.714      0.667       0.69      0.462
                   car        378        762      0.789      0.881      0.893      0.486
                 motor        378         50       0.85       0.68      0.777      0.366
                person        378        111      0.676       0.64      0.654      0.263
                 rider        378         76      0.569      0.313      0.464       0.16
         traffic light        378        139       0.76       0.46      0.597      0.193
          traffic sign        378         61       0.58      0.565      0.568       0.28
                 train        378         27      0.826      0.879      0.879      0.586
                 truck        378         31      0.694      0.742      0.792      0.511
```

Fig 4 YOLOV5s improved version

MCS YoloV5s: The MCS YoloV5s, presented in this study, integrates a coordinate attention module for the aggregation of spatial and cross-channel information. Furthermore, it utilizes a multiscale tiny object identification framework to augment sensitivity, hence boosting the recognition of dense small objects. The use of the Swin Transformer architecture significantly amplifies the network's emphasis on contextual spatial information [40].



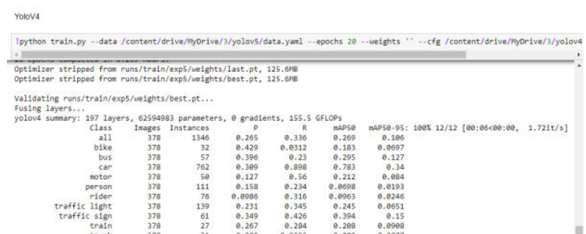
```
MCS YoloV5

!python train.py --data /content/drive/MyDrive/3/yolov5/data.yaml --epochs 20 --weights '' --cfg /content/yolov5/models/hub/yol
Optimizer stripped from runs/train/exp3/weights/last.pt, 14.4MB
Optimizer stripped from runs/train/exp3/weights/best.pt, 14.4MB

Validating runs/train/exp3/weights/best.pt...
Fusing layers...
YOLOv5s-transformer summary: 162 layers, 7037863 parameters, 0 gradients, 15.6 GFLOPs
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 12/12 [00:04<00:00,  2.43it/s]
                   all        378       1346      0.333       0.15      0.141       0.05
                  bike        378         32          1          0     0.0265    0.00466
                   bus        378         57          0          0     0.0407     0.0138
                   car        378        762      0.354      0.795      0.671      0.264
                 motor        378         50      0.122        0.1     0.0968     0.0264
                person        378        111      0.114    0.00901     0.0284    0.00751
                 rider        378         76      0.165     0.0526     0.0487     0.0114
         traffic light        378        139      0.203     0.0719     0.0775     0.0245
          traffic sign        378         61      0.295      0.393      0.298       0.11
                 train        378         27     0.0802     0.0741     0.0737     0.0229
                 truck        378         31          1          0     0.0516     0.0143
```

Fig 5 MCS YOLOV5s

YoloV4: YoloV4 represents an advancement in the Yolo series, prioritizing both velocity and precision. It incorporates technologies such as CSPDarknet53 as a backbone, PANet, and SAM block to enhance object detection.



```
YoloV4

!python train.py --data /content/drive/MyDrive/3/yolov5/data.yaml --epochs 20 --weights '' --cfg /content/drive/MyDrive/3/yolov4
Optimizer stripped from runs/train/exp5/weights/last.pt, 125.6MB
Optimizer stripped from runs/train/exp5/weights/best.pt, 125.6MB

Validating runs/train/exp5/weights/best.pt...
Fusing layers...
yolov4 summary: 197 layers, 62594983 parameters, 0 gradients, 155.5 GFLOPs
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 12/12 [00:06<00:00,  1.72it/s]
                   all        378       1346      0.265      0.336      0.269      0.106
                  bike        378         32      0.429     0.0312      0.183     0.0697
                   bus        378         57      0.396       0.23      0.295      0.127
                   car        378        762      0.309      0.898      0.783       0.34
                 motor        378         50      0.127       0.56      0.212      0.084
                person        378        111      0.158      0.234     0.0698     0.0193
                 rider        378         76     0.0986      0.316     0.0963     0.0246
         traffic light        378        139      0.231      0.345      0.245     0.0651
          traffic sign        378         61      0.349      0.426      0.394       0.15
                 train        378         27      0.267      0.284      0.208     0.0908
                 truck        378         31      0.289     0.0323      0.202     0.0877
```

Fig 6 YOLOV4

YoloV3: YoloV3 is a predecessor in the Yolo series, distinguished by a tri-stage detection methodology. It utilizes a Darknet-53 backbone and forecasts bounding boxes at various sizes. YoloV3 achieves a harmonious equilibrium between precision and velocity in object detecting endeavors.

2581-4575

International Journal For Recent Developments in Science & Technology

Crossref

A Peer Reviewed Research Journal

