REAL TIME TRANSFORMER BASED OBJECT DETECTION USING YOLOv8.

MAIN PROJECT REPORT

Submitted by

VISHNU V NAIR

REG NO : TKM21CSCE08

In partial fulfillment for the award of the degree of

MASTER OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING

Under the guidance of
Asst. Prof. Thushara A

Thangal Kunju Musaliar College of Engineering
Kerala

Declaration

I undersigned hereby declare that the project report on **"REAL TIME TRANSFORMER BASED OBJECT DETECTION USING YOLOv8"**, submitted for partial fulfillment of the requirements for the award of the degree of Bachelor of Technology of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by us under the supervision of **Asst. Prof. Thushara A**, Assistant Professor of the Computer Science and Engineering Department, TKMCE. This submission represents our ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that we have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in our submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University

Place: Kollam
Date: 11/07/23

VISHNU V NAIR

Thangal Kunju Musaliar College of Engineering
Dept. of Computer Science & Engineering

C E R T I F I C A T E

This is to certify that,this report titled ***REAL TIME TRANSFORMER BASED OBJECT DETECTION USING YOLOv8*** is a bonafide record of the **Main Project** presented by **VISHNU V NAIR(TKM21CSCE08)**, under our guidance and supervision, in partial fulfillment of the requirements for the award of the degree, **M.Tech Computer Science & Engineering** in **APJ Abdul Kalam Technological University** .

Coordinator

Dr. Aneesh G Nath
Associate Professor
Computer Science & Engineering

Guide

Prof. Thushara A
Assistant Professor
Computer Science & Engineering

Head of the Department

Dr. Dimple A Shajahan
Professor
Computer Science & Engineering

ACKNOWLEDGEMENT

A successful main project is a fruitful culmination of efforts by many people, some directly involved and some others indirectly, by providing support and encouragement. Firstly I would like to thank the almighty for giving me the wisdom and grace for making my seminar project a memorable one. I thank him for steering me to the shore of fulfillment under his protective wings.

I express my sincere gratitude to **Dr. T A Shahul Hameed**, Principal of T.K.M College of Engineering for giving me an opportunity to present my main project. I would like to thank **Dr. Dimple A Shajahan**, Professor and Head of the Department, CSE, TKMCE, for her constant support and encouragement throughout the work.

With a profound sense of gratitude, I would like to express my heartfelt thanks to my guide **Prof. Thushara A** , Assistant Professor, CSE, TKMCE for their expert guidance, cooperation and immense encouragement. I also extend my thanks to the entire faculty members and staffs of the Department of Computer Engineering, TKMCE, who has encouraged me throughout this work.

I also express my thanks to my loving parents, brother and friends, for their support and encouragement in the successful completion of this main project work.

VISHNU V NAIR

Abstract

Real time Object detection is a computer vision task that involves identifying and localizing objects of interest within an image or video. Many challenges need to be addressed in object detection, including occlusions, scale variations, clutter in the background, deformations and variations of objects, limited data, real-time processing demands, imbalanced classes, and the need to adapt to new object categories. This project proposes a Transformer-based object detection model to tackle the aforementioned challenges. The proposed model utilizes Transformers, originally designed for natural language processing, to address object detection challenges. The model leverages the self-attention mechanism in Transformers for feature extraction rather than relying on convolutional neural networks. This allows the models to effectively capture global and local features and learn complex spatial relationships between objects. Furthermore, the fully connected layers in the conventional object detection method are replaced with a Transformer-based detection head in the proposed models. This modification allows the model to utilize the strengths of Transformers in processing the extracted features and generating precise bounding box predictions. Also, the model can learn complex object representations and handle object occlusion, scale variation, and other challenging scenarios more effectively. This adaptation enhances the model's capability to detect and localize objects in various real-world applications accurately. The performance of the proposed Transformer-based object detection model is evaluated through experiments on widely recognized object detection benchmarks like COCO. Additionally, proprietary datasets like Next wealth are used to gauge the model's performance. The results of these evaluations exhibit significant enhancements in metrics such as mean average precision and localization accuracy compared to the other state-of-art methods. The Transformer-based object detection models demonstrate promising outcomes, showcasing improved accuracy and their capability to handle challenging scenarios and complex object interactions effectively..

Contents

List of Figures

Chapter 1

Introduction

Real-time object detection is a very important topic in computer vision, as it is often a necessary component in computer vision systems. For example, multi-object tracking , autonomous driving , robotics , medical image analysis , etc. The computing devices that execute real-time object detection is usually some mobile CPU or GPU, as well as various neural processing units (NPU) developed by major manufacturers. For example, the Apple neural engine (Apple), the neural compute stick (Intel), Jetson AI edge devices (Nvidia), the edge TPU (Google), the neural processing engine (Qualcomm), the AI processing unit (MediaTek), and the AI SoCs (Kneron), are all NPUs. Some of the above mentioned edge devices focus on speeding up different operations such as vanilla convolution, depth-wise convolution, or MLP operations. Here the real-time object detector proposed mainly hopes that it can support both mobile GPU and GPU devices from the edge to the cloud. Traditionally, object detection models are built on convolutional neural networks (CNNs). CNNs are able to learn local features from images, which are helpful for object detection. However, CNNs are not as well-suited for learning long-range dependencies between data points.Recently, transformer-based models have shown promising results for object detection. Transformers are neural networks that are able to learn long-range dependencies between data points. This makes them well-suited for tasks such as object detection, where it is important to be able to identify objects that are located far apart in an image.The transition from convolution to transformer-based object detection has been driven by several factors. First, transformers have been shown to achieve better accuracy than CNNs on a variety of natural language processing (NLP) tasks. . This suggests that transformers may also be able to achieve better accuracy on object detection tasks.Second, transformers are more efficient than CNNs. This is because transformers do not require the use of specialized hardware, such as GPUs. This makes them more scalable and easier to deploy in real-world applications.Third, transformers are more flexible than CNNs. This is because transformers can be used to model a wider range of relationships between data points. This makes them more suitable for complex object detection tasks.As a result of these factors, transformer

based object detection models are becoming increasingly popular. They are now being used in a variety of applications, such as self-driving cars and video surveillance.accuracy on object detection tasks.Second, transformers are more efficient than CNNs. This is because transformers do not require the use of specialized hardware, such as GPUs. This makes them more scalable and easier to deploy in real-world applications.Third, transformers are more flexible than CNNs. This is because transformers can be used to model a wider range of relationships between data points. This makes them more suitable for complex object detection tasks.As a result of these factors, transformer based object detection models are becoming increasingly popular. They are now being used in a variety of applications, such as self-driving cars and video surveillance. The contributions are summarized as follows:(1) Introduced the C3TR module instead of the C2F module. The C2F module is a convolutional block that is used in the YOLOv8 object detection model. The C3TR module is a transformer-based block that is used in the proposed model. The C3TR module is able to learn long-range dependencies between data points, which helps to improve the accuracy of the model.(2)Replaced the head part with a transformer-based detection head called RT-DETR. The head part of an object detection model is responsible for generating bounding box predictions. The original YOLOv8 model uses a CNN-based head. The proposed model uses a transformer-based head called RT-DETR. The RT-DETR head is able to learn long-range dependencies between data points, which helps to improve the accuracy of the model.

Chapter 2

Related Works

2.1 DETR

DETR is a transformer-based object detection model [3]. DETR uses a transformer encoder-decoder architecture to learn the relationships between different parts of an image. It has been shown to achieve state-of-the-art results on the COCO object detection benchmark.
The architectural details of DETR are as follows:

- The transformer encoder is a stack of self-attention layers. Each self-attention layer attends to different parts of the image, which allows the model to learn the relationships between different parts of the image.
- The transformer decoder is a stack of decoder layers. Each decoder layer takes as input the output of the transformer encoder and predicts the bounding boxes and class labels of the objects in the image.
- DETR uses a sparse attention mechanism to reduce the computational cost of the model. The sparse attention mechanism only attends to a subset of the input data, which makes the model more efficient

2.2 Sparse RCNN

Sparse RCNN uses a sparse attention mechanism to reduce the computational cost of transformer-based object detection models. It has been shown to achieve comparable results to DETR on the COCO object detection benchmark.
The architectural details of Sparse RCNN are as follows:

- The backbone of Sparse RCNN is a ResNet-50 CNN. The ResNet-50 CNN extracts features from the image.
- The transformer encoder is a stack of self-attention layers. Each self-attention layer attends to different parts of the feature maps extracted by the ResNet-50 CNN.

