A PROPOSED METHOD FOR INCREASING THE ACCURACY OF OBJECT DETECTION UNDER RAIN CONDITIONS USING DEEP LEARNING

2022
MASTER THESIS
COMPUTER ENGINEERING

Faris Kareem HALYUT

Thesis Advisor
Assist. Prof. Dr. Adnan Saher MOHAMMED

A PROPOSED METHOD FOR INCREASING THE ACCURACY OF OBJECT DETECTION UNDER RAIN CONDITIONS USING DEEP LEARNING

Faris Kareem HALYUT

T.C.
Karabuk University
Institute of Graduate Programs
Department of Computer Engineering
Prepared as
Master Thesis

Assist. Prof. Dr. Adnan Saher MOHAMMED

KARABUK
June 2022

I certify that in my opinion the thesis submitted by Faris Kareem HALYUT titled "A PROPOSED METHOD FOR INCREASING THE ACCURACY OF OBJECT DETECTION UNDER RAIN CONDITIONS USING DEEP LEARNING " is fully adequate in scope and in quality as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Adnan Saher MOHAMMED ....
Thesis Advisor, Department of Computer Engineering

This thesis is accepted by the examining committee with a unanimous vote in the Department of Computer Engineering as a Master of Science thesis. June 21, 2022

Examining Committee Members (Institutions)		Signature
Chairman	: Prof.Dr. Fatih Vehbi ÇELEBİ (AYBU)	....
Member	: Assist.Prof.Dr. Adnan Saher MOHAMMED (KBU)	....
Member	: Assist. Prof. Dr. Caner ÖZCAN (KBU)	....

The degree of Master of Science by the thesis submitted is approved by the Administrative Board of the Institute of Graduate Programs, Karabuk University.

Prof. Dr. Hasan SOLMAZ ....
Director of the Institute of Graduate Programs

“I declare that all the information within this thesis has been gathered and presented in accordance with academic regulations and ethical principles and I have according to the requirements of these regulations and principles cited all those which do not originate in this work as well.”

Faris Kareem HALYUT

ABSTRACT

M. Sc. Thesis

A PROPOSED METHOD FOR INCREASING THE ACCURACY OF OBJECT DETECTION UNDER RAIN CONDITIONS USING DEEP LEARNING

Faris Kareem HALYUT

Karabuk University
Institute of Graduate Programs
The Department of Computer Engineering

Thesis Advisor:
Assist. Prof. Dr. Adnan Saher MOHAMMED
June 2022, 75 pages

Detecting objects utilizing deep learning is one of the most essential deep learning and computer vision applications where one can learn image features in normal weather conditions and different rain conditions. Therefore, a deep convolutional neural network (CNN) has become more important for object detection. Rain is a common and major factor in degrading image quality and decreasing object detection accuracy. The main aim of this work is to remove rain streaks and improve the quality of image/video for object detection with high accuracy, increase the reliability of the algorithm and decrease the error rate in the object detection process, in normal conditions and different rain conditions (light, medium and heavy), compared to the methods used in previous work mentioned in the research.

Firstly, the quality of the images is improved and removed rain streaks by a de-raining algorithm that uses the Deep Detail Network (DDN) method. Then the way deep learning is the main object detector, through using the YOLO to detect objects identify their type. YOLOv3 and tiny-YOLOv3 have been determined from the literature review as the most suitable and efficient algorithms for object detection in real-time after improving the image quality. The performance of these algorithms has been calculated and compared with each other in terms of accuracy. To evaluate the effectiveness of the devised approach (DDN+YOLOv3), the Dice Coefficient (F1-score), Recall, Precision and accuracy were computed. Using the proposed model combined of Deep Detail Network (DDN) to de-rain with YOLOv3 technique, the mean of F1-score of 95.02%, Recall of 97.22%, a Precision of 92.92% and accuracy of 90.51% were attained. Our presented approach is more flexible and accurate to detect objects under different rain conditions according to the results of the experiments that we obtained. And it is considered the best way to obtain high accuracy for detecting objects, compared to machine learning methods that use the same proposed idea.