```
In [ ]: !python train.py --img 416 --batch 16 --epochs 20 --data /content/drive/MyDrive/3/yolov5/data.yaml --weights yolov3.pt
Optimizer stripped from runs/train/exp/weights/last.pt, 123.5MB
Optimizer stripped from runs/train/exp/weights/best.pt, 123.5MB

Validating runs/train/exp/weights/best.pt...
Fusing layers...
Model summary: 190 layers, 61545895 parameters, 0 gradients, 154.7 GFLOPs
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 12/12 [00:06<00:00,  1.79it/s]
                   all        378       1346      0.833      0.778      0.805      0.429
                  bike        378         32      0.922      0.906       0.94      0.409
                   bus        378         57      0.761      0.667      0.684      0.467
                   car        378        762       0.82      0.871      0.908      0.508
                 motor        378         50      0.893      0.831      0.899      0.462
                person        378        111      0.797       0.64      0.724      0.337
                 rider        378         76      0.813      0.776      0.794      0.308
         traffic light        378        139      0.795      0.642       0.67      0.256
          traffic sign        378         61      0.687      0.684       0.67      0.344
                 train        378         27      0.961      0.923        0.9      0.594
                 truck        378         31       0.88      0.839      0.865      0.608
Results saved to runs/train/exp
```

Fig 7 YOLOV3

YoloV3-tiny: YoloV3-tiny is a streamlined variant of YoloV3, designed for expedited inference on devices with limited resources. It compromises a degree of precision for enhanced speed, rendering it appropriate for real-time applications.

Yolov3-tiny

```
!python train.py --img 416 --batch 16 --epochs 20 --data /content/drive/MyDrive/3/yolov5/data.yaml --weights yolov3-tiny.pt
Optimizer stripped from runs/train/exp2/weights/last.pt, 17.5MB
Optimizer stripped from runs/train/exp2/weights/best.pt, 17.5MB

Validating runs/train/exp2/weights/best.pt...
Fusing layers...
Model summary: 38 layers, 8687482 parameters, 0 gradients, 12.9 GFLOPs
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 12/12 [00:04<00:00,  2.45it/s]
                   all        378       1346      0.701      0.593      0.657      0.292
                  bike        378         32      0.682        0.5      0.603      0.284
                   bus        378         57      0.645      0.439      0.593      0.318
                   car        378        762       0.73      0.828       0.85      0.422
                 motor        378         50      0.775      0.756      0.758      0.328
                person        378        111      0.651      0.369      0.451      0.165
                 rider        378         76       0.63      0.461      0.504      0.149
         traffic light        378        139      0.736      0.518      0.578      0.169
          traffic sign        378         61      0.697      0.603      0.677      0.318
                 train        378         27      0.731      0.815      0.801      0.375
                 truck        378         31      0.735      0.645      0.753      0.391
```

Fig 8 YOLOV3-tiny

Yolo V7: YOLOv7, an improved version, combines elements from YOLOv4, Scaled YOLOv4, and YOLO-R. The Extended Efficient Layer Aggregation Network (E-ELAN) improves learning, and Compound Model Scaling lets you alter width, depth, and resolution independently. With its speed, versatility, and accuracy in real-time object identification, YOLOv7 meets the project's needs.

```
!python yolov7/train.py --batch 32 --cfg cfg/training/yolov7-tiny.yaml --epochs 20 --data /content/data.yaml --weights 'yolov7/y
     Epoch   gpu_mem       box       obj       cls     total    labels  img_size
     19/19     4.57G   0.04382    0.0131  0.009903   0.06683        52       640: 100% 36/36 [00:22<00:00,  1.61it/s]
               Class      Images      Labels           P           R      mAP@.5  mAP@.5:.95: 100% 5/5 [00:05<00:00,  1.01s/i
t]
                 all         320        1140       0.594       0.695       0.664       0.316
                bike         320          26       0.616       0.808       0.748       0.344
                 bus         320          49       0.573       0.612       0.609       0.374
                 car         320         640       0.636       0.917       0.865       0.437
               motor         320          44        0.79       0.773       0.798       0.333
              person         320         100       0.513         0.7       0.625       0.252
               rider         320          63        0.35        0.46       0.298      0.0978
       traffic light         320         120       0.606       0.617       0.626       0.211
        traffic sign         320          54       0.506       0.741       0.702       0.288
               train         320          21       0.684       0.619       0.654       0.354
               truck         320          23       0.668         0.7       0.712       0.467
20 epochs completed in 0.152 hours.