- The transformer decoder is a stack of decoder layers. Each decoder layer takes as input the output of the transformer encoder and predicts the bounding boxes and class labels of the objects in the image.
- Sparse RCNN uses a sparse attention mechanism to reduce the computational cost of the model. The sparse attention mechanism only attends to a subset of the feature maps extracted by the ResNet-50 CNN, which makes the model more efficient.

2.3 FairMOT

FairMOT (Fairness-aware Multi-Object Tracking) is a transformer-based object tracking model . FairMOT uses a transformer encoder-decoder architecture to learn the relationships between different frames of a video. It has been shown to achieve state-of-the-art results on the MOTChallenge object tracking benchmark.
The architectural details of FairMOT are as follows:

- The backbone of FairMOT is a ResNet-50 CNN. The ResNet-50 CNN extracts features from the video.
- The transformer encoder is a stack of self-attention layers. Each self-attention layer attends to different parts of the feature maps extracted by the ResNet-50 CNN.
- The transformer decoder is a stack of decoder layers. Each decoder layer takes as input the output of the transformer encoder and predicts the bounding boxes and class labels of the objects in the video.
- FairMOT uses a fairness-aware attention mechanism to reduce the bias in the model. The fairness-aware attention mechanism ensures that the model pays attention to all objects in the video, regardless of their size, shape, or appearance.

2.4 ViT-FRCNN

ViT-FRCNN uses a Vision Transformer (ViT) encoder as the backbone of its object detection model. It has been shown to achieve comparable results to

DETR on the COCO object detection benchmark.
The architectural details of ViT-FRCNN are as follows:

- The backbone of ViT-FRCNN is a ViT encoder. The ViT encoder extracts features from the image.
- The transformer decoder is a stack of decoder layers. Each decoder layer takes as input the output of the ViT encoder and predicts the bounding boxes and class labels of the objects in the image.
- ViT-FRCNN uses a sparse attention mechanism to reduce the computational cost of the model. The sparse attention mechanism only attends to a subset of the feature maps extracted by the ViT encoder, which makes the model more efficient.

2.5 Swin Transformer

The Swin Transformer consists of a stack of Swin Transformer blocks. Each Swin Transformer block is composed of two sub-blocks: a shifting window attention block and a feedforward block.

The shifting window attention block performs self-attention on a shifted window of image patches. The shifting window is a technique that allows the Swin Transformer to learn long-range dependencies between image patches that are far apart. The architectural details of Swin Transformer are as follows:

- Hierarchical architecture: The Swin Transformer uses a hierarchical architecture to learn long-range dependencies between image patches. This hierarchical architecture allows the Swin Transformer to learn dependencies between image patches that are far apart.
- Shifting window attention: The Swin Transformer uses a shifting window attention block to perform self-attention on a shifted window of image patches. This shifting window allows the Swin Transformer to learn long-range dependencies between image patches that are far apart..

Chapter 3

Methodology

3.1 Model architecture

3.1.1 The reason for choosing YOLOv8 as the baseline

YOLO is currently the most popular real-time object detector, which can be widely accepted for the following reasons: a)Lightweight network architecture. b) Effective feature fusion methods. c) The detection results are more accurate. yolov8 is designed to combine the advantages of many real-time object detectors. It still adopts the idea ofCSP in YOLOv5[14], feature fusion method (PAN-FPN)[15][16] and SPPF module. Its main improvement is: a) It provided a brand new SOTA model, including P5 640 and P6 1280resolution object detection networks and YOLACT's instance segmentation model[10]. Inorder to met the needs of different projects, it also designed models of different scales based on the scaling coefficient like YOLOv5. b) On the premise of retaining the original idea of YOLOv5, the C2f module is designed by referring to the ELAN structure in YOLOv7[12]. c)The detection head part also used the current popular method (separating the classification and detection heads)[17]. Most of the other parts were still based on the original idea of YOLOv5. d) YOLOv8 classification Loss used BCE Loss, The regression Loss was of the form CIOU Loss + DFL, VFL proposes an asymmetric weighting operation[11]. DFL: The position of the box is modeled as a general distribution. Let the network quickly focus on the distribution of the location close to the target location, and make the probability density near the location as large as possible, as shown in formula(1).siis the output of sigmod for the network,yiandyi+1are interval orders, y is label. Compared to the previous YOLO algorithm, YOLOv8 is very extensible. It is a framework that can support previous versions of YOLO, and can switch between different versions, so it is easy to compare the performance of different versions.

$$DFL_{(s_i,s_{i+1})} = ((y_{i+1} - y)log(s_i) + (y - y_i)log(s_{i+1})) \tag{3.1}$$

YOLOv8 used Anchor-Free instead of Anchor-Base. V8 used dynamic

TaskAlignedAssigner for matching strategy. It calculates the alignment degree of Anchor-level for each instance using Equation(2),sis the classification score, u is the IOU value,andare the weight hyperparameters. It selects m anchors with the maximum value (t) in each instance as positive samples, and selects the other anchors as negative samples, and then trains through the loss function. After the above improvements, YOLOv8 is 1% more accurate than YOLOv5, making it the most accurate detector so far

$t=s\times u$

The key feature of YOLOv8 is that it is extensible. Yolov8 is designed to work with all versions of YOLO and switch between them, making it easy to compare their performance,which is a great benefit for researchers working on YOLO projects. Therefore, YOLOv8version was selected as the baseline.

3.2 The network structure of YOLOv8

The Backbone part of YOLOv8 is basically the same as that of YOLOv5, and the C3module is replaced by the C2f module based on the CSP idea. The C2f module learned from the ELAN idea in YOLOv7, and combined C3 and ELAN to form the C2f module, so that YOLOv8 could obtain more abundant gradient flow information while ensured lightweight.At the end of backbone, the most popular SPPF module was still used, and three MaxPools of size 5×5 were passed in serial, and then each layer was concatenation, so as to guarantee the accuracy of targets in various scales while ensuring lightweight simultaneously.

In the Neck part, the feature fusion method used by YOLOv8 is still PAN-FPN, which strengthens the fusion and utilization of feature layer information at different scales. The authors of YOLOv8 used two upsampling and multiple C2f modules together with the final decoupled head structure to compose the Neck module. The idea of decoupling the head in YOLOx, was used by YOLOv8 for the last part of the neck. It combined confidence and regression boxes to achieve a new level of accuracy.

YOLOv8 can support all versions of YOLO, and can switch between different versions at will. It can also run on various hardware platforms (CPU-GPU), which has strong flexibility. Figure 1 shows the YOLOv8 network architecture diagram.

