Enhanced Safety of Autonomous Driving in Real-World Adverse Weather conditions via Deep Learning-Based Object Detection

by

Biwei Zhang

A thesis submitted
in partial fulfillment of the requirements for the degree of
Master of Systems Science and Engineering

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa
Ottawa, Ontario, Canada K1N 6N5

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Recognizing and categorizing objects in adverse weather conditions presents significant challenges for autonomous vehicles. To enhance the robustness of object detection systems, this thesis introduces an innovative approach that leverages sensors and deep learning-based solutions for object detection at various levels within a traffic circle. The proposed approach improves the effectiveness of single-stage object detectors, aiming to advance performance in autonomous racing environments by reducing instances of false detection and increasing recognition rates. The enhanced framework is based on the one-stage object detection model and incorporates multiple lightweight backbones.Additionally, attention mechanisms are integrated to further refine the object detection process. Our proposed model demonstrates superior performance compared to state-of-the-art methods on the DAWN dataset, achieving a mean average precision (mAP) of 99.1%, surpassing the previous result of 84.7%.

Object detection is one of the most fundamental challenges in computer vision. Over the past decade, the rapid evolution of deep learning has led researchers to extensively experiment with and enhance the performance of object detection, as well as related tasks such as object classification, localization, and segmentation using deep models. Despite these advancements, identifying and classifying objects in challenging weather conditions remains a significant difficulty for autonomous vehicles.

To improve the robustness of object detection systems, in this thesis we introduces an innovative approach for detecting objects at different levels by leveraging sensors and deep learning-based solutions within a traffic circle. We proposed an enhanced framework for the one-stage object detection model, incorporating multiple lightweight backbones such as ShuffleNetV2, GhostNet, MobileNetv3-Small, and VoVNet. We also integrated computer vision attention mechanisms, including SE Block, CBAM Block, and ECA Block. Additionally, we present a comparative analysis of the proposed single-stage object detectors, various YOLOv5 variants. Our results achieved a 99.1% Intersection over Union (IOU) mAP with an IOU threshold of 0.5, exceeding the state-of-the-art results of 94.7%. The suggested approach enhances the effectiveness of single-stage object detectors, aiming to improve performance in autonomous racing environments by reducing false detection and increasing recognition accuracy.

As a result, this research advances the efficiency of single-stage object detectors, significantly boosting their performance in autonomous racing scenarios under adverse weather conditions. This thesis presents significant enhancements to the performance of one-stage object detectors through the application of pruning and quantization methods. These improvements are designed to ensure safe operations, support global traffic management, and

enhance the mechanism and efficiency of practical 2D object detection under compromised visibility conditions.

Keywords: Object Detection, Bounding Boxes, Adverse weather, Self-driving Cars, YOLOv5,Neural Network Architecture, Single Shot Detection (SSD),LiDAR Technology, Sensor Fusion, Multi-Object Detection, CNN (Convolutional Neural Network), Image Segmentation, Detection Accuracy, Real-Time Processing.

Acknowledgements

I would like to express my deepest gratitude to my supervisor Dr.Burak Kantarci for his guidance and endless support to bring me into a field of Computer Vision and Image Processing at the Smart Connected Vehicles Innovation Centre (SCVIC), University of Ottawa. His knowledge and passion inspired me to become a researcher and their supervision empowers my skills and directions in Artificial Intelligence, whose insight and knowledge into the subject matter steered me through this research. I am deeply indebted to his encouragement and patience throughout the duration of my research and studies. I am grateful for the opportunity to work with them as well as for everything he has done for me.

I would like to thank Dr. Murat Simsek for guiding me to his important publications and for stimulating questions on Artificial Intelligence and Image Processing problems. The meetings and conversations were vital in inspiring me to think outside the box, from multiple perspectives to form a comprehensive and objective critique. I could not have done this without his support and guidance, thank you Dr. Murat.

Despite our great distance, I would like to express my profound gratitude to the members of the SCVIC for their unwavering support. They have stood by my shoulder through all of my challenges and have been my strongest supporters in every decision I have made. Their absolute belief in me has been the most important aspect in getting me to where I am.

Dedication

To the experience we never had and paths we were came across. To the people and friends we have along the way.

"It eluded us then, but that's no matter –tomorrow we will run faster, stretch out our arms farther.... And one fine morning– So we beat on, boats against the current, borne back ceaselessly into the past."

- F. Scott Fitzgerald

Table of Contents

List of Tables

List of Figures

Publications of the Candidate During MASc Studies

Outcomes of the thesis:

- **B. Zhang**, M. Simsek, M. Kulhandjian, B. Kantarci "Enhancing the Safety of Autonomous vehicles in Adverse Weather by Deep Learning-Based Object Detection" *MDPI Perception Sensors for Road Applications.*

Chapter 1

1. Introduction

The promise of various long-term benefits makes the idea of autonomous mobility quite appealing. These include reduced workloads related to driving and navigation, improved accessibility for people with impairments, increased safety standards that lead to fewer traffic accidents, and possible cost savings on infrastructure. However, one major challenge that has surfaced recently is the need to improve self-driving performance in harsh, worldwide weather scenarios. To improve passenger safety and dependability, manufacturers must put in place systematic, efficient, and precise detection systems before Autonomous Vehicles (AVs) and self-driving cars are seen on public roads worldwide. For workers on farms, construction sites, mines, and other locations where heavy-duty off-highway equipment is used, safety is crucial. The requirement to identify items and enhance overall safety has grown as manufacturers have pushed toward automating systems and equipment and as machines have become larger. For an autonomous automobile to sense its surroundings and produce a digital map, it needs sensory input devices like cameras, radar, and lasers. Fig. 1.1 shows the imaging where automobiles do object identification, which is what we will be concentrating on.

In AVs, software plays a crucial role in managing vast amounts of data so that users can comprehend their internal and external surroundings and make the most informed decisions. In this intricate system, sophisticated hardware and software components must function together. However, it is challenging to deconstruct software features and assign distinct roles to each component due to the complicated network of interconnected algorithms. Nonetheless, current research highlights that the four primary components of any AV's software architecture are perception, prediction, planning, and control.

1. Understanding the surroundings and making educated decisions are made possible

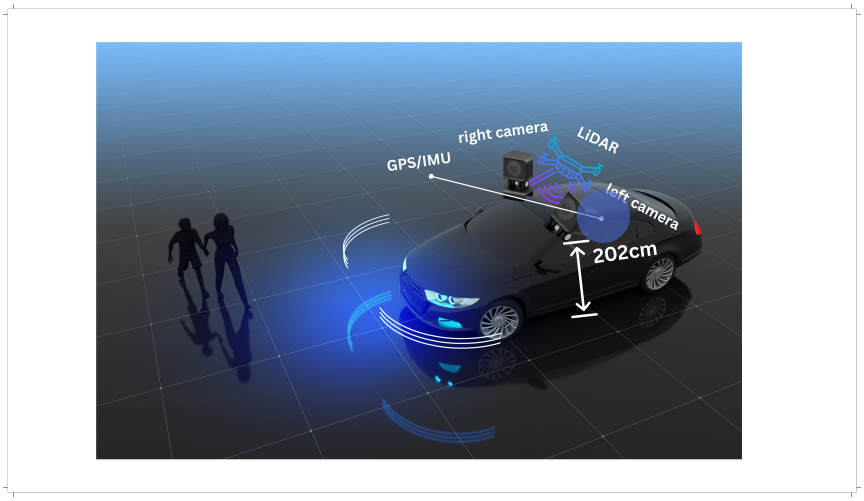


Figure 1.1: Sensor configuration for the collection of A2*D and 3D vehicle platform data. The A*STAR autonomous driving vehicle consists of a spinning Velodyne LiDAR and two colour PointGrey Chameleon3 cameras on either side of the LiDAR [1].

by the perception module, which transforms unprocessed sensor data into a coherent picture. Its responsibilities include determining the position of an item, identifying it, producing high-definition maps, and reconstructing the environment in three dimensions. Computer vision applications, in particular, leverage machine learning techniques in their sophisticated perception modules.

2. The prediction module is in charge of projecting the future behaviour of other entities in the traffic atmosphere. It assesses the likelihood of another vehicle's path at a junction, for example, to prevent collisions.

3. The navigation path selection process in the planning module makes use of an environment model that is created by observation and forecast. It usually distinguishes between actions at a lower level, like choosing a suitable path during a lane change, and decisions at a higher level, like changing lanes.

4. In the end, the control module sends vital signals to the hardware of the car, monitoring important parts such as the brakes, throttle, and steering torque to maintain the desired course.

The ability to identify and warn machine operators when there could be a human, animal, or inanimate item in their path is being improved by advancements in sensors and

other components utilized in many types of safety systems. As robots grow more automated, this capacity will be especially crucial since they will need to be able to observe their surroundings and know they are pursuing a safe path. For this reason, recent research work and technologies have improved detection capabilities. For instance, researchers have developed an end-to-end neural network in conjunction with Internet-of-Things (IoT) detects objects with high accuracy (99.1%) in 2D and 3D [4]. The state-of-the-art method outperforms current methods as the team aims to forge a path toward 2D and 3D detection systems for AVs[5].

AVs necessitate a diverse array of sensors for two primary purposes: (a) mapping and comprehending their environment and (b) ensuring redundancy, thereby ensuring that sensors' fields of view overlap in case of failure or malfunction. These sensors fall into two categories: exteroceptive (such as cameras, LIDARs, and RADARs) and proprioceptive (including GNSS and IMU). The former serves the function of collecting data about the external environment, while the latter is dedicated to providing information about the vehicle's relative positioning.

1. Cameras are used to capture images of the environment and turn them into a three-dimensional representation. They do, however, perform worse when there is less light, whether from inclement weather or insufficient sunshine. A special challenge for cameras is depth perception.

2. RADARs are good at estimating distances and speeds accurately, especially in low visibility conditions. Having said that, they struggle to recognize things' forms precisely, which makes it challenging to distinguish, for example, between an animal and a human.

3. LIDARs, or light detection and ranging systems, employ pulsed lasers and analysis of the reflected light to identify objects and determine distances. They are frequently regarded as the foundation of autonomous vehicle (AV) technology because of their capacity to produce precise three-dimensional maps of the surroundings of the vehicle. Like other sensors, LIDARs are restricted in low-visibility weather.

4. For precise car localization, the Global Navigation Satellite System (GNSS) is essential. It is usually used in conjunction with an Inertial Measurement Unit (IMU) to help the vehicle identify its orientation and angular rate with more accuracy.

1.1 Motivation

Autonomous, secure, and reliable car navigation in a variety of environments requires the use of a wide range of technologies related to signal processing, image processing, deep learning, edge computing, artificial intelligence (AI), and the Internet of Things (IoT). To ensure a safe driving experience, AVs need to be able to distinguish and identify their surroundings, as well as identify any threats to the safety of their occupants [6]. Finding and locating interesting objects in a given picture is the aim of object detection. Its close ties to other computer vision applications have piqued the curiosity of the community. The object recognition problem was approached using some traditional methods before notable progress was made in the field of deep learning. These methods were built with feature representations that were manually constructed. The inevitability of handcrafted elements limited the performance of traditional approaches.

In the age of deep learning, object detection has increasingly gained prominence. Three categories of algorithms are used in classical computer vision for object detection: single-stage detectors, two-stage detectors, and multiple-stage detectors. In this thesis titled Understanding a Real-Time Object Detection Network: You Only Look Once (YOLOv5), this thesis analyzes one of the earliest single-stage detectors. YOLOv5, an anchor-less design that reduces object detection to a simple regression problem, is a breakthrough for the object detection regime. It is much faster than popular two-stage detectors, such as fast-RCNN. The primary characteristics of YOLOv5 are its speed, faster version, and network understanding of generalized object representation, all of which contribute significantly to the emergence of object model problems [7].

To the knowledge, AVs rely on their sense of surroundings to function reliably and securely. Using the state-of-the-art (SOA) perception system YOLOv5 Object Detection Algorithms as the baseline, it achieves high-reliability recognition of important small things (pedestrians, bicycles, automobiles, motorbikes, buses, cyclists, and trucks) in the weather-adverse vehicle environment. This stream of work and study will be beneficial for signalized junction optimization, traffic operations throughout the winter, and global weather-adverse traffic management.

1.2 Objectives

Recent one-stage object detectors achieve good performance on datasets such as MS COCO[1] [8] and PASCAL VOC [2]. However, they cannot consider possible relations between image regions. The current one-stage detectors treat each image region separately. They are unaware of distinct image regions due to small receptive fields when image size is considered. They depend solely on high-quality local convolutional features to detect objects successfully. However, this is not the way the human visual system works. Humans can reason to carry out visual tasks with the help of acquired knowledge. Many techniques have been put out to simulate human object-detecting thinking. These techniques are often complex and rely on two-stage detection systems. As a result, real-time apps cannot use them.

Finding a reliable and effective approach for the state-of-the-art 2D one-stage object detection techniques is the main objective of this thesis to describe the structure as simply as possible. Different objects on the roadways with numerical results as well as visual outputs. Then, we analyze the most efficient training scheme for structure detection models with different backbones. An efficient annotation bootstrapping strategy is introduced to overcome the challenges of labelling table structure, and problems are discussed in detail. The thesis proposes the one-stage object detection model to detect table structure and create the most efficient output for semantic modelling.

1.3 Contributions

Following a thorough literature review, experiments were conducted to increase the 2D object detection level in the definition table structure and to find robust deep-learning models for detecting objects in weather-adverse conditions. We proposed the One-Stage Detection Multi-Backbone Compression Framework. We proposed the enhanced framework of the one-stage object detection model along with multiple lightweight backbones including ShuffleNetV2 [10], GhostNet [11], MobileNetv3-Small [12], and VoVNet [13], the Computer Vision Attention Mechanism. The primary contributions we have made can be outlined as follows:

- Propose the enhancements of the one-stage object detection module YOLOv5 as

[1]Microsoft COCO: Common Objects in Context

[2]The PASCAL VOC (Visual Object Classes) dataset [9] is a well-known object detection, segmentation, and classification dataset.

the baseline with multiple lightweight backbones such as ShuffleNetV2, GhostNet, VoVNet, and computer vision attention mechanisms, e.g.:SE Block [14], CBAM Block [15], ECA Block [16], Pruning, Quantization and Distillation for GhostNet on the weather adverse issues. A new architecture is proposed that can model and use semantic relations between image regions to predict bounding boxes and class probabilities.

- Incorporate Transformer Prediction Heads (TPH) into YOLOv5 to locate objects in densely populated scenes.
- Simplify the framework with the Pruning method, quantization, and distillation specifically tailored for addressing adverse weather-related issues to reduce the computational cost and size of the model.
- Enhance YOLOv5 with CBAM to improve the network's capacity to recognize regions of interest in photos with broad region coverage. To improve the ability to classify small categories of objects, we use a self-trained classifier, and our proposed framework achieves 99.7% AP, which beats the baseline method of [17] by 5%.

1.4 Thesis Outline

This thesis investigates the quantitative indicators exceed the state-of-the-art outcomes within this section of the thesis framework. An analysis is undertaken to examine the impact of compressed modelling on the average increase in precision for each type of object. The following background provides further details on the structure of this five-chapter thesis.

Chapter 1 of the thesis discusses the potential long-term benefits of autonomous mobility, such as reduced driving workloads, improved accessibility, enhanced safety, and cost savings on infrastructure. However, it emphasizes the challenge of improving AV performance in harsh weather conditions, highlighting the importance of effective detection systems for AV safety and reliability. AVs use a combination of sensory input devices and software components to perceive surroundings, predict behavior, plan navigation paths, and control vehicle hardware. Recent advancements in sensors and safety systems have enhanced AVs' hazard detection capabilities, critical for their safe operation.

The chapter also covers the diverse array of sensors required for AVs, including exteroceptive sensors for environmental data and proprioceptive sensors for vehicle positioning. The motivation section underscores the need for AVs to navigate various environments

safely using technologies like signal processing, image processing, deep learning, edge computing, AI, and IoT. Object detection is highlighted as a key focus, with YOLOv5 noted for its speed and efficiency. The thesis aims to improve object detection models for AVs in adverse weather, contributing a One-Stage Detection Multi-Backbone Compression Framework, enhancements to YOLOv5 with multiple lightweight backbones, and Transformer Prediction Heads (TPH) for better detection in densely populated scenes.

Chapter 2 explores both conventional and deep learning-based object detection methods, particularly their use in low-visibility road environments. It categorizes object detection tasks into image classification, object localization, and object detection, with an additional focus on object segmentation. The chapter emphasizes advancements in deep learning models like the YOLO variants, highlighting their speed and accuracy. It contrasts two-stage detectors such as R-CNN with one-stage detectors like YOLO and SSD, evaluating them using metrics like Mean Average Precision (mAP) and Intersection over Union (IoU). The evolution from YOLOv1 to YOLOv5 is detailed, showcasing improvements in network structure, training techniques, and overall performance.

The literature review also addresses the challenges of object detection in low visibility, emphasizing enhancements to YOLOv5 for better performance. Traditional methods focus on optimizing YOLOv5 through techniques like pruning and quantization to improve performance on low-capability devices.

The chapter highlights YOLOv5's unified model for its suitability in deep learning-based object detection, particularly in challenging scenarios like low-visibility road environments due to high frame rates and efficient detection capabilities. YOLOv5's architecture, including features like the SPP layer and BiFPN neck, improves efficiency while reducing convolutional layer parameters. Despite newer versions like YOLOv6, YOLOv7, and YOLOv8, the focus remains on YOLOv5 due to its extensive research foundation and balance of speed, accuracy, and simplicity.

Chapter 3 provides a detailed exploration of the enhancements made to the YOLOv5 architecture, aimed at improving object detection performance under challenging environmental conditions such as snow, rain, fog, and sandstorms. The chapter begins by outlining the fundamental components of YOLOv5, including its input processing, backbone (featuring focus, convolution, bottleneck CSP, and SPP modules), and neck (utilizing PANet for feature aggregation). It then introduces novel algorithmic advancements like the integration of transformer encoder blocks and the Convolutional Block Attention Module (CBAM) to enhance feature representation and improve the model's ability to detect objects of varying sizes and scales. Additionally, the chapter discusses the application of pruning techniques like EagleEye and network slimming to optimize computational effi-

ciency without sacrificing detection accuracy. Overall, these enhancements are designed to make YOLOv5 more robust and effective for real-time object detection in adverse weather conditions.

Chapter 4 evaluates the performance of YOLOv5 in comparison to the proposed one-stage detection framework. Initially, YOLOv5 is trained on the use-case data, and the results are contrasted with a relevant baseline model. This evaluation incorporates both standard metrics and those specific to our use case. Subsequently, we introduce an alternative corner grouping strategy and conduct a thorough evaluation of this approach. The datasets used, performance criteria, and ablation studies are detailed with numeric results presented in this chapter.

Chapter 5 gives a conclusion of all the approaches described in the study and bestows a conclusion and bibliography. This chapter refines the research problem and how the thesis improves the problems and presents the quantitative results findings, rephrases the quantitative indicators that are beating the state-of-the-art results. Additionally, there is an exploration of future directions and open issues that have arisen in conjunction with the results. Most importantly, this chapter underscores the significance of the thesis by discussing forthcoming steps and approaches, shedding light on its enduring value soon.

1.5 Quantitative Metrics

A detection-based metric seeks to quantify the object-detecting capabilities of the system. In this thesis, we evaluate the performance of the proposed model using a range of metrics to provide a comprehensive assessment. The most current results will be compared with the Confusion Matrix, Mean Average Precision (mAP), Intersection over Union (IoU), Recall, F1 Score, Frames Per Second (FPS), and Precision-Recall (PR) curves. The metrics shown below are computed to gauge the effectiveness and efficiency of the object detection system:

$$2 \cdot \frac{Precision * Recall}{Precision + Recall} = F1\ Score \tag{1.1}$$

$$\frac{Area\ of\ Overlap}{Area\ of\ Union} = IOU \tag{1.2}$$

Intersection over Union (IOU)

Intersection over Union (IOU) is used to assess the object detection algorithm. By computing the overlaps between the ground truth and the expected bounding box, it establishes how similar the predicted box is to the ground truth. If the IOU of a bounding box is less than the specified threshold, it is not taken into consideration.

The IOU computes the overlaps between two boundaries. This is the process by which we determine the degree of correspondence between our estimated border and the real object boundary, or ground truth. We established an IOU 0.5 threshold across many datasets to identify whether a forecast is a true positive or a false positive.

Mean Average Precision (mAP)

Mean Average Precision (mAP) is a metric used to evaluate object detection models such as Fast R-CNN [18], YOLO [19], Mask R-CNN [20], etc. The mean of average precision (AP) values are calculated over recall values from 0 to 1. In ImageNet [21], the AUC (Area under the ROC Curve) method is used. So even if they all follow the same principle in measurement AP, the exact calculation may vary according to the datasets. Fortunately, development kits are available to calculate this metric.

Precision and Recall

Calculate the area under the PR and mAP curves to get the average accuracy. The PR Curve is simplified to a single scalar number using the AP. The average precision is high when recall or precision are both low; it is low throughout a variety of confidence threshold values when both are low. Mean average precision ($mAP_{0.95}$) and mean average precision ($mAP_{0.5}$) are the two measurements that we mostly concentrate on. These evaluation metrics are represented by the following equation:

$$\frac{TP}{TP + FP} = Precision \tag{1.3}$$

$$\frac{TP}{TP + FN} = Recall \tag{1.4}$$

$$\frac{1}{n_{class}} P \int_0^1 P(R)dR = mAP_{0.5} \tag{1.5}$$

$$\frac{1}{N_{class}} P \sum_{i=1}^{n_{class}} = mAP_{0.5} \tag{1.6}$$

As you can see above, the initials TP, FP, and FN stand for true positives, false positives, and false negatives, respectively.

Confusion Matrix

A confusion matrix is a table employed to visually represent and quantify the performance of a classification algorithm by illustrating where the algorithm classified a value in comparison to the ground truth. In this study, the confusion matrix will be utilized as an additional evaluation tool during training, complementing other features such as real-time training and evaluation. The calculation of the confusion matrix relies on the detection outputs of the model, contrasting them with the ground truth labels. The test results shown in the terminal are dependent on the Intersection over Union (IOU) threshold settings and the given confidence level, which is vital to remember. To generate a confusion matrix, we need four attributes:

- True Positives (TP): The model predicted a label and matches correctly as per ground truth.
- True Negatives (TN): The model does not predict the label and is not a part of the ground truth.
- False Positives (FP): The model predicted a label, but it is not a part of the ground truth (Type I Error).
- False Negatives (FN): The model does not predict a label, but it is part of the ground truth. (Type II Error).

Table 1.1: Prediction result of Binary Classifier: if there is a match between the Ground Truth and the Prediction, it is True/False. Based on the outcome of the forecast, set Positive or Negative.

		Predicted label	
		+	-
True Label	+	True Postive	False Negative
	-	False Postive	True Negative

Chapter 2

Literature Review

The literature review explores traditional object detection methods and deep learning-based approaches specifically applied to road scenarios. The object detection module's deployment and implementation in low-visibility environments are the main areas of focus and challenge. Because of this, a Table 2.3 outlining the approach and a comprehensive explanation of the dataset are employed. The implementation of deep learning models is shown, reporting the precise Intersection Over Union (IoU) findings obtained at the end.

The general term object recognition describes a group of related computer vision tasks aimed at object recognition in digital pictures. Predicting the class of a single item in a picture is known as image classification. The process of locating one or more items and defining a bounding box around their extent is known as object localization. These two tasks are combined in object detection, which locates and categorizes one or more things in an image.

When object recognition is mentioned in talks between users or professionals, object recognition is usually mentioned. In the framework of this study, the terms object recognition and image classification are interchangeably used to describe the processes by which an algorithm ascertains the classes of items contained in an image and locates each object in the picture. Consequently, these three computer vision tasks may be distinguished from one another:

- Predict the kind or class of an object in a picture using image classification. An image containing just one item, such a picture, is the input. A class label is the result. For example, one or more numbers that are mapped to class labels.

- Object localization involves finding items in a picture and using a bounding box to show where they are. Input: An image, like a picture, that contains one or more items. One or more bounding boxes, for example, those with a point, width, and height, are the output.
- Object detection involves identifying the locations and classes of objects in an image using bounding boxes, resulting in a class label and corresponding bounding boxes (defined by a point, width, and height) for each detected object.

In the realm of computer vision tasks, an additional layer is introduced through object segmentation, also recognized as "object instance segmentation" or "semantic segmentation". Instead of using a more generic bounding box, this technique involves accurately recognizing object instances by defining the individual pixels that belong to the item.

This detailed analysis emphasizes that object identification is a broad category of complex computer vision problems. Current progress in tackling image identification problems frequently stems from working on projects like the ones the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [22] highlights.

The three kinds of difficulties are represented by distinct challenges in the annual ILSVRC academic competition. In order to allow for future expansion, its objective is to encourage specific, independent growth at each level. The three relevant task types from the 2015 ILSVRC review report are listed below for your reference:

- Image classification: algorithms produce a list of object categories present in the image.
- Single-object localization: a list of the item categories found in the picture is generated by algorithms, together with an axis-aligned bounding box that shows the location and size of a single instance of each object type.
- Object detection: an axis-aligned bounding box displaying the position and dimensions of each object category's occurrences is displayed alongside an algorithmic list of the image's object categories.

Single-object localization represents a simplified iteration of the more encompassing concept of object localization, narrowing down the localization tasks to objects of a single type within an image. This restriction is made under the assumption that dealing with a singular object type makes the task inherently less complex.

The object detection analysis in 2D object detection is the main emphasis of this thesis. The two sub-tasks that make up object detection are localization, which locates an item in an image (or video frame), and classification, which gives an object a class, such as "pedestrian", "vehicle", or "traffic light". Fig. 2.1 illustrates a taxonomy of the state-of-the-art deep learning-based object detectors. The taxonomy of these object detectors is discussed in this section.

Two-stage deep learning-based object detectors employ the two-step process of object classification and region recommendations. During the region proposal stage, the object detector in an input picture proposes many Regions of Interest (ROIs) with a high likelihood of having items of interest in the picture.

After removing the unnecessary things, the next step is categorizing the objects within the most potential ROIs. Well-known the two-stage detectors include R-CNN, Fast R-CNN, and Faster R-CNN. In contrast, one feed-forward neural network in one-stage object detectors achieves both bounding box construction and object categorization in the same step. These detectors are quicker than two-stage detectors, though sometimes less precise. YOLO [19], SSD [23], RetinaNet [24], and EfficientNet [25] are a handful of the popular single-stage detectors.

Fig. 2.1 illustrates the difference between the two types of object detectors. Both types of object detectors are typically evaluated using the mAP and IoU accuracy metrics. mAP is the mean of the ratio of precision to recall for individual object classes, with a higher value indicating a more accurate object detector. The overlap between the ground truth bounding box and the anticipated bounding box is measured by IoU. In formal terms, IoU is the ratio of the area of union and the area of overlap between the (bounding and ground truth) boxes. Fig. 2.1 illustrates the IoU of an object detector prediction and the ground truth. A very accurate IoU is shown in Fig. 2.1 (a), whereas a less accurate IoU is shown in Fig. 2.1 (b).