Optimizer stripped from yolov7-tiny/run1/weights/last.pt, 12.3MB
Optimizer stripped from yolov7-tiny/run1/weights/best.pt, 12.3MB
```

Fig 9 YOLOV7

Faster RCNN: Faster R-CNN (Region-based Convolutional Neural Network) is a dual-phase object identification system. It utilizes a Region Proposal Network (RPN) to identify regions of interest and subsequently classifies those regions.

```python
        target["iscrowd"] = iscrowd

        if self.transforms is not None:
            sample = self.transforms(image=img,
                                     bboxes=target['boxes'],
                                     labels=labels)

        img = sample['image']
        target['boxes'] = torch.tensor(sample['bboxes'])
        target['labels'] = torch.tensor(sample['labels'])
        if target['boxes'].ndim == 1:
            target['boxes'] = torch.as_tensor([[0, 0, 640, 640]], dtype=torch.float32)
            target['labels'] = torch.zeros((1,), dtype=torch.int64)
        return img, target

    def __len__(self):
        return len(self.imgs)

def get_model_bbox(num_classes):
    # load an instance segmentation model pre-trained on COCO
    model = torchvision.models.detection.fasterrcnn_resnet50_fpn(pretrained=True)

    # get number of input features for the classifier
    in_features = model.roi_heads.box_predictor.cls_score.in_features
    # replace the pre-trained head with a new one
    model.roi_heads.box_predictor = FastRCNNPredictor(in_features, num_classes)

    return model

def get_transform(train):
    if train:
        return A.Compose([
            # A.Flip(p=0.5),
            # A.RandomResizedCrop(height=640,width=640,p=0.4),
            # # A.Perspective(p=0.4),
            # A.Rotate(p=0.5),
            # # A.Transpose(p=0.3),
            ToTensorV2(p=1.0)],
            bbox_params=A.BboxParams(format='pascal_voc',min_visibility=0.4, label_fields=['labels']))
```

Fig 10 Faster RCNN

AD-Faster RCNN: AD-FRCNN (Adaptive Dynamic Faster R-CNN) improves object detection performance by adding a dynamic region proposal network, a visual attention scheme for feature generation, and an adaptive dynamic training module [42].

AD-FasterRCNN

```python
def get_model_bbox(num_classes):
    # load an instance segmentation model pre-trained on COCO
    model = torchvision.models.detection.fasterrcnn_resnet50_fpn_v2(pretrained=True)

    # get number of input features for the classifier
    #in_features = model.roi_heads.box_predictor.cls_score.in_features
    # replace the pre-trained head with a new one
    #model.roi_heads.box_predictor = FastRCNNPredictor(in_features, num_classes)

    return model

def get_transform(train):
    if train:
        return A.Compose([
            # A.Flip(p=0.5),
            # A.RandomResizedCrop(height=640,width=640,p=0.4),
            # # A.Perspective(p=0.4),
            # A.Rotate(p=0.5),
            # # A.Transpose(p=0.3),
            ToTensorV2(p=1.0)],
            bbox_params=A.BboxParams(format='pascal_voc',min_visibility=0.4, label_fields=['labels']))
    else:
        return A.Compose([ToTensorV2(p=1.0)],
                         bbox_params=A.BboxParams(format='pascal_voc', min_visibility=0.5, label_fields=['labels']))

def reset_weights(m):
  '''
    Try resetting model weights to avoid
    weight leakage.
  '''
```

Fig 11 AD-FasterRCNN

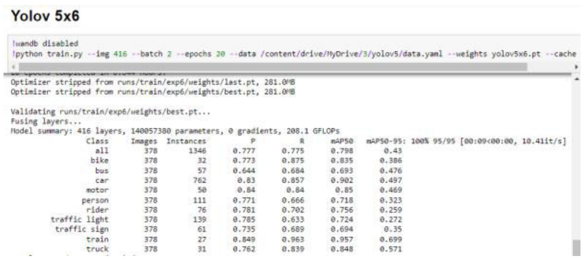
International Journal For Recent Developments in Science & Technology

2581-4575

Crossref

A Peer Reviewed Research Journal

Yolo V5x6, A fast and accurate form of the YOLO object detection model is optimized for this project. Its grid-based bounding box and class probability prediction gives it six times the processing capacity. Fast inference and precise object identification are essential for autonomous driving technology in varied road circumstances, and this computing increase is essential for satisfying project requirements.



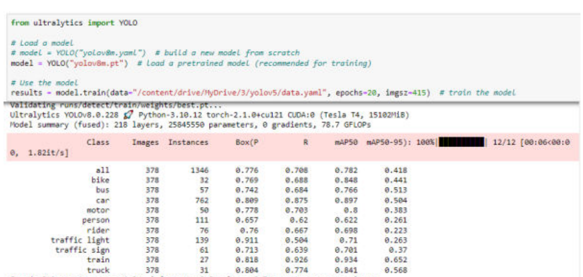
Yolov 5x6