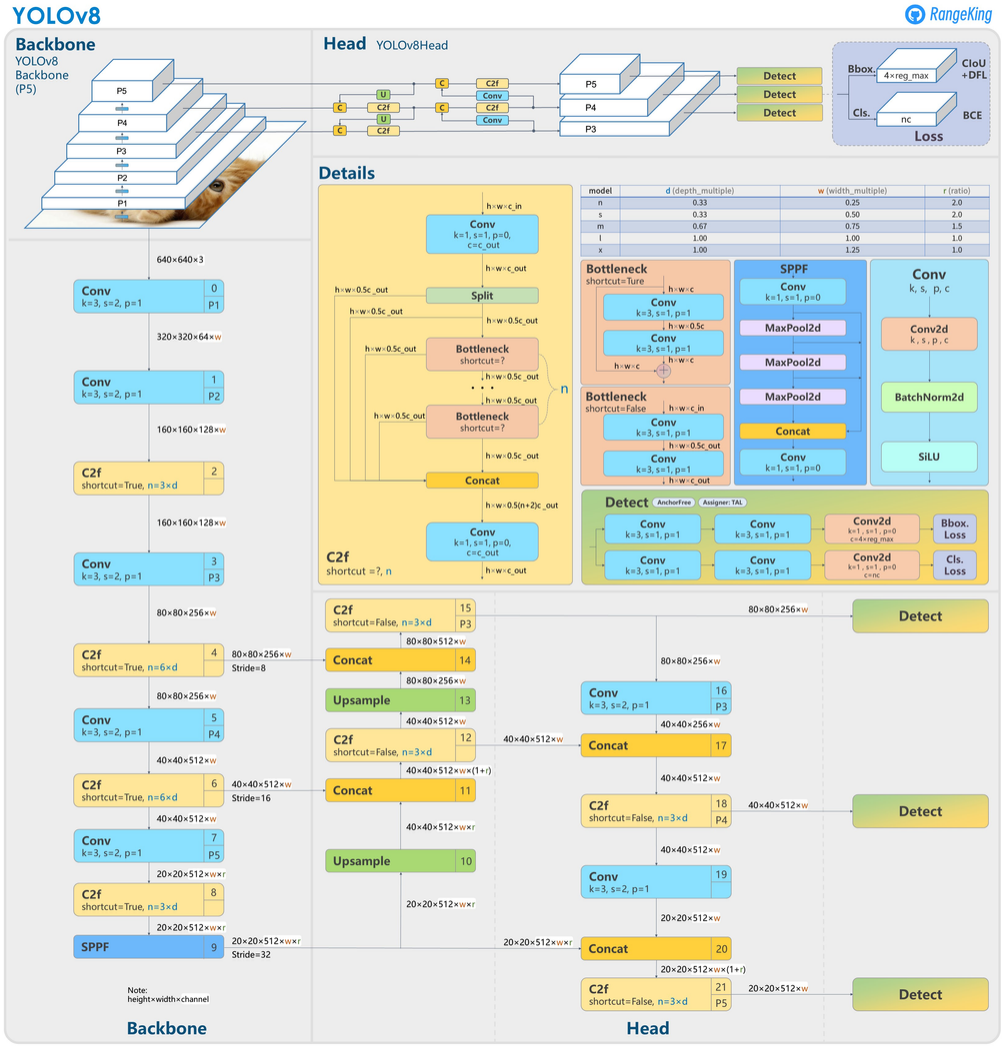

Figure 3.1: Yolov8 Architecture

3.3 The proposed Transformer based YOLOv8 algorithm

The proposed architecture replaces the detection head of YOLOv8n with RTDETR and some of the C2F blocks with C3TR blocks. RTDETR is a transformer-based detection head that has been shown to be more accurate than the YOLOv8 detection head. C3TR blocks are transformer-based blocks that have been shown to be more efficient than C2F blocks.

The proposed architecture is introduced because it aims to improve the accuracy and efficiency of YOLOv8n. The RTDETR detection head is more accurate than the YOLOv8 detection head, and the C3TR blocks are more

efficient than the C2F blocks.

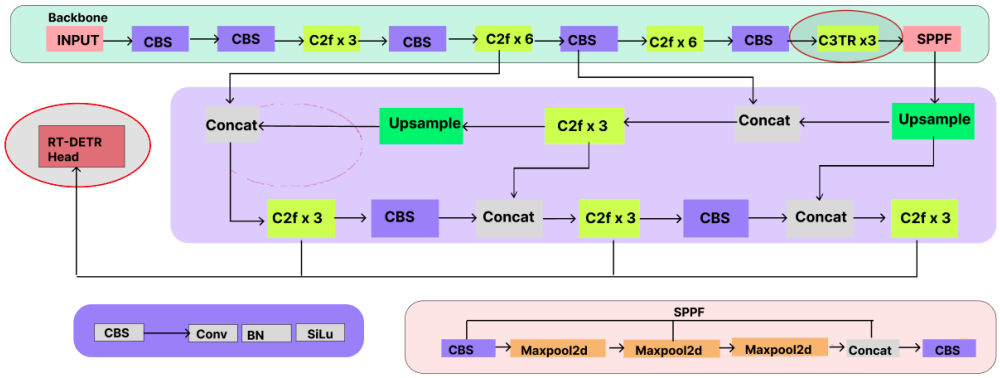

Figure 3.2: Proposed architecture

3.3.1 C2F blocks vs. C3TR blocks

- C3TR blocks are more accurate than C2F blocks because they use a hierarchical attention mechanism that allows them to learn long-range dependencies between different parts of the image.
- C2F blocks are convolutional blocks that use a traditional convolution operation to extract features from the input image. This operation is efficient, but it does not allow the blocks to learn long-range dependencies between different parts of the image.
- C3TR blocks are transformer-based blocks that use a hierarchical attention mechanism to learn long-range dependencies between different parts of the image. This mechanism allows the blocks to learn relationships between features that are far apart in the input image. This can help the blocks to better identify objects in the image and to predict their bounding boxes more accurately.

3.3.2 RT-DETR HEAD

RT-DETR consists of a backbone, a hybrid encoder and a transformer decoder with auxiliary prediction heads. The overview of the model architecture is illustrated in Fig. 3.4. Specifically, leverage the output features of the last three stages of the backbone S3, S4, S5 as the input to the encoder. The

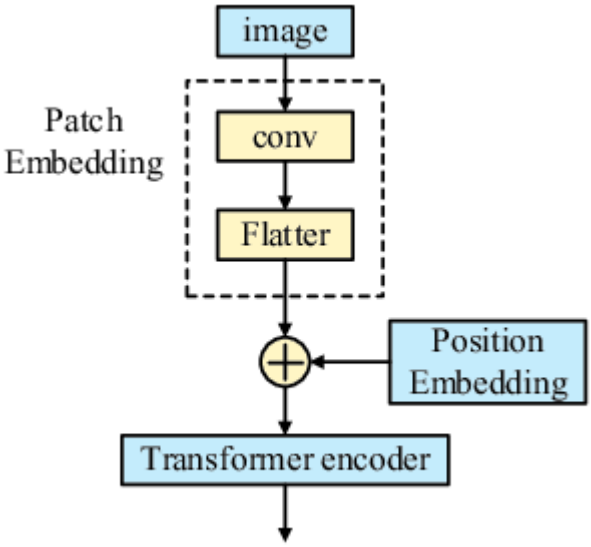

Figure 3.3: C3TR

hybrid encoder transforms multi-scale features into a sequence of image features through intra-scale interaction and cross-scale fusion . Subsequently, the IoU-aware query selection is employed to select a fixed number of image features from the encoder output sequence to serve as initial object queries for the decoder . Finally, the decoder with auxiliary prediction heads iteratively optimizes object queries to generate boxes and confidence scores.

Efficient Hybrid Encoder

Computational bottleneck analysis To accelerate training convergence and improve performance, Zhu et al. [24] suggest introducing multi-scale features and propose the deformable attention mechanism to reduce computation. However, although the improvement in the attention mechanism reduces the computational overhead, the sharply increased length of input sequence still causes the encoder to become a computational bottleneck, hampering the real time implementation of DETR. As reported in [26], the encoder accounts for 49% of the GFLOPs but contributes only 11% of the AP in Deformable-DETR [25]. Analyzed the computational redundancy present in the multi-scale transformer encoder and design a set of variants to prove that the simultaneous interaction of intra-scale and cross-scale features is computationally inefficient .
High-level features are extracted from low-level features that contain rich se-

mantic information about objects in the image. Intuitively, it is redundant to perform feature interaction on the concatenate multi-scale features. To verify this opinion, designed a range of variants with different encoders,.The set of variants gradually improves model accuracy while significantly reducing computational cost by decoupling multi-scale feature interaction into two-step operations of intra-scale interaction and cross-scale fusion . We first remove the multiscale transformer encoder in DINO-R50 [25] as baseline A.Next, different forms of encoder are inserted to produce a series of variants based on baseline A.