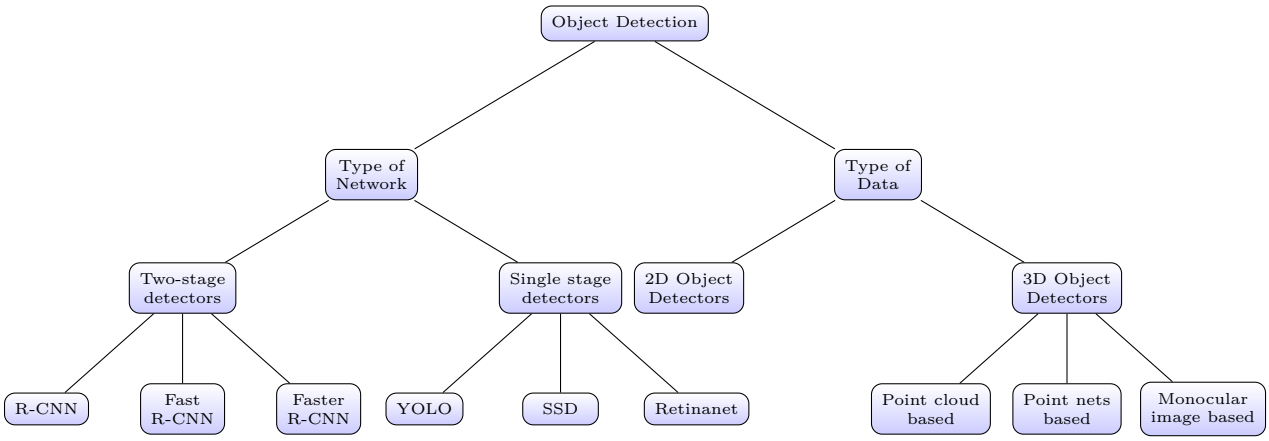


Figure 2.1: Taxonomy of Object Detectors.

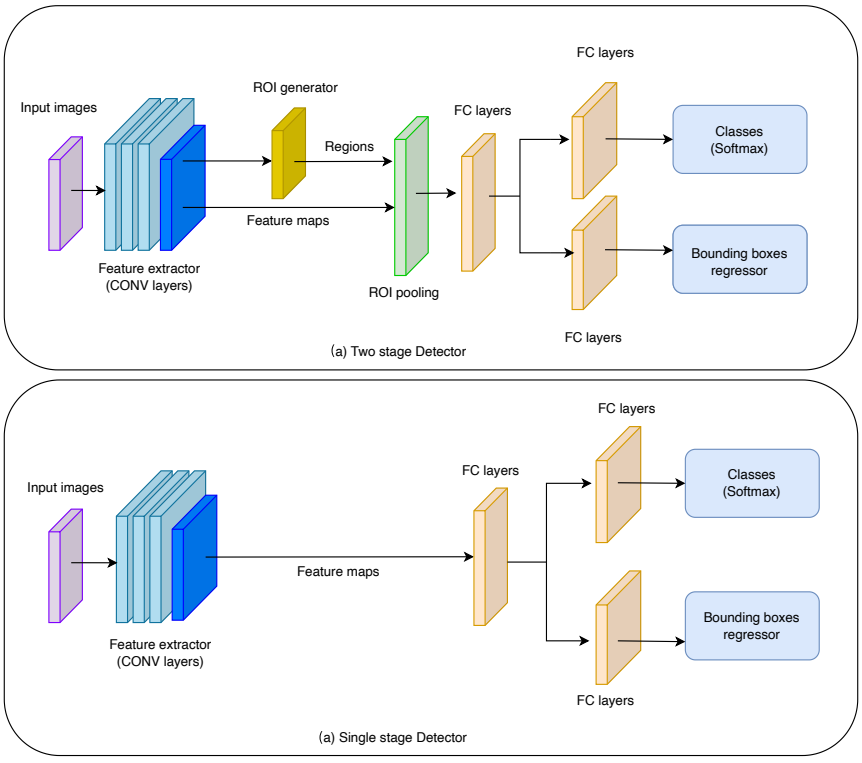


Figure 2.2: Classification of Object Detectors.

R-CNN is one of the first deep learning-based object detectors and used an efficient selective search algorithm for ROI proposals as part of a two-stage detection. Fast R-CNN solved some problems in the R-CNN model, such as low inference speed and accuracy. A Convolutional Neural Network (CNN) is fed the input picture in the Fast R-CNN model to produce an ROI projection and feature map. These ROIs are then mapped to the feature map for prediction using ROI pooling. Unlike R-CNN, instead of feeding the ROI as input to the CNN layers, Fast R-CNN uses the entire image directly to process the feature maps to detect objects. Like Fast R-CNN, Faster R-CNN adopted a similar strategy, except instead of feeding the ROI to the ROI pooling layer and the feature map, which were subsequently reshaped and used for prediction, it used a separate network.

Single-stage object detectors such as YOLO [19], You only look once, are faster than two-stage detectors as they can predict objects on an input with a single pass. The first YOLO variant, YOLOv1, learned generalizable representations of objects to detect them faster. In 2017, YOLOv2 [26] improved upon YOLOv1 by adding batch normalization, a high-resolution classifier, and use of anchor boxes to create bounding boxes instead of using a fully connected layer like YOLOv1. In 2018, YOLOv3 [27] was proposed with a 53-layered backbone-based network that used an independent logistic classifier and binary cross-entropy loss to predict overlapping bounding boxes and smaller objects. Single-Shot Detector (SSD) [23] models were proposed as a better option to run inference on videos and real-time applications as they share features between the classification and localization tasks on the whole image, unlike YOLO models that generate feature maps by creating grids within an image. While the YOLO models are faster than SSD, they trail behind SSD models in accuracy. Even though YOLO and SSD models provide good inference speed, they have a class imbalance problem when detecting small objects. This issue was addressed in the RetinaNet [24] detector that used a focal loss function during training and a separate network for classification and bounding box regression.

In 2020, YOLOv4 [28] introduced two important techniques: 'bag of freebies' which involves improved methods for data augmentation and regularization during training and 'bag of specials' which is a post-processing module that allows for better mAP and faster inference. YOLOv5 [29], which was also introduced in 2020, proposed further data augmentation and loss calculation improvements. It also used auto-learning bounding box anchors to adapt to a given dataset. Another variant called YOLOR, You Only Learn One Representation, [30] was proposed in 2021 and used a unified network that encoded implicit and explicit knowledge to predict the output. With a single model, YOLOR [31] can carry out multitask learning tasks including feature embedding, multi-label picture classification, and object identification. A decoupled head method, which is anchor-free, enables the network to conduct classification and regression utilizing different networks,

as demonstrated by the YOLOX [32] model, which was also suggested in 2021. YOLOX features a faster inference rate and fewer parameters than the YOLOv4 and YOLOv5 [32] models. Finally, the timeline of the presented state-of-the-art solutions in this study for object detection are provided in Fig. 2.3 and Fig. 2.4

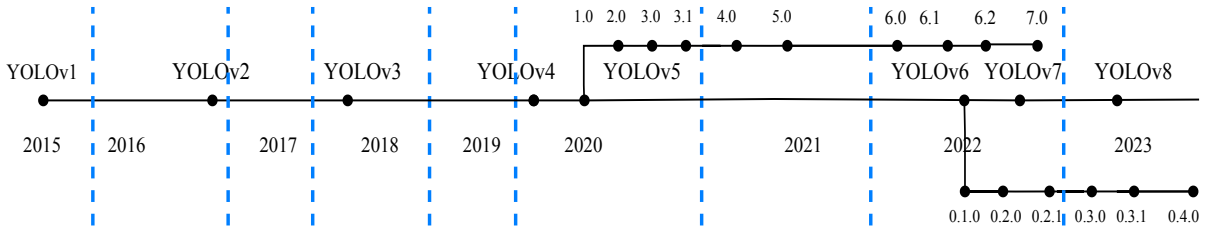


Figure 2.3: YOLO release timeline. YOLOv5 and YOLOv6 have ten and six released variants, respectively.

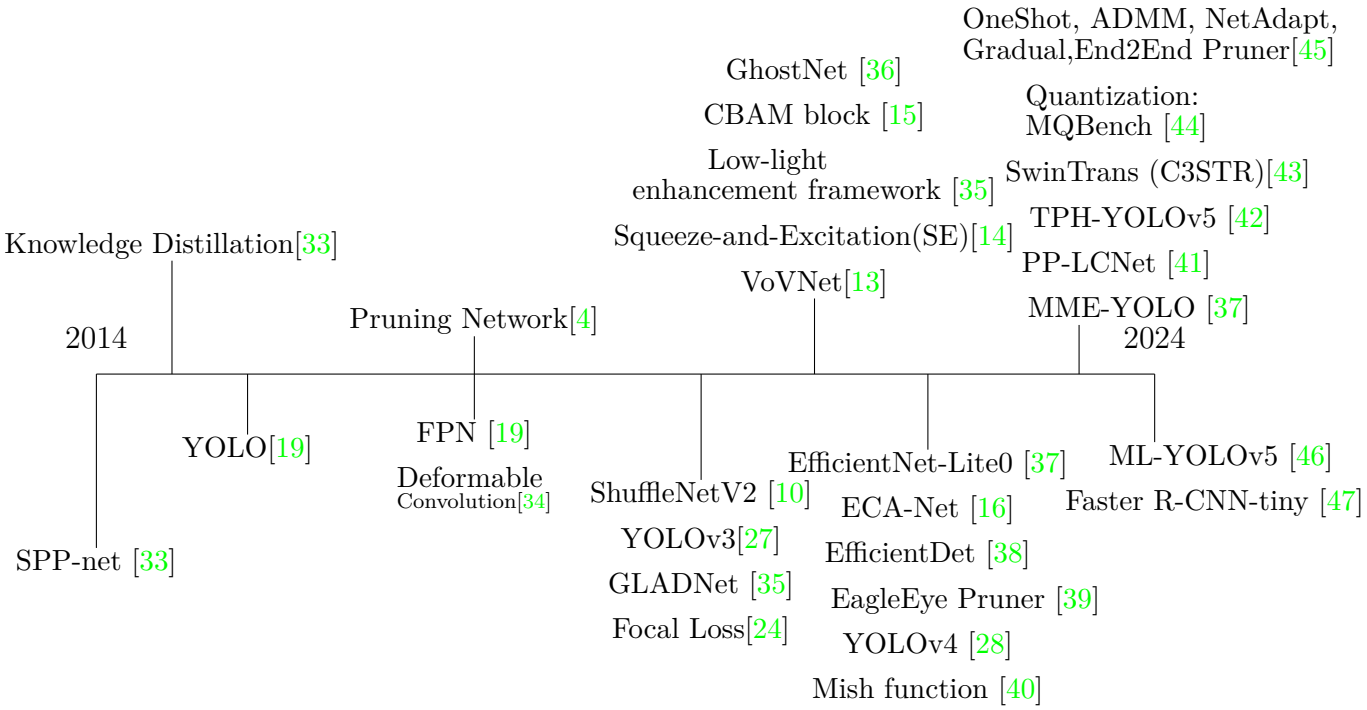


Figure 2.4: Brief history of presented algorithms with the timeline.

2.1 Traditional Object Detection Algorithms

Numerous research projects have attempted to enhance YOLO object detectors over time. Since its launch, eight versions have been issued to improve YOLO's accuracy and efficacy. Although YOLO's apparent benefits have led to its widespread adoption in many fields, there are issues when using it on devices with low capabilities.

In this section, the methodologies used in the training session are described to enhance the one-stage detection methodology YOLOv5 by treating each model as an independent entity and modifying its structure individually at all scales.

In the first step, YOLOv5 is selected to establish the baseline. The data argumentation and transformation techniques are not employed in the data preparation for the baseline model implementation since the augmented dataset will have the biases of the original dataset, it is crucial to create early tactics to prevent bias. The majority of data augmentation processes change the distribution of data as training goes on. This creates a "data

bias" in the data distribution between the enhanced and original data, which could make it challenging to properly utilize the augmented data. It can be hypothesized that data augmentation may not be evident if the model is trained well enough for any identifiable difference. Despite many encouraging results, it is not the case that data augmentation always improves generalization errors. Particularly, shows that training by augmented data will lead to a smaller robust error but potentially a larger standard error. The drawback is that, at the initial phase of our research, there are just too numerous combinations of augmentations for anyone to find the ideal one.

Section 2.1 discusses the traditional object detection approaches. The suggested IoU boundary box regression loss function, which relies on the YOLO object identification method, aims to increase the form consistency of the actual and predicted boxes. Push loss is also included to improve the boundary box regression loss function and increase the number of occluded items that may be detected.

For many years, object detection on the road has been a challenging area of research due to its fundamental nature. An essential component of automated driving technology is road object identification. A model with better detection accuracy promotes safer driving practices. The issue of tiny and obscured items being missed in road object detection is significant. For this reason, lowering the object's missed rate is crucial to driving safely.

The main goal of object detection involves identifying and localizing objects of interest from different categories within some given image. Object detection is the basis of many other advanced computer vision tasks ranging from semantic segmentation to object tracking to activity recognition. Lately, state-of-the-art results in object detection tasks have been attained using deep learning-based methods like Convolutional Neural Networks (CNNs). As a result of the advancements in computational power and cutting-edge algorithms, object detection has become more accurate, enabling a wide range of real-world applications. The issues of feature extraction, classification, and localization in object identification are lessened when CNNs are used as part of traditional object detection techniques. Single-stage and two-stage detection are the two common approaches used for object detection. In the former case, the method creates a collection of area suggestions and classifies them as backgrounds or objects; in the latter case, the algorithm first predicts the bounding boxes and class probabilities for objects. Unlike Faster RCNN and RFCN as two-stage object detection methods, single-stage ones, like YOLOv5, SSD, EfficientDet [38], and RetinaNet [24], typically use one Fully Convolutional Neural Network (FCNNs) [48] to detect objects' classes and spatial locations without intermediate steps.

A state-of-the-art architecture for object detection consists of a two-stage detector, which separates the detection process into two stages: region suggestion and classification.

The two-stage detection models also known as the multiple-stage detection models are one of the most fundamental and widely researched challenges in computer vision object detection. The most representative two-stage object detector is the R-CNN (Region-Based Convolutional Neural Networks) family, which also includes Fast R-CNN, Faster R-CNN, and Mask-R-CNN. The two-stage detector is challenging to train and understand, even if the R-CNN series produces results on object detection accuracy (mAP) that are comparatively good. The objective of the undertaking is to create many bounding boxes around each object in the image, a crucial problem in many domains, including autonomous driving.

Multi-stage models are often the foundation of one subset of object detectors. One model is used to extract areas of objects derived from R-CNN work, while another model is used to categorize and improve the object's localization. These techniques are renowned for being extremely effective yet rather sluggish.

Furthermore, the dynamic anchor box mechanism is employed to enhance the confidence label's correctness and the object detection model's label inaccuracy in the absence of an anchor box. Numerous tests on the DAWN dataset show how effective the suggested method is. On the DAWN dataset, the improved YOLOX-s achieved 88.9% mAP and 91.0% mAP, compared to 2.77% and 4.24% for the baseline version improvements; on the same dataset, the improved YOLOv5 achieved 89.1% mAP and 91.4% mAP, compared to 2.30% and 4.10% for the baseline version improvements.

Since its publication in 2016, YOLOv5 has garnered significant interest among various single-stage object identification techniques. The primary idea behind YOLOv5 is to divide an input image into a grid of cells and predict bounding boxes and class probabilities for each cell. YOLOv5 treats object detection as a regression problem. Also, since it uses a single neural network for object detection and classification, it can be optimized jointly for both tasks, which enhances the whole detection performance. YOLOv1 was equipped with a simple structure containing 24 convolutional layers and two fully connected layers at the end to deliver probabilities and coordinates. Since its introduction, YOLO has undergone several improvements and variations. In 2017, YOLOv2 was published with improvements in the performance led by using multi-scale training, anchor boxes, batch normalization, Darknet-19 [26] architecture, and a modified loss function. Following that, Redmon and Farhadi *et al.* introduced YOLOv3, which employs a feature pyramid network, convolutional layers with anchor boxes, Spatial Pyramid Pooling (SPP) block [49], Darknet-53 architecture, and an improved loss function. Unlike previous versions, YOLOv4 was introduced by different authors. A. Bochkovskiy *et al.* [28] enhanced YOLO's performance by utilizing CSPDarknet53 architecture [50], Bag-of-Freebies, Bag-of-Specials, Mish activation function [40], Weighted-Residual-Connections (WRC) [51], Spatial Pyramid Pooling

(SPP) [49], and Path Aggregation Network (PANet) [52].

2.2 Object Detection Algorithms in Low Visibility

As part of the effort to enhance the one-stage detection methodology YOLOv5, we have detailed the approaches utilized in poor visibility settings in this section. We achieve this by considering each model as a distinct entity and altering its structure consistently across all sizes. As a first step, we selected the optimal baseline model, YOLOv5.

Since pruning and quantization are often employed as modular compression strategies, we concentrate on them in this thesis. Knowledge distillation necessitates the employment of two models or the modification of the target network's structure. We examine the pruning and quantization techniques applied to YOLOv5 compression in the last several years and compare the outcomes using keywords related to compression. Since YOLOv5 is the most recent version and has sufficient research on pruning and quantization, we have decided to concentrate on it. Despite recent improvements over YOLOv5, the applied compression algorithms of the current versions of YOLO remain inadequate for review. Numerous analyses of neural network compression techniques have been conducted; in this study, however, we focus on the practical applications of these techniques on YOLOv5. All of the effort that went into quantizing and trimming YOLOv5 is provided, along with the findings from all angles. In general, changes in memory footprint, power consumption, FLOPs, inference time, FPS, and accuracy may be used to represent the compression outcomes.

For the implementation of the baseline model, we chose the ideal one-stage object detection model YOLOv5 in the first stage to set the baseline. We chose not to overuse the data argumentation and data transformation techniques in the data preparation as the augmented dataset will carry the biases of the existing dataset, and it is important to develop initial strategies to avoid bias in an early stage [53]. Most data augmentation operations alter the data distribution during the training progress. This imposes a data distribution bias as the "data bias" between the augmented data and the original data, which may make it difficult to fully leverage the augmented data [5]. It can be hypothesized that data augmentation may not be evident if the model is trained well enough for any identifiable difference. Despite many encouraging results, it is not the case that data augmentation will always improve generalization errors [54]. Particularly, [53] showed that training by augmented data will lead to a smaller robust error but potentially a larger standard error. Also, the downside is that there are way too many augmentation variations for anyone to identify the right combination in the first step of our investigation.

YOLOv5 is a family of compound-scaled object detection models trained on the COCO dataset and includes simple functionality for Test Time Augmentation (TTA), model ensembling, hyper-parameter evolution, and export to ONNX, CoreML and TFLite. Glenn Jocher released YOLOv5 with many differences and improvements. (Notably, Glenn is the creator of mosaic augmentation, which is an included technique in what improved YOLOv4.) The release of the improved YOLOv5 includes five different models' sizes: YOLOv5s (small), YOLOv5m (medium), YOLOv5l (large), YOLOv5x (extra-large). Each of these scales applies a different multiplier to the depth and width of the model, meaning the overall structure of the model remains constant, but the size and complexity of each model are scaled. In the experiments, the changes are applied to the structure of the models individually across all the scales and treat each one as a different model to evaluate the effect.

$$Pr(Classi|Object) * Pr(Object) * IoU = Pr(Classi) * IoU \tag{2.1}$$

The final predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor.

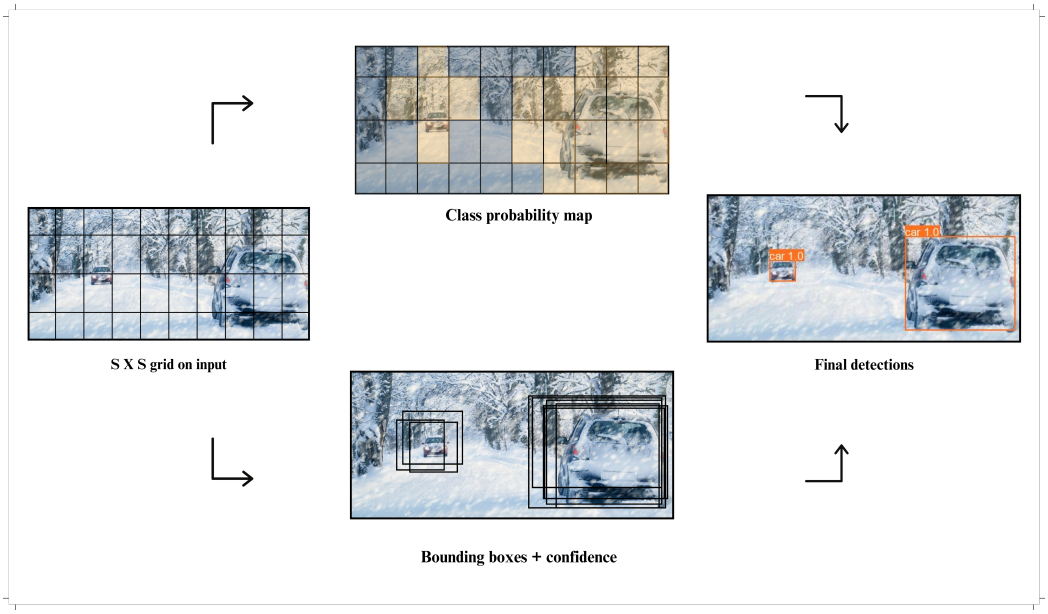


Figure 2.5: The procedure outlines the workflow of YOLOv5. Each image is segmented into uniformly sized boxes, and bounding boxes are drawn. The width of each line within the box corresponds to the confidence ratings.

Table 2.1: The performance results of YOLOv5 baseline training process.

Methodology	**Precision(%)**	**Recall(%)**	**F1**	**mAP**$_{0.5}$(%)	**mAP**$_{0.95}$(%)
YOLOv5 - L	0.85	0.90	0.86	0.90	0.90
YOLOv5 - M	0.92	0.81	0.88	0.93	0.74
YOLOv5 - N	0.87	0.80	0.91	0.90	0.74
YOLOv5 - S	0.89	0.83	0.80	0.88	0.75
YOLOv5 - X	0.90	0.85	0.80	0.90	0.69

Table 2.2: The performance results during the enhanced YOLOv5 training process.

Paper	**Methodology**	**V**	**P(%)**	**R(%)**	**F1**	**mAP**$_{0.5}$(%)	**mAP**$_{0.95}$(%)
[4]	TPH-Slimming Pruned (Our)	5s	0.94	0.81	0.89	96.8%	76.1%
[55]	YOLOv5x (Our)	5s	0.75	**0.99**	0.66	**99.1%**	**90.0%**
[55]	YOLOv5s (Our)	5m	0.95	0.81	0.88	98.5%	80.9%
[55]	YOLOv5s+YOLOv5xP2CBAM (Our)	5s	**0.97**	0.80	**0.91**	98.1%	73.5%
[55]	YOLOv5xP2+YOLOv5s (Our)	5s	0.90	0.83	0.80	97.2%	75.1%
[17]	YOLOv4-tiny	-	0.82	0.52	0.64	49.9%	29.1%
[17]	YOLOv3-tiny	-	0.80	0.59	0.68	51.3%	25.4%
[17]	YOLOv5-baseline	-	0.91	0.82	0.90	84.7%	65.1%

2.3 Overview of Proposed One-Stage Approaches

This section 2.3 initially examines and discusses the overview of object detection algorithms. Afterwards, it explores a conversation about object detection algorithms designed for situations with poor visibility. In the end, an overview of the proposed one-stage methods is provided.

A single convolutional network simultaneously predicts multiple bounding boxes and class probabilities for those boxes. YOLOv5 trains on full images and directly optimizes detection performance. This unified model has several benefits over traditional methods of object detection. The YOLOv5 architecture has brought huge cutting-edge advancements to the traffic perception problem, and Fig. 2.6 indicates the YOLOv5 Architecture. First, YOLOv5 is extremely fast. Since we frame detection as a regression problem, we do not need a complex pipeline. We simply run our neural network on a new image at test time to

predict detection. Our base network runs at 45 frames per second with no batch processing on a Titan X GPU and a fast version runs at more than 150 frames per second (FPS). This means we can process streaming video in real-time with less than 25 milliseconds of latency.

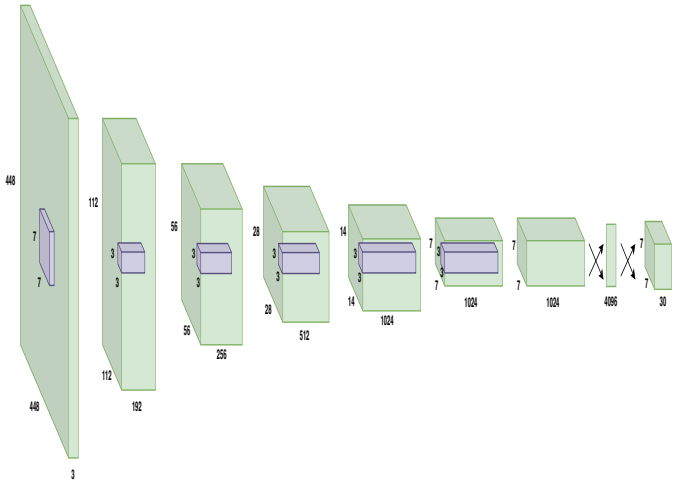


Figure 2.6: YOLOv5 Architecture is generated by Visual Keras [2] and ChatGPT [3]

Second, YOLOv5 reasons globally about the image when making predictions. Unlike sliding window and region proposal-based techniques, YOLOv5 sees the entire image during training and test time so it implicitly encodes contextual information about classes as well as their appearance. Fast R-CNN [18], a top detection method, mistakes background patches in an image for objects because it can not see the larger context. YOLOv5 makes less than half the number of background errors compared to Fast R-CNN.

Third, YOLOv5 learns generalizable representations of objects. When trained on natural images and tested on the artwork, YOLOv5 outperforms top detection methods like DPM [56] and R-CNN by a wide margin. Since YOLOv5 is highly generalizable it is less likely to break down when applied to new domains or unexpected inputs.

Our network uses features from the entire image to predict each bounding box. It predicts all bounding boxes across all classes for an image simultaneously. This means our network reasons globally about the full image and all the objects in the image. The YOLOv5 design enables end-to-end training and real-time speeds while maintaining high average precision. The system divides the input image into an $S \times S$ grid. If the center of an object falls into a grid cell, that grid cell is responsible for detecting that object.

Each grid cell predicts B bounding boxes and confidence scores for those boxes. These confidence scores reflect how confident the model is that the box contains an object and how accurate it thinks the box is that it predicts. Formally we define confidence as $Pr(Object) \times IoU$. The confidence score should be zero if no object exists in that cell. Otherwise, we want the confidence score to equal the IoU between the predicted box and the ground truth.

Each bounding box consists of 5 predictions: x, y, w, h, and confidence. The (x, y) coordinates represent the box's center relative to the grid cell's bounds. The width and height are predicted relative to the whole image.

Finally, the confidence prediction represents the IoU between the predicted box and any ground truth box. Each grid cell also predicts C conditional class probabilities, $Pr(Class_i \mid Object)$. These probabilities are conditioned on the grid cell containing an object. We only predict one set of class probabilities per grid cell, regardless of the number of boxes B. At test time we multiply the conditional class probabilities and the individual box confidence predictions.

As sharing features improved two-stage detectors to have similar computational cost to single-stage detectors. These works are highly dependent on previous works and mostly build on the previous pipeline as a baseline. Therefore, it is important to understand all the main algorithms in two-stage detectors.