```
!wandb disabled
!python train.py --img 416 --batch 2 --epochs 20 --data /content/drive/MyDrive/3/yolov5/data.yaml --weights yolov5x6.pt --cache
Optimizer stripped from runs/train/exp6/weights/last.pt, 281.0MB
Optimizer stripped from runs/train/exp6/weights/best.pt, 281.0MB

Validating runs/train/exp6/weights/best.pt...
Fusing layers...
Model summary: 416 layers, 140057380 parameters, 0 gradients, 208.1 GFLOPs
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 95/95 [00:09<00:00, 10.41it/s]
                   all        378       1346      0.777      0.775      0.798       0.43
                  bike        378         32      0.773      0.875      0.835      0.386
                   bus        378         57      0.644      0.684      0.693      0.476
                   car        378        762       0.83      0.857      0.902      0.497
                 motor        378         50       0.84       0.84       0.85      0.469
                person        378        111      0.771      0.666      0.718      0.323
                 rider        378         76      0.781      0.702      0.756      0.259
         traffic light        378        139      0.785      0.633      0.724      0.272
          traffic sign        378         61      0.735      0.689      0.694       0.35
                 train        378         27      0.849      0.963      0.957      0.699
                 truck        378         31      0.762      0.839      0.848      0.571
```

Fig 12 YOLOV5x6

YOLOv8, The YOLO series' top performer detects many objects simultaneously by gridding pictures and estimating bounding boxes and class probabilities. It supports Object Detection, Instance Segmentation, and Image Classification with a user-friendly API and high accuracy and speed. New architecture with C2f modules and an anchor-free head improves efficiency and versatility. For this project, YOLOv8 was chosen for robust, real-time object recognition.



```
from ultralytics import YOLO

# Load a model
# model = YOLO("yolov8m.yaml")  # build a new model from scratch
model = YOLO("yolov8m.pt")  # load a pretrained model (recommended for training)