Hybrid design The encoder consists of two modules, the Attention-based Intrascale Feature Interaction (AIFI) module and the CNNbased Cross-scale Feature-fusion Module (CCFM). AIFI further reduces computational redundancy based on variant D, which only performs intra-scale interaction on S5. Applying the self-attention operation to high level features with richer semantic concepts can capture the connection between conceptual entities in the image, which facilitates the detection and recognition of objects in the image by subsequent modules. Meanwhile, the intra-scale interactions of lower-level features are unnecessary due to the lack of semantic concepts and the risk of the risk of duplication and confusion with interactions of high-level features. To verify this view, performed the intra-scale interaction on S5 in variant D, and the experimental results are reported in Tab. 3, see row DS5 . Compared to the vanilla variant D, DS5 significantly reduces the latency (35% faster) but delivers an improvement in accuracy (0.4% AP higher). This conclusion is crucial for the design of real-time detectors. CCFM is also optimized based on variant D, inserting several fusion blocks composed of convolutional layers into the fusion path

The role of the fusion block is to fuse the adjacent features into a new feature, and its structure . The fusion block contains N RepBlocks, and the two-path outputs are fused by element-wise add. Formulated this process as follows:

$$Q = K = V = Flatten(S5)$$

$$\text{F5} = \text{Reshape}(\text{Attn}(\text{Q, K, V}))$$

$$\text{Output} = \text{CCFM}(S_3, S_4, F_5) \quad (3.2)$$

where Attn represents the multi-head self-attention, and Reshape repre-

sents restoring the shape of the feature to the same as S5, which is the inverse operation of Flatten.

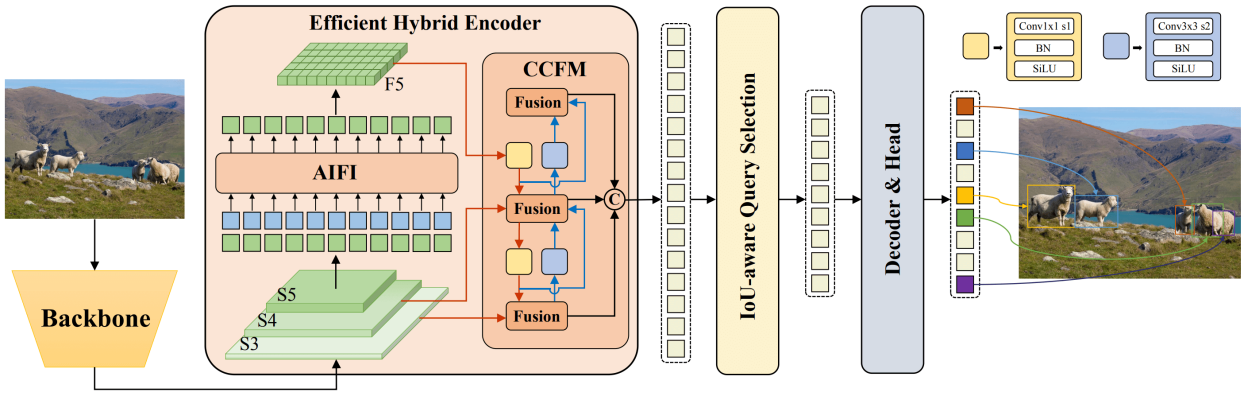

Figure 3.4: The RT-DETR model architecture diagram shows the last three stages of the backbone S3, S4, S5 as the input to the encoder. The efficient hybrid encoder transforms multiscale features into a sequence of image features through intrascale feature interaction (AIFI) and cross-scale feature-fusion module (CCFM). The IoU-aware query selection is employed to select a fixed number of image features to serve as initial object queries for the decoder. Finally, the decoder with auxiliary prediction heads iteratively optimizes object queries to generate boxes and confidence scores

3.4 IoU-aware Query Selection

The object queries in DETR are a set of learnable embeddings, which are optimized by the decoder and mapped to classification scores and bounding boxes by the prediction head. However, these object queries are difficult to interpret and optimize because they have no explicit physical meaning. Subsequent works [18, 7, 22, 20, 21] improve the initialization of object query and extend it to content query and position query (anchor). Among them, [22, 20, 21] all propose query selection schemes, which have in common that they utilize the classification score to select top K features from the encoder to initialize object queries (or only position queries [40]). However, due to the inconsistent distribution of classification score and location confidence, some predicted boxes have high classification scores but are not close to GT boxes, which results in boxes with high classification scores and low IoU scores being selected, while boxes with low classification scores and high IoU scores are discarded. This impairs the performance of the detector. To address this issue, we propose IoU-aware query selection by constraining the model to produce high classification scores for features with high IoU scores and low classification scores for features with low IoU scores during training. Therefore, the prediction boxes corresponding to the top K encoder features selected by the model according to the classification score have both high classification scores and high IoU scores.So the reformulated optimization objective of the detector as follows:

$$L(\hat{y}, y) = Lbox(\hat{b}, b) + Lcls(\hat{c}, \hat{b}, y, b) = Lbox(\hat{b}, b) + Lcls(\hat{c}, c, IoU) \tag{3.3}$$

where and y denote prediction and ground truth, $\hat{y}$ ={$\hat{c}$, $\hat{b}$} and y = {c, b}, c and b represent categories and bounding boxes, respectively. We introduce the IoU score into objective function of the classification branch (similar to VFL [41]) to realize the consistency constraint on the classification and localization of positive samples.

3.5 Experiments And Results

3.5.1 Experimental Settings

Trained transformer based yolov8 object detection model on the next wealth dataset. Model is trained for 100 epochs, using a batch size of 16 and an image size of 640x640 pixels.Saved the model weights to the runs/detect/ directory after each epoch. Used the SGD optimizer with a learning rate of 0.01, which will decrease linearly over time. Incorporated automatic mixed precision training and overlap masks during training. Validate the model on the val split of the dataset after each epoch. Saved the model predictions in JSON format or in hybrid format. And set the intersection over union (IoU) threshold to 0.7 and the maximum number of detections to 300. Useed half-precision training or the Deep Neural Network (DNN) backend. Ploted the training and validation losses and accuracies to a file.Model is trained on an RTX 3090 Ti GPU.

3.6 Datasets

Two object detection datasets for training: the COCO dataset and the Next Wealth dataset. The Next Wealth dataset is a proprietary dataset that consists of only 10 classes. It contains 22,997 images captured from different environmental conditions.

3.6.1 Coco dataset

COCO dataset is a popular dataset used for object detection. It contains over 330,000 images, each annotated with bounding boxes and segmentation masks for 80 different object categories. This makes it a valuable resource for training and evaluating object detection models.The images and their annotations are stored in separate folders, and the annotations are stored in a JSON file. The JSON file contains a list of objects for each image, along with the bounding boxes and segmentation masks for each object.

3.6.2 Next wealth dataset

The Next Wealth dataset is a proprietary dataset that contains 22,997 images and 10 classes: person, vehicle, train, car, motorcycle, bicycle, truck, other vehicle, bus, and auto rickshaw. The images were taken from different CCTV cameras and contain images from different conditions.

3.7 Evaluation metrics

3.7.1 Precision

Precision in object detection is the fraction of predicted bounding boxes that actually contain an object. It is calculated as follows:

$$P = \frac{TP}{TP + FP} \tag{3.4}$$

where:

TP is the number of true positives, which are predicted bounding boxes that contain an object and also match a ground truth bounding box. FP is the number of false positives, which are predicted bounding boxes that do not contain an object but match a ground truth bounding box.

Precision is important because it measures how accurate the model's predictions are. A high precision means that the model is good at identifying objects that are actually present in the image. However, a high precision can also mean that the model is not detecting all of the objects in the image.

3.7.2 Recall

Recall is another important metric for object detection. Recall measures the fraction of ground truth objects that are detected by the model. It is calculated as follows:

$$R = \frac{TP}{TP + FN} \tag{3.5}$$

where:

FN is the number of false negatives, which are ground truth bounding boxes that are not detected by the model.

Recall is important because it measures how complete the model's predictions are. A high recall means that the model is good at detecting all of the objects in the image. However, a high recall can also mean that the model is generating a lot of false positives.