Compared to the other region proposal classification networks such as, Fast-RCNN which perform detection on various region proposals and thus end up performing prediction multiple times for various regions in an image, YOLO architecture is more like FCNN (Fully Convolutional Neural Network) and passes the image $(n \times n)$ once through the FCNN and output is $(m \times m)$ prediction. This architecture is splitting the input image in a $(M \times M)$ grid and for each grid generation two bounding boxes and class probabilities for those bounding boxes. Note that the bounding box is more likely to be larger than the grid itself. A single Convolutional Network simultaneously predicts multiple bounding boxes and class probabilities for those boxes. YOLOv5 trains on full images and directly optimizes detection performance.

In this section, the methodologies used in the training session are presented as we aim to improve the one-stage detection methodology YOLOv5 by changing the structure of the models individually across all the scales and treating each one as a different model.

[1]The framework implements new features: (1) remove the CBAM module in the positions of 14 and 19 in Fig. 3.2 (2) reduce the number of channels associated with P2 head in original YOLOv5 from 256 to 128.(3) add SPP layer in between the backbone and the neck.4) carry out output after CBAM (5) only

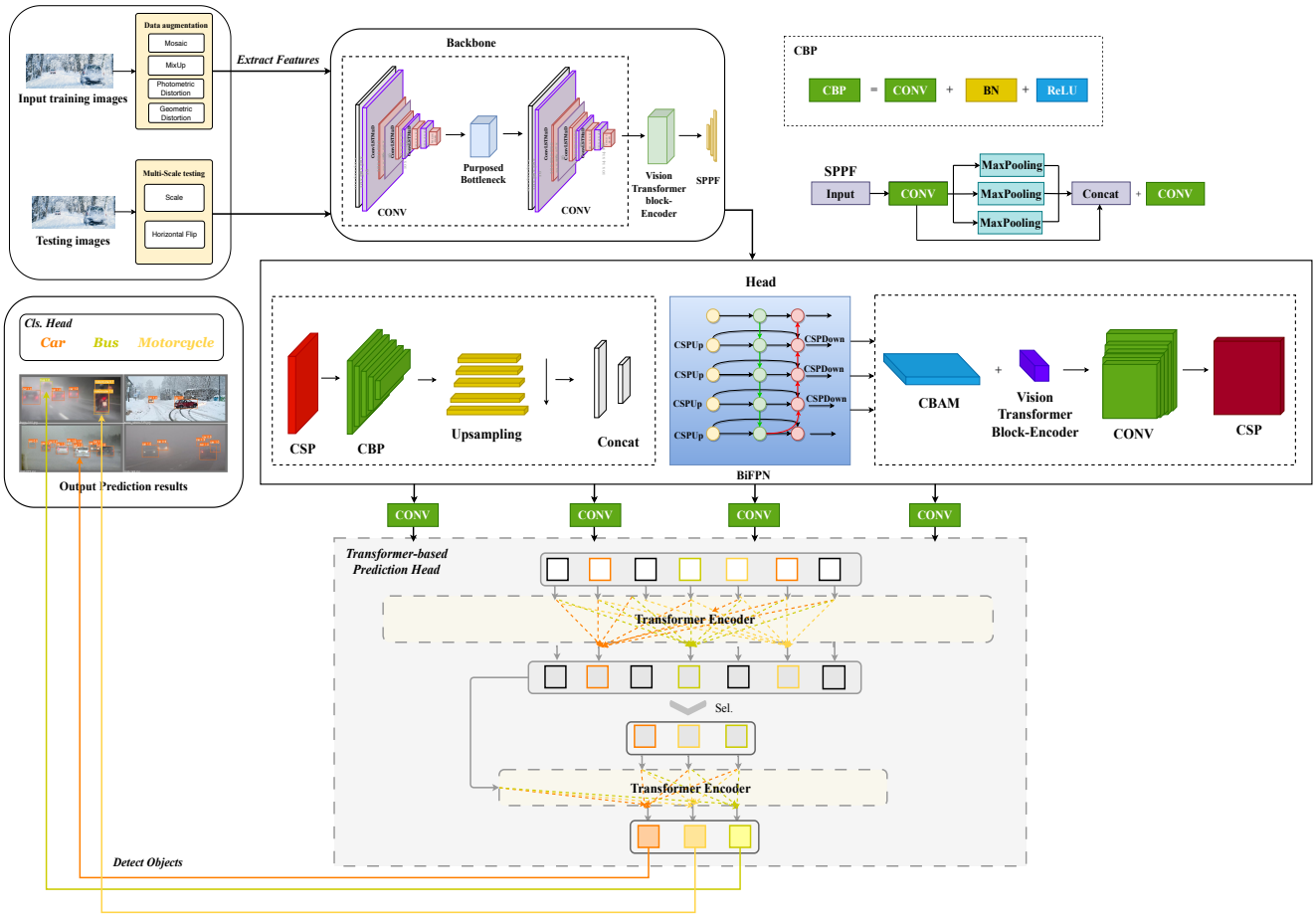


Figure 2.7: the Proposed Framework of one-stage detection module. [1]

In 2020, Ultralytics released YOLOv5 in five sizes, ranging from micro to extra big. YOLO saw significant changes, including automatic hyper-parameter tuning and a new backbone architecture. With an additional Cross Stage Partial (CSP) technique [57], the new CSPDarknet53 structure, built on the foundation of Darknet53, is employed by YOLOv5 in the Backbone. Utilizing CSP-PAN and a quicker SPP block variant (SPPF) [49], the Neck design of YOLOv5 adopts these technologies. Utilizing the Head structure of YOLOv3 [27], the output is produced. The structure of YOLOv5l is illustrated in Fig. 2.7 in which CSPDarknet53 contains C3 blocks which are the CSP-fused modules. CSP

retain backbone and last layer of output TransBlock (6) Adopt BiFPN as neck. These changes ensure a lighter implementation while lowering the parameters of the Convolutional Layer.

strategy partitions the feature map of the base layer into two parts and then merges them through a cross-stage hierarchy. Therefore, the C3 module can effectively handle redundant gradients while improving the efficiency of information transfer within residual and dense blocks. C3 is a simplified version of BottleNeckCSP and is currently used in the latest variant of YOLOv5. For comparison, the design of C3 and BottleNeckCSP blocks is depicted in Fig. 2.7. Overall, these modifications have enabled YOLOv5 to achieve state-of-the-art performance on several benchmarks for object detection, including the COCO dataset. Also, different model sizes provide the user with the opportunity to choose based on their need. In 2022, YOLOv6 [58] was released by Meituan, featuring enhancements led by a Bi-directional Concatenation (BiC) module, an anchor-aided training (AAT) strategy, and a new backbone and neck design. After a very short period, introduced by the original authors, YOLOv7 [59] was a breakthrough. Wang *et al.* proposed a Bag-of-Freebies, a compound model scaling method, and an extended ELAN architecture to expand, shuffle, and merge cardinality. The Bag-of-Freebies refers to a planned re-parameterized convolution inspired by ResConv, an extra auxiliary head in the middle layers of the network for Deep Supervision, and a soft label assigner guiding both the auxiliary head and the lead head by the lead head prediction. Lastly, Ultralytics presented YOLOv8 [60] in 2023 with several alterations in the backbone, neck, and head; A C2f module is used instead of C3; a decoupled head provides the output; and the model directly predicts the center of the object instead of the anchor box. While YOLOv6, YOLOv7 and YOLOv8 are more featured models, this work focuses on YOLOv5 as more well-established studies have been done on it. However, this investigation can be expanded to newer versions of YOLO, especially YOLOv8.

The following Table 2.3 is the reference list of state-of-the-art work related to the concern of adverse weather as this thesis discovers and explores the strategies of self-attention mechanism and attention-guided mechanism to solve the object detection challenges in weather-adverse conditions.

2.4 Rationale Behind Choosing YOLO

In section 2.4, the rationale for selecting the YOLO (You Only Look Once) one-stage detector architecture for object detection lies in its numerous compelling advantages, highlighting its versatility and benefits in various applications. This informed the decision to integrate YOLO into our proposed framework.

Compared to traditional two-stage detectors, YOLO provides an appealing alternative by delivering high accuracy without sacrificing real-time processing speed. These attributes

make YOLO a top choice for contemporary object detection tasks, aiming to enhance efficiency and responsiveness in dynamic operational scenarios.

To be specific, YOLO excels in real-time image processing, making it ideal for applications like autonomous vehicles where quick decision-making is crucial for safety. Unlike traditional multi-stage detectors, YOLO's single-pass efficiency enhances speed. While two-stage detectors are accurate, they often lag in speed. With iterations like YOLOv3 to YOLOv5, it achieves high precision and recall, making it versatile for environments from surveillance to industrial automation. Its end-to-end training simplifies integration by combining localization and classification within a single network, reducing complexity and boosting efficiency. Additionally, the YOLO family offers models tailored to various computational needs, allowing developers to select the best option for their projects, ensuring optimal performance across different use cases.

2.4.1 Speed and Real-Time Processing

Where YOLO truly excels is in its speed, uniquely capable of processing images in real-time. Traditional object detection algorithms, while accurate, often depend on multi-stage processing, which can be significantly slower and may not meet real-time requirements. YOLO, on the other hand, is renowned for its exceptional speed, performing object detection with a single pass through the network. This makes it ideal for applications requiring rapid decision-making based on precise object detection.

Unlike traditional approaches that involve complex multi-stage pipelines, YOLO's single-pass processing is highly efficient. This efficiency is crucial for applications like AVs, where swift decision-making based on real-time detection of pedestrians, vehicles, and obstacles is essential for ensuring safety and responsiveness.

2.4.2 Accuracy and Performance

The two-stage detectors are renowned for their accuracy in image recognition tasks but often entail more complex multi-stage pipelines for object detection. While they can achieve high accuracy, they may lag behind YOLO in terms of speed. YOLO's architecture has evolved through iterations such as YOLOv3, YOLOv4, and YOLOv5 to achieve high precision and recall rates in detecting objects across different scales and orientations.

This balanced performance makes YOLO a versatile choice for diverse environments and scenarios, ranging from surveillance systems to industrial automation.

2.4.3 End-to-End Training and Unified Framework

YOLO's end-to-end training approach significantly simplifies the deployment and integration of object detection systems. By integrating localization and classification tasks into a unified neural network architecture, YOLO effectively reduces computational complexity and enhances overall system efficiency. This streamlined approach not only accelerates the development process but also improves the accuracy and reliability of object detection across various applications.

Moreover, the unified framework of YOLO facilitates seamless adaptation to challenging datasets, such as those encountered in adverse weather conditions or autonomous vehicle environments. This adaptability is crucial for ensuring robust performance in scenarios where environmental factors may affect visibility or object detection accuracy.

In essence, YOLO's end-to-end training not only optimizes system efficiency and accuracy but also enhances its versatility and reliability in diverse real-world applications, ranging from safety-critical autonomous vehicles to surveillance systems in challenging environments.

2.4.4 Versatility and Adaptability

The YOLO family stands out for its exceptional versatility and adaptability, offering a diverse array of models specifically designed to cater to varying computational resources and performance demands. This inherent flexibility empowers developers to select the most suitable YOLO variant for their distinct project requirements, whether these pertain to resource-constrained embedded systems or high-performance computing environments. Furthermore, YOLO's architecture is highly amenable to fine-tuning and customization, which enhances its ability to address unique challenges and achieve optimal performance metrics tailored to specific use cases.

2.4.5 Comparison with Other Approaches

While there are alternative object detection architectures such as Faster R-CNN, SSD, RetinaNet, and EfficientDet, YOLO stands out for its combination of speed, accuracy, and simplicity. These attributes make YOLO particularly well-suited for applications where real-time processing, high detection accuracy, and ease of implementation are critical factors.

In conclusion, the rationale behind choosing YOLO for object detection revolves around its capability to deliver real-time performance without sacrificing accuracy, its streamlined end-to-end training process, adaptability to diverse computing environments, and proven effectiveness across various applications. These factors collectively position YOLO as a leading choice in the field of computer vision and object detection.

Table 2.3: Comparison of the state-of-the-art solutions for object detection.

Ref.	Methodology	Datasets	Benefits	Future Improvements	Features
[12]	MobileNetV3-small architecture	Cityscapes Dataset.	Lightweight backbone	Advanced the detection speed and accuracy.	Expand diversity of small samples in training data for improving accuracy in learning.
[10]	ShuffleNetV2	ImageNet 2012 classification Dataset.	Lightweight backbone	Advance detection speed for small-scale road object detection.	Enrich shallow scale features. Improve context interaction.
[11]	GhostNet	CIFAR-10, ImageNet ILSVRC 2012 Dataset, MS COCO.	Lightweight backbone	Potentially higher recognition performance under GhostNet	Achieve higher recognition performance.
[25]	EfficientNet-Lite0	CIFAR-10,CIFAR-100, Birdsnap, Stanford Cars,Flowers, etc.	Lightweight backbone	Advance detection speed.	Light-weight; faster network.
[41]	PP-LCNet	Cityscapes Dataset.	Lightweight backbone	Advance detection speed	Improve performance of lightweight models on multiple tasks and network accuracy at nearly constant latency.
[42]	TPH-YOLOv5	VisDrone2021 Dataset.	Lightweight backbone	Improve detection accuracy.	Achieve state-of-the-art performance; improve the accuracy of object detector.
[43]	SwinTrans (C3STR)	COCO2017, COCO2014, BDD100k, Visdrone, Hand.	Transformer Block Module	Advance detection speed and accuracy.	Better speed accuracy trade-off. Generailzable approach and design.
[34]	Module: Deformable Convolution	ImageNet dataset.	Advance the detection speed and accuracy.	Deformable ConvNets.	Improve accuracy of object detection using object boundary and various deformation object through an offset added to the position of each pixel point.
[4]	Pruning: Network Slimming	CIFAR and SVHN datasets.	Pruning method.	Advance detection speed and accuracy.	Distil knowledge in an ensemble models into a single model. Decrease computational cost with no accuracy loss.
[39]	Pruning: EagleEye	COCO dataset.	Pruning method	Advance detection speed.	Extract image features into visual embeddings, pass alongside text embeddings to BERT.
[44]	Quantization: MQBench	ImageNet dataset.	Quantization algorithms	Advance the detection speed and accuracy.	Reduce computational overhead, improve empirical robustness of deep neural networks(DNNs).
[33]	Knowledge Distillation	MNIST and ImageNet.	Model Compression method	Advance detection speed and accuracy.	Lighter model with less compute requirements; superior performance and higher accuracy; feasible for scarce training data.
[39]	EagleEye Pruner	ImageNet; CIFAR-10.	Pruning method	Improve efficiency of pruning by adaptive batch normalization.	Reduce computational overhead, improve empirical robustness of deep neural networks(DNNs).
[45]	OneShot, ADMM, NetAdapt, Gradual, End2End Pruner	MNIST dataset.	Pruning method	Advance detection speed and accuracy.	Less time consuming leads to optimal results.

Chapter 3

Methodology and Algorithm Design

The objective of Chapter 3 is to comprehensively elucidate the adopted methodology, focusing prominently on the enhancements introduced to the original YOLOv5 architecture. This chapter highlights methodological advancements and algorithmic modifications aimed at enhancing the model's resilience in the challenging environmental conditions. These improvements specifically address varying levels of image noise and adverse weather conditions such as snow, rain, fog, and sandstorms.

3.1 Algorithm

The proposed framework in the thesis is based on YOLOv5, the fifth iteration in the YOLO (You Only Look Once) series of object detection algorithms, and was introduced in [19] in June 2020. Similar to its predecessors, YOLOv5 is a single-stage detector network, which allows for faster and more stable processing compared to other object detection algorithms, making it particularly suitable for real-time applications. This section 3.1 provides a comprehensive overview of the enhanced algorithm, detailing each step from input processing to final output.

The structure of enhanced architecture consists of 4 modules: input, backbone, neck, and prediction. The backbone network of YOLOv5 includes focus, convolution, bottleneck CSP (Cross Stage Partial), and spatial pyramid pooling (SPP) modules. The focus module processes input images of shape $3 \times 640 \times 640$, where 3 represents the three color channels (typically red, green, and blue), and 640×640 denotes the image's width and height in pixels. It is adaptable to different image sizes and channel numbers. The primary goal of

the focus module is to enhance training speed using the hard-swish activation function, a modified version of the swish activation function that substitutes the sigmoid function with a piecewiselinear hard equivalent [61], as described in equations 3.1 and 3.2. Here, Σ is the sigmoid function, and ReLU6 is a variant of the rectified linear unit (ReLU) activation function, capped at a maximum value of 6.

$$swish(x) = x\sigma(x) = \frac{x}{1 + e^{-x}}, \tag{3.1}$$

$$h - swish(x) = x\frac{ReLU6(x + 3)}{6}, \tag{3.2}$$

The bottleneck CSP module improves feature extraction by utilizing a residual network, concatenating deeper features with shallow ones. The SPP module generates a fixed-size feature vector from varying size feature maps using three parallel max-pooling layers.

The neck network ensures detection accuracy across different object scales and sizes by employing a path aggregation network (PANet) [52] to enhance feature propagation regardless of scale. Modules 17, 20, and 23 of the PANet, also known as P3, P4, and P5, output feature maps for small, medium, and large-scale objects, respectively. P3 produces a feature map of 80×80 pixels, P4 outputs 40×40 pixels, and P5 outputs 20×20 pixels.

The detection network consists of three layers corresponding to the features from P3, P4, and P5, each applying three anchor boxes per scale to generate a vector with class probability, objectness score, and predicted bounding box coordinates.

YOLOv5's architecture was tailored to detect road objects and process input images with dimensions of $3 \times 416 \times 416$. Initially, YOLOv5 was released in four variants: YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x. These variants differ in network depth and parameter count, representing small (s), medium (m), large (l), and extra-large (x) models, respectively. All these variants were utilized in this study.

3.1.1 Input Image Processing

In the enhanced model for input image processing, the input image is divided into an $S \times S$ grid, where S represents the number of divisions along each axis of the image. This grid division is a fundamental part of the YOLO architecture, facilitating the localization and detection of objects within the image.

In the new approach, the input image is partitioned into an $S \times S$ grid, where each grid cell in the $S \times S$ grid is responsible for predicting the presence of objects whose centers fall within that specific cell. For instance, if an object's center lies within a particular grid cell, that cell is tasked with predicting the bounding box coordinates, objectness score (confidence score that an object is present), and class probabilities for the object. This approach ensures that every part of the image is examined for potential objects.

To improve the original method, the input layer uses a combination of conventional enhancement techniques, such as rotation, flipping, and blurring, along with advanced methods like Mosaic augmentation [28]. Furthermore, settings like adaptive anchors and adaptive image scaling are incorporated. All dataset images, which vary in length, width, and proportions, are resized to a uniform 640×640 pixels. This standardization is designed to reduce the number of training parameters and speeds up both the training and inference processes, particularly effective in mitigating noise in images. The enhanced input image processing in the YOLO architecture, utilizing grid division, conventional and advanced augmentation techniques, adaptive settings, and uniform resizing, significantly improves object detection performance by enhancing localization accuracy, robustness, and noise mitigation.

3.1.2 Feature Extraction

The enhanced model utilizes CSPDarknet as its foundational architecture for feature extraction, encompassing components such as Focus, CSP, and SPP. Focus operates by slicing the feature map, integrating width (w) and height (h) information into the channel (c) dimension. This operation involves splicing and stacking units 2 pixels apart along each channel, resulting in a reduction of width and height by half and a fourfold increase in the number of channels. This optimization reduces Floating Point Operations Per Second (FLOPS) and enhances inference speed.

CSP within CSPDarknet encompasses two structures: one incorporating residuals and another without residuals. This design innovation addresses critical challenges in large-scale convolutional neural networks by optimizing gradient information propagation throughout the network architecture. By facilitating improved gradient flow from the initial layers to the feature maps' end, CSP reduces model parameters and FLOPS, thereby enhancing both speed and accuracy in inference tasks.

SPP complements CSPDarknet by pooling feature maps across multiple scales and concatenating results in the channel dimension. This process significantly enlarges the receptive field, thereby improving the alignment between anchor points and feature maps.

SPP's integration ensures that the enhanced model can effectively capture context at varying scales, enhancing its capability for precise object detection and localization in complex visual datasets. Thus, the utilization of CSPDarknet for feature extraction, incorporating Focus, CSP with and without residuals, and SPP, optimizes gradient flow, reduces model parameters, enhances receptive fields, and improves inference speed and accuracy in object detection tasks.

3.1.3 Anchor Boxes

In the enhanced model, anchor boxes play a crucial role in predicting object locations within an image. The upgraded algorithm begins by defining a set of anchor boxes with various aspect ratios and scales to capture different object shapes and sizes. The input image is divided into an $S \times S$ grid, where each grid cell is associated with multiple anchor boxes. For each grid cell an anchor box, the model predicts the bounding box coordinates relative to the anchor box, including offsets for the center coordinates (x,y), width, and height.

Additionally, the model predicts an objectness score and class probabilities for each bounding box. During training, a composite loss function is used, comprising localization loss, confidence loss, and classification loss. To refine the predictions and eliminate redundant detection, non-maximum suppression (NMS) [62] is applied, which filters, sorts, and suppresses overlapping boxes based on their objectness scores and Intersection over Union (IoU) thresholds. The final output consists of accurately predicted bounding boxes with confidence scores and class probabilities, enabling precise and efficient real-time object detection.

To further refine the original model, the enhanced solution utilizes anchor boxes to predict bounding box coordinates for objects in an image, improving efficiency and accuracy in object detection. The predefined aspect ratios and scales help the model adapt to various object sizes and shapes.

3.1.4 Prediction Outputs

In the enhanced model, each grid cell is responsible for predicting a set number of bounding boxes, which are associated with predefined anchor boxes of different shapes and sizes.

For every anchor box, the network predicts specific parameters: x and y represent offsets from the grid cell's coordinates, determining the center of the predicted bounding box.

The parameters w and h denote log-space adjustments that scale the width and height of the anchor box relative to its predefined size. Additionally, the network assigns an objectness score to each predicted bounding box, indicating the likelihood of an object being present. Furthermore, class probabilities are predicted for each bounding box, indicating the probability of the object belonging to various predefined classes or categories. As a result, the prediction outputs in our proposed method, including bounding box offsets, objectness scores, and class probabilities assigned to predefined anchor boxes, improve object detection performance by accurately localizing objects, assessing their presence likelihood, and assigning class probabilities for precise classification.

3.1.5 Bounding Box Calculation

The bounding box calculation in enhanced YOLOv5 is a crucial step where the model predicts the locations and dimensions of bounding boxes for objects within an image, transforming the raw network outputs into meaningful coordinates.

YOLOv5 utilizes predefined anchor boxes—prior boxes with fixed shapes and sizes. Each grid cell is associated with multiple anchor boxes, designed to capture different aspect ratios and scales of objects. The bounding box coordinates are derived by applying predicted offsets to these anchor boxes. To refine these predictions and eliminate redundant detection, NMS is employed. This process involves filtering out bounding boxes with objectness scores below a certain threshold, sorting the remaining boxes by their objectness scores, and then suppressing overlapping boxes. Starting with the highest-scoring bounding box, any boxes with an IoU above a specified threshold are discarded, retaining only the most accurate detection.

In summary, the bounding box calculation in enhanced methodology involves predicting offsets and adjustments for anchor boxes, determining the center coordinates and dimensions of the bounding boxes, and refining these predictions through NMS. This process enables YOLOv5 to generate accurate and reliable object detection efficiently.

3.1.6 Post-Processing

To streamline detection and ensure accuracy, the enhanced framework employs NMS through several key steps.

Firstly, the model filters out boxes with low objectness scores, ensuring that only those with a significant likelihood of containing objects are considered.

Next, the remaining boxes are sorted by their objectness scores in descending order to prioritize high-confidence predictions. The model then iteratively selects the box with the highest score and removes any overlapping boxes that exceed a specified IoU threshold. This step ensures that only the most relevant and non-overlapping detections are retained.

Additionally, the enhanced model applies a confidence threshold to further refine its predictions. Boxes with confidence scores below a predefined threshold are discarded, enhancing the overall precision of object detection results. As a result, these steps collectively help in reducing redundancy, enhancing the precision of detected objects, and ensuring that only the most confident and non-overlapping detections are considered.

3.1.7 Dense Prediction Output Result

The final output of the enhanced model consists of a set of bounding boxes, each associated with a confidence score and class probabilities. After processing the input image through the neural network, the model predicts bounding boxes relative to predefined anchor boxes. These predictions include the coordinates of the bounding box center, width, height, objectness score, and class probabilities. Post-processing steps, such as NMS, are applied to eliminate redundant detection and retain the most accurate bounding boxes. The resulting output provides precise locations and classifications of detected objects within the image, making it suitable for real-time object detection applications.

Compared with the original YOLOv5, the enhanced model uses a composite loss function including Localization Loss, which measures errors in bounding box coordinates; Confidence Loss, which measures errors in the objectness score; and classification loss, which measures errors in class probability predictions. Gradients are calculated and propagated back through the network to update the weights. The entire image is processed in a single forward pass through the network, and the network outputs are interpreted to form the final detection. Thus, the dense prediction output, featuring precise bounding box coordinates, confidence scores, and class probabilities processed with a composite loss function and optimized through efficient forward and backward propagation, enhances object detection performance by accurately identifying and classifying objects in real-time applications, facilitated by streamlined post-processing steps like NMS for robust detection.

3.2 Method

The YOLO series is a deep learning-based target detection framework that achieves efficient target detection by detecting all objects in a single forward propagation. YOLOv7 and YOLOv5 were different versions of YOLO, with YOLOv7 being the newer version. In terms of computational efficiency and accuracy, YOLOv7 improved relative to YOLOv5. However, YOLOv5 was much faster than YOLOv7 in terms of training and inference and had a lower memory footprint. This made YOLOv5 more advantageous in object detection application scenarios.

Therefore, we chose to use the original YOLOv5 architecture as a foundation for the enhancements. The network structure of the proposed YOLOv5 algorithm. Firstly, at the input of the network, we use a shallow bottleneck to down-sample the picture while increasing the number of channels of the feature map. By this it would be possible to retain as much detailed information about the target as possible. Next, the proposed efficient spatio temporal interaction module was used as the residual feature learning module for Backbone. It could better combine deep semantic information with shallow semantic information. In the neck section, we added a spatial pyramidal convolution module to the 160×160 layers as the detection head for very small targets. Finally, recursive gated convolution was introduced into the PANet structure to enhance spatial localization information.

3.2.1 Level of Signal-To-Noise Ratio

We introduce the Signal-to-Noise Ratio (SNR) as a fundamental metric for evaluating noisy levels in datasets affected by adverse weather conditions. SNR plays a crucial role in our efforts to develop a new framework aimed at detecting radar signals with low SNR in challenging environmental settings. Widely utilized in signal processing, SNR serves as a primary indicator of image quality, representing the ratio between the signal, which encapsulates essential image information, and the noise, which comprises unwanted variations introduced by adverse weather phenomena such as fog, snow, or sensor imperfections.

In this study, experiments are conducted to validate the efficacy of our proposed model. We evaluate its performance using real-world road images, and the bench-marking against several state-of-the-art methodologies. The restoration of synthetic datasets is quantitatively assessed using established metrics such as PSNR (Peak Signal-to-Noise Ratio) [63] and SSIM (Structural Similarity Index) [64], widely recognized for their effectiveness in evaluating image restoration techniques. Specifically, SNR calculations are derived from

the mean measured depth of fog, rain, or snow effects divided by the standard deviation of the error between observed and estimated levels, providing a robust measure of noise levels in adverse weather conditions.