Keywords: Deep learning, Object detection, Rain streaks, YOLO, Deep Detail Network.
Science Code : 92431

ÖZET

Yüksek Lisans Tezi

DERİN ÖĞRENMEYLE YAĞMUR DURUMLARINDA NESNE TESPİTİN DOĞRLIĞINI ARTIRMAK İÇİN ÖNERİLEN BİR YÖNTEM

Faris Kareem HALYUT

Karabük Üniversitesi
Lisansüstü Eğitim Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Tez Danışmanı:
Dr. Öğr.Üyesi. Adnan Saher MOHAMMED
Haziran 2022,75 sayfa

Derin öğrenme kullanılarak nesnelerin algılanması, normal hava koşullarında ve farklı yağmur koşullarında görüntü özelliklerinin öğrenilebileceği en temel derin öğrenme ve bilgisayarla görü uygulamalarından biridir. Bu nedenle, nesne tespiti için derin bir evrişimli sinir ağı (CNN) daha önemli hale geldi. Yağmur, Görüntü kalitesini ve nesne algılama doğruluğunu azaltmada yaygın ve önemli bir faktördür. Bu çalışmanın temel amacı, normal ve farklı yağmur koşullarında, nesne tespitinde yağmur çizgilerini ortadan kaldırmak ve görüntü/video kalitesini yüksek doğrulukla iyileştirmek, algoritmanın güvenilirliğini artırmak ve nesne algılama sürecindeki hata oranını azaltmaktır (hafif, orta, ağır) aramada daha önce bahsedilen eserlerde kullanılan yöntemlerle karşılaştırıldığında.

İlk olarak, Derin Ayrıntı Ağı (DDN) yöntemini kullanan bir yağmur azaltma algoritması ile görüntülerin kalitesi iyileştirilir ve yağmur izleri kaldırılır. Daha sonra, derin öğrenmenin yolu, nesnelerin türlerini tanımlamak için YOLO'yu kullanarak ana nesne dedektörüdür. YOLOv3 ve tiny-YOLOv3, görüntü kalitesini iyileştirdikten sonra gerçek zamanlı olarak nesne tespiti için en uygun ve verimli algoritmalar olarak literatür taramasından belirlenmiştir. Bu algoritmaların performansları hesaplanmış ve doğruluk açısından birbirleriyle karşılaştırılmıştır. Tasarlanan yaklaşımın (DDN+YOLOv3) etkinliğini değerlendirmek için Zar Katsayısı (F1-skor), Geri Çağırma, Kesinlik ve doğruluk hesaplandı. YOLOv3 tekniği ile yağmurdan arındırmak için Derin Ayrıntı Ağı (DDN) ile birleştirilmiş önerilen model kullanılarak, F1 skorunun ortalaması %95,02, Geri Çağırma %97,22, %92,92 Hassasiyet ve %90,51 doğruluk elde edildi. Sunulan yaklaşımımız, elde ettiğimiz deneylerin sonuçlarına göre farklı yağmur koşullarında nesneleri algılamak için daha esnek ve doğrudur. Aynı önerilen fikri kullanan makine öğrenme yöntemlerine kıyasla, nesneleri algılamak için yüksek doğruluk elde etmenin en iyi yolu olarak kabul edilir.

Anahtar Kelimeler: Derin öğrenme, Nesne algılama, Yağmur çizgileri, YOLO, Derin Ayrıntı Ağı, Görüntü İşleme.
Bilim Kodu : 92431

ACKNOWLEDGMENT

I thank God Almighty, who inspired me strength and patience first, and I proudly bow to beloved parents, my dear father, who gave everything to make me a man of honor, and my dear mother who stood with me with her prayers and blessings, my dear brothers, and my little sister. Thank you very much for all. My thanks and gratitude to my supervisor, assistant professor, Dr. Adnan Saher Mohammed, who spared no effort in providing advice, and I wish him good health and success. I also express my thanks and gratitude to Karabuk University and all the wonderful professors and colleagues, thank you to all my good friends. I dedicate this thesis to my beloved country, great Iraq, and to my second country, the great Turkey, that embraced science students. And I did not feel the alienation of the homeland.