The ideal object detection model would have a high precision and a high recall. However, in practice, it is often necessary to trade off between precision and recall. For example, a model that is designed to detect as many objects as possible may have a lower precision than a model that is designed to avoid false positives.

The choice of whether to prioritize precision or recall depends on the specific application. For example, if the goal is to detect as many objects as possible, then recall may be more important than precision. However, if the goal is to avoid false positives, then precision may be more important than recall.

3.7.3 Mean Average Precision

Average Precision (AP) is the average accuracy of the model.Mean average precision (mAP) is a metric for evaluating object detection models performance. It is one of the key evaluation

metrics alongside other measurements like precision and recall that gives a comprehensive description of how well the model detects various objects.

$$AP = \int_0^1 p(r)dr \tag{3.6}$$

$$mAP = \frac{1}{k}\sum_{i=1}^{k} AP_i \tag{3.7}$$

3.8 Evaluation of Model Performance

The provided data showcases the performance metrics of two models: YOLOv8n-trained and our model. These models were evaluated on various classes of objects in a dataset consisting of 2553 images.
In terms of overall performance, our model achieved an impressive mean Average Precision (mAP) of 0.805, which indicates its ability to accurately detect and classify objects in the given dataset. The YOLOv8n-trained model also demonstrated strong performance with an mAP of 0.792. When focusing

Notably, our model demonstrated exceptional performance in detecting "trains" with a precision of 1.0 and an mAP of 0.995. In contrast, the YOLOv8n-trained model achieved a slightly lower precision of 0.962 and an mAP of 0.804. This indicates that our model is highly proficient in identifying trains within the given dataset. Furthermore, our model outperformed the YOLOv8n-trained model in detecting "cars," "motorcycles," "buses," and "auto-rickshaws" in terms of precision, recall, and mAP scores. These results highlight the superior object detection capabilities of our model across multiple vehicle categories. Although our model generally outperformed the YOLOv8n-trained model, it is worth mentioning that the YOLOv8n-trained model achieved higher recall scores for the "person" and "bus" classes. This suggests that the YOLOv8n-trained model might be better at capturing a higher proportion of instances within these specific classes, although our model achieved higher precision scores overall. In summary, our model demonstrates excellent performance in object detection and classification tasks, surpassing the YOLOv8n-trained model in various categories. The results indicate its effectiveness in accurately identifying and localizing objects, particularly individuals and vehicles.

Table 3.1: YOLOv8n-trained vs. Our Model

			Trained Model				Our Model			
Class	Images	Instances	Box(P)	R	mAP 50	mAP 50-95	Box(P)	R	mAP 50	mAP 50-95
all	2553	40810	0.878	0.711	0.792	0.581	0.844	0.772	0.805	0.583
person	2553	12294	0.885	0.627	0.74	0.43	0.803	0.773	0.86	0.54
vehicle	2553	1134	0.761	0.617	0.709	0.378	0.646	0.725	0.715	0.447
train	2553	17	0.962	1	0.804	0.87	0.955	1	0.995	0.995
cars	2553	11108	0.935	0.871	0.829	0.704	0.907	0.992	0.947	0.735
motorcycle	2553	9062	0.95	0.806	0.888	0.623	0.881	0.928	0.938	0.638
bicycle	2553	21	0.855	0.238	0.266	0.125	0.999	0.143	0.413	0.177
truck	2553	5409	0.902	0.877	0.824	0.628	0.896	0.933	0.95	0.719
other vehicle	2553	187	0.726	0.511	0.618	0.437	0.7	0.567	0.648	0.486
bus	2553	376	0.888	0.778	0.841	0.649	0.786	0.875	0.894	0.677
auto-rickshaw	2553	1202	0.916	0.782	0.808	0.611	0.869	0.854	0.899	0.663