For instance, in the analysis of foggy images, determining SNR involves scrutinizing the contrast between the signal, representing pertinent visual data, and the noise, encompassing undesirable elements like fog-induced haze or inherent sensor noise. This analytical process typically involves pixel-level examination within the image dataset and the application of mathematical formulations tailored to assess SNR.

Where μ represents the average pixel value (signal) and σ denotes the standard deviation of the pixel values (noise). The images will be loaded and converted to gray-scale, followed by the computation of the SNR. For example, the calculated values for the foggy image in the DAWN dataset are indicated in Algorithm 1, the mean pixel value, representing the signal, is 117.12, while the standard deviation, representing the noise, is 61.66. This results in a SNR of 1.90. An SNR of 1.90 indicates that the signal level is only marginally higher than the noise level in this foggy image, which corresponds with the reduced clarity and contrast typically observed in foggy conditions.

The SNR indicates that the signal level is only marginally higher than the noise level, aligning with the image's visual characteristics of reduced clarity and contrast due to fog. The Table 3.1 presents the mean SNR values for different weather conditions observed in the DAWN dataset. Each weather condition: Snow, Rain, Fog, and Sandstorm has a corresponding SNR value, reflecting the ratio of signal clarity to noise interference in image data affected by these conditions. Lower SNR values for Snow with 4.74 and Rain with 5.28 suggest higher levels of noise relative to signal, indicating more challenging image quality in snowy and rainy environments. In contrast, higher SNR values such as Fog with 7.17 imply clearer and more distinguishable images under foggy conditions. These SNR measurements provide critical insights into how varying weather phenomena impact image quality, guiding the development of robust image processing techniques tailored to different environmental challenges encountered in real-world scenarios.

The system can effectively detect noise levels associated with different weather conditions, as indicated by their respective Mean Signal-to-Noise Ratios (SNR). For snow, the system can handle an SNR of at least 4.74, while for rain, it can manage an SNR of 5.28. In foggy conditions, the detection limit is higher, with an SNR of 7.58, and for sandstorms, the system can detect noise levels with an SNR of 5.39. These values represent the minimum SNR levels required for effective noise detection in each weather condition.

Combining the analysis from Table 4.3 with the above findings, it becomes evident that images characterized by an SNR of 7.58 exhibit exceptional precision in object detection

Classes	SNR (dB)
Snow	4.74
Rain	5.28
Fog	7.58
Sandstorm	5.39

Table 3.1: Mean SNR for the proposed models capable of detecting minimal targets in DAWN under various weather conditions.

Algorithm 1 Calculate SNR of an Image

1: Load the image from *image_path*
2: Convert the image to a NumPy array *image_data*
3: Convert *image_data* to grayscale using weights [0.2989, 0.5870, 0.1140]
4: Calculate the mean signal (brightness) as *signal*
5: Calculate the noise (standard deviation) as *noise*
6: Compute SNR as *signal* / *noise*
7: **return** SNR

tasks at a confidence threshold of 0.50, indicating robust performance even in challenging weather conditions like fog. This correlation underscores the importance of SNR in maintaining high accuracy and reliability in our proposed image processing and object detection model.

As a result, SNR serves not only to objectively evaluate image processing techniques but also to steer efforts towards mitigating noise and improving the dependability of object detection and recognition in challenging weather conditions.

3.2.2 Evaluation and Fine-Tuning

Evaluate the performance of the modified YOLOv5 model on a separate test set that includes various adverse weather conditions. Fine-tune the model parameters and hyperparameters to achieve optimal detection accuracy and robustness.

Chapter 3 delineates the traditional approach to tackling the challenges inherent in object detection, steering clear of specific dataset dependencies. At the core of this approach lies the development of a one-stage object detection model, with a particular focus on introducing and leveraging YOLOv5. The study delves into the intricacies of enhancing YOLOv5 at a high level, dissecting its architecture and exploring avenues for improve-

ment. The subsequent sections delve deeper into the nuanced aspects of different backbones, necks, and heads, elucidating how modifications in these components can bolster the overall detection performance. Furthermore, it elucidates advancements in model integration techniques to enhance the state-of-the-art solutions further. The section culminates in an exploration of neural network fine-tuning methodologies, showcasing how iterative improvements based on the enhanced YOLOv5 structures can propel the performance of object detection systems.

The methodologies used in the training session are presented as this thesis aims to improve the one-stage detection methodology YOLOv5 by changing the structure of the models individually across all the scales and treating each one as a different model. The study chooses the ideal baseline model YOLOv5 in the first stage to set the baseline.

As seen in Fig. 3.8, the workflow of the proposed methodology is illustrated to make the content discussed from section 3.2.3 to section 3.2.3 easy to follow. The structural framework of the proposed one-stage detection module can be seen in Fig. 2.7 in detail. The investigation of this study includes a detailed study of the basic structure, including heads, necks, and backbones. Some approaches are discussed to model integration to improve the performance using the state-of-the-art detectors.

3.2.3 Overview of Traditional Approaches

As a fundamental problem, object detection has been an active area of research for many years. The main goal of object detection involves identifying and localizing objects of interest from different categories within some given image. Object detection is the basis of many other advanced computer vision tasks ranging from semantic segmentation to object tracking to activity recognition.

Typically, there are two ways to identify objects, first single-stage detection and two-stage detection. As opposed to the latter, where the one-stage algorithm first creates a collection of area recommendations before classifying them as objects or backgrounds, the former explicitly predicts the bounding boxes and class probabilities for objects. Single-stage approaches, such as YOLO [19], SSD [23], EfficientDet [38], and RetinaNet [24], generally employ one Fully Convolutional Neural Network (FCNN) [48] to identify objects' classes and spatial placements without the usage of intermediary steps, in contrast to two-stage methods, Faster RCNN [47] and R-FCN [65].

We have used the framework of YOLOv5 for our proposed dataset because it is the improved method for smaller objects in weather-adverse conditions. For example, Fig. 3.1,

demonstrate the design of original YOLOv5 consists of three main components: the backbone, the neck, and the output. Each part plays a crucial role in the overall functionality and performance of the network.

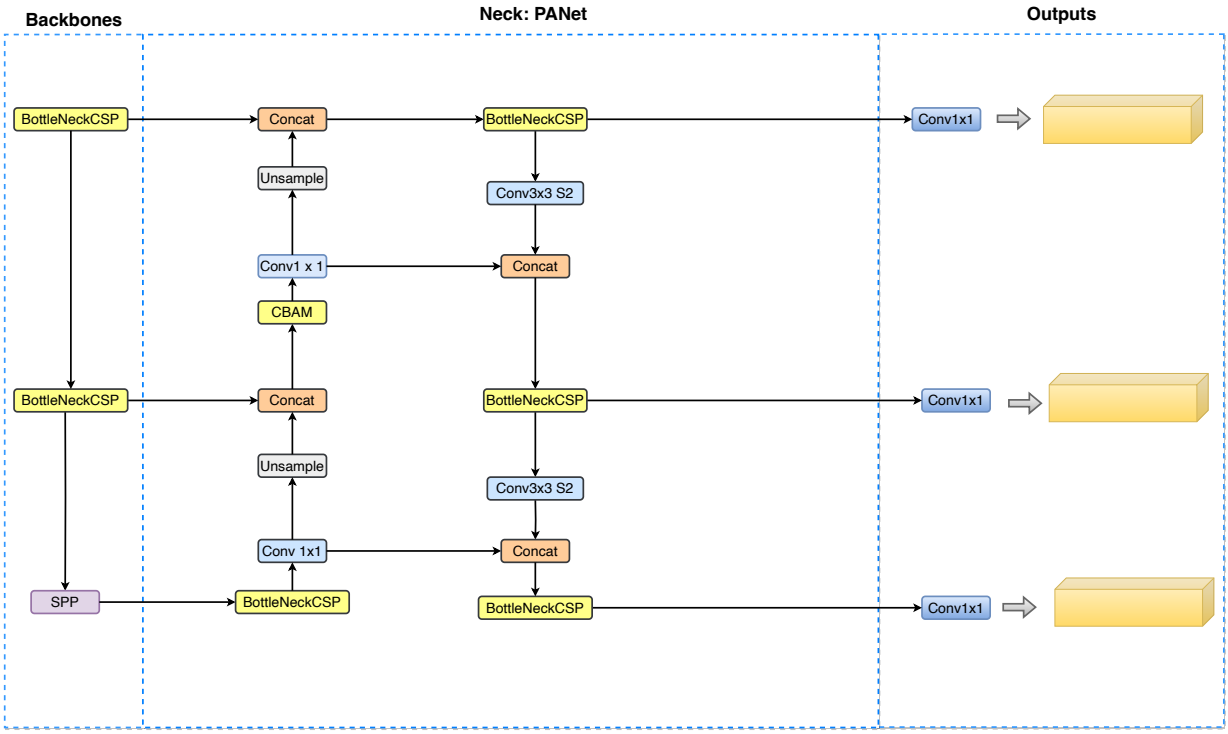


Figure 3.1: The design of YOLOv5. The network's three main parts are the output, the neck, and the backbone. The output section employs these feature maps to identify things, and the neck part combines the feature data gathered from the input images to build three feature map scales. Getting feature information from the input photos is the main focus of the backbone portion.

Detection Head

Original YOLOv5 employs a multi-scale detection strategy, utilizing three detection heads to predict bounding boxes, objectness scores, and class probabilities across three different scales (small, medium, large). This approach ensures that objects of varying sizes are accurately detected. To constrain the objectness scores and class probabilities between 0 and 1, YOLOv5 applies a sigmoid activation function, which helps in normalizing the

predictions and improving the model's performance. Refer to Fig. 3.1,the Head section consists of three predictors. The images undergo feature extraction and feature fusion, ultimately resulting in three different sizes of detection heads. These heads are used to output targets of varying sizes: large, medium, and small.

The Detection Neck functions as a feature aggregator, combining and mixing the features extracted by the Backbone to prepare for the subsequent step in the Detection Head. Upon these aggregated features, the Head module performs object detection.

The YOLOv5 model primarily adopts the CSPDarkNet53 structure, utilizing the Convolutional (Conv) layer, C3 layer, and SPPF (Spatial Pyramid Pooling Fast) layer in the backbone network. The Conv layer comprises convolution, Batch Normalization(BN), and the SiLU activation function [66]. By employing residual connections, the C3 module effectively reduces model parameters, thereby improving inference speed. SPPF, an improved version of the original SPP module, replaces the max-pooling layers of size 5×5, 9×9, and 13×13 with three max-pooling layers of size 5×5. This modification ensures the fusion of features with different receptive fields while further enhancing operational speed.

Neck

In the improved model, the Neck component incorporates the PANet, which builds upon the Feature Pyramid Network (FPN) by introducing bottom-up pathways. This design choice allows the neck to create a feature pyramid that aggregates features from different stages of the backbone network, thereby enhancing the representation of features for detecting objects across multiple scales.

Following the top-down feature fusion in FPN, the introduced bottom-up pathways facilitate the transmission of positional information from lower-level feature maps to deeper ones. This approach significantly improves the model's ability to localize objects at various scales and complexities within an image. YOLOv5 offers five derivative models—YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x—all sharing the same underlying architecture but differing in terms of width and depth to accommodate different computational resources and performance requirements.

Backbone

The backbone network utilized in YOLOv5 is CSPDarknet53, an enhanced variant of the Darknet architecture. CSPDarknet53 is specifically designed to improve gradient flow and computational efficiency. It achieves this through a series of convolutional layers that systematically extract features from the input image, ensuring a comprehensive and hierarchical representation of the data.

To stabilize and accelerate the training process, batch normalization is employed throughout the network. This technique helps mitigate internal covariate shift and allows for higher learning rates. Furthermore, the Leaky ReLU activation function is applied to introduce non-linearity, which is crucial for the model's ability to learn complex patterns effectively, thereby enhancing overall performance.

3.2.4 Overview of the Enhanced Model

Section 3.2.3 discussed the overview of the enhanced model. Original YOLO is a regression-based target identification system. Once the input images or videos are sent into the deep network, YOLO completes the prediction of the object's categorization and position based on the loss function calculation, turning the target detection issue into a regression problem.

YOLOv5 [29] is based on the YOLO detection architecture and uses the excellent algorithm optimization strategy in the field of Convolutional Neural Networks (CNNs) in recent years, such as auto-learning bounding box anchors, mosaic data augmentation, the cross-stage partial network, and so on; they are responsible for different functions in different locations of the YOLOv5 architecture.

In the architecture, YOLOv5 consists of four main parts: input, backbone, neck, and output. The input terminal mainly contains the preprocessing of the data, including mosaic data augmentation and adaptive image filling. To adapt to different datasets, YOLOv5 integrates adaptive anchor frame calculation on the input, so that it can automatically set the initial anchor frame size when the dataset changes. The backbone network mainly uses a cross-stage partial network (CSP) [67] and spatial pyramid pooling (SPP) [49] to extract feature maps of different sizes from the input image by multiple convolution and pooling. BottleneckCSP is used to reduce the amount of calculation and increase the speed of inference, while the SPP structure realizes the feature extraction from different scales for the same feature map, and can generate three-scale feature maps, which helps improve the detection accuracy.

In the neck network, the FPN [68] and PANet are used. The FPN structure conveys strong semantic features from the top feature maps into the lower feature maps. At the same time, the PANet [52] structure conveys strong localization features from lower feature maps into higher feature maps. These two structures jointly strengthen the feature extracted from different network layers in Backbone fusion, which further improves the detection capability.

As a final detection step, the head output is mainly used to predict targets of different sizes on feature maps. The enhanced YOLOv5 consists of four architectures, named

YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x. The main difference between them lies in the number of feature extraction modules and convolution kernels at specific locations on the network. The network structure of YOLOv5 is shown in Fig. 3.2.

Based on the design of the original YOLOv5 architecture, we have implemented the following Algorithm 2 in the enhanced framework.

Algorithm 2 Enhancements in the YOLOv5-based approach.

1: **Completed:** Replacement of backbone with GhostNet [11]
2: **Completed:**Replacement of backbone with ShuffleNetv2 [10]
3: **Completed:** Replacement of backbone with MobileNetv3Small [12]
4: **Completed:** EagleEye [39]'s support for YOLOv5 series pruning
5: **Completed:** MQBench's quantitative [44] support for YOLOv5 series
6: **Completed:** Replacement of backbone with EfficientNetLite-0 [25]
7: **Completed:** Replacement of backbone with PP-LCNet-1x [41]
8: **Completed:** SwinTrans-YOLOv5 (C3STR)
9: **Completed:** Slimming support for YOLOv5 series pruning

Algorithm 3 Calculate SNR in an Image

Require: Image I
1: Convert the image to grayscale I_{gray} (if it's in color)
2: Calculate the mean pixel value μ of I_{gray} (signal)
3: Calculate the standard deviation σ of I_{gray} (noise)
4: Compute the SNR using the formula:

$$\text{SNR} = \frac{\mu}{\sigma}$$

5: **return** SNR

The proposed framework adds an extra prediction head based on YOLOv5 to detect targets of different scales. Then, transformer prediction heads (TPH) were used to replace the original prediction heads by exploring the prediction potential of Self-Attention. At the same time, we integrated the Convolutional Block Attention model (CBAM) [15] to find the attention area in dense scenes.

The architecture of YOLOv5 introduces an additional head, a transformer prediction head (TPH) [69], and a Convolutional Block Attention Module (CBAM) [15]. Four prediction heads are named tiny, small, medium, and large heads, and the branches before these

heads are tiny, small, medium, and large paths. Therefore, the new TPH promotes the

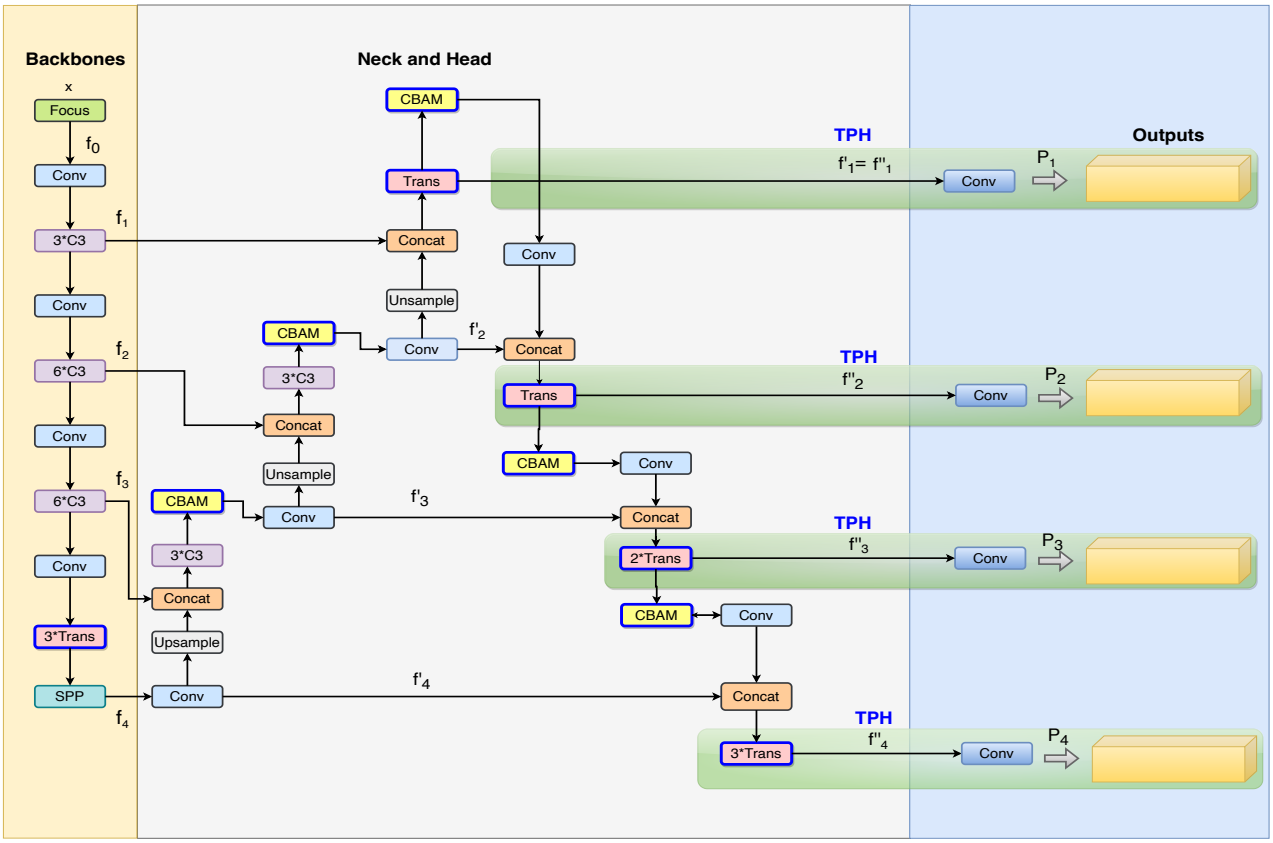


Figure 3.2: The architecture of the improved YOLOv5; Blue highlights are the blocks are integrated.

For analytical purposes, we establish the mathematical representation of the flow chart and model structure of YOLOv5. By designating the input photos as x, we obtain f_i equal to Focus(x). The output from the backbone, consisting of four features, is indicated as f_i, where i equals 1,2,3,4. Equation (3.3) is used to obtain these four features:

$$f_i = B_i(f_{i-1}), \quad i \in \{1, 2, 3, 4\}, \tag{3.3}$$

Combinations of a convolutional layer are represented by B_1, B_2, and B_3; these com-

binations include 3, 6, or 9 CSPbottleneck modules. Three transformer modules, an SPP module, and a convolutional layer are combined into B_4 [49]. B_i indicates the various blocks in the backbone. In the neck section, four features are denoted as f_i', where i equals 1,2,3,4. The following four features can be articulated:

$$f_i' = \begin{cases} N_i(f_i, f_{i+1}'), & i \in \{1,2,3\} \\ Conv(f_i), & i = 4 \end{cases}, \tag{3.4}$$

Let N_i represent different blocks, $Conv$ represent Conv layers, and N_i be expressed as follows in Equation (3.5):

$$N_i(f_i, f_{i+1}') = UpBlock(Concat(f_i, Upsampling(f_i' + 1))). \tag{3.5}$$

The synthesis of several modules is represented by the UpBlock. In N_2 and N_3, the UpBlock comprises three CSP bottleneck modules, a CBAM module, and a convolutional layer. Meanwhile, in N_1, the CBAM and transformer modules are included with the UpBlock [15]. With i equal to 1, 2, 3, 4, the features are listed before the last four convolution layers as f''. Thus, Equation (3.7) is derived as follows:

$$f_i'' = \begin{cases} f_1', & i = 1 \\ H_i(f_i, f_{i-1}''), & i \in \{2,3,4\} \end{cases}, \tag{3.6}$$

Like N_i, H_i denotes distinct blocks.

$$H_i(f_i, f_{i-1}'') = DownBlock(Concat(f_i, Conv(f_{i-1}''))), \text{for } i \in \{1,2,3,4\}. \tag{3.7}$$

The Down-Block signifies the amalgamation of various modules. A CBAM module plus one, two, or three transformer modules are present in H_2, H_3, and H_4 of the Down-Block. As described in the Equation 3.9, you can obtain a definitive forecast following your acquisition of f_i''.:

$$p_i = Conv(f_i''), \text{where i equals to 1,2,3,4.} \tag{3.8}$$

Here, p_i represents the set of four output predictions derived from various prediction heads.

In conclusion, we employ a methodology distinct from that of CNNs in the component of enhanced model. Bounding box and class probabilities are directly predicted from feature maps using convolutional layers rather than fully linked layers. A 3D tensor with the bounding box coordinates, objectness scores, and class predictions for every grid cell is produced using 1x1 convolutions at the network's conclusion. Notably, the most significant enhancement in this study involves replacing the original prediction heads with Transformer Prediction Heads (TPH), leveraging their self-attention mechanism to enhance prediction capabilities. Additionally, the addition of the fourth TPH to YOLOv5 specifically targets improved performance in detecting minimal targets within noisy images.

Proposed Small Objects Prediction Head

To recognize small objects, the suggested system adds an additional prediction head. The four-head structure can mitigate the adverse effect of violent object scale variation when combined with the other three prediction heads.

Adding a transformer prediction head to YOLOv5 involves integrating transformer layers into the existing architecture to improve the capture of long-range dependencies and contextual information. Starting with the standard YOLOv5 model, including the backbone for feature extraction and the neck for feature aggregation, transformer layers are introduced after the neck. These layers use positional encoding and self-attention mechanisms to process the aggregated feature maps. The enriched feature maps are then passed to an additional prediction head, which works alongside the original detection heads to predict bounding boxes, objectness scores, and class probabilities.

In the default neck section, shown in Fig. 3.2, a combination of FPN and PANet ideas is used. The feature maps are fused with feature information extracted from Backbone's C3, C4 and SPPF layers, respectively. Firstly, high-level features are fused with underlying features using the FPN network to leverage both high resolution and rich semantic information for improved detection of small objects. The PANet structure then fused different levels of feature information to retain shallow location features, enhancing overall network performance. This process resulted in three pre-detection scales of 80×80, 40×40, and 20×20. We introduced a spatial pyramidal convolution module as the detection head for very small targets in the default Neck section. This module includes a 3×3 convolution, SPPF module, `SPPF_1` module, and C3 module. The `SPPF_1` module featured a 1×1 convolution and three parallel standard convolutions with kernel sizes of 3×3, 5×5, and 7×7. The spatial pyramidal convolution module used the Backbone's P2 layer output as input. After processing through the SPPF and `SPPF_1` blocks, the number of channels in

the feature map was adjusted, and the feature map was concatenated. The output was then used as a 160×160 detection head. Finally, a recursive gated convolutional was embedded on top of the default Neck.

The outputs from both the original and transformer prediction heads are combined, and NMS is applied to refine the final detection. This modification improves the model's performance, particularly in scenarios involving complex backgrounds and small objects, by leveraging the transformer's capability to understand intricate contextual relationships within the input data.

A high-definition and lower-level feature map that is more perceptive to small objects is employed to construct the prediction head of transformers, as seen in Fig. 3.2. Despite higher computing and memory costs, the performance of identifying tiny objects has improved by embedding the detection head. In the original YOLO design, the four-head structure reduces the negative impact of violent object size variation more than the other three prediction heads did. Thus, the innovative transformer prediction head notably enhances performance in detecting object classes such as bicycles and pedestrians. By combining both low-quality and high-quality data inputs with multi-layer, superior features, the supplemental head improves sensitivity to tiny objects.

In conclusion, the proposed small objects prediction head improves object detection performance in adverse weather images by enhancing the model's sensitivity to small objects through advanced feature aggregation and transformer layers, mitigating the challenges posed by varying scales and environmental conditions.

Transformer Encoder Block

Built upon the CSPDarknet53 design, YOLOv5 utilizes an SPP structure along with PANet for its neck and head in YOLO detection. Inspired by [70], transformer encoder blocks replace YOLOv5's convolutional and CSP bottleneck components. The framework is illustrated in Fig. 3.2.

According to this frame's traditional procedure. Utilizing transformer encoder blocks, we have adjusted a few CSP bottleneck modules such that they generate attention between each pair of pixels, therefore utilizing contextual information. The transformer encoder module improves the capacity to record various local data. With its self-attention mechanism, it can investigate the possibilities of feature representation.

A feed-forward neural network and a multi-head attention block are the two major blocks that make up the Transformer Encoder (MLP). In addition to preventing overfitting, LayerNorm and Dropout layers aid in improved network convergence. In addition to helping the current node focus on the pixels in front of it, multi-head attention can assist

it in obtaining the context's semantics. The transformer encoder blocks in CSPDarknet53 gather a wealth of contextual information that is both global and plentiful when compared to the original bottleneck block. Transformer encoders come in two sub-layer configurations. A multi-head attention layer makes up the first sub-layer, while a fully connected layer makes up the second one (MLP). Every sub-layer has a residual link between them. The capacity to record various local data is increased by transformer encoder blocks. Additionally, it may investigate the possibility of feature representation using a self-attention technique. Transformer encoder blocks provide high-density occluded object performance on the VisDrone2021 dataset [71].

Inspired by the vision transformer [70], we replace some CSPbottleneck modules with transformer encoder modules and we also apply transformer encoder blocks in the head part to form the Transformer Prediction Head (TPH) and the end of the backbone to utilize contextual information by generating attentions between every pixel pairs. Because the feature maps at the network's end have low resolution, deploying TPH on low-resolution feature maps can decrease the expensive computation and memory costs. When the resolution of input images is enlarged, some TPH blocks at early layers will be removed to make the training process available.The transformer encoder module increases the ability to capture different local information. It can also explore the feature representation potential with its self-attention mechanism [72].

As a result, the proposed transformer encoder block enhances object detection performance in adverse weather images by leveraging its self-attention mechanism to effectively capture contextual information across pixel pairs, improving the model's ability to discern intricate details and features amidst challenging environmental conditions.