CONTENTS

Page

Page

LIST OF FIGURES

Page

LIST OF TABLES

<u>Page</u>

ABBREVIATIONS

CNNs	:	Convolutional Neural Networks
SVM	:	Support Vector Machine
YOLO	:	You Only Look Once
R-CNNs	:	Regional Convolutional Neural Networks
DCNNs	:	Deep Convolutional Neural Networks
VGG16	:	Visual Geometry Group
LiDAR	:	Light Detection and Ranging
COCO	:	Common Objects in Context
SPP-Net	:	Spatial Pyramid Pooling
Res-Net	:	Residual Neural Network
SSD	:	Single Shot Multi-Box Detector
HOG	:	Histogram of Oriented Gradients
SIFT	:	Scale-Invariant Feature Transform
MCG	:	Multiscale Combinatorial Grouping
NMS	:	Non-maximum Suppression
RGB	:	Red Green Blue
RPA	:	Region Proposal Arranges
ROI	:	Region of Interest
Conv	:	Convolutional
FC	:	Fully-Connected Layer
SS	:	Selection Search
FPN	:	Feature Pyramid Network
GANs	:	Generative Adversarial Networks
SAR	:	Synthetic Aperture Radar
IOU	:	Intersection Over Union
PSNR	:	Peak Signal to Noise Ratio
SSIM	:	Structural Similarity Method
DDN	:	Deep Detail Network

PART 1

INTRODUCTION

Deep learning is a subset of machine learning that mimics brain functions and is used to real-world challenges in conventional learning. Object detection and recognition are important techniques in computer vision and deep learning techniques that deal with the detecting states of observable objects in video and images of a certain class (such as humans, animals, automobiles, and so on). The goal of object detection is to determine the type of object, its position, and its categorization. From examples of important applications that use the process of object detection are autonomous cars, it can categorize and recognize things in real-time thanks to the computer vision method. An autonomous vehicle can sense and react to its surroundings to navigate without involving a person [1][2]. Detecting objects and recognizing them is essential to aid the vehicle to detect the impediments and determine their future route [2]. So, detecting object techniques are required.

Even though several machine learning and Convolutional Neural Networks (CNNs), Support Vector Machines (SVMs), the You Only Look Once (YOLO) model, and Regional Convolutional Neural Networks are examples of deep learning algorithms (R-CNNs), must be chosen carefully for object detection applications example autonomous driving because real-time object detection and recognition are required. Because machines cannot recognize things in images as quickly as humans can, it is critical that the algorithms be fast and precise, and that the objects be detected in real-time, so that vehicle and other applications controllers can solve optimization issues at a rate of at least one per second [3].

Object location is shown in Figure 1.1. It is a small-scale object representation of real-world street/intersection locations. Objects in the conditions of rain are completely different from the pure environment in the city, and the image has varying features depending on the weather, the surrounding conditions, and the items' sizes in reference to the other objects in the image. Detecting objects under the rain is difficult to distinguish objects, as the problem of lack of detection can be addressed in addition to performing tasks for long periods with a minimum of errors.

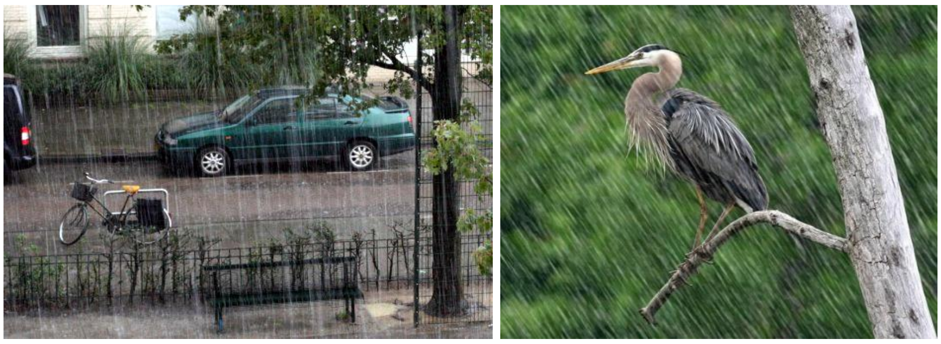


Figure 1.1. Objects under rain [4].