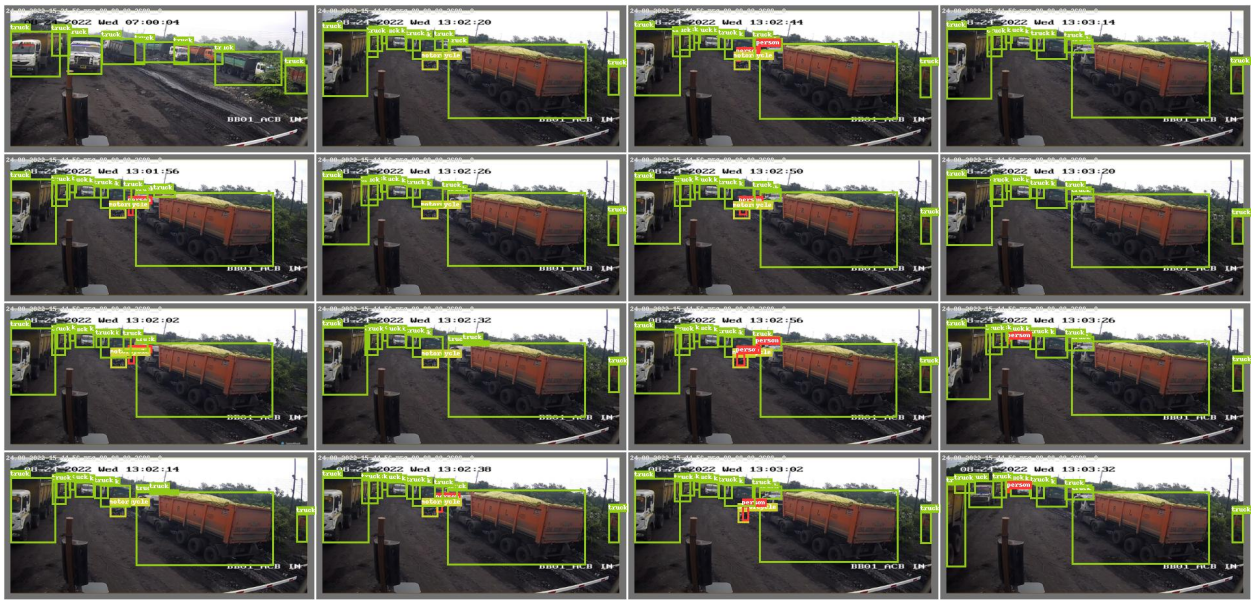

Figure 3.5: Validation batch 0 labels

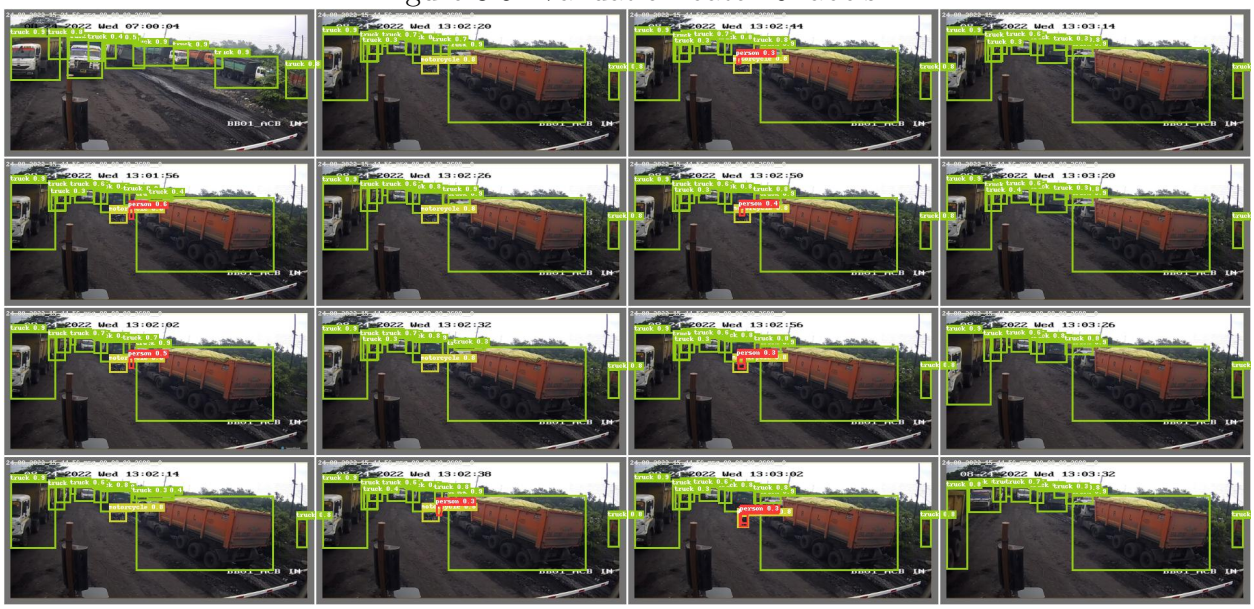

Figure 3.6: Validation batch 0 prediction

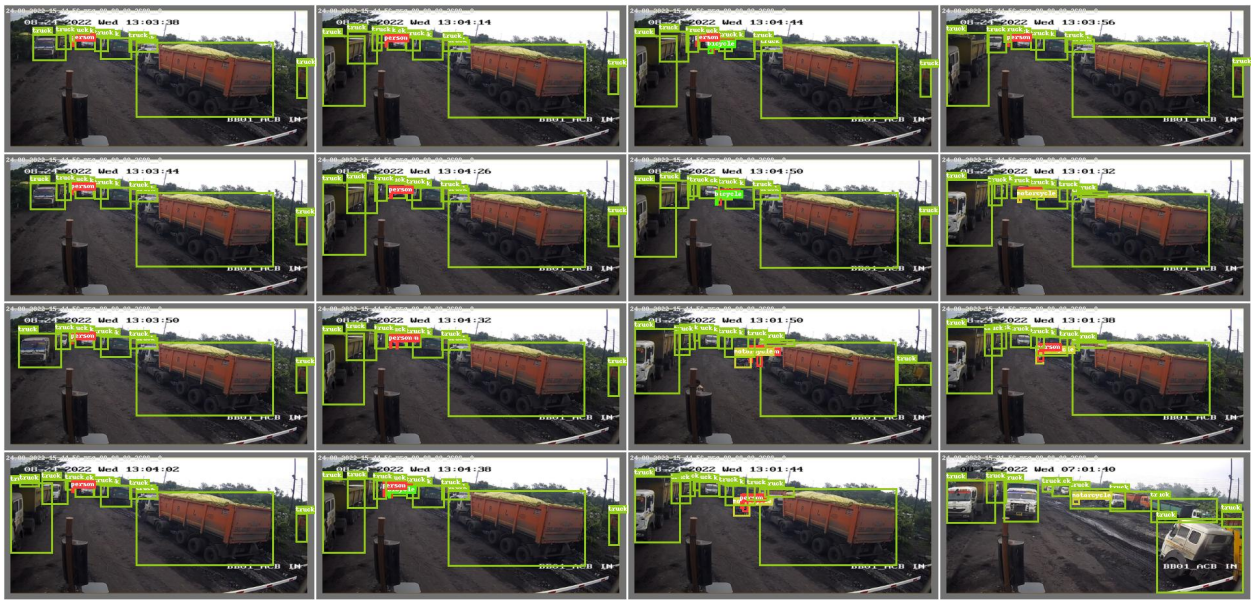

Figure 3.7: Validation batch 1 labels

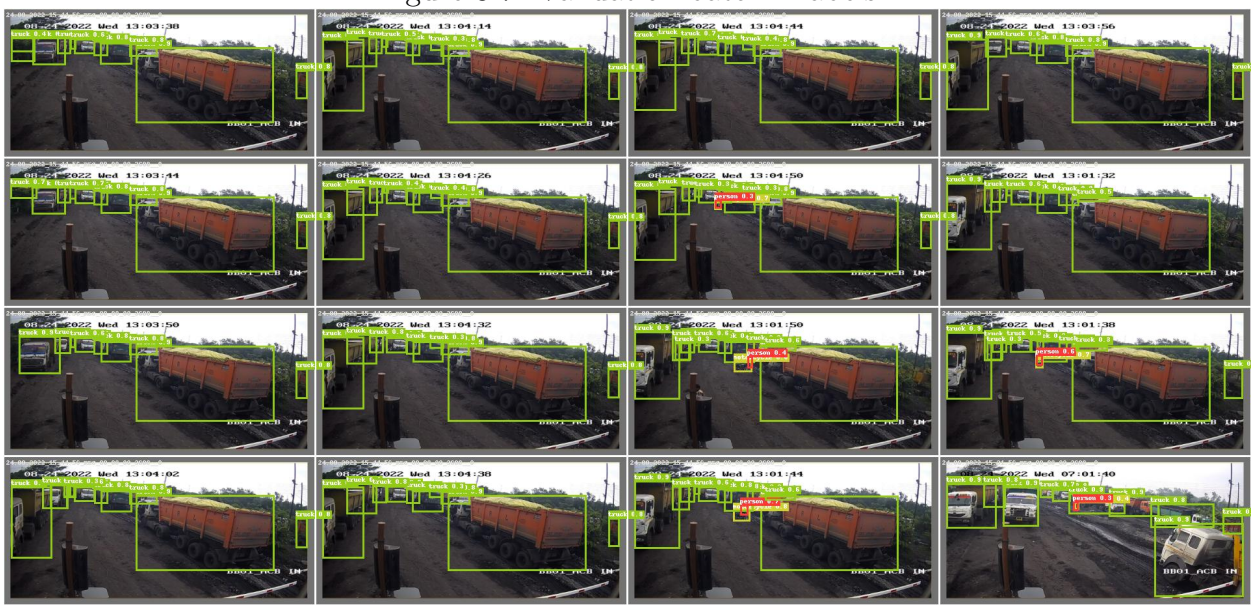

Figure 3.8: Validation batch 1 prediction

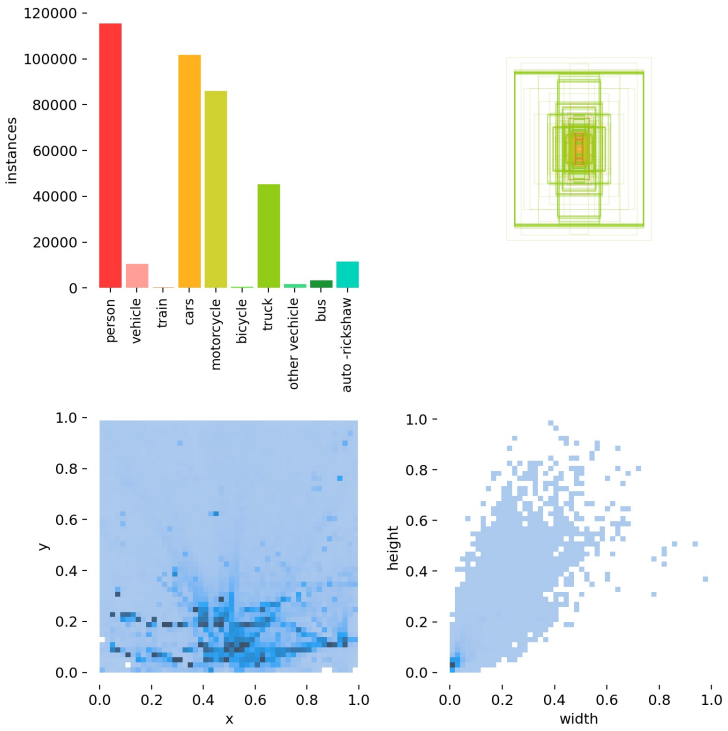

Figure 3.9: Labels

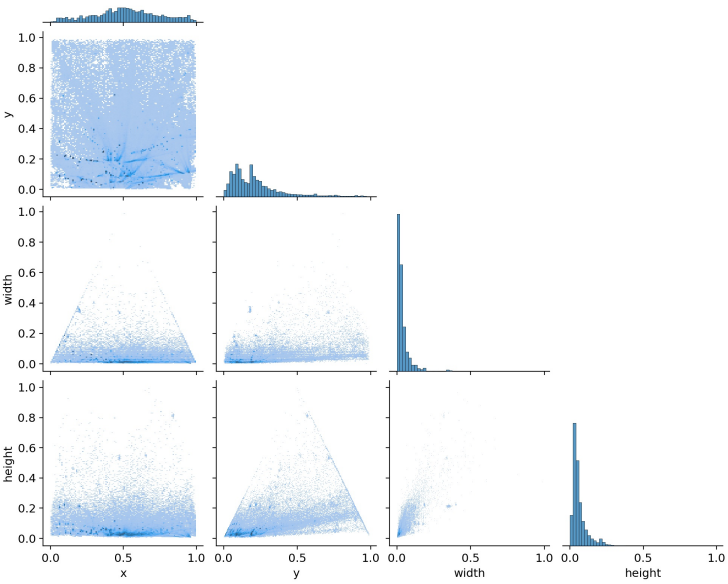

Figure 3.10: Labels Correlation

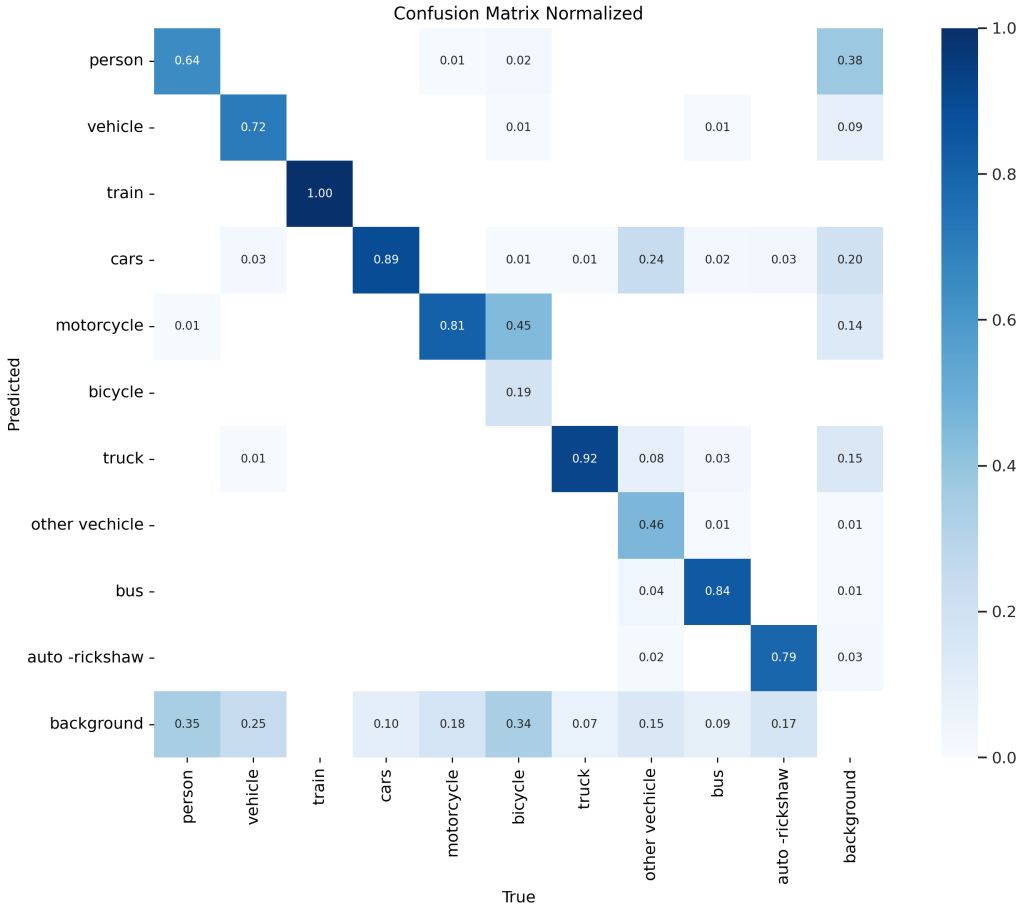

Figure 3.11: Confusion Matrix

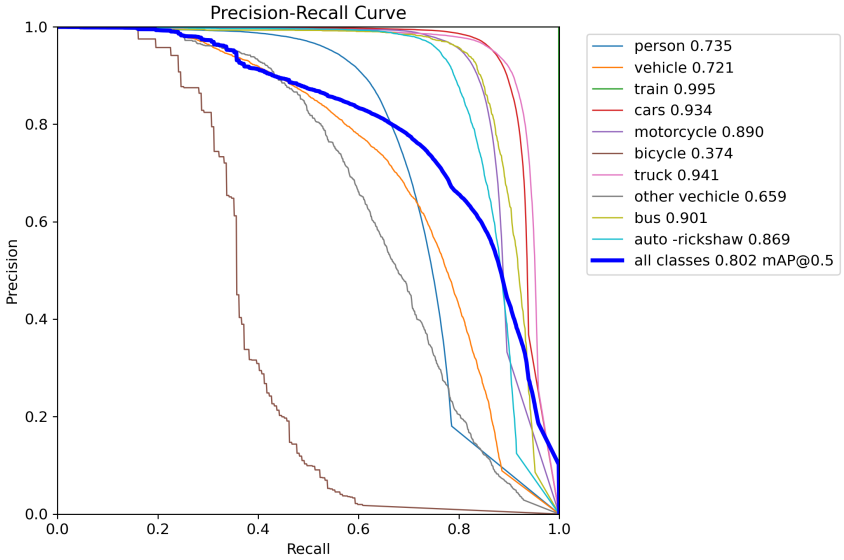

Figure 3.12: PR Curve

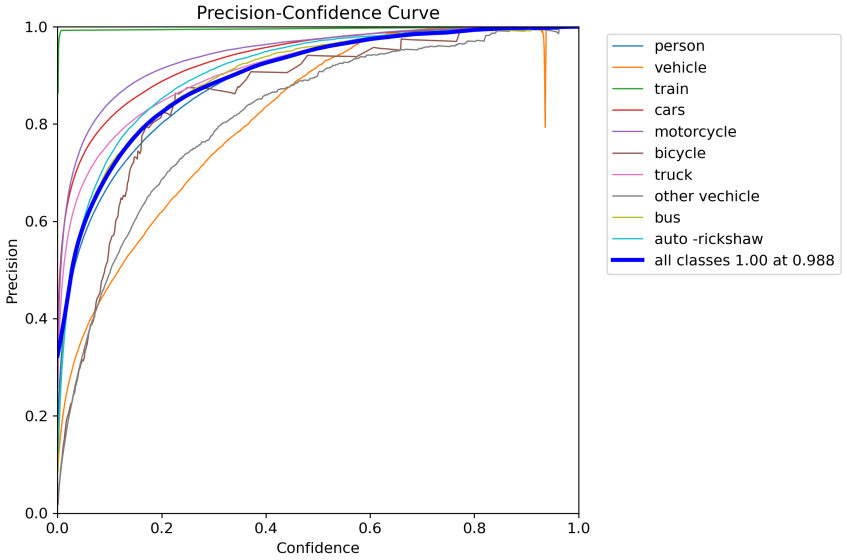

Figure 3.13: P Curve

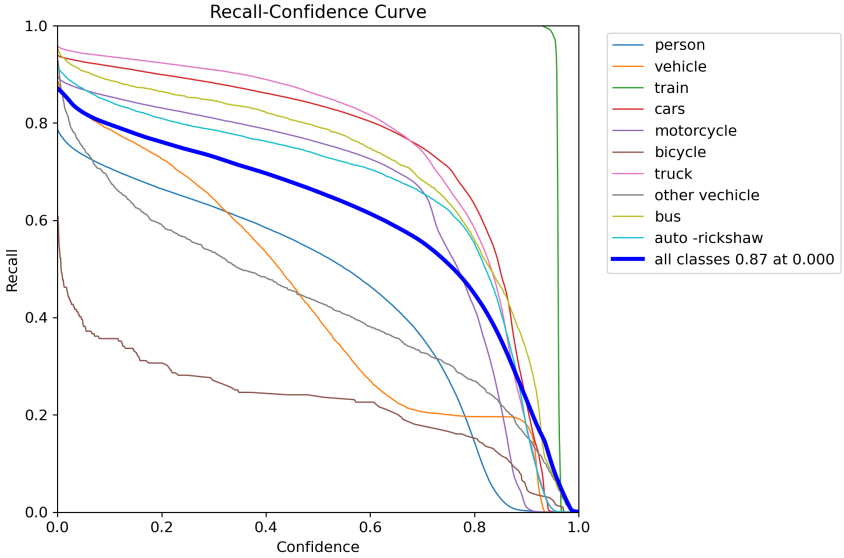

Figure 3.14: R Curve

Chapter 4

Conclusion

YOLOv8 is a state-of-the-art object detection algorithm that was first released in 2021. It is based on the YOLOv3 algorithm, but it has been significantly improved in a number of ways. One of the most significant improvements to YOLOv8 is the replacement of the detection head with RTDetR. RTDetR is a more recent and advanced object detection algorithm than the one used in YOLOv3. It is based on the RetinaNet algorithm, which is known for its high accuracy. RTDetR also uses a technique called focal loss, which helps to improve the accuracy of object detection at low object confidences.
Another significant improvement to YOLOv8 is the replacement of some of the C2F blocks with C3TR blocks. C3TR is a block that was introduced in the YOLOv5 family of object detectors. It is a more efficient block than the C2F block, which was used in YOLOv3. This is because C3TR uses a different convolution kernel size, which allows it to extract more features from the input image.
The combination of these two changes resulted in a significant improvement in the performance of YOLOv8. The MAP (mean average precision) score was increased , which means that YOLOv8 is now able to detect objects more accurately.
In addition to the improvements mentioned above, there are a number of other potential improvements that could be made to YOLOv8. These include using a larger backbone network, such as ResNet50 or ResNet101 or by using different loss function, such as smooth L1 loss or Huber loss or different data augmentation strategy.
RTDetR is a more recent and advanced object detection algorithm than the one used in YOLOv3. It is based on the RetinaNet algorithm, which is known for its high accuracy. RTDetR also uses a technique called focal loss, which helps to improve the accuracy of object detection at low object confidences. Focal loss is a type of loss function that is designed to give more weight to difficult examples. This means that the model will be more likely to learn to detect objects that are difficult to see.