Swin Transformer Block

To address the challenges of detecting objects of varied sizes in adverse weather conditions, the Swin-Transformer is integrated into the enhanced architecture, forming a unique network with improved inference times per instance. The integration of Swin Transformer Blocks improves object detection performance in adverse weather images by effectively aggregating hierarchical features and leveraging window-based multi-head self-attention to capture long-range dependencies, thus improving the model's robustness and accuracy in detecting objects of varied sizes and in challenging environmental conditions.

As illustrated in Fig. 3.3, the Swin-Transformer creates a hierarchical representation

[1]Multi-head self-attention, which mimics the interactions between feature representations, is necessary to capture the inter-feature linkages inside the Transformer.

[2]Structure Diagram of Swin Transformer.

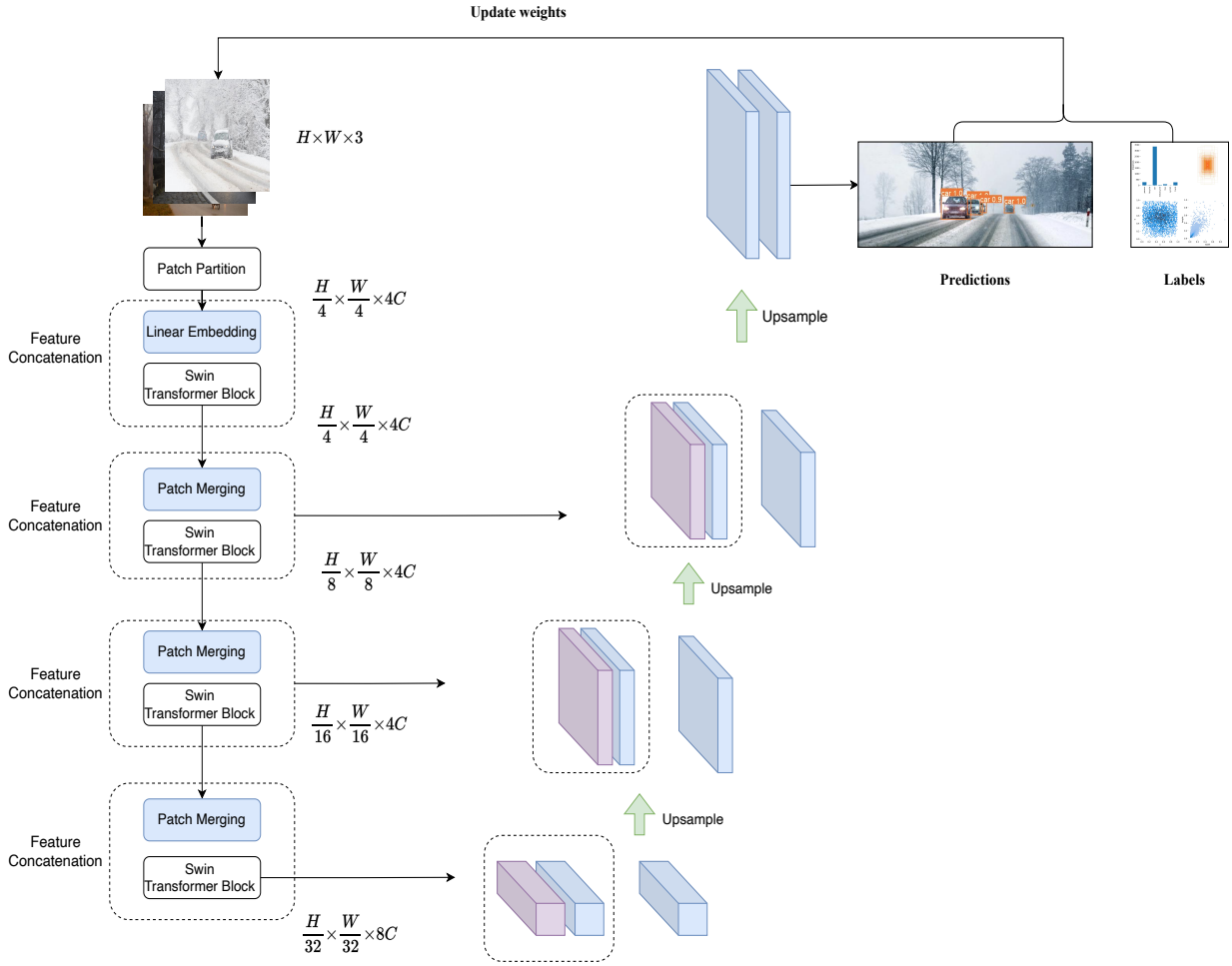


Figure 3.3: A diagram of the Swin-Transformer in the proposed layout.[1]

by progressively integrating nearby small patches into deep transformer layers. This approach utilizes a patch partition method, dividing the RGB input image into small, non-overlapping patches, as described by Liu *et al.* in [73]. Each patch is treated as a unique entity, with its properties defined by concatenating the raw pixel values of the RGB image. For each 4×4 patch, 48 features ($4 \times 4 \times 3$) are collected. These raw value features, denoted as C in Fig. 3.3, are projected to a random dimension using a linear embedding layer.

The decoding process begins with up-sampling blocks, starting from the lowest-resolution feature map. These blocks up-sample the feature map and concatenate it with the corresponding skip connection from the encoder. The decoder comprises three main components:

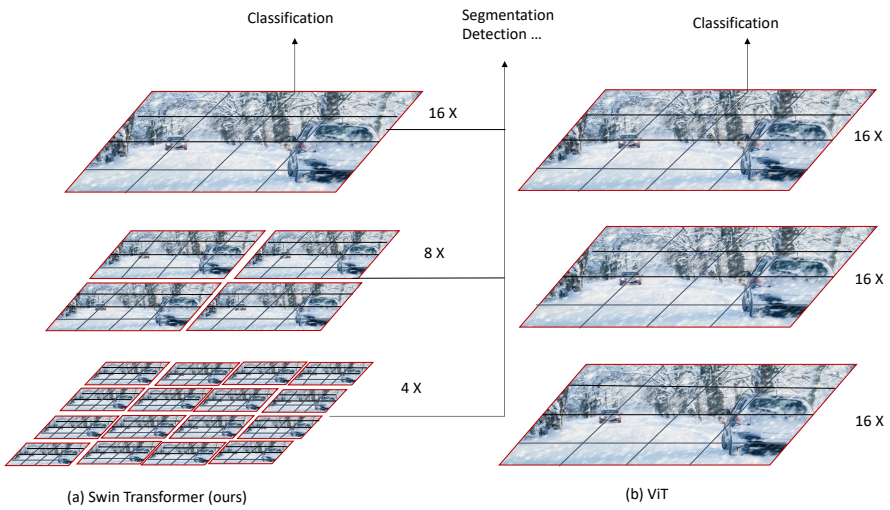


Figure 3.4: The architecture of the Swin Transformer in the Enhanced model.[2]

a 3×3 convolutional layer, a ReLU activation layer, and a 2X up-sampling operator. This design follows the U-Net architecture by encoding the decoder's features with the skip connections.

The cascaded up-sampling method is used to restore the resolution of the previous layer. During the encoder phase, feature maps are generated, and multiple cascaded up-sampling blocks are utilized to achieve full resolution. Each block includes an $H \times W$ resolution up-sampling layer, batch normalization, ReLU activation, and two 3×3 convolutional layers. The combination of the encoder and decoder forms the traditional U-shaped design, as depicted in Fig. 3.3.

To enhance YOLOv5's sensitivity to small objects in noisy images, the architecture can be improved by incorporating Swin-Transformer blocks in the backbone for feature extraction. The neck section should utilize a fusion-concat method for feature fusion, while NMS is employed to remove redundant detection boxes. This implementation aims to effectively detect small objects by leveraging the strengths of Swin-Transformers and advanced feature fusion techniques.

The objective of the first stage is to classify vertical greenery images as either clear or blurry, as depicted in stage 1 of Fig. 3.3. To achieve this, we added a LayerNorm layer, a global pooling layer, and a fully connected layer with a Sigmoid activation after the Swin Transformer. The Swin Transformer utilizes Window Multi-Head Self-Attention (W-

MSA) for global modeling and feature extraction. Initially, the network processes RGB three-channel images, which are not optimal for self-attention computation. Therefore, it is essential to first perform Patch Partition on the images for block processing. These block-processed images then pass through four stacked hierarchical structures, constructing feature maps of different resolutions and extracting the corresponding image features.

The classification results of the images are ultimately output through a fully connected layer with a Sigmoid activation function. Images predicted to be clear are then forwarded to the subsequent stages. The four stacked hierarchical structures, as shown in the figure, comprise different components. The first stage includes a Linear Embedding layer and two Swin Transformer Blocks, while Stages 2 to 4 feature a Patch Merging layer and varying numbers of Res-Swin Transformer Blocks.

As illustrated in Fig. 3.4, the Swin Transformer constructs a hierarchical representation by progressively merging adjacent patches in deeper transformer layers, beginning with small patches (gray outline). This model facilitates dense predictions by utilizing hierarchical feature maps. Linear computational complexity is achieved by calculating self-attention (red outline) locally within the non-overlapping windows of the image partition, rather than across all patches of the entire image. Since the number of patches within each window remains constant, the complexity scales linearly with the image size.

This enhanced architecture with swin transformer enables effective feature aggregation using skip connections at various resolution levels. As a result, the encoder's final output consists of multi-level feature maps with higher-level features at lower resolutions than the preceding ones, leading to superior performance in detecting objects in adverse weather conditions.

Convolutional Block Attention Module

Woo *et al.* originally introduced the CBAM in [15]. It is a simple attention module that enhances the performance of deep neural networks. As a result, the network's capacity to identify regions of interest in images with a large coverage area is enhanced by CBAM deployment in YOLOv5. Firstly, the most widely used CNN architecture may incorporate the thin CBAM module, which is entirely trainable. To apply feature modifications, CBAM multiplies the attention maps it creates for two different channel dimensions and areas by the input feature map when it receives a feature map.

As described in [15], the CBAM generates attention maps sequentially across two dimensions: spatial and channel attention modules. The CBAM attention mechanism combines the channel attention module and the spatial attentions module to process the input character layer of the channel attention module and of the spatial attentions module re-

spectively. These attention maps are then multiplied with the input feature map. The structure of the CBAM module integrated into our framework is depicted in Fig. 3.5. The CBAM attention mechanism within YOLOv5 architecture has proven to be effective, as it helps in extracting attention areas that support our detector, thereby improving object detection accuracy, particularly for small target objects.

The channel attention module depicted in Fig. 3.5 utilizes spatial information from average pooling and max pooling operations to derive average pooling features $F^C_{average}$ and max pooling features F^C_{max}. These spatial descriptors are subsequently processed by a shared network to produce the Channel Attention Map $M_c \in \mathbb{R}^{C\times1\times1}$.

A shared network comprises a Multi-Layer Perceptron (MLP) with a hidden layer designed to minimize parameter overhead. The hidden layer's activation size is set to $R^{C/r\times1\times1}$, where r denotes the reduction rate. The computation process follows Equation 3.9, where b represents the sigmoid function, $W_0 \in \mathbb{R}^{C\times C/r}$, $W_1 \in \mathbb{R}^{C\times C/r}$, and both W_0 and W_1 share the MLP's weights. W_0 is first activated by the ReLU function.

Unlike channel attention modules, the spatial attention module in Fig. 3.5 focuses on determining the significance of different spatial regions within the input data. Initially, the module aggregates channel information from a feature map through two pooling operations, generating $F^savg \in \mathbb{R}^{1\times H\times W}$ and $F^smax \in \mathbb{R}^{1\times H\times W}$, which represent the average and maximum pooled features across channels, respectively. Subsequently, normalization is performed using the sigmoid function, as detailed in Equation 3.10, where b denotes the sigmoid function and $F^{7\times7}$ denotes a compact kernel of size 7×7.

$$\begin{aligned} M_c(F) &= \sigma(MLP(AvgPool(F)) + MLP(MaxPool(F))) \\ &= \sigma(W_1(W_0F^C_{avg})) + W_0(F^C_{max})) \end{aligned} \tag{3.9}$$

$$\begin{aligned} M_c(F) &= \sigma(f7x7([AvgPool(F); MaxPool(F)]) \\ &= \sigma(f7x7([F^C_{avg}; F^C_{max}])) \end{aligned} \tag{3.10}$$

Finally, as shown in Fig. 3.2, to enhance YOLOv5's sensitivity to small objects, the Convolutional Block Attention Module (CBAM) is integrated into its architecture. CBAM is added to the backbone, neck, and head sections of YOLOv5, improving the model's ability to focus on relevant features. The integration begins with the backbone, where CBAM modules are inserted after significant convolutional layers. Each CBAM module sequentially derives attention maps along the channel and spatial dimensions, multiplying these maps by the input feature maps to emphasize important regions. In the neck, CBAM modules are strategically placed to enhance feature fusion, and in the head, they refine

prediction features. The integration of CBAM enhances object detection performance, particularly in adverse weather conditions, by leveraging its capability to focus on relevant spatial and channel-wise attention areas. This enhancement improves the model's accuracy in detecting small objects and is validated through training with an updated loss function, followed by rigorous validation and testing on separate datasets, demonstrating superior performance compared to the original model.

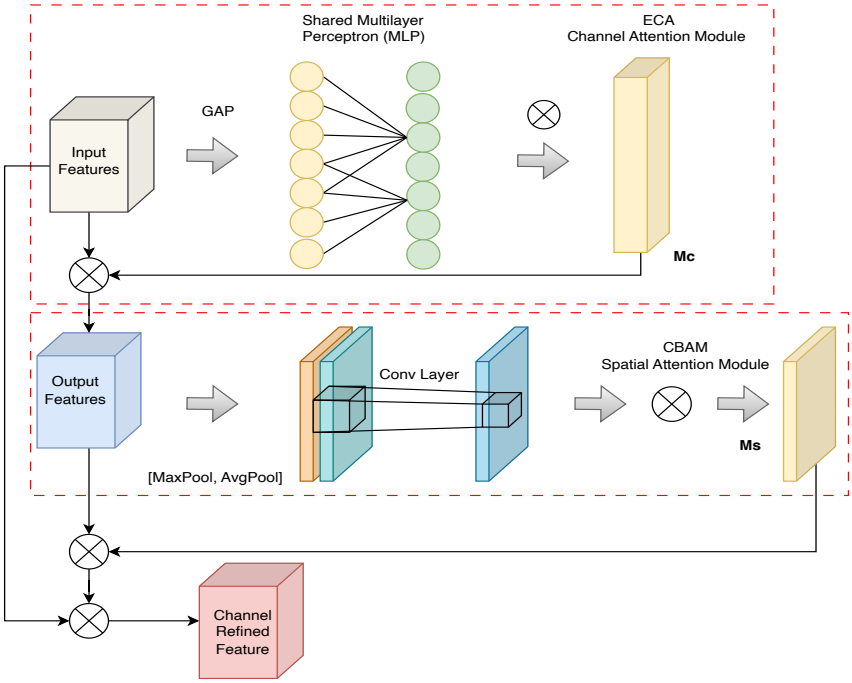


Figure 3.5: The architecture of CBAM in the enhanced YOLOv5. The feature maps passing through CBAM involve the utilization of two consecutive sub-modules, with the additional incorporation of residual paths.

Pruning Procedure and Strategy

The proposed methodology integrates EagleEye [39], a neural network pruning technique, to enhance performance in low-visibility environments.

The pruning method EagleEye incorporates a straightforward yet effective evaluation mechanism utilizing adaptive batch normalization to reveal significant correlations between pruned Deep Neural Network (DNN) architectures [74] and their eventual accuracy [39].

This correlation enables rapid identification of pruned candidates with potentially high accuracy, eliminating the need for extensive fine-tuning. Moreover, this module is versatile enough to integrate and enhance existing pruning algorithms. EagleEye's simplicity contrasts with its achievement of state-of-the-art pruning performance, surpassing many more intricate approaches.

To enhance the performance of the suggested detector, we will apply the EagleEye pruning technique to reduce the number of parameters and calculations in the YOLO model. This substantial association allows us to swiftly eliminate candidates without affecting them and prune those with the highest potential accuracy. EagleEye works better than any of the trimming algorithms in our tests.

This generalization is illustrated graphically in Fig.3.6. This approach should show the sub-net potential to choose the optimal pruning option and execute the pruning plan. EagleEye highlights that the traditional pruning process often incurs significant time during the fine-tuning phase, primarily due to the BN layer's substantial impact on model accuracy. The BN layer's parameters are frequently fine-tuned to adapt the pruned model, aiming to improve detection accuracy. In contrast, the EagleEye method utilizes Adaptive batch normalization, continually adjusting BN layer parameters within the pruned network using samples from a subset of the training data. This approach yields a pruned network model optimized for enhanced detection performance.

As documented in [39], EagleEye achieves a peak accuracy of 70.9%, demonstrating a superior performance of 1.3% to 3.8% over other methodologies when applied to a streamlined version of MobileNetV1 [75]. The study provides detailed insights into these findings. During ImageNet trials, approximately 50% of the FLOPs across all operations were reduced through pruning. This reduction is attributed to the fact that pruning a channel effectively eliminates both incoming and outgoing connections associated with that channel. EagleEye consequently achieves state-of-the-art pruning results, surpassing many more complex approaches.

The primary objective of this research is to explore a diverse array of sub-networks that are randomly selected to meet specific constraints. Subsequently, there is a rapid update and calibration of the mean and variance parameters of the BN layer, followed by comprehensive accuracy testing and calibration using a validation set. The sub-network exhibiting the highest accuracy is identified as possessing the optimal architecture and is further refined through fine-tuning to achieve enhanced performance. To implement EagleEye pruning within the enhanced framework, the process begins with loading the model parameters into YOLOv5n. Subsequently, the model undergoes training, followed by the generation of multiple pruning schemes through random selection. The BN layer param-

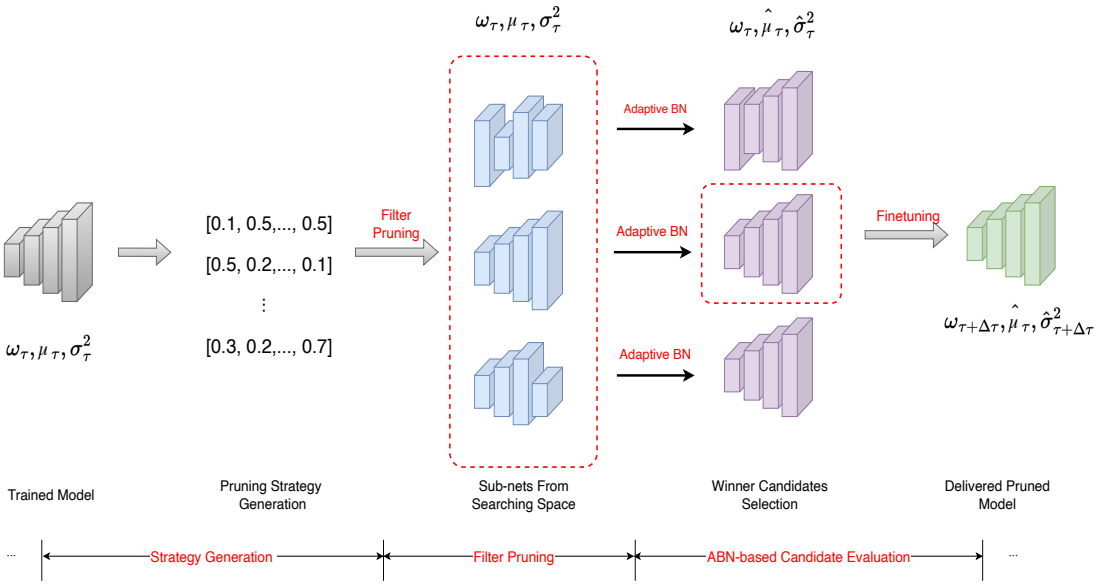


Figure 3.6: The pipeline of EagleEye Pruner.

eters are then adjusted according to various pruning strategies. Each pruning strategy's accuracy is evaluated to determine the most effective approach. Finally, fine-tuning is conducted to restore and potentially enhance the model's accuracy after pruning adjustments.

Thus, the incorporation of EagleEye, a neural network pruning technique, into the enhanced model improves object detection performance in adverse weather conditions. By systematically reducing model complexity and computational demands, while concurrently enhancing or maintaining accuracy through adaptive batch normalization and swift evaluation of pruned architectures, EagleEye optimizes the model's efficiency and effectiveness in challenging environmental scenarios.

Network Slimming

Inspired by network slimming [4], we provide a novel learning approach for the enhanced framework with the goal of preserving accuracy while simultaneously reducing the runtime memory footprint and model size. In the initial training process, we implemented the practical way of network slimming based on [4]. Fig. 3.7 indicates the flow diagram for the network slimming process, utilizing the scaling factor of the BN layer to gauge model pruning. It identifies and prunes convolution channels below a pre-defined threshold, thereby reducing algorithmic complexity. Subsequently, a retraining phase refines the

pruned model to recover accuracy, yielding a streamlined detection network.

Fig. 3.7 illustrates the channel pruning process: post-sparse training, convolution channels are pruned based on the model's altered structure compared to the original. While this structural change initially diminishes the model's detection capabilities and mapping power, fine-tuning enables the relearning of neural network parameters aligned with the current structure, thereby restoring and enhancing detection accuracy.

The enhanced model integrates an optimization solution alongside a slimming pruner to improve object identification accuracy, even when using the YOLOv5 baseline. Initially, large and broad networks serve as input models for the network slimming process. This process generates thin and compact models with equivalent accuracy by automatically identifying and pruning irrelevant channels during training.

For the sparsity training, the idea is to use the *gamma* scaling factor to multiply the output from every channel. We then focus on sparsity regularization while optimizing scaling factors and network weights. Because of this, we first adjusted the pruned network before removing channels that were distinguished by tiny factors. Our strategy aims to provide training:

$$L = \sum_{(x,y)} l(f(x, W), y) + \lambda \sum_{\gamma \in \Gamma} g(\gamma) \tag{3.11}$$

The first term in Equation 3.11, where g is a penalty term arising from the sparsity of the scaling factor and *gamma* balances the two components, represents the typical training loss of the CNNs. W represents the trainable weight, while x and y stand for the training input and target, respectively. The L1 norm is used in the experiment, and $g(s)$ is set to the absolute value of s. Sparsity is often achieved with it. The first term signifies the loss of the CNN network during standard training. Here, l is the adjustment coefficient, and $g(\cdot)$ denotes the sparsity penalty applied to the scaling factor. The optimization methods with the L1 penalty period will be swapped out for sub-gradient descent. Another method is to substitute the smooth L1 penalty for the L1 penalty to avoid using the sub-gradient at unsmooth places [76]. Pruning effectively removes all outgoing and incoming connections from a channel. Compared to more intricate methods, it produces pruning with better performance.

In the formula, W represents the weight of the trainable model, (x, y) denotes the input and output of the model during training. The initial term corresponds to the loss of the CNN network during standard training, λ is an adjustment coefficient, and $g(\gamma)$ denotes the sparse penalty applied to the scaling factor. This chapter employs the L1 norm as the penalty term, illustrated in Equation 3.12.

$$g(\Upsilon) = |\Upsilon| \tag{3.12}$$

Further, batch normalization, widely used as an enhancement technique in CNN networks, is usually placed before the activation layer. This approach improves the network's generalization performance and speeds up its convergence rate. The associated formula is depicted in Equation 3.13:

$$\hat{z} = \frac{Z_{in}\mu_B}{\sigma_B^2 + \varepsilon}; z_{out} = \Upsilon\hat{z} + \beta \tag{3.13}$$

In this context, B denotes the current minimum number of batches, with μ_B and δB representing the mean and standard deviation of B respectively. The parameters g and b are the trainable affine transformation coefficients for scale and displacement. z_{in} and z_{out} refer to the input and output of the BN layer.

For channel pruning and fine-tuning of the enhanced model, Inspired by the network-slimming pruning algorithm [4], we incorporate the enhanced framework with a pruning algorithm specifically designed for lightweight object detection networks. This algorithm prunes the convolutional channels within the detection network, making it directly applicable to convolutional neural network-based object detection. Additionally, the resulting pruned model can be utilized for rapid detection tasks without the need for specialized hardware or underlying libraries.

The detailed process of the pruning algorithm proposed is shown in Algorithm 4 and Fig. 3.8. It uses the scaling factor of the BN layer as the measure of model pruning and the convolution channel below the preset threshold to reduce the volume size of the algorithm by the pruning operation, then train the trimming algorithm to recover the accuracy and finally obtain a lightweight detection network. Fig. 3.8 shows the specific process of the channel pruning algorithm. Following sparse training, the lower convolution channels are pruned. This pruning process modifies the network structure in comparison to the original model, while the neural network parameters remain unchanged. Consequently, the pruned model's target detection and mapping capabilities are diminished. To mitigate this, the pruned model undergoes fine-tuning, allowing it to relearn the neural network parameters in accordance with the new network structure. This fine-tuning process restores the model's detection accuracy and enhances its mapping performance.

In conclusion, the one-stage detection model consists of three main stages: sparse training, model pruning, and model recovery. Initially, our enhanced model undergoes sparse training. Following this, channel pruning is applied to the network model based on

Algorithm 4 A channel pruning algorithm based on the scaling factor

Input: Original model
Output: Lightweight compression model
$bn\_weights = \gamma_1, \gamma_2, \gamma_3, \ldots, \gamma_n$, Initialize $Epochs = 200$
// Turn on the sparse training session
while epoch $\leq$ Epochs **do**
    $g(Y) \leftarrow |Y|$
    objective function $L \leftarrow \sum_{(x,y)} l(f(x, W), y) + \lambda \sum_{Y \in r} g(Y)$
    Output of the BN $z_{out} \leftarrow Y\hat{z} + \beta$
    Update $\gamma_1, \gamma_2, \gamma_3, \ldots, \gamma_n$
    $bn\_weights \leftarrow \gamma_1, \gamma_2, \gamma_3, \ldots, \gamma_n$
end while
Initialize $prune\_ratio = 0.9$
To $\gamma_1, \gamma_2, \gamma_3, \ldots, \gamma_n$ Sort by channel number
if $c_1, c_2, c_3, \ldots, c_n$ $channel\_id < len(channels)$ $prune\_ratio\times$ **then**
    Trim the channel $c_1, c_2, c_3, \ldots, c_i$
end if
Fine-tuning of the pruning network

a specific pruning strategy. Finally, fine-tuning is performed to obtain the final lightweight, compressed model.

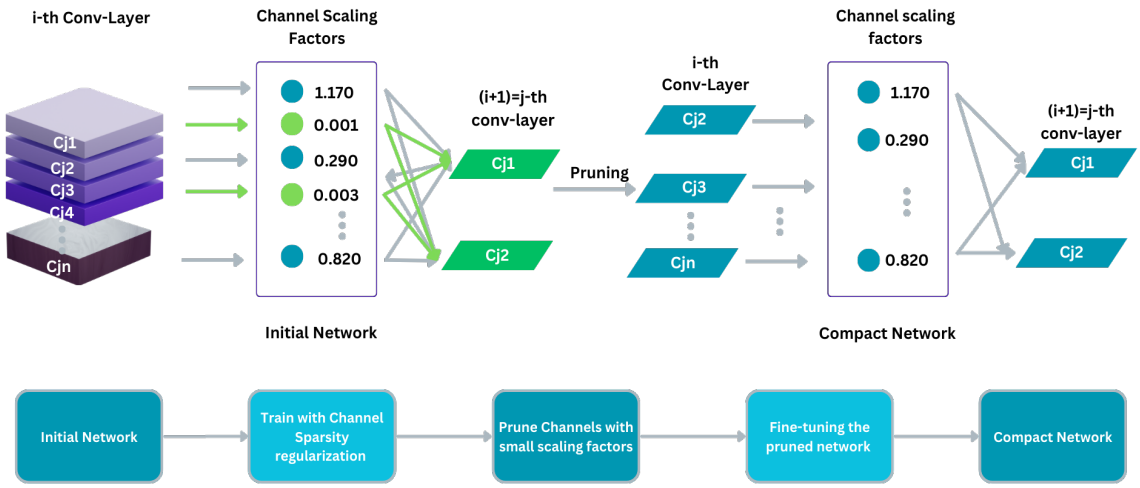


Figure 3.7: Network Slimming procedure.