Video surveillance is becoming a widely used technology in the smart city paradigm to help improve the quality of human life in the digital age, where pedestrian detection is one of the key components for people-centered smart city applications such as wellbeing, security, traffic guiding, and drones, among others. Though lesser resolution can save costs and processing time, there is little information on how resolution affects detection accuracy [5], Figure 1.2 shows the images with rain which needs to boost object detection quality and accuracy.

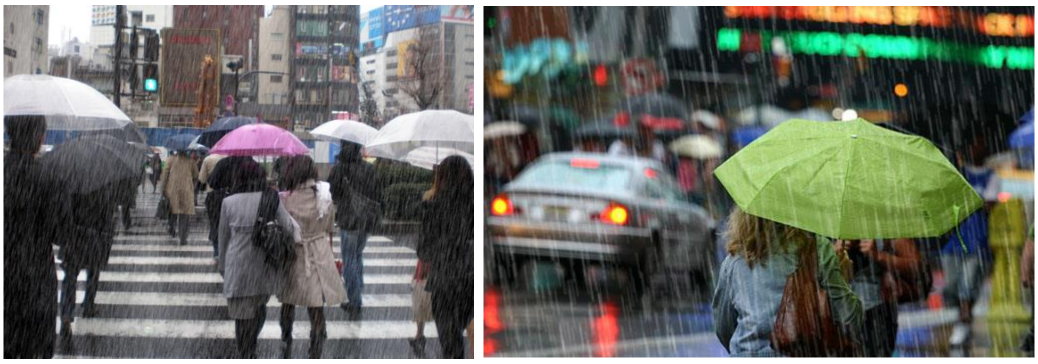


Figure 1.2. Images with noise (rain) [4].

Deep convolutional neural networks have become more popular as deep learning techniques have progressed are crucial for detecting objects. Deep learning-based object recognition algorithms could deal with low-level and high-level picture characteristics, as different from the classic handcrafted feature-based methods. These algorithms learn picture characteristics that are more representative than handmade features. As a result, the focus of this review is on object recognition methods according to the deep convolutional neural networks, whereas the classic object detection techniques are also briefly discussed. Object identification based on deep learning was examined and analyzed.

Object detection is the process of identifying and categorizing entities using rectangular bounding boxes. It aims at finding the detected objects followed by categorizing them. Object detection and object categorization, in addition to semantic and instance segmentation, have certain connections. Figure 1.3 illustrates the details. The first is object detection [6] which is then followed by semantic segmentation [6][7].