# Use the model
results = model.train(data="/content/drive/MyDrive/3/yolov5/data.yaml", epochs=20, imgsz=415)  # train the model
Validating runs/detect/train/weights/best.pt...
Ultralytics YOLOv8.0.228 🚀 Python-3.10.12 torch-2.1.0+cu121 CUDA:0 (Tesla T4, 15102MiB)
Model summary (fused): 218 layers, 25845550 parameters, 0 gradients, 78.7 GFLOPs
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 12/12 [00:06<00:00,  1.82it/s]
                   all        378       1346      0.776      0.708      0.782      0.418
                  bike        378         32      0.769      0.688      0.848      0.441
                   bus        378         57      0.742      0.684      0.766      0.513
                   car        378        762      0.809      0.875      0.897      0.504
                 motor        378         50      0.778      0.703        0.8      0.383
                person        378        111      0.657       0.62      0.622      0.261
                 rider        378         76       0.76      0.667      0.698      0.223
         traffic light        378        139      0.911      0.504       0.71      0.263
          traffic sign        378         61      0.713      0.639      0.701       0.37
                 train        378         27      0.818      0.926      0.934      0.652
                 truck        378         31      0.804      0.774      0.841      0.568
Speed: 0.1ms preprocess, 4.2ms inference, 0.0ms loss, 0.8ms postprocess per image
```

Fig 13 YOLOV8

4. EXPERIMENTAL RESULTS

Precision: The extent of events or tests that are accurately sorted out of the multitude of ones that are marked as sure is called precision. Subsequently, coming up next is the recipe for deciding the precision:

Precision = True positives/ (True positives + False positives) = TP/(TP + FP)

$$\text{Precision} = \frac{\textit{True Positive}}{\textit{True Positive} + \textit{False Positive}}$$

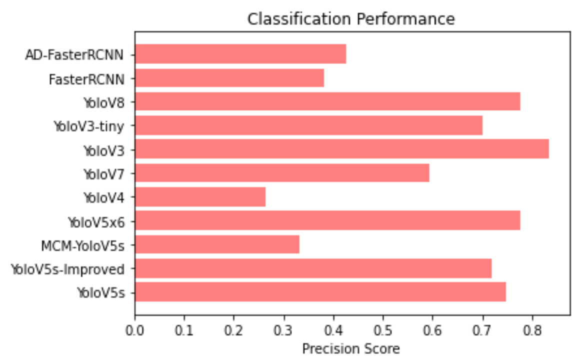


Fig 14 Precision comparison graph

Recall: In ML, recall is a proportion of how well a model can track down all examples of a particular class. This measurement reveals insight into how well a model catches occasions of a specific class, as it addresses the proportion of appropriately anticipated positive perceptions to the all-out genuine up-sides.

$$Recall = \frac{TP}{TP + FN}$$

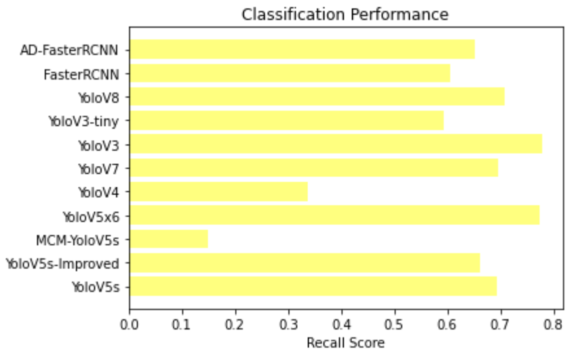


Fig 15 Recall comparison graph

mAP: Positioning quality measurements incorporate Mean Average Precision (MAP). It considers both the amount and positioning of appropriate ideas. To get MAP at K, we take the normal of all clients' or alternately inquiries' Average Precision (AP) at K and normal it out.

$$mAP = \frac{1}{n}\sum_{k=1}^{k=n} AP_k$$

$$AP_k = \text{the AP of class } k$$

$$n = \text{the number of classes}$$

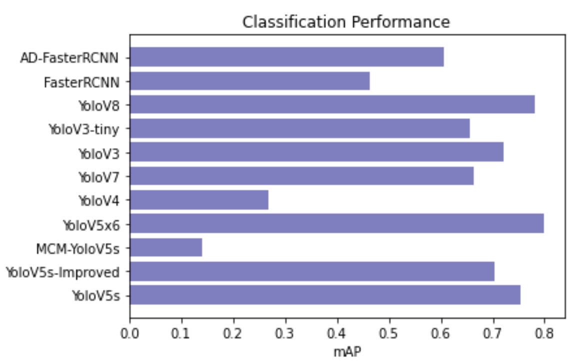


Fig 16 mAP comparison graph

	ML Model	Precision	Recall	mAP
0	YoloV5s	0.747	0.694	0.754
1	YoloV5s-Improved	0.719	0.661	0.704
2	MCM-YoloV5s	0.333	0.150	0.141
3	**YoloV5x6**	**0.777**	**0.775**	**0.798**
4	YoloV4	0.265	0.336	0.269
5	YoloV7	0.594	0.695	0.664
6	YoloV3	0.833	0.778	0.720
7	YoloV3-tiny	0.701	0.593	0.657
8	**YoloV8**	**0.776**	**0.708**	**0.782**
9	FasterRCNN	0.382	0.606	0.463
10	AD-FasterRCNN	0.427	0.653	0.605

Fig 17 Performance Evaluation table

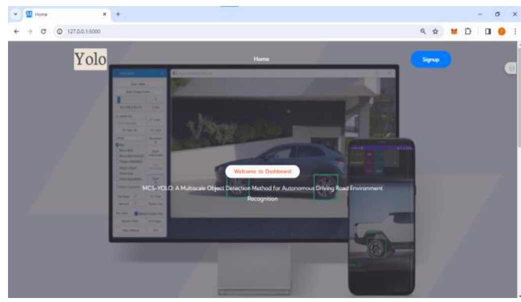


Fig 18 Home page

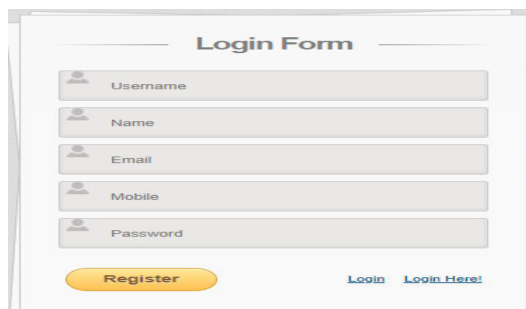


Fig 19 Registration page

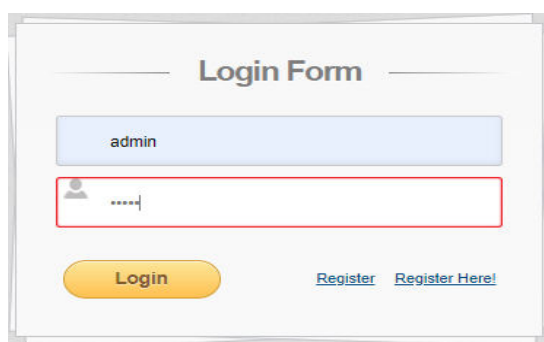


Fig 20 Login page

2581-4575
International Journal For Recent Developments in Science & Technology