To integrate fine-tuning steps in network slimming for model compression and save training time, we fine-tune the pruned model while introducing a teacher network to guide the learning of the student network. The distilled network model can be pruned iteratively to achieve better results. The integration of network slimming aims to reduce the computational complexity and memory footprint while maintaining or even enhancing object detection accuracy, particularly in challenging weather conditions. This process involves identifying and pruning convolutional channels based on their importance, as measured by the scaling factor of BN layers. After pruning, a retraining phase fine-tunes the model to recover its accuracy, resulting in a more efficient and streamlined detection network capable of performing well under adverse environmental conditions.

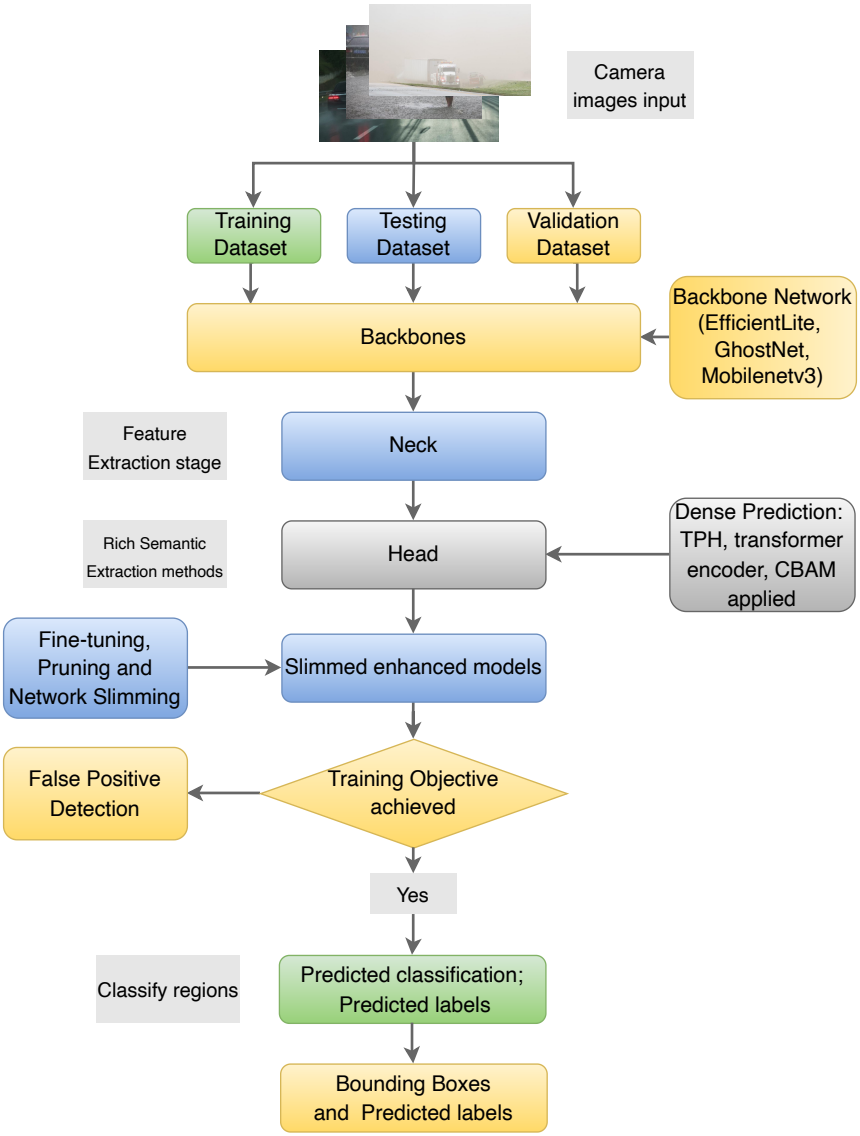


Figure 3.8: Workflow in the proposed one-stage detection model framework.

Chapter 4

Results and Discussions

Chapter 4 assesses the performance of YOLOv5 alongside the proposed one-stage detection framework. Initially, YOLOv5 undergoes training on the use-case data, and the outcomes are contrasted with those of a relevant baseline model. This evaluation incorporates both standard metrics and those specific to our use case. Subsequently, we introduce and advocate for an alternative corner grouping strategy, followed by a comprehensive evaluation of this approach.

4.1 Experimental Setup

This study employs the refined DAWN dataset to assess our model's performance, reporting metrics such as mAP (averaged across all 10 IoU thresholds ranging from 0.5 to 0.90), and AP_{50} and AP_{95}. Throughout the training process, we conducted a comparative analysis of various IoU methodologies, including basic models, enhanced YOLOv5, enhanced YOLOv5xP2, and enhanced YOLOv5 with the P2CBAM module. After the training, we will delve into discussions on the Mean Average Precision (mAP), the Precision-Recall (PR) curve, and the F1 curve.

The DAWN dataset is trained on a machine with four GPUs using NVIDIA-SMI version 470.63.01, driver version 470.63.01, and CUDA version 11.4. Overall, we have the metrics $mAP_{0.5}$ is 0.99252, and $mAP_{0.5:0.95}$ is 0.85658, which is higher 4.552% than the target paper. To achieve a higher performance than the target paper based on the YOLOv5 basic model. Our objective was to surpass the performance outlined in the target paper based on the YOLOv5 basic model. To do this, we improved the base model by including

pruning techniques, adding batch norm synchronization, and putting multi-scale training solutions into practice. Additionally, we adopted the slimming methodology, incorporating techniques such as channel sparsity regularization and scale sparse rate during training. The results obtained through these enhancements surpass the benchmarks set in the target paper.

4.2 Performance Measurement

The proposed one-stage detection framework has been assessed using the baseline YOLOv5 results. Initially, the enhanced model is trained on our redefined data, and the outcomes are compared with those of a corresponding baseline model.

4.3 Dataset

While many datasets, including KITTI [77], Cityscapes [78], and BDD100K[79], exist in the field of autonomous driving, it is challenging for these datasets to simultaneously meet the requirements of varying severe weather, resolutions, target scales, and target numbers. In contrast, the DAWN dataset better satisfies the needs for sample realism and extreme environment variation; as a result, we finally focused the analysis on the open-source dataset DAWN.

A vehicle's ability to detect adverse weather is described in the DAWN dataset. A diverse range of traffic flows and a variety of traffic environments, including freeways, interstates, and cities, are the main topics of the Nature Dataset. The DAWN dataset was collected from people, cars, buses, trucks, motorcycles, and bicycles during fog, rain, snow, and dust storms. For the studies on autonomous vehicle identification in inclement weather, this dataset is helpful. Since there aren't enough motorcycles in this dataset, we ignore it and concentrate on recognising people, cars, buses, trucks, and bicycles.

The DAWN dataset encompasses 1000 images captured in authentic traffic settings, categorized into four classes based on weather conditions: fog, snow, rain, and sandstorms. Each image in the dataset is meticulously annotated with object bounding boxes, facilitating applications in autonomous driving and video surveillance scenarios. This thesis will utilize a subset of 505 images randomly selected from the dataset, ensuring representation across all six classes. The weather-adverse dataset is shown graphically in Table 4.1.

Table 4.1: DAWN Dataset: Vehicle Detection in Adverse Weather Nature: dataset categories and quantities.

		Objects					
Training Stages	**Num of Images**	**Person**	**Bicycle**	**Car**	**Motorcycle**	**Bus**	**Truck**
Training set	328	162	550	1515	0	21	91
Validating set	106	28	193	226	9	5	14
Testing set	71	9	74	203	1	0	7
Total	505	199	817	1944	10	26	112

Furthermore, object bounding boxes are marked for scenarios such as autonomous driving and video surveillance in the DAWN dataset. The DAWN dataset may help to clarify how bad weather affects the functioning of vehicle identification systems. To align with the software configuration, we re-defined the DAWN dataset on workstations [80].

To make training the YOLOv5 model easier, we also re-annotate the dataset. Training, validation, and test data sets were created from the 505 chosen pictures, with a ratio of 6:2:1. To reduce the possibility of bias arising from "samples" in the training, validation, and testing sets, we performed the whole splitting process using a single Python script.

4.4 Implementation Details

All experiments detailed in this thesis are conducted on a computing system featuring a 5-core Intel(R) Xeon(R) W-2295 CPU running at a clock speed of 3.00 GHz, coupled with an RTX 3070 graphics card. The experimental environment utilized PyTorch 1.8.1, Python 3.8.8, and CUDA 11.4. For the training of the proposed model on the DAWN dataset, the input image size was set to 512×512, with a batch size of 2, a learning rate of 0.01, and an initial IoU detection threshold of 0.50.

The training process comprised 450/500 rounds. The AP, representing the area under both the mAP and the PR curve, was employed as a comprehensive evaluation metric. The AP condenses the information from the PR curve into a single scalar value, indicating high AP when both precision and recall are elevated, and conversely, low AP when either of them is diminished across various IoUs confidence threshold values. Tables 4.2 and 4.2 provide detailed information on the hardware and software configurations used in the experiments. Table 1 outlines the specific hardware components, including their

specifications and performance metrics. Table 2 lists the software configurations, detailing the operating systems, application versions, and any additional tools or libraries utilized during the experimental procedures.

Parameters	**Values**
Processor	5-core Intel(R)
CPU	Xeon(R) W-2295 CPU
RAM	0.01
Memory	62.5 GiB
Graphics	5 GiB
OS Type	64-bit
Disk	2.5 TB
Graphics	NVIDIA GeForce RTX 3070/PCIe/SSE2
GNOME	Version 3.28.2

Table 4.2: Hardware Configuration of the proposed methodology in the experiments.

It is generally described as the harmonic mean of the two. Harmonic mean is another way to calculate an "average" of values, generally described as more suitable for ratios (such as precision and recall) than the traditional arithmetic mean.

4.5 Evaluation Metrics

To effectively evaluate the performance of the enhanced YOLOv5s algorithm, this thesis primarily uses Precision, Recall, F1 score, and AP as evaluation metrics. Additionally, PR curves are plotted to further assess the algorithm's performance.

The formula for Precision is

$$Precision = \frac{TP}{TP + FP} \tag{4.1}$$

The precision rate indicates the percentage of samples identified as targets by the target detection algorithm that are actually targets. Here, TP (true positive) represents the

Parameters	Values
Python Version	Python 3.8.8
Image size	512×512
initial learning rate	0.01
final OneCycleLR learning rate	0.1
momentum	0.937
IoU detection threshold	0.50
IoU training threshold	0.5
Batch size	4
box loss gain	0.05
cls loss gain	0.3
Training epochs	450/500

Table 4.3: Software Configuration of the proposed methodology in the experiments.

number of positive samples correctly predicted as positive, FN (false negative) represents the number of positive samples incorrectly predicted as negative, and FP (false positive) represents the number of negative samples incorrectly predicted as positive.

The formula for Recall is

$$Recall = \frac{TP}{TP + FN} \tag{4.2}$$

Recall indicates how many of the samples that are actually targeted are accurately detected. FN denotes the number of false negative cases, or the number of samples for which the predicted negative cases are actually positive cases.

The F1-confidence curve is a measure combining both precision and recall. The formula used for the F1-score in this case is Equation 4.3 [1].

$$F1\,Score = 2 \cdot \frac{Precision \cdot Recall}{Precision} + Recall \tag{4.3}$$

Precision and recall are generally negatively correlated; as precision increases, recall tends to decrease. The $F1$ score provides a balanced measure that considers both precision

[1]The F1 score is defined as the harmonic mean of precision and recall.

and recall, offering a comprehensive evaluation of the detection model. A higher $F1$ score indicates better model performance.

The formula for calculating the average precision (AP) is:

$$AP_i = \frac{TP + TN}{TP + FN + FP} \tag{4.4}$$

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i \tag{4.5}$$

The relationship between precision and recall is depicted by a PR curve. The area under the PR curve represents the AP for the target. AP measures the algorithm's average accuracy for a specific class of targets, whereas mAP is the average of AP values across all classes.

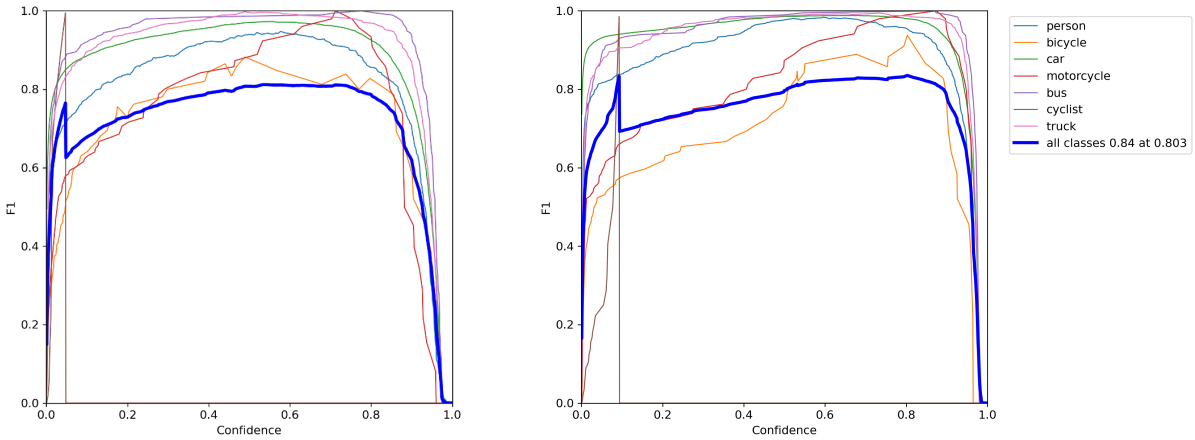
4.6 Model Training

In this thesis, the parameters and results of both original YOLOv5 and the proposed model are evaluated against the baseline model using COCO object recognition training models. The training data is employed to fine-tune all parameters, ensuring optimal performance for each model. This comparison facilitates a thorough assessment of the proposed method's effectiveness and improvements.

4.7 Performance Results

In the Fig. 4.1, the highest F1 value of 0.90 is matched by the confidence value of 0.666, which maximizes accuracy and recall. The graph shows higher confidence values and F1 scores as the epoch increases to 500, and the mAP@0.5:0.95 index continues to rise. In the experiment, we train the input data with YOLOv5 using the default settings of the parameters. Subsequently, we trained the improved YOLOv5 algorithm with various backbones and necks. Notably, the enhanced method, configured for smaller target sizes, exhibited superior P-values and mAP) compared to alternative configurations. The outcomes of the improved method surpassed the performance of other advanced object detection systems by showing the graphs of comparison and analysis.

The enhanced method was used to generate the Precision-Recall (PR) curves before and after training of initial and improved data sets. The PR curves encapsulate the area



(a) (b)

Figure 4.1: F1 curves for training the proposed YOLOv5 in small and large sizes target, where (a) is YOLOv5s and (b) is YOLOv5x.

formed by Precision (P) and Recall (R). In Fig. 4.2 and section (a), a clear trend is observed where the areas of the three plots sequentially expand. A confidence of 0.95 and 0.81 has been obtained that maximizes accuracy and recall, corresponding to an F1 value of 0.883.

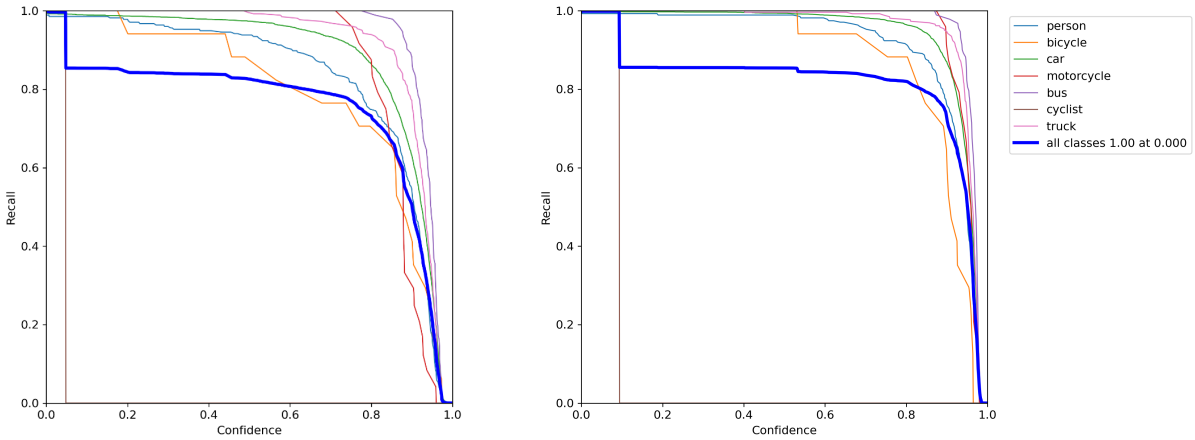


Figure 4.2: Recall curves for training the proposed YOLOv5 in small and large sizes target, where (a) is YOLOv5s and (b) is YOLOv5x.

In the second training session, we trained for the enhanced YOLOv5 model with the extra-large target, employing the default methodology for the initial training process. Consequently, the default methodology with the extra-large(x) models within our enhanced YOLOv5 algorithm has facilitated the attainment of optimized results on the proposed dataset.

4.8 Quantitative Evaluation

Section 4.8 validates the efficacy of the proposed method in target detection by employing crucial evaluation metrics such as AP_{50}, AP_{95}, mAP_{50}, and mAP_{95}. This section uses important assessment indicators to confirm that the recommended strategy for target identification works. Our modified YOLOv5x approach has improved the performance of mAP@0.5, mAP@0.95 and Recall in identifying large objects, with scores of 0.99, 0.90 and 0.99 in Table 4.4. These metrics are much better than the comparable values for the baseline YOLOv5, which are 0.90, 0.69, and 0.85 compared to the baseline data in Table 2.1. The suggested approaches yield outcomes similar to those of the original YOLOv5, as demonstrated by our illustrated outcomes in the test datasets, as presented in Fig. 4.9. Specifically, our technique shows distinct improvements over the suggested YOLOv5 compared to our framework in all other detection scenarios.

Furthermore, Table 4.4 shows that the improved techniques are in comparison to the

Table 4.4: The performance results during the enhanced YOLOv5 training process.

Methodology	**Precision(%)**	**Recall(%)**	**F1**	**mAP**$_{0.5}$(%)	**mAP**$_{0.95}$(%)
TPH-Slimming Pruned (Our)	0.94	0.81	0.89	96.8	76.1
YOLOv5x (Our)	0.75	**0.99**	0.66	**99.1**	**90.0**
YOLOv5s (Our)	0.95	0.81	0.88	98.5	80.9
YOLOv5s+YOLOv5xP2CBAM (Our)	**0.97**	0.80	**0.91**	98.1	73.5
YOLOv5xP2+YOLOv5s (Our)	0.90	0.83	0.80	97.2	75.1
YOLOv4-tiny[17]	0.82	0.52	0.64	49.9	29.1
YOLOv3-tiny[17]	0.80	0.59	0.68	51.3	25.4
YOLOv5-baseline[17]	0.91	0.82	0.90	84.7	65.1
B-T (5)-N-CBAM (EIOU)[17]	0.97	0.92	0.94	94.7	72.3
B-T (5)-N-CBAM (CIOU)[17]	0.92	0.91	0.91	94.3	71.0

benchmark in the state-of-the-art. When YOLOv5-baseline findings from upgraded approaches are compared in Table 2.1, mAP@0.95 has risen by 24.9%, while mAP@0.5 is expanding by 14.4%. Precision and recall increased by 6 and 17 percent, respectively. As a result, the proposed detecting head progresses in maintaining the characteristics of tiny objects. In addition, involution effectively improves channel information, and CBAM Block draws attention to significant details as it extracts them from the spine.

Algorithm 5 and Fig. 4.3 shows some examples of the visualized detection results by our approach in extra large targets during training and validation in the training YOLOv5 in different weather conditions:

Algorithm 5 Architecture Modification

1: Remove CBAM modules 14 and 19 from the architecture.
2: Reduce the number of channels connected to P2 to 128.
3: Insert the Spatial Pyramid Pooling (SPP) module before the output head, ensuring that the SPP kernel size decreases as the pixels of P decrease.
4: Check the output after the CBAM module.
5: Retain only the backbone and TransBlock from the final layer's output.
6: Use a weighted bi-directional feature pyramid network (BiFPN) [38] as the neck in the neural network structure.

The suggested approach efficiently identifies a greater number of objects while sig-

nificantly reducing processing time. This is achieved through our resilient strategy and weather-resistant image processing techniques, such as connected component labeling instead of using Canny edge detection and contouring.

Some examples of object detection using improved YOLOv5. Including but not limited to sandy, snowy, pedestrians, vehicles, foggy pedestrians, cars, etc. The objects in the figure are labelled with yellow orange and red anchor boxes.

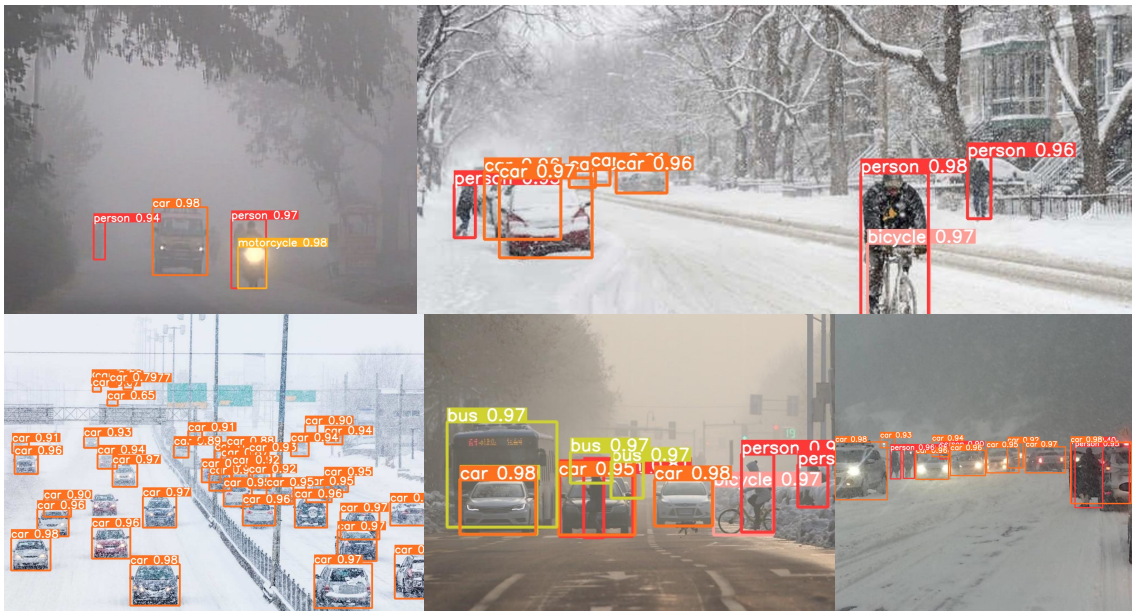


Figure 4.3: The proposed method's detection results are broken down by the number of epochs used for training and validation in each class using the training YOLOv5.

4.9 Performance Comparison

The idea of section 4.9 is to provide a single metric that weights the two ratios precision and recall in a balanced way, requiring both to have a higher value for the F1-score value to rise.

Fig. 4.4 and 4.5 show the performance of five methods at different IoU thresholds. The F1-score of all deep learning models drops significantly when the IoU threshold increases from 0.5 to 0.9, with the 31%, 45% and 28% decreases for Faster-RCNN, YOLOv3 and CenterNet respectively. The bounding box of a RoI is predicted by statistical regression

in the high-layer feature map of the Convolutional neural network (CNN), where one pixel in this abstract feature map corresponds to a pixel block in the original image. That is, a minor change of the predicted coordinates in the abstract feature map will lead to a large change in the exact position in the original image. Therefore, deep learning models either detect more elements with loose bounding boxes or detect fewer elements with accurate bounding boxes.

We employed TensorBoard to monitor the metric curves of the model training data (mAP@0.5, mAP@0.5:0.95, Precision (%), Recall(%)) in real-time during the model training, and we performed a real-time visual inspection of the main performance metrics of the algorithm before and after the improvement, as shown in Fig. 4.4 and Fig. 4.5. With the increase in epoch, the model performance metrics keep changing. Among them, mAP@0.5, Precision and Recall level off at an epoch of 450, but the mAP@0.5:0.95 index still has an increasing trend. To extract as many features as possible and achieve better performance, we finally set Epoch to 500 for comprehensive consideration. Compared with normal images, it is more difficult to train images in bad weather, we can see that the YOLOv5-improve algorithm has the best results in all four metrics, which also proves the superiority of our improved algorithm.

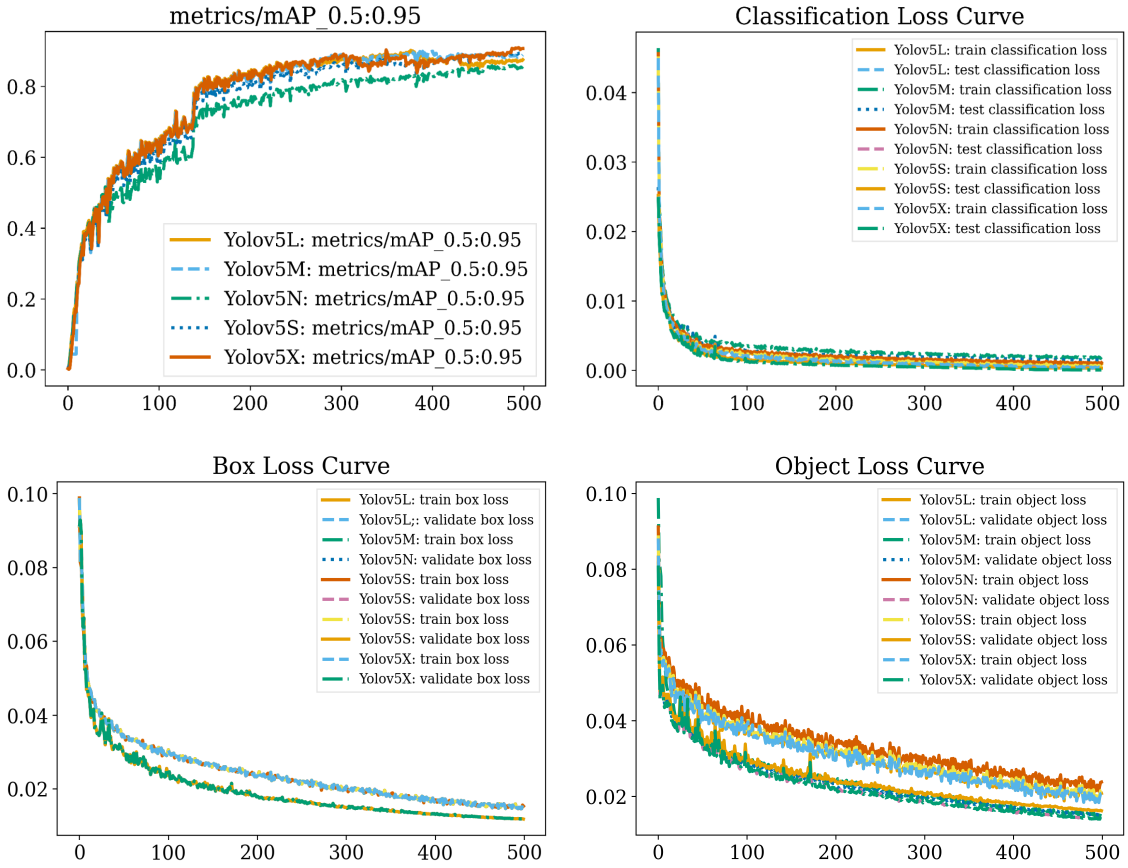


Figure 4.4: The results of training YOLOv5 for extra large size model with the Convolutional Block Attention Module (CBAM).