C3TR is a block that was introduced in the YOLOv5 family of object detectors. It is a more efficient block than the C2F block, which was used in YOLOv3. This is because C3TR uses a different convolution kernel size, which allows it to extract more features from the input image. The convolution kernel size is the size of the filter that is used to convolve the input image. A larger convolution kernel size will extract more features from the input image, but it will also be more computationally expensive.
The improvements made to YOLOv8 have made it a more powerful object detection algorithm. It is now able to detect objects more accurately and efficiently than previous versions of the algorithm. This makes it a valuable tool for a variety of applications, such as self-driving cars, video surveillance, and medical imaging.

REFERENCES

[1] Alexey Bochkovskiy, Chien-Yao Wang, and HongYuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934, 2020. 1, 2

[2] Jocher Glenn. Yolov5 release v7.0. https://github. com/ultralytics/yolov5/tree/v7.0, 2022. 1, 2, 3, 4, 7, 8

[3] Jocher Glenn. Yolov8. https : / / github . com / ultralytics/ultralytics/tree/main, 2023. 1, 2, 3, 4, 7, 8

[4] YOLOv1: Joseph Redmon, Santosh Divvala, Ross Girshick, Alejandro Farhadi. "You Only Look Once: Unified, Real-Time Object Detection." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016. Paper: https://arxiv.org/abs/1506.02640.

[5] YOLOv2: Joseph Redmon, Ali Farhadi. "YOLO9000: Better, Faster, Stronger." arXiv preprint arXiv:1612.08242 (2016). Paper: https://arxiv.org/abs/1612.08242

[6] YOLOv3: Joseph Redmon, Ali Farhadi. "YOLOv3: An Incremental Improvement." arXiv preprint arXiv:1804.02976 (2018). Paper: https://arxiv.org/abs/1804.02976.

[7] Shilong Liu, Feng Li, Hao Zhang, Xiao Yang, Xianbiao Qi, Hang Su, Jun Zhu, and Lei Zhang. Dab-detr: Dynamic anchor boxes are better queries for detr. arXiv preprint arXiv:2201.12329, 2022. 1, 2, 3, 6, 7.

[8] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time objectdetectors. 2022, pp. 1–15.

[9] W. Liu et al., SSD: Single shot multibox detector. 2016, Computer Vision–ECCV, vol. 9905 LNCS, pp. 21–37

[10] D. Bolya, C. Zhou, F. Xiao, and Y. Lee Jae. Yolact: Real-time Instance Segmentation. 2019, Proceedings of the IEEE/CVFinternational conference on computer vision, pp. 9157–9166.

[11] Y. Cao, K. Chen, C. C. Loy, and D. Lin. Prime Sample Attention in Object Detection. 2020, Proceedings of the IEEE/CVFconference on computer vision and pattern recognition, pp. 11583-11591