Crossref
A Peer Reviewed Research Journal

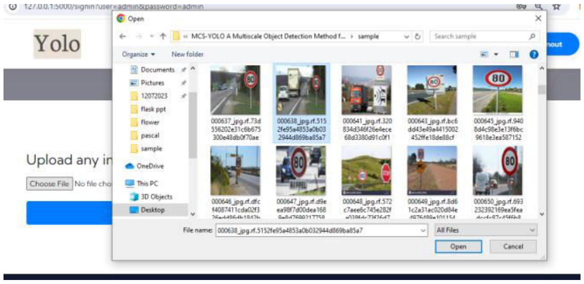


Fig 21 Input image folder

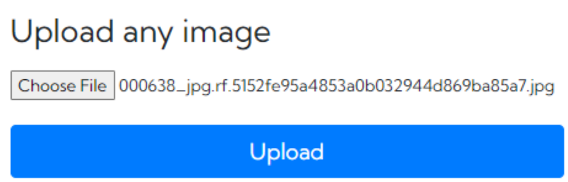


Fig 22 Upload input image

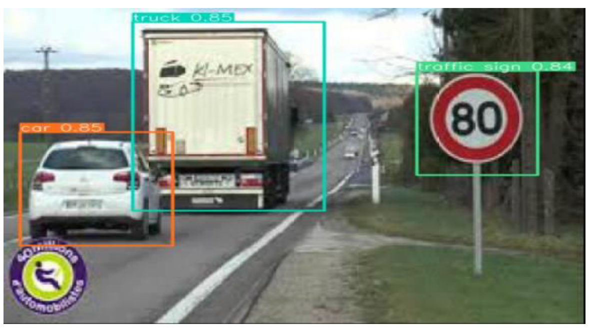


Fig 23 Predict result for given input

5. CONCLUSION

In conclusion, our research presents the MCS-YOLO algorithm, demonstrating its efficacy and superiority in object identification for autonomous driving. Utilizing a coordinate attention module, a multiscale tiny object detection framework, and the Swin Transformer, the technique markedly enhances detection precision and velocity. Ablation studies and comparison trials on the BDD100K dataset [41] highlight its significant performance improvements compared to previous techniques. Future pursuits entail the application of MCS-YOLO to the Multiple Object Tracking (MOT) task, guaranteeing its versatility and resilience in diverse autonomous driving contexts. This research tackles the urgent necessity to improve safety in autonomous driving in light of increasing incidents and traffic congestion. We enhance autonomous driving by transforming environmental perception using advanced deep learning algorithms, such as the upgraded YOLOv5s and the novel MCS-YOLOv5s [25,46]. Comparative assessments using benchmarks, investigation of sophisticated models, and integration with the Flask framework and SQLite for user testing demonstrate our dedication to technological excellence. Ultimately, the benefits encompass users and communities, as our approach fosters safer mobility, improved efficiency, less pollution, and further progress in autonomous driving technology.

6. FUTURE SCOPE

Future initiatives involve augmenting object identification proficiency through the integration of radar and LiDAR sensors for a thorough comprehension of the surroundings. Enhancing real-time processing entails utilizing improved hardware acceleration, parallel processing, and model compression to address dynamic situations. Investigating the seamless integration of edge computing seeks to decentralize processing, minimize latency, and improve flexibility, particularly in resource-limited or time-critical situations. Maintaining leadership in improvements necessitates ongoing investigation and incorporation of cutting-edge algorithms and structures, guaranteeing adaptability to new obstacles in autonomous driving technology [42].

REFERENCES

[1] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, M. Sokolsky, G. Stanek, D. Stavens, A. Teichman, M. Werling, and S. Thrun, ‘‘Towards fully autonomous driving: Systems and algorithms,’’ in Proc. IEEE Intell. Vehicles Symp. (IV), Jun. 2011, pp. 163–168.

[2] J. Janai, F. Güney, A. Behl, and A. Geiger, ‘‘Computer vision for autonomous vehicles: Problems, datasets and state of the art,’’ Found. Trends Comput. Graph. Vis., vol. 12, no. 1–3, pp. 1–308, 2020.

[3] Y. Wang, ‘‘Overview on key technology of perceptual system on selfdriving vehicles,’’ Auto Electr. Parts., vol. 2016, no. 12, pp. 12–16, Dec. 2016.

[4] M. Kuderer, S. Gulati, and W. Burgard, ‘‘Learning driving styles for autonomous vehicles from demonstration,’’ in Proc. IEEE Int. Conf. Robot. Autom. (ICRA), May 2015, pp. 2641–2646.

[5] R. Girshick, J. Donahue, T. Darrell, and J. Malik, ‘‘Rich feature hierarchies for accurate object detection and semantic segmentation,’’ in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2014, pp. 580–587.