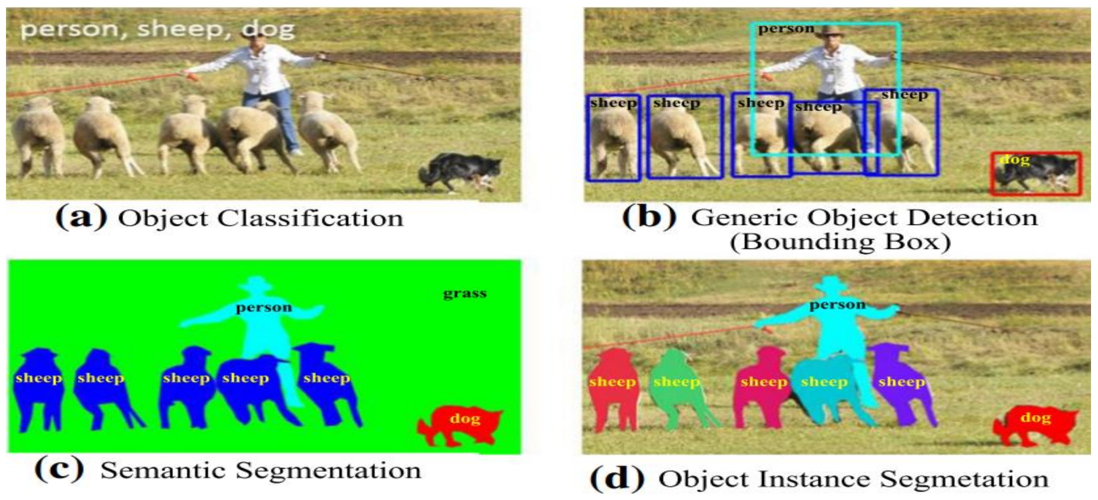


Figure 1.3. Object recognition [8].

In addition, it explains the pedestrian detection [9][1] with logo detection [10] followed by the video detection [11][12]. Furthermore, object detection [13][14], and medical image detection [15] are also explained. All these are examples of important applications of object detection in scientific research and practical industrial production. The research and use of deep neural networks has been impeded in recent decades due to a lack of computing resources, datasets, and basic ideas [16].

Traditional object identification algorithms have a basic design that is separated into three parts. The first is a region selector. The second part can be a feature extractor, and the last is represented by a classifier (Figure 1.4 and Figure 1.5). Traditional object detection, despite its maturity, has flaws. The sliding window [17] depends on the region selection approach, for starters with a high processing complexity and bigger window redundancy. Then, because of the morphological variation of appearances and the diversity of the backdrop, manually developing strong characteristics is difficult. Building ensemble systems and using modest modifications of current approaches yielded very tiny advances between 2010 and 2012 [7]. Learning is possible with deep convolutional neural networks visual characteristics at all levels, from low to high [18] [19]. As a result, the researchers progressively shifted their focus to the DCNNs.

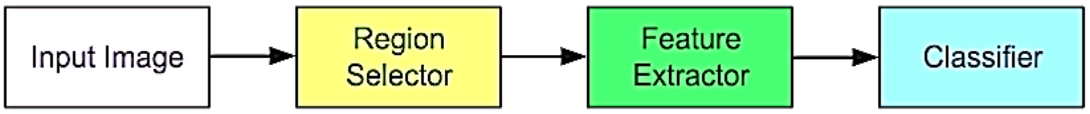


Figure 1.4. Traditional object detection techniques' fundamental design [20].

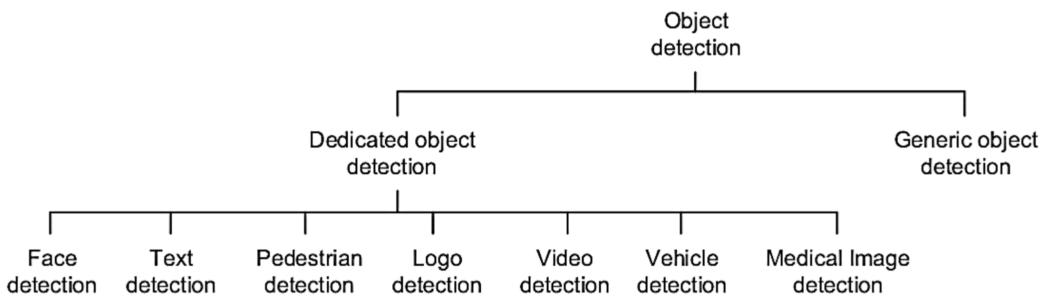


Figure 1.5. Object detection applications can be classified as general or specialized [20].

Object detection has been on the rise in recent years. First, detection accuracy is continually enhanced to meet the demands of a wide range of difficult circumstances. Second, the detection speed has been increased to meet the needs of real-time system applications maintaining accuracy. As a result, future studies should examine this trade-off [21][22] which is critical in order to get cutting-edge outcomes.

1.1. PROBLEM STATEMENT

Different weather conditions, like as rain, have a significant impact on the accurate identification and detection of objects, as well as the determination of their position, resulting in image distortion and the inability to identify objects or small objects. Although the approach of deep learning techniques to object detection is the same for some many applications such as construction vehicles, the drones, and self-driving cars, when it comes to autonomous movements, there exist unique challenges for the detect objects in terms of conditions, it requires high detection accuracy. As a result, more research is needed to improve the quality of images and video and to select the best deep learning model to detect existing state-of-the-art deep learning models and track objects in different environments, as very little research has been done in this field to far.