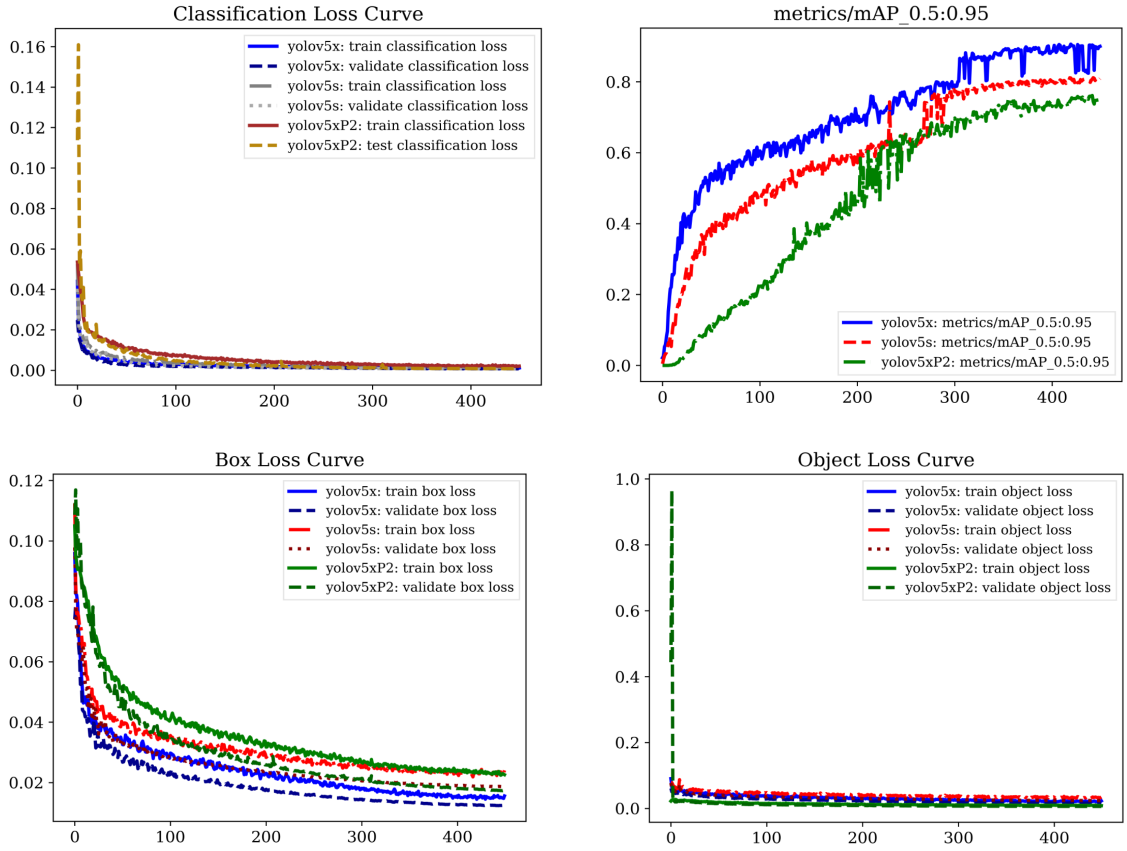


Figure 4.5: The results of training YOLOv5 for extra large size model with the Convolutional Block Attention Module (CBAM).

Fig. 4.4 depicts the outcomes of training the YOLOv5s model and the enhanced model on the DAWN dataset. The graph includes the loss of the bounding box, the mean value of target detection loss, and the mean value of classification loss on both the training and validation sets, along with the evaluation metrics employed in this thesis. In contrast to Fig. 4.5, the improved model considerably increases the recall rate along with improving the detection accuracy of small objects.

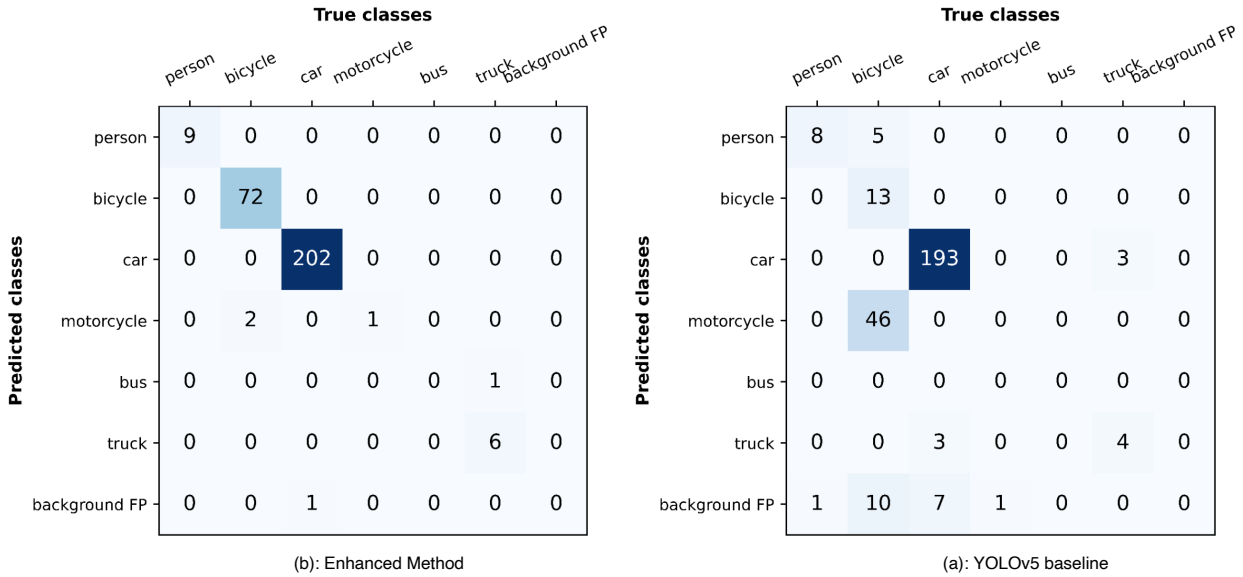


Figure 4.6: The confusion matrix comparison for baseline and enhanced methods.

Fig. 4.6 shows the confusion matrices of our proposed algorithms for comparative study on the validation dataset. We generated the results with an IoU of 50%. Our proposed one-stage deep learning-based approach has predicted the images based on the cars, bicycles and or trucks in weather-adverse conditions. Moreover, the results confirm that our proposed methodology has superiority over the baseline YOLOv5 algorithms in classifying the DAWN data. The result illustrates the confusion matrix of the proposed approach. It can be noted that the Deep learning-based approach has resulted in detecting more than 4.4% in cars, 80% in bicycles and 42% in trucks respectively.

4.10 Navigating the trade-off in bounding-box accuracy

The evaluation of the COCO dataset follows standard metrics, as referenced in [8]. Our analysis includes reporting AP metrics at various intersections over union (IOU) thresholds: 0.5 (AP0:5), 0.95 (AP0:95), and the average across ten equidistant IOU thresholds from 0.5 to 0.95 (AP). Table 4.5 presents the corresponding outcomes for both object detection (bounding box-based) and instance segmentation (mask-based) formulations of the problem.

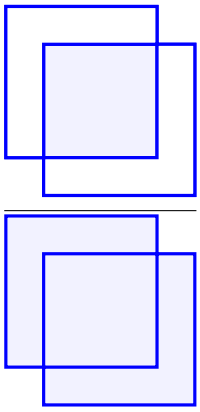
Equation 4.6 [2], as given IoU scores for each channel, our results show that each example has different IoU scores, as each is represented different and has individual activation throughout all channels in the network. Therefore, by using a set of data samples to find the optimal channel subset, the judgment of the selection criteria becomes more accurate.

[2]Visualized of IoU equation.

For the redefined DAWN dataset, we used the results from each class in the training set to compute the IoU score for each channel.

Fig. 4.7 based on Equation 4.6 shows the comparison of different IoU measured criteria and a reduction in the accuracy of various convolution layers, differentiating all three methods. The proposed method votes for the higher IoU scores, comparing these scores among all examples in the baseline model, and assigns a voting score to compute a measure of relevance that identifies the most critical channels.

$$IoU = \frac{\text{Area of Overlap}}{\text{Area of Union}} = \quad (4.6)$$



Besides, Fig. 4.7 and Table 4.5 demonstrate the efficacy of five approaches across varying IoU thresholds. The F1 scores of all models exhibit a notable decline as the IoU threshold rises from 0.5 to 0.9. In particular, the baseline model decreased by 16%, and the enhanced model was reduced by 19%. In this study, statistical regression within the convolutional neural network's high-layer feature map is to foresight the bounding box for a Region of Interest (RoI). In this abstract feature map, each pixel in the original image correlates to a pixel block. In other words, a slight modification in the expected coordinates of the abstract feature map will result in a noticeable shift in the exact location within the original image.

Considering the values at the lowest IoU threshold examined (0.5), the improved YOLO outperforms the YOLO baseline by significant margins. For the AP evaluation in Table 4.5, the improved YOLO's AP0:5-value is 8.1 points higher than the baseline (0.99 vs. 0.92). With an AP0:70 of 0.91 and performance values at the IoU threshold of 0.70, YOLO demonstrated improvement while maintaining a higher proportion of detection with bounding boxes classed as accurate. Still, when looking at the F1-Score evaluation, improved YOLO achieves a significantly higher precision of 83.5, compared to 66.5 for the baseline model. This observation can be expanded for the recall scores at higher IoU thresholds:

Table 4.5: Comparison Performance at different IoU Thresholds

Methodologies	**Metrics**	**IoU Threshold (%)**				
		$AP_{0.5}$	$AP_{0.6}$	$AP_{0.7}$	$AP_{0.8}$	$AP_{0.9}$
YOLO baseline model	Precision	0.92	0.87	0.81	0.74	0.68
	F1-Score	0.88	0.84	0.80	0.75	0.69
	Recall	0.81	0.79	0.76	0.73	0.69
YOLO enhanced model	Precision	0.99	0.95	0.91	0.85	0.79
	F1-Score	0.91	0.88	0.85	0.81	0.75
	Recall	0.88	0.86	0.84	0.80	0.76

our improved solution still gains an advantage over our baseline method as detection accuracy requirements increase. This is illustrated in Fig. 4.7, which visualizes the same results shown in Table 4.5.

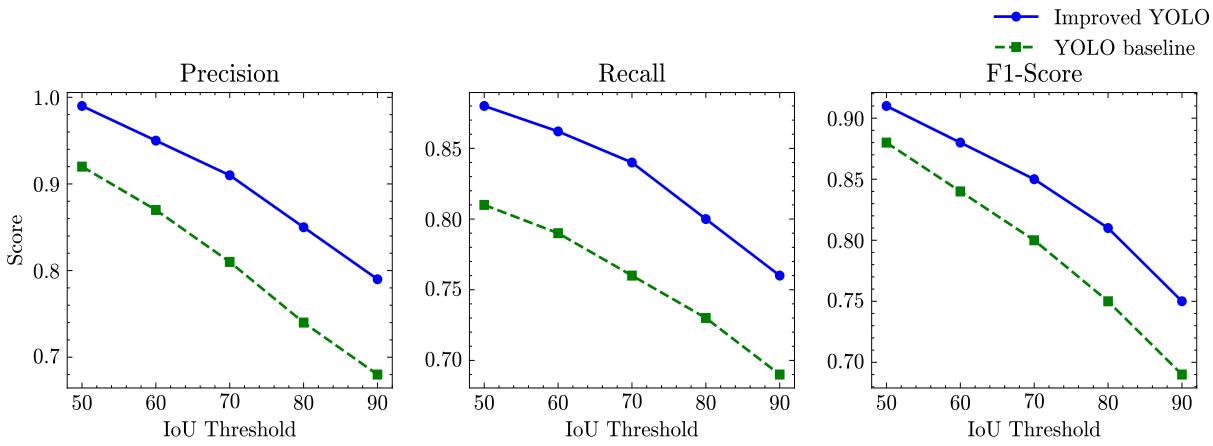


Figure 4.7: Comparison of improved and baseline models at different IoU thresholds (%)

Because our workflow requires accurate object prediction to be used in the context of weather-adverse object recognition, the results show that the suggested framework's predictions appear to be very accurate, as the average accuracy in the bottom part of Fig. 4.7 remains high as the IoU accuracy requirements increase. Either a higher proportion of components with loose bounding boxes or a lower proportion of items with correct bounding boxes are detected by deep learning models [81]. Nevertheless, when the IoU threshold rises, the F1-scores of YOLOv5 and improved YOLOv5 decrease.

However, deep learning models either detect more elements with loose bounding boxes

or detect fewer elements with accurate bounding boxes [81]. In contrast, the F1-score of YOLOv5 and YOLOv5 improved does drop as significantly, as that of deep learning models as the IoU threshold increases. The deep learning model demonstrates the capability to identify a greater number of components; however, its bounding box lacks precision. In other words, the deep learning approach exhibits a high recall rate, but the accuracy of bounding box localization is insufficient.

4.11 Comparison Experiment of different IoU thresholds

It can be seen in Fig. 4.8, the graph demonstrates the graph of AP (%) vs threshold IoU compared with other object detection techniques such as YOLO5x, YOLO5s, and YOLO5xP2 on customer DAWN dataset.

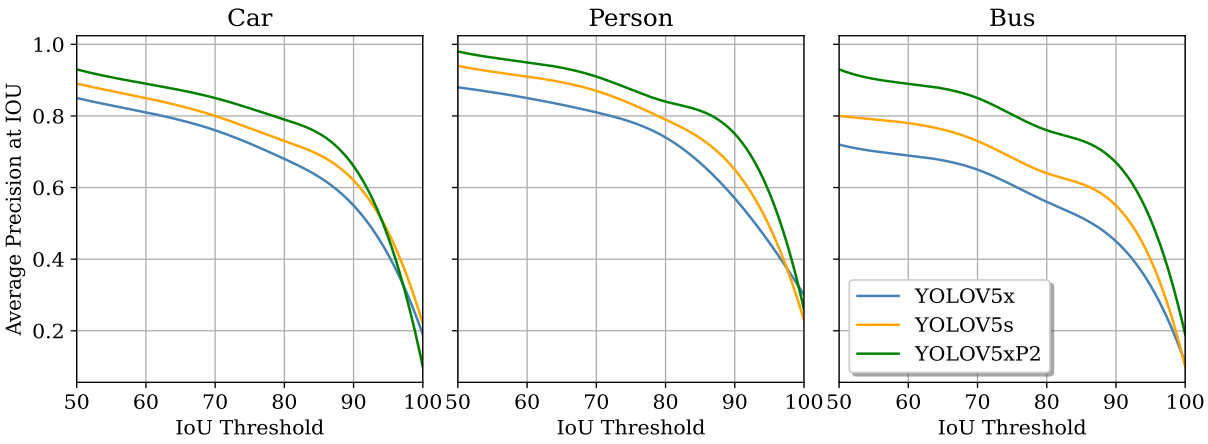


Figure 4.8: Average Precision of Improved models at different IoU thresholds (%)

As more false negative samples are excluded as the IoU threshold is lowered, the model's performance and detection gradually improve. The most notable increase in the model's object identification skills can be observed by the precision peaking at IoU 0.5 when the IoU threshold is set. Consequently, our suggested framework's YOLOv5XP2 shows better resilience in every study class.

4.12 Ablation Study

The sigmoid-weighted linear unit activation function (SiLU) [82] served as the foundation for the enhanced model. According to Liu, *et al.* in [83], models with limited activation, for example, ReLUX, are amenable to quantization. Nevertheless, models with unbounded activation functions, such as SiLU or Hard-Swish, are not. Therefore, with ReLU activated, we retrained the models. Changing the activation from SiLU to ReLU is observed to decrease by approximately 1 to 2%. To further position these models as embedded-friendly, we have quantified them. The example of YOLOv5 for small targets in the ReLU function demonstrates that these models may be quantized with a minor drop of around 1.2 % accuracy.

Specifically, Post Training Quantization (PTQ) [84] is used to create the findings below, instead of Quantization Aware Training [85]. An example of inference results for row and column detection is displayed in Fig.4.6. The models operated well at the lower IoU thresholds, with F1 scores of 93.4% and 94.7%, respectively, the models performed well at the lower IoU thresholds. With F1 scores of 57.4% and 58.2%, respectively, the graph displayed positive outcomes at the 90% IoU threshold. Inference results for separate models are presented in Fig. 4.1 and Fig. 4.2. These graphs demonstrate that individual models detect more than a combined model does.

Table 4.6: Comparison results of ablation experiments on DAWN dataset: Vehicle Detection in Adverse Weather Nature Dataset.

Methodologies	**Solutions**	**mAP@0.5**	**mAP@0.5:0.9**
TPH-Slimming Pruned	Our proposed	0.94(↑2.15)	0.81(↑2.15)
YOLOv5x	Our proposed	0.75 (↑1.10)	0.99 (↑1.15)
YOLOv5s	Our proposed	0.95(↑0.10)	0.81 (↑0.15)
YOLOv5s+ YOLOv5xP2CBAM	Our proposed	0.97 (↑2.15)	0.80 (↑2.15)
YOLOv5xP2+YOLOv5s	Our proposed	0.90(↑0.39)	0.83 (↑0.05)
B-T (5)-N-CBAM (EIOU) [5]	State-of-the-art	0.97 (↑1.15)	0.92(↑1.15)
B-T (5)-N-CBAM (CIOU) [5]	State-of-the-art	0.95(↑0.35)	0.91(↑1.15)

4.13 Detection Results Comparison

The outcomes on the test set of the baseline and enhanced models are shown in Fig. 4.9. For large targets, both methodologies can be identified accurately. The recognition

confidence of our proposed method, which indicates that the improved YOLOv5 has a more enhanced ability to recognize in terms of foreground probability as opposed to YOLOv5, is significantly greater compared with YOLOv5. In some scenes with dense targets, as shown in Fig. 4.9 (c). Because of the challenging weather conditions and instances of vehicle or pedestrian overlap and occlusion, the YOLOv5 algorithm has missed detection. However, the improved YOLOv5 still accurately identifies its target.

This study undertakes a comparative analysis with the most recent YOLO family of object detection algorithms, with a primary emphasis on assessing the algorithm's detection accuracy and speed. The objective is to showcase the enhanced performance of the YOLOv5x algorithm compared to other algorithms. Specifically, in (c), the proposed framework exhibits enhanced capabilities in detecting small objects within weather adverse conditions.

Hence, based on the findings, our trained weights exhibit superior performance in detection, even for the smallest or farthest objects, compared to the weights of the YOLOv5 algorithm. The advanced detection capabilities demonstrated by our work under challenging weather conditions contribute to enhancing the perception of AVs. Additionally, to realize the potential of autonomous driving globally, addressing autonomy in adverse weather conditions is essential. Our work specifically focuses on perception in a wide spectrum of challenging weather conditions, contributing to the broader applicability of autonomous driving technology.

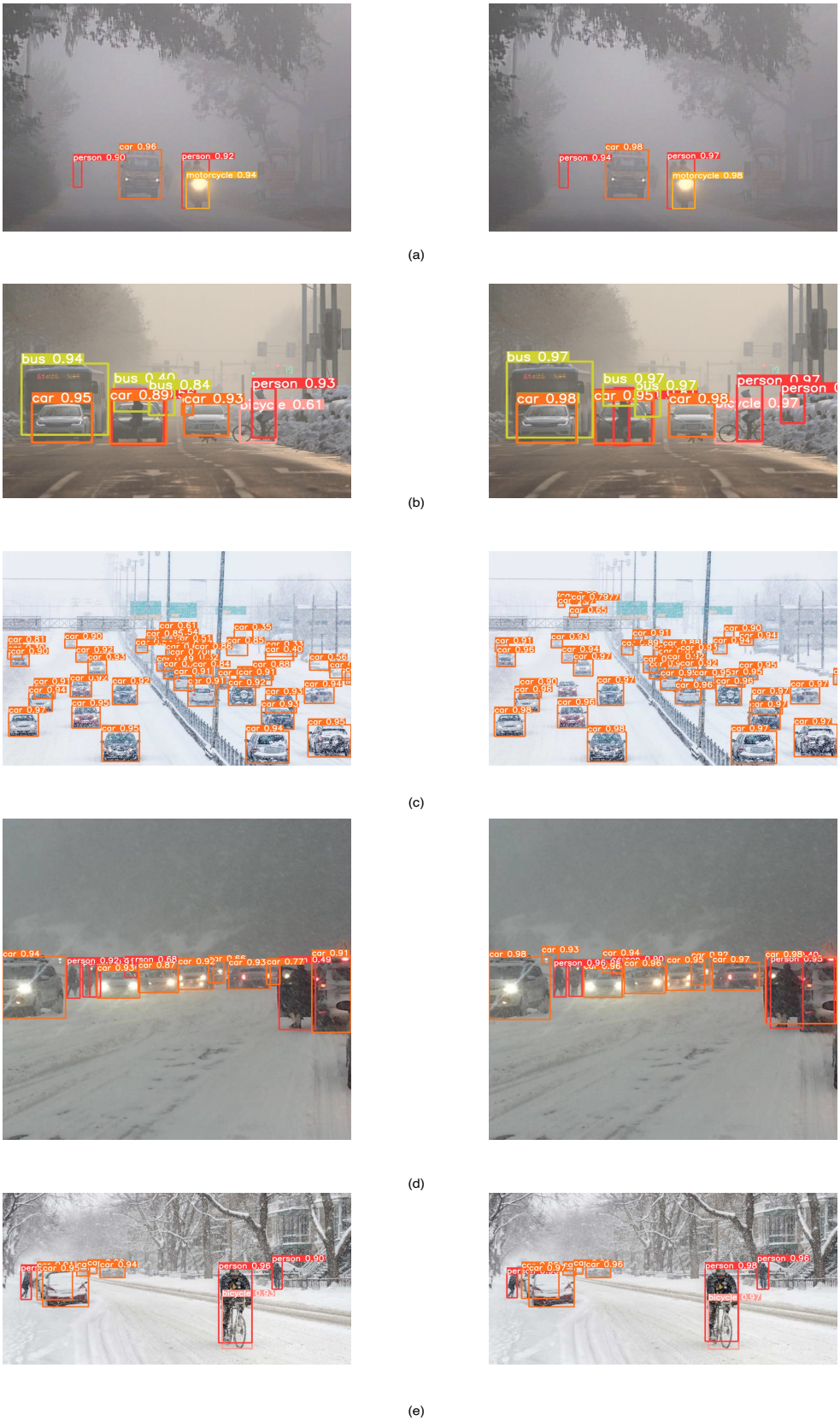


Figure 4.9: The detection outcomes for each class include many epochs from the training and validation stages. The outcomes for YOLOv5 are on the left; on the right are the results of the upgraded model.

4.14 Extra Prediction Head

If a small object detection head is added, the original YOLOv5s' layer count increases from 701 to 750, and their GFLOPs go from 119.0 to 266.0. Of course, this results in more calculations, but the mAP improvement is also very significant. It is evident that the enhanced computation has advantages for the recognition of tiny items using our suggested model.

4.15 Transformer Encoder Block

The model's total number of layers drops from 719 to 705, and its GFLOPs drop from 259.0 to 237.3, following the use of the transformer encoder block. In addition to increasing mAP, using Transformer Encoder Blocks can help shrink the network's size. Concurrently, it contributes to the identification of large and dense things.

4.16 Model Ensemble

The mAP of our five distinct models' outcomes in each category was given, and we compared it with the fusion model shown in Fig. 4.6. To make each model distinct, we vary the weight of each category and utilize various input picture sizes in the training phrase. For the output of the final ensemble model to be roughly balanced.

1. Improved-YOLOv5-1 uses the input image size of 520 and all categories have equal weights.

2. Improved-YOLOv5-2 uses the input image size of 510 and all categories have equal weights.

On the DAWN dataset, some detections were made. We have selected a few photographs to serve as a representative of the test findings. The outcome of huge objects, small objects and dense objects are shown in Fig. 4.6.

For the impact of the model ensemble, we list the mAP of the final results of our five different models in each category and compare them with the fusion model in Table 4.6. In the training phase, we use different input image sizes and change the weight of each category to make each model unique. The final ensemble model can get a relatively balanced result.

4.17 Effect of Extra Prediction Head

The layer count of the original YOLOv5x increases from 607 to 719 and GFLOPs from 219.0 to 259.0 with the addition of a small object-detecting head. In addition to increasing computation, this additionally results in a mAP improvement. The performance of TPH-YOLOv5 in identifying tiny objects is demonstrated in Fig. 4.6, indicating that the computation increase is justified.

4.18 Summary

In this chapter 4, the findings from the ablation study provide valuable insights into developing more robust object detection models in weather-adverse conditions. This methodology outperforms an F1-score and AP at the IoU threshold from an intersection over union (IoU) threshold of 0.5 (AP0:5), 0.75 (AP0:75) and averaged for ten equidistant IoU thresholds from 0.5 to 0.95 (AP). We add some cutting-edge techniques, i.e. transformer encoder block, convolutional block attention module (CBAM) and some experienced tricks to YOLOv5 and form a state-of-the-art detector called TPH-YOLOv5 as our improved YOLO solution, which is especially competent at object detection in weather adverse scenarios. We refreshed the record of the DAWN dataset, our experiments showed that the improved YOLO module achieved the state-of-the-art performance in the DAWN dataset.

Chapter 5

Conclusions and Future Works

The section 5 introduces a novel framework called the One-Stage Detection Multi-Backbone Compression Framework, which excels at object detection in adverse weather conditions. The study integrates advanced techniques, including the transformer encoder block, CBAM module, and various practical enhancements, into a robust YOLOv5 model. To enhance the object detector's accuracy, we experimented with different characteristics and incorporated several effective ones. This work significantly improves 2D object detection in severe weather conditions.

In this thesis, our contributions begin with the introduction of an enhanced framework that integrates multiple lightweight backbones, such as ShuffleNetV2, GhostNet, and VoVNet. Furthermore, we incorporate advanced computer vision attention mechanisms, including the squeeze-and-excitation (SE) block, CBAM, and efficient channel attention (ECA) block, aimed at enhancing object detection performance in challenging environments.

In terms of the advancements in the object localization, this study integrated the transformer prediction heads (TPH) into YOLOv5, significantly improving object localization, particularly in adverse scenes.