[6] K. He, X. Zhang, J. Sun, and S. Ren, ‘‘Spatial pyramid pooling in deep convolutional networks for visual recognition,’’ IEEE Trans. Pattern Anal. Mach. Intell., vol. 37, no. 9, pp. 1904–1916, Jan. 2015.

[7] R. Girshick, ‘‘Fast R-CNN,’’ in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Dec. 2015, pp. 1440–1448.

[8] S. Ren, K. He, R. Girshick, and J. Sun, ‘‘Faster R-CNN: Towards realtime object detection with region proposal networks,’’ IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[9] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, ‘‘Mask R-CNN,’’ in IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 2980–2988.

[10] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, ‘‘You only look once: Unified, real-time object detection,’’ in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 779–788.

[11] J. Redmon and A. Farhadi, ‘‘YOLO9000: Better, Faster, stronger,’’ in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Honolulu, HI, USA, Jul. 2017, pp. 6517–6525.

[12] J. Redmon and A. Farhadi, ‘‘YOLOv3: An incremental improvement,’’ 2018, arXiv:1804.02767.

[13] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, ‘‘YOLOv4: Optimal speed and accuracy of object detection,’’ 2020, arXiv:2004.10934.

[14] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, ‘‘YOLOX: Exceeding Yolo series in 2021,’’ 2021, arXiv:2107.08430.

[15] C. Li, L. Li, H. Jiang, K. Weng, Y. Geng, L. Li, Z. Ke, Q. Li, M. Cheng, W. Nie, Y. Li, B. Zhang, Y. Liang, L. Zhou, X. Xu, X. Chu, X. Wei, and X. Wei, ‘‘YOLOv6: A single-stage object detection framework for industrial applications,’’ 2022, arXiv:2209.02976.

[16] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, ‘‘YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors,’’ 2022, arXiv:2207.02696.

[17] W. Liu, ‘‘SSD: Single shot multibox detector,’’ in Proc. Eur. Conf. Comput. Vis. Amsterdam, The Netherlands, 2016, pp. 21–37.

[18] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, ‘‘DSSD : Deconvolutional single shot detector,’’ 2017, arXiv:1701.06659.

[19] Z. Li and F. Zhou, ‘‘FSSD: Feature fusion single shot multibox detector,’’ 2017, arXiv:1712.00960.

[20] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, ‘‘Swin transformer: Hierarchical vision transformer using shifted Windows,’’ in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2021, pp. 9992–10002.

[21] Z. Liu, H. Hu, Y. Lin, Z. Yao, Z. Xie, Y. Wei, J. Ning, Y. Cao, Z. Zhang, L. Dong, F. Wei, and B. Guo, ‘‘Swin transformer v2: Scaling up capacity and resolution,’’ in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2022, pp. 12009–12019.

[22] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, ‘‘A ConvNet for the 2020s,’’ in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2022, pp. 11976–11986.

[23] Y. Chen et al., ‘‘DW-YOLO: An efficient object detector for drones and self-driving vehicles,’’ Arabian J. Sci. Eng., vol. 48, pp. 1–10, May 2022.

[24] Y. Zhou, S. Wen, D. Wang, J. Meng, J. Mu, and R. Irampaye, ‘‘MobileYOLO: Real-time object detection algorithm in autonomous driving scenarios,’’ Sensors, vol. 22, no. 9, p. 3349, Apr. 2022.

[25] H. Wang and W. Zang, ‘‘Research on object detection method in driving scenario based on improved YOLOv4,’’ in Proc. IEEE 6th Inf. Technol. Mechatronics Eng. Conf. (ITOEC), Mar. 2022, pp. 1751–1754.

[26] D. Tian, ‘‘SA-YOLOv3: An efficient and accurate object detector using self-attention mechanism for autonomous driving,’’ IEEE Trans. Intell. Transp. Syst., vol. 23, no. 5, pp. 4099–4110, May 2022.

[27] A. Gupta, K. Illanko, and X. Fernando, ‘‘Object detection for connected and autonomous vehicles using CNN with attention mechanism,’’ in Proc. IEEE 95th Veh. Technol. Conf., (VTC-Spring), Jun. 2022, pp. 1–6.