1.2. OBJECTIVES OF THESIS

The primary objectives of this thesis are as follows:

- Determining of appropriate and high-efficiency deep learning models (Res-Net) for de-raining and (YOLOv3 and YOLOv3-tiny) for object detection and recognition in real time.
- Improving the quality of the image or video taken from the environment by removing the noise using deep detail network (DDN) in different rain conditions to detect the object with high accuracy.
- Evaluate the performance of candidate deep learning models in object detection.
- Comparison and display the performance of selected deep learning models for de-raining and object detection.

1.3. CONTRIBUTION

The following are the research's contributions:

- Implementation of deep learning algorithms to detect objects. The model was trained and evaluated using three from a different set of datasets used to detect objects using the 12-layer YOLO model and its predictive speed, which can predict up to 45 frames per second.
- Dealing with an image in rain conditions difficult the identification of the type of objects when conducting the detection process, most previous research works to improve the detection of an object only without removing the rain streaks.
- In rainy conditions, the process of removing rain streaks is carried out by training de-raining algorithm and then object detection by YOLO to obtain high accuracy in the detection process. This means combining two methods to obtain one result, which is the higher accuracy of the detection process under the rain.
- When conducting the process of removing rain streaks only by Deep Detail Network (DDN)[4], it gets high image quality compared to other methods used.
- The use of this modern technology in many applications of computer vision to detect traffic signals and objects to be detected after training the algorithm in normal conditions and rain conditions.

1.4. ORGANIZATION OF THESIS

This thesis discusses cutting-edge neural networks for real-time object recognition in clear weather and rainy conditions, and it analyzes their performance against a collection of data. This thesis has six chapters, and the rest of the thesis is arranged as follows:

Chapter 2. A literature review focuses on methods used to de-raining and detector objects from neural networks.

Chapter 3. Explains theoretical background about object detection algorithms.

Chapter 4. Explains the research questions formulated for this thesis and research methodologies selected to answer them.

Chapter 5. Details about the results of the experiment and discussed.

Chapter 6. Contained the research conclusion and future work.

PART 2

LITERATURE REVIEW

With the fast advancement of deep learning networks and their applications in detect of objects and recognition and because of the great importance of object detection technology with high accuracy in all conditions for use in several areas such as self-driving cars, monitoring security, detection systems for the blind and construction vehicles. However, there are many important considerations for detecting and distinguishing objects at high speed: The network's runtime, the size of the network model, and, most crucially, the network's resiliency are all key factors to consider. Neural networks must function under a variety of circumstances seen on city roadways. As a result, they must be resistant to noise generated by the surroundings.
In this section, a review of the relevant literature is conducted to find out the most important applications of object detection on based deep learning. The performance of CNNs is primarily determined by the training dataset, network design employed and applying image transformations to existing training data to produce more training data that some researchers concentrate on.

Montserrat et al. [23] focused on using several data augmentation strategies are utilized to expand the training dataset. Then, using an enhanced dataset, Faster R-CNN is trained for high accuracy object detection and recognition. Therefore, this research mainly focuses on the detection of objects in the wild (trade logos) and the detection of objects captured by the camera installed on the computer system. Achieved an accuracy of 94.32%.

The Deep Convolutional Neural Network is among the most advanced artificial intelligence frameworks, and Yang Luo et al [24] published an OpenCV implementation of it, their architecture sought to make three key achievements: a real-time object identification system, a low-power framework that can be used in wearable electronics and a framework that can run on a variety of computing hardware. The framework's performance was compared to that of the framework that used the YOLO V2 benchmark.

Arcos-Garcia et al. [25] introduced a highly efficient system for recognition of traffic lights in two stages: use A LINX Mobile Mapper system for cloud data collection and analysis to detect traffic lights by reflective materials for the light which represented 3D point. Using the GTSRB dataset and it was the accuracy 99.71. The bitmap image mapping on RGB pictures was then classified using a deep neural network.