For the framework simplification, this thesis employs pruning, quantization, and distillation methods specifically tailored to address adverse weather-related issues, thereby reducing the computational cost and size of the model.

Finally, we integrated the CBAM into YOLOv5. To improve the network's ability to recognize regions of interest in images with broad region coverage, we augment the YOLOv5 with CBAM. Additionally, a self-trained classifier is utilized for enhanced classification of

object categories.Our proposed framework achieves an impressive 99.7% average precision (AP), outperforming the baseline method by 5%.

Therefore, the proposed framework represents a promising solution for enhancing a wide range of vision-based applications, particularly those operating in challenging weather conditions. Its applicability spans various domains, including UAV-based object detection [86; 87], pedestrian safety alert systems [88; 89], and intelligent transportation systems leveraging vehicle-to-infrastructure (V2I) communication [90]. Our approach effectively addresses the challenges posed by adverse weather, thereby enhancing the performance and reliability of these critical applications.

5.1 Current Challenges

The study objective is accomplished, and the research presented in the thesis improves the ability of self-driving cars to identify impediments in hazardous weather conditions. For example, traffic flow monitoring and potential danger identification in industrial environments are only two scenarios in which the YOLOv5-based improved technique is utilized to improve safety.

The state-of-the-art detector known as TPH-YOLOv5 is created by combining many cutting-edge approaches, such as transformer encoder block, CBAM, and various optimization methods, with the conventional one-stage method YOLOv5. This detector excels at object recognition in settings where drones collect it. We updated the DAWN dataset records, and the tests revealed that TPH-YOLOv5 performed at the cutting edge in the DAWN dataset. To enhance the accuracy of the object detector, we tested various attributes and implemented a select few. This study will improve the experience of researchers and developers in processing and analyzing drone instances.

In the realm of AVs and advanced machine learning models, understanding and mitigating risks and unresolved issues is critical for ensuring safety, efficiency, and reliability. This section delves into several key areas where improvements and further research are essential, particularly focusing on quantization techniques for machine learning models, the robustness of 3D object detection in adverse weather conditions, and the application of vision transformers for structural identification.

5.2 Future work

Section 5.2 outlines potential avenues for future research and development stemming from the findings and methodologies presented in this thesis. The identified enhancements

and frameworks have laid a solid foundation, but there are several areas where further exploration and refinement could significantly advance the field of vision-based object detection, particularly in challenging environments such as adverse weather conditions and resource-constrained settings.

The proposed future work include improvements in transformer architectures, advancements in 3D object detection methodologies, exploration of vision transformers for structural identification, and optimization of quantization techniques in YOLOv5. These areas represent promising directions to enhance the robustness, efficiency, and applicability of vision systems across various real-world applications.

5.2.1 Transformer Improvements

Quantizing FP32 [91] to INT8 is not a smooth transformation, while it is not particular to YOLO, and it may compromise the optimality of the outcomes if the gradient terrain is severe. Furthermore, because doing so would severely impair the model's performance, it is challenging to employ PyTorch Quantization (PTQ) to obtain a low-bit (4-bit) precision. The ready-to-use quantization modules such as TensorRT [92], PTQ, and ONNX quantization [93] are currently popular. However, because of their 8-bit precision restriction, these modules cannot achieve extremely low precision. However, as the goal in this study is to identify new quantization techniques applied to YOLOv5, such research is not covered.

Regarding applied studies on quantizing one stage object detection model, more research is conducted using Quantization Aware Training (QAT) [85] with a wide range of precision from 1 bit to 8 bits. However, there is a gap in focusing on accelerating training time as well as inference time, especially because training YOLOv5 on a new dataset is computationally and time-consuming. As a solution, more work can be done employing integer-only quantization since hardware throughput is much higher when operations are performed using integer numbers. For instance, TITAN RTX can perform around 23 times more operations per second when the data type is INT4 rather than FP32.

Additionally, PTQ methods still fall back when lower than 8-bit precision is investigated and needed, which presents the opportunity for future work in this area. Hence, we recommend some approaches that can be applied to YOLOv5 to fill the above-mentioned gap. A PTQ algorithm, AdaRound [94], is proposed to more efficiently round weights for quantization. It accomplishes the state-of-the-art performance with as low as 4-bit precision without a noticeable drop in accuracy (<1%). Yao *et al.* proposed HAWQ-V3 [95], a mixed-precision integer-only quantization method that reaches INT4 or INT4/INT8 uniformly mapped quantization. AdaQuant [96] proposed a PTQ quantization scheme to minimize the quantization errors of each layer or block separately by optimizing its pa-

rameters based on a calibration set. It can attain the state-of-the-art quantization with INT4 precision, which leads to a negligible accuracy drop. The authors presented a quantization method that exclusively utilizes integer-based operations and eliminates redundant instructions during inference. In [97], the effects of quantization on the loss landscape are assessed, and a new PTQ technique that directly minimizes the loss function can achieve 4-bit precision, yielding near-full-precision baseline accuracy.

5.2.2 Using 3D Object Detection in Weather Adverse Conditions

We anticipate delving deeper into a cutting-edge technique for physically correct snowfall and wet surface modelling in winter sceneries created from clean LiDAR data. Additionally, we have demonstrated the efficacy of the suggested methodology by evaluating the augmentation using seven distinct 2D object-detecting techniques and consistently obtaining gains.

In the upcoming study, for instance, we will include that multi-modal sensor fusion is among the most promising methods since it makes use of the advantages of various sensors to supply redundant detection data.

5.2.3 Vision Transformers for Identifying Structures

With an accuracy ranging from 1 bit to 8 bits, more studies are being done on quantizing YOLOv5 through practical investigations using QAT. However, there is little focus on accelerating the training and inference duration, especially as YOLOv5 training demands a significant amount of computation and work when using a new dataset. Given that operations on integer numbers provide significantly higher hardware throughput, integer-only quantization is a viable alternative for additional work. For instance, TITAN RTX can perform almost 23 times more operations per second when the data type is INT4 rather than FP32.

5.2.4 Quantization in YOLOv5

Prior to trimming the YOLOV5, several compensations should be made [29]. For example, it does not take into account trimming the YOLOV5 head, the concatenate layer, or the up-sampling layer.

To allow the inputs to have a varied number of channels, it also ignores the shortcut connection in the Bottle-Neck module [98]. Further research in this area should focus on filter-based and kernel-based pruning strategies since they simplify the pruning process by not altering the number of output channels. Since hardware throughput is significantly

better when operations are conducted using integer numbers, more work may be done utilizing integer-only quantization as a solution. When the data type is INT4 instead of FP32, for example, TITAN RTX may execute about 23 times more operations per second. Furthermore, when investigating or requiring lower than 8-bit precision, the Post-training Quantization (PTQ) approaches [84] still revert, which offers a chance for further research in this field.

Therefore, we suggest a few methods that may be used with YOLOV5 to close the aforementioned gap. AdaRound [99], a PTQ method, may be suggested in further work to round weights for quantization more effectively.

5.3 Challenges and Risks in AI-Driven Automotive Systems

As AVs continue to evolve, they promise transformative benefits in transportation efficiency and safety. However, this technological advancement also introduces new and complex risks that must be carefully managed. One of the primary challenges lies in understanding and predicting the behavior of AI systems that drive these vehicles. Unlike traditional vehicles where human error is a major risk factor, AVs driven by AI present unique challenges in assessing and mitigating risks.

The opacity of AI algorithms and the difficulty in accessing and interpreting their source code pose significant hurdles in evaluating their reliability and safety. This section explores several key emerging risks associated with AVs, including behavioral unpredictability, cybersecurity vulnerabilities, connectivity issues, and internal hardware and software failures. Each of these risks not only presents technical challenges but also raises critical questions about regulatory frameworks, ethical considerations, and public trust in autonomous technologies.

Moreover, the shift towards AI-driven mobility necessitates a shift in how risks are identified, assessed, and managed. While advancements in AI offer promising capabilities, they also underscore the need for transparent and accountable practices in developing and deploying autonomous systems. This section examines these issues to provide a comprehensive understanding of the evolving landscape of AV risks and the strategies needed to navigate them effectively.

5.3.1 Emerging AV Risks

AVs represent a significant advancement in transportation technology, where driving responsibilities are increasingly entrusted to AI systems. However, this transition introduces

several emerging risks that need careful consideration. One of the critical challenges is the evaluation of AI's behavior in diverse driving scenarios, compounded by the complexity of accessing and interpreting AI source code.

Object detection, a fundamental capability in AVs, exemplifies these challenges. AI-driven object detection algorithms must accurately identify and classify objects such as pedestrians, vehicles, and obstacles in real-time. Evaluating the performance and safety implications of different object detection algorithms across various AV platforms becomes inherently challenging due to the proprietary nature of AI models and the lack of transparency in their inner workings.

Moreover, the reliance on black box methods in AI development, where outputs are analyzed without fully understanding the underlying software and hardware components, poses risks. This approach limits the ability to comprehensively assess the robustness and reliability of object detection systems in AVs. Clear box approaches, which advocate for transparent access and analysis of AI components, are crucial for mitigating these risks by enabling thorough evaluation and accountability in algorithm performance.

Addressing these challenges is essential not only for enhancing the safety and effectiveness of AV technologies but also for fostering public trust and regulatory compliance. As object detection continues to evolve in autonomous driving, navigating these emerging risks will require interdisciplinary collaboration, ethical considerations, and robust frameworks for AI governance.

5.3.2 Behavioural Risk

In traditional driving environments, human error—resulting from intentional wrongdoing, careless conduct, driver exhaustion, or distraction—significantly contributes to driving risks. The introduction of AI into driving activities shifts this paradigm, presenting unique challenges in risk evaluation and management. Understanding and predicting AI behavior in diverse driving situations is complex, particularly due to the inherent difficulty in accessing and interpreting AI source code.

Object detection, a critical component of AI in AVs, exemplifies these challenges. Object detection algorithms must accurately identify and classify various objects such as pedestrians, vehicles, and obstacles in real-time to ensure safe driving. However, the opacity of AI models and the use of black box methods, where AI outputs are assessed without full transparency into the underlying software and hardware functions, complicate the evaluation of algorithmic performance and safety.

This lack of transparency makes it difficult to comprehensively assess the reliability and robustness of object detection systems across different AV platforms. Evaluating

the behavioral risks associated with object detection algorithms—such as their response to complex scenarios or unexpected events—requires methodologies that enable deeper insights into AI decision-making processes.

5.3.3 Risk of Connectivity

If networked driving becomes an essential component of the AV, network failures will increase the likelihood of a connection (refer to the section on automobile connectivity). For instance, a car that receives software updates from the cloud may see a decrease in performance in the event of a connectivity issue.

5.3.4 Cybersecurity Risk

Malicious cyber-attacks could affect all three of the above risks. Most malicious attacks fall into the following categories:

1. (Previously present for conventional automobiles) The intention was to steal the vehicle.
2. Take over the car and demand a ransom.
3. Data theft (personal).
4. Turn off features to cause the car to crash or stop moving altogether.

The attack surface grows along with autonomy and connection, raising the possibility of malevolent attacks. While precautions are necessary, widespread cyberattacks on connected automobiles have not yet occurred. As the automobile industry grows more software-dependent and networked, businesses and companies within it are dedicating more resources to cybersecurity.

Human-machine interaction (HMI) risk may differ throughout vehicles classified as Levels 1 through 4. This risk has two primary causes or origins.

The ADAS warning signals on a car are not always promptly and accurately responded to by a human driver. This often occurs when a motorist is unable to use the technology correctly or has an insufficient grasp of it. The existence of ADAS in the car contributes to human drivers' overconfidence.

5.3.5 Risky Behaviour

Human error is the primary source of driving errors in conventional automobiles, and it is commonly ascribed to driver distraction, fatigue, intentional misbehaviour, and risky behaviour.

On the other hand, when AI takes over driving responsibilities, it poses a unique set of concerns. Assessing this risk is difficult because of barriers like restricted access to the source code and the difficulty of understanding and decoding it. It is difficult to predict AI behaviour in a variety of driving conditions.

Furthermore, assessing the hazards of various algorithms within and between providers is a difficult task. The black box strategy is a possible substitute for the transparent box approach, which entails gaining access to and analyzing software components and their functionality. This approach entails analyzing the AI's result in terms of how a car behaves, ignoring the fundamental functions and duties of hardware and software elements.

5.3.6 Risk of AI

AI is being utilized more and more for a wide range of complicated tasks, yet many of the models are opaque, making it hard to comprehend and put your faith in them. Understanding the logic underlying the judgments made by AI models is often crucial. Consequently, there is a growing demand for Explainable Artificial Intelligence (XAI) methods [100] to enhance consumers' trust in AI models .

The AI takes over driving responsibilities. This risk is difficult to assess as it might be difficult to have access to the source code and to understand and interpret it. It is challenging to forecast AI's behaviour in different driving scenarios. Moreover, comparing the risks of different algorithms between and within providers may be challenging (e.g., figuring out which one offers the safest set of responses). This clear box approach, which involves accessing and analyzing the software components and their performance, may be substituted by a black box method, which looks at the AI's output in terms of a vehicle's behaviour while ignoring the underlying tasks and functions of software and hardware components.

5.3.7 Internal Risk

Internal risk is the potential for auto hardware, such as standard parts like brakes and steering wheels and automation-enabling gear like radars and cameras, to malfunction or fail. To ensure that the car can still function in the event of a malfunction or breakdown, redundant systems are required. For example, overlapping field views between a radar and

a camera enable two sensors to monitor the same area. An internal component, such as an algorithmic mismatch with a program version, can also contribute to software failure.

5.3.8 Cyber Risk

Malicious cyber-attacks may affect all three of the previously listed risks. Most malicious assaults fall into one of these categories:

1. The aim is to take the automobile (this used to be true for conventional autos).
2. The desire to steal the car (formerly present for conventional autos).
3. Unauthorized data acquisition (personal).
4. Turn off features that will crash the car or make it unmovable.
5. Turn off characteristics that might cause a car to crash or make it incapable of moving.

As autonomy and connectedness increase, so does the attack surface, increasing the risk of malicious attacks. Though major cyberattacks on connected cars have not yet happened, this is something for which you should be ready. Businesses and enterprises within the automotive sector are allocating more funds towards cyber security as the industry becomes increasingly software-dependent and networked. There are two primary sources, or origins, of varying degrees of Human-Machine Interaction (HMI) danger affecting vehicles from Levels 1 to 4.

1. AV risk may also change in accordance with changes in the goal functions that direct the vehicle's decision-making. Depending on how fragile the cargo is, delivery AVs could choose to give more weight to "progress towards the goal"—that is, speed—than to comfort, whereas school bus AVs might put comfort and safety ahead of speed.
2. The human driver acts over-confidently when operating an automobile with ADAS (Advanced Driver Assistance Systems).

Bibliography

[1] Q.-H. Pham, P. Sevestre, R. S. Pahwa, H. Zhan, C. H. Pang, Y. Chen, A. Mustafa, V. Chandrasekhar, and J. Lin, "A*3d dataset: Towards autonomous driving in challenging environments," 2019.

[2] P. Gavrikov, "visualkeras." https://github.com/paulgavrikov/visualkeras, 2020.

[3] OpenAI, "Chatgpt." https://openai.com/research/chatgpt, 2022.

[4] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang, "Learning efficient convolutional networks through network slimming," 2017.

[5] J. Yao, X. Fan, B. Li, and W. Qin, "Adverse weather target detection algorithm based on adaptive color levels and improved yolov5," *Sensors*, vol. 22, no. 21, 2022.

[6] M. Antonakakis, A. Tzavaras, K. Tsakos, E. G. Spanakis, V. Sakkalis, M. Zervakis, and E. G. Petrakis, "Real-time object detection using an ultra-high-resolution camera on embedded systems," in *2022 IEEE International Conference on Imaging Systems and Techniques (IST)*, pp. 1–6, 2022.

[7] M. Simsek, B. Kantarci, and A. Boukerche, "Utility-aware legitimacy detection of mobile crowdsensing tasks via knowledge-based self organizing feature map," *IEEE Transactions on Mobile Computing*, vol. 22, no. 6, pp. 3706–3723, 2023.

[8] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft coco: Common objects in context," 2015.

[9] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International Journal of Computer Vision*, vol. 111, pp. 98–136, Jan. 2015.

[10] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "Shufflenet v2: Practical guidelines for efficient cnn architecture design," 2018.

[11] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "Ghostnet: More features from cheap operations," 2020.

[12] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q. V. Le, and H. Adam, "Searching for mobilenetv3," 2019.

[13] Y. Lee, J. won Hwang, S. Lee, Y. Bae, and J. Park, "An energy and gpu-computation efficient backbone network for real-time object detection," 2019.

[14] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-excitation networks," 2019.

[15] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," 2018.

[16] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, "Eca-net: Efficient channel attention for deep convolutional neural networks," 2020.

[17] J. Yao, X. Fan, B. Li, and W. Qin, "Adverse weather target detection algorithm based on adaptive color levels and improved yolov5," *Sensors (Basel, Switzerland)*, vol. 22, 2022.

[18] R. Girshick, "Fast r-cnn," 2015.

[19] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 2016.

[20] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," 2018.

[21] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

[22] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[23] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, *SSD: Single Shot MultiBox Detector*, p. 21–37. United States of America: Springer International Publishing, 2016.

[24] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," 2018.

[25] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," 2020.

[26] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," 2016.

[27] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," 2018.

[28] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," 2020.

[29] Ultralytics, "YOLOv5: A state-of-the-art real-time object detection system." https://docs.ultralytics.com, 2021. Accessed: insert date here.

[30] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, "You only learn one representation: Unified network for multiple tasks," 2021.

[31] H.-S. Chang, C.-Y. Wang, R. R. Wang, G. Chou, and H.-Y. M. Liao, "Yolor-based multi-task learning," 2023.

[32] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "Yolox: Exceeding yolo series in 2021," 2021.

[33] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015.

[34] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," *CoRR*, vol. abs/1703.06211, 2017.

[35] W. Wang, C. Wei, W. Yang, and J. Liu, "Gladnet: Low-light enhancement network with global awareness," in *Automatic Face & Gesture Recognition (FG 2018), 2018 13th IEEE International Conference*, (China), pp. 751–755, IEEE, 2018.

[36] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "Ghostnet: More features from cheap operations," *CoRR*, vol. abs/1911.11907, 2019.

[37] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *CoRR*, vol. abs/1905.11946, 2019.

[38] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," 2020.

[39] B. Li, B. Wu, J. Su, G. Wang, and L. Lin, "Eagleeye: Fast sub-net evaluation for efficient neural network pruning," 2020.

[40] D. Misra, "Mish: A self regularized non-monotonic activation function," 2020.

[41] C. Cui, T. Gao, S. Wei, Y. Du, R. Guo, S. Dong, B. Lu, Y. Zhou, X. Lv, Q. Liu, X. Hu, D. Yu, and Y. Ma, "Pp-lcnet: A lightweight cpu convolutional neural network," 2021.

[42] X. Zhu, S. Lyu, X. Wang, and Q. Zhao, "Tph-yolov5: Improved yolov5 based on transformer prediction head for object detection on drone-captured scenarios," *CoRR*, vol. abs/2108.11539, 2021.

[43] 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," *CoRR*, vol. abs/2103.14030, 2021.

[44] Y. Li*, M. Shen*, J. Ma*, Y. Ren*, M. Zhao*, Q. Zhang*, R. Gong*, F. Yu, and J. Yan, "Mqbench: Towards reproducible and deployable model quantization benchmark," *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, 2021.

[45] H. Ding, J. Pu, and C. Hu, "Tinyneuralnetwork: An efficient deep learning model compression framework." https://github.com/alibaba/TinyNeuralNetwork, 2021.

[46] T. Wang, Y. Zhai, Y. Li, W. Wang, G. Ye, and S. Jin, "Insulator defect detection based on ml-yolov5 algorithm," *Sensors*, vol. 24, no. 1, p. 19, 2024.

[47] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," 2016.

[48] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," 2015.

[49] K. He, X. Zhang, S. Ren, and J. Sun, *Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition*, p. 346–361. China: Springer International Publishing, 2014.

[50] C.-Y. Wang, H.-Y. M. Liao, I.-H. Yeh, Y.-H. Wu, P.-Y. Chen, and J.-W. Hsieh, "Cspnet: A new backbone that can enhance learning capability of cnn," 2019.

[51] F. Shen and G. Zeng, "Weighted residuals for very deep networks," 2016.

[52] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," 2018.

[53] X. Lang, Z. Ren, D. Wan, Y. Zhang, and S. Shu, "Mr-yolo: An improved yolov5 network for detecting magnetic ring surface defects," *Sensors*, vol. 22, no. 24, 2022.

[54] S. Anzaroot, A. Passos, D. Belanger, and A. McCallum, "Learning soft linear constraints with application to citation field extraction," 2014.

[55] G. Jocher, "YOLOv5 by Ultralytics," May 2020.

[56] R. Girshick, F. Iandola, T. Darrell, and J. Malik, "Deformable part models are convolutional neural networks," 2014.

[57] C.-Y. Wang, H.-Y. M. Liao, I.-H. Yeh, Y.-H. Wu, P.-Y. Chen, and J.-W. Hsieh, "Cspnet: A new backbone that can enhance learning capability of cnn," 2019.

[58] 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.

[59] 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.

[60] G. Jocher, A. Chaurasia, and J. Qiu, "YOLO by Ultralytics," Jan. 2023.

[61] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q. V. Le, and H. Adam, "Searching for mobilenetv3," 2019.

[62] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-iou loss: Faster and better learning for bounding box regression," 2019.

[63] F. A. Fardo, V. H. Conforto, F. C. de Oliveira, and P. S. Rodrigues, "A formal evaluation of psnr as quality measurement parameter for image segmentation algorithms," 2016.

[64] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[65] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region-based fully convolutional networks," 2023.

[66] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," 2017.

[67] C. Wang, H. M. Liao, I. Yeh, Y. Wu, P. Chen, and J. Hsieh, "Cspnet: A new backbone that can enhance learning capability of CNN," *CoRR*, vol. abs/1911.11929, 2019.

[68] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," 2017.

[69] X. Zhu, S. Lyu, X. Wang, and Q. Zhao, "Tph-yolov5: Improved yolov5 based on transformer prediction head for object detection on drone-captured scenarios," in *2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pp. 2778–2788, 2021.

[70] 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 16x16 words: Transformers for image recognition at scale," 2021.

[71] M. L. Mekhalfi, C. Nicolò, Y. Bazi, M. M. A. Rahhal, N. A. Alsharif, and E. A. Maghayreh, "Contrasting yolov5, transformer, and efficientdet detectors for crop circle detection in desert," *IEEE Geoscience and Remote Sensing Letters*, vol. 19, pp. 1–5, 2022.

[72] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2023.

[73] 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," 2021.

[74] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[75] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," 2017.

[76] G.-X. Yuan, K.-W. Chang, C.-J. Hsieh, and C.-J. Lin, "A comparison of optimization methods and software for large-scale l1-regularized linear classification.," *Journal of Machine Learning Research*, vol. 11, pp. 3183–3234, 08 2010.

[77] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[78] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," 2016.

[79] 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," 2020.

[80] S. Katyal, S. Kumar, R. Sakhuja, and S. Gupta, "Object detection in foggy conditions by fusion of saliency map and yolo," in *2018 12th International Conference on Sensing Technology (ICST)*, pp. 154–159, 2018.

[81] A. Kumar, Z. J. Zhang, and H. Lyu, "Object detection in real time based on improved single shot multi-box detector algorithm," *EURASIP Journal on Wireless Communications and Networking*, vol. 2020, no. 1, p. 204, 2020.

[82] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," 2017.

[83] "Higher accuracy on vision models with efficientnet-lite." https://blog.tensorflow.org/2020/03/higher-accuracy-on-vision-models-with-efficientnet-lite.html. Accessed: 2020-03-16.

[84] Z. Liu, Y. Wang, K. Han, S. Ma, and W. Gao, "Post-training quantization for vision transformer," 2021.

[85] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," 2017.

[86] N. A. Khan, N. Jhanjhi, S. N. Brohi, R. S. A. Usmani, and A. Nayyar, "Smart traffic monitoring system using unmanned aerial vehicles (uavs)," *Computer Communications*, vol. 157, pp. 434–443, 2020.

[87] L. Lu and F. Dai, "Accurate road user localization in aerial images captured by unmanned aerial vehicles," *Automation in Construction*, vol. 158, p. 105257, 2024.

[88] P. Kohli and A. Chadha, "Enabling pedestrian safety using computer vision techniques: A case study of the 2018 uber inc. self-driving car crash," *ArXiv*, vol. abs/1805.11815, 2018.

[89] X. Li, H. Cui, J. Rizzo, E. Wong, and Y. Fang, "Cross-safe: A computer vision-based approach to make all intersection-related pedestrian signals accessible for the visually impaired," in *Advances in Computer Vision - Proceedings of the 2019 Computer Vision Conference CVC* (K. Arai and S. Kapoor, eds.), Advances in Intelligent Systems and Computing, pp. 132–146, Springer Verlag, 2020. Publisher Copyright: © 2020, Springer Nature Switzerland AG.; Computer Vision Conference, CVC 2019 ; Conference date: 25-04-2019 Through 26-04-2019.

[90] L. Lu and F. Dai, "Automated visual surveying of vehicle heights to help measure the risk of overheight collisions using deep learning and view geometry," *Computer-Aided Civil and Infrastructure Engineering*, vol. 38, 04 2022.

[91] H. Wu, P. Judd, X. Zhang, M. Isaev, and P. Micikevicius, "Integer quantization for deep learning inference: Principles and empirical evaluation," 2020.

[92] X. Xia, J. Li, J. Wu, X. Wang, X. Xiao, M. Zheng, and R. Wang, "Trt-vit: Tensorrt-oriented vision transformer," 2022.

[93] J. Bai, F. Lu, K. Zhang, *et al.*, "Onnx: Open neural network exchange." https://github.com/onnx/onnx, 2019.

[94] M. Nagel, R. A. Amjad, M. van Baalen, C. Louizos, and T. Blankevoort, "Up or down? adaptive rounding for post-training quantization," 2020.

[95] Z. Yao, Z. Dong, Z. Zheng, A. Gholami, J. Yu, E. Tan, L. Wang, Q. Huang, Y. Wang, M. W. Mahoney, and K. Keutzer, "Hawqv3: Dyadic neural network quantization," 2021.

[96] I. Hubara, Y. Nahshan, Y. Hanani, R. Banner, and D. Soudry, "Improving post training neural quantization: Layer-wise calibration and integer programming," 2020.

[97] Y. Nahshan, B. Chmiel, C. Baskin, E. Zheltonozhskii, R. Banner, A. M. Bronstein, and A. Mendelson, "Loss aware post-training quantization," 2020.

[98] J. Park, S. Woo, J.-Y. Lee, and I. S. Kweon, "Bam: Bottleneck attention module," 2018.

[99] M. Nagel, R. A. Amjad, M. van Baalen, C. Louizos, and T. Blankevoort, "Up or down? adaptive rounding for post-training quantization," 2020.

[100] S. Ali, T. Abuhmed, S. El-Sappagh, K. Muhammad, J. M. Alonso-Moral, R. Confalonieri, R. Guidotti, J. Del Ser, N. Díaz-Rodríguez, and F. Herrera, "Explainable artificial intelligence (xai): What we know and what is left to attain trustworthy artificial intelligence," *Information Fusion*, vol. 99, p. 101805, 2023.