[28] H. Wang, Y. Xu, Y. He, Y. Cai, L. Chen, Y. Li, M. A. Sotelo, and Z. Li, ‘‘YOLOv5-fog: A multiobjective visual detection algorithm for fog driving scenes based on improved YOLOv5,’’ IEEE Trans. Instrum. Meas., vol. 71, pp. 1–12, 2022.

[29] Y. Li, J. Wang, J. Huang, and Y. Li, ‘‘Research on deep learning automatic vehicle recognition algorithm based on RES-YOLO model,’’ Sensors, vol. 22, no. 10, p. 3783, May 2022.

[30] J. Hu, L. Shen, and G. Sun, ‘‘Squeeze-and-excitation networks,’’ in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 7132–7141.

[31] S. Woo, J. Park, J. Y. Lee, and I. S. Kweon, ‘‘CBAM: Convolutional block attention module,’’ in Proc. Eur. Conf. Comput. Vis. (ECCV), Sep. 2018, pp. 3–19.

[32] Y. Liu, Z. Shao, and N. Hoffmann, ‘‘Global attention mechanism: Retain information to enhance channel-spatial interactions,’’ 2021, arXiv:2112.05561.

[33] X. Pan, C. Ge, R. Lu, S. Song, G. Chen, Z. Huang, and G. Huang, ‘‘On the integration of self-attention and convolution,’’ in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2022, pp. 815–825.

[34] Q. Hou, D. Zhou, and J. Feng, ‘‘Coordinate attention for efficient mobile network design,’’ in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2021, pp. 13708–13717.

[35] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, ‘‘Feature pyramid networks for object detection,’’ in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 936–944.

[36] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, ‘‘Path aggregation network for instance segmentation,’’ in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 8759–8768.

[37] G. Ghiasi, T.-Y. Lin, and Q. V. Le, ‘‘NAS-FPN: Learning scalable feature pyramid architecture for object detection,’’ in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 7036–7045.

[38] S. Qiao, L.-C. Chen, and A. Yuille, ‘‘DetectoRS: Detecting objects with recursive feature pyramid and switchable atrous convolution,’’ in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2021, pp. 10213–10224.

[39] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, ‘‘An image is worth 16×16 words: Transformers for image recognition at scale,’’ 2020, arXiv:2010.11929.

[40] J. Park, S. Woo, J. Lee, and I. Kweon, ‘‘BAM: Bottleneck attention module,’’ presented at the 29th Brit. Mach. Vis. Conf., Newcastle, U.K., Sep. 2018.

[41] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, ‘‘BDD100K: A diverse driving dataset for heterogeneous multitask learning,’’ 2018, arXiv:1805.04687.

[42] Y. Zhou, S. Wen, D. Wang, J. Mu, and I. Richard, ‘‘Object detection in autonomous driving scenarios based on an improved faster-RCNN,’’ Appl. Sci., vol. 11, no. 24, Dec. 2021, Art. no. 11630.

[43] L. Li and X. Li, ‘‘H-YoLov3: High performance object detection applied to assisted driving,’’ in Proc. Asia Conf. Algorithms, Comput. Mach. Learn. (CACML), Mar. 2022, pp. 462–467.

[44] F. Yang, X. Zhang, S. Zhang, C. Li, and H. Hu, ‘‘Design of real-time vehicle detection based on YOLOv4,’’ in Proc. Int. Conf. Control, Autom. Inf. Sci. (ICCAIS), Oct. 2021, pp. 824–829.

[45] Y. Li and F. Yu, ‘‘CDMY: A lightweight object detection model based on coordinate attention,’’ in Proc. IEEE 10th Joint Int. Inf. Technol. Artif. Intell. Conf. (ITAIC), Jun. 2022, pp. 1258–1263.

[46] Q. Luo, J. Wang, M. Gao, Z. He, Y. Yang, and H. Zhou, ‘‘Multiple mechanisms to strengthen the ability of YOLOv5s for real-time identification of vehicle type,’’ Electronics, vol. 11, no. 16, p. 2586, Aug. 2022.