In another interesting study by Zhou et al. [26], a set of data is proposed for recognition of traffic lights in snow and ice environment (ITSRB) and detection standard in the same harsh environment (ITSDB) compatible with COCO2017. The data contains images of different sizes and under different weather conditions. The durability and quality of Libar-RCNN and HRNetv2p have been tested under the same harsh conditions in the snow environment compared to Faster-RCNN. The performance of Liber-RCNN is more robust and accurate in detecting and recognizing traffic signals. Therefore, a network based on the classification of (PEA-Net) traffic lights has been proposed, the accuracy was achieved 93.57% in ITSRB.

Chadalawada et al. [27] presented an adaptable fuzzy-based network architecture that is used in conjunction with Deep Convolutional Neural Networks to accomplish extremely efficient object detection. It was concluded that YOLOv3 is the best algorithm in real-time object detection and construction vehicle tracking for long-range pictures with poor contrast and changeable, noisy backgrounds. Achieved an accuracy of 89.51%by VOLOv3, 84.07% by Faster R-CNN, and 74.05% by YOLOv3-tiny.

The authors of [28][29] investigated the influence of noise in the input data and aimed to improve CNN accuracy by purposely adding noisy pictures during the training phase. Dresossi et al. [30] proposed a paradigm by comparing with classical augmentation techniques to study the case of object detection based on deep neural networks (DNNs) in self-driving. The researchers used a data set of type ImageNet 2012 and compared it with a data set of type ILSVRC. The results of the dual-channel were in pre 1 64.98, pre 2 48.48, and pre 3 27.93[28].

Maturana et al. [31] a 3D Convolutional Neural Network was suggested (CNN) technology called ‘Vox-Net' that uses LiDAR data and RGB-D bounding boxes to accomplish accurate and efficient item recognition. They compared their method to publicly available state-of-the-art evaluations and determined that it beat them in terms of object categorization accuracy in real-time.

Due to instability in deep neural networks, such as distortions in the image, which can impede the embedding feature, which affects the performance of computer vision tasks. Zheng et al. [32] suggest a training method to achieve stability in deep neural networks against distortions in the image input, that is, stability training makes the deep neural network more powerful through the model to be stable on the input images with different disturbances and this method has high performance in the case of noise. Precision for stability training on the original image was 75.1% on the ImageNet dataset.

Redmon et al. [3] recommended that detection and recognition be treated as a regression issue, and they created the YOLO integrated algorithm to predict region proposals and conditional probabilities automatically from a whole image in a single assessment. An input picture is split into a collection of grid cells in YOLO, for each cell responsible for identifying objects with centers are within that cell. The algorithm was trained on data of type PASCAL VOC 2007 the performance of YOLO conducted by the researchers was 63.4% represented average precision (MAP) as well as on the VOC2012 dataset 57.9% MAP. The researchers also used a combined model between Fast-RCNN and YOLO, and it performed well in the object detection process by 2.3%.

The beginning and middle phases of the Faster R-CNN network, which are Fast R-CNN, CNN feature extraction, and the expensive per-region computation, slow down the network. To address this problem, in used cutting-edge technical innovations to make improvements in the feature extraction step and introduced as the newer network. This network can recognize items from a wide range of categories with the same accuracy as its competitors but using less computing power. The performance of Fast R-CNN conducted by the researchers was MAP 68.4% on VOC 2012 and MAP 68.8% on a dataset from type VOC 2010. This research was done by R. Girshick [33].

Kaiming He et al. [34] proposed an ROI Align layer rather than the ROI pooling layer, which achieves pixel-level alignment using bilinear interpolation. The ROI Align can also boost the object detection branch's accuracy. The experiment shows the Mask R-CNN with ResNet-101- FPN [35] as the backbone network outperforms Faster RCNN with G-RMI in this experiment.

Shrivastava A et al. [36] proposed the skip model to combine high-level features with low-level features, but low-level features require accurate top-down contextual information. Inspired by the optical path, the researchers' suggestion was to make top-down adjustments to incorporate finer details into the object detection framework. A standard bottom-up Conv-Net network with a top-down modulated (TDM) network connected using lateral connections is responsible for modifying the filters of the bottom layers of the network and the top-down network defines contextual information and low-level features. TDM provides a proposed COCO test standard. The object detection competition at COCO 2016 was won by Mask RCNN. When compared to Inception-ResNet-v2-TDM Faster RCNN, its AP is 3 points higher.

Tanvir et al. [37] suggested modifying the loss function and adding a spatial pyramid pooling layer and inception module with 1×1 convolution kernels to a YOLOv1 neural network for object recognition. The efficiency of the upgraded network can be shown in the Pascal VOC dataset from 2007 to 2012, with detection rates of 65.6% and 58.7%, respectively.

S. Jana et al. [38] introduced Deep-Test, which is an advanced approach to testing for deep learning systems in many areas of safety and security including autonomous driving cars. Existing deep learning test relies heavily on manually categorized data, which frequently fails to notice incorrect behavior of sparse inputs. The Deep-Test that researchers submitted to evaluate autonomous driving DNNs using enriched data and detecting erroneous behaviors. To evaluate DNNs for resilience, they used weather situations such as rain, fog, and illumination conditions. Nonetheless, the induced artificial rainstorm was not produced realistically.

Mazin Hnewa et al. [39] The researchers proposed object detection under rain conditions in self-driving vehicles based on the YOLO algorithm and rain removal. The researchers used the BDD 100k data set and the images used were from the car window for public streets.

Rui Qian et al. [40] suggested a raindrop removal approach based on a single photograph. The method uses a generative adversarial network (GAN), in which the generative network constructs an attention map using an attentive-recurrent network, which is then applied to the input image via a contextual auto-encoder to build a raindrop-free image. The validity of the produced output is then assessed globally and locally by the discriminative network. It is necessary to be able to validate locally, the results were on their dataset. The researchers achieved the image quality PSNR 31.57 and SSIM 0.9023 by the suggested model (AA+AD).

Yan Huang et al. [41] suggested a method for removing raindrops from single photos that employs a deep neural network. The researchers have introduced a mechanism that allows important features to be individually tested in the network to deal with raindrops of different sizes, in addition to using network-neutral features to restore image quality, the researchers describe in their paper the effectiveness of the neural network and its components. The researchers achieved the image quality PSNR 30.72 and SSIM 0.9262. This research used the same dataset [40].

Teena Sharma et al. [42] presented processing for the captured image that is under the influence of dust, rain and snows, such as images taken from inside cars for traffic monitoring while driving and object detection using YOLO. In their work, the researchers used Roboflow datasets. This approach is used without removing noise, so it may cause errors during the detection process. The value of the accuracy for car identification was 72.3%. Yuhang Liu et al. [43] the researchers proposed a method to detect pedestrians under rain conditions with the removal of rain lines using the YOLO deep learning model on real and artificial rain data, and the results of average precision are 21.1%, 48.1%, and 60.9%. Table 2.1 shows details of all related works that were remembered in this study.

Table 2.1 Summary literature review.

Method	Dataset	Results	Reference
VGG16 and ZF	FlickrLogos-32 dataset	Accuracy 94.32% Accuracy 92.91%	Montserrat et al. [23]
CNN with 3 STNs	GTSRB dataset	Accuracy 99.71% F1-score 93.4%	Arcos-Garcia et al.[25]
PFANet	GTSRB dataset	Accuracy 93.57%	Zhou et al. [26]
YOLOv3, Faster R-CNN and YOLOv3-tiny	VOC2007 and VOC2012	Accuracy 89.51% Accuracy 84.07% Accuracy 74.05%	Chadalawada et al. [27]
Deep Neural Networks	ImageNet	Precision 75.1%	Zheng et al. [32]
Fast R-CNN + YOLO	PASCAL VOC 2012	MAP 70.7%	Redmon et al. [3]
Fast R-