[12] H. Liu, X. Duan, H. Chen, H. Lou, and L. Deng. DBF-YOLO:UAV Small Targets Detection Based on Shallow Feature Fusion.2023,IEEJ Transactions on Electrical and Electronic Engineering, doi: 10.1002/tee.23758

[13] H. Liu, F. Sun, J. Gu, and L. Deng. SF-YOLOv5: A Lightweight Small Object Detection Algorithm Based on Improved FeatureFusion Mode. 2022, Sensors, vol. 22, no. 15, pp. 1–14.

[14] C. Y. Wang, H. Y. Mark Liao, Y. H. Wu, P. Y. Chen, J. W. Hsieh, and I. H. Yeh. CSP-Net: A new backbone that can enhance learningcapability of CNN. 2020, Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, vol.2020-June, pp. 1571–1580.

[15] T. Lin, R. Girshick, K. He, B. Hariharan, S. Belongie. Feature Pyramid Networks for Object Detection. 2017, Proceedings of theIEEE conference on computer vision and pattern recognition, pp. 2117-2125.

[16] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia. Path Aggregation Network for Instance Segmentation. 2018, Proceedings of the IEEEconference on computer vision and pattern recognition, pp. 8759–8768.

[17] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun. YOLOX: Exceeding YOLO Series in 2021. 2021, pp. 1–7.

[18] Yingming Wang, Xiangyu Zhang, Tong Yang, and Jian Sun. Anchor detr: Query design for transformer-based detector. In Proceedings of the AAAI conference on artificial intelligence, volume 36, pages 2567–2575, 2022. 1, 2, 3, 6, 7.

[19] Shangliang Xu, Xinxin Wang, Wenyu Lv, Qinyao Chang, Cheng Cui, Kaipeng Deng, Guanzhong Wang, Qingqing Dang, Shengyu Wei, Yuning Du, et al. Pp-yoloe: An evolved version of yolo. arXiv preprint arXiv:2203.16250, 2022. 1, 2, 4, 7, 8.

[20] Zhuyu Yao, Jiangbo Ai, Boxun Li, and Chi Zhang. Efficient detr: improving end-to-end object detector with dense prior. arXiv preprint arXiv:2104.01318, 2021. 3, 6, 7.

[21] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. In The Eleventh International Conference on Learning Representations, 2022. 1, 2, 3, 5, 6, 7, 8.

[22] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. arXiv preprint arXiv:2010.04159, 2020. 1, 2, 3, 5, 6, 7.

[23] C. Li, C. Xu, C. Gui, and M. D. Fox, "Level set evolution without reinitialization: A new variational formulation," in IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, 2005, pp. 430–436.

[24] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. arXiv preprint arXiv:2010.04159, 2020. 1, 2, 3, 5, 6, 7

[25] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end

object detection. In The Eleventh International Conference on Learning Representations, 2022. 1, 2, 3, 5, 6, 7, 8.

[26] Junyu Lin, Xiaofeng Mao, Yuefeng Chen, Lei Xu, Yuan He, and Hui Xue. Dˆ 2etr: Decoder-only detr with computationally efficient cross-scale attention. arXiv preprint arXiv:2203.00860, 2022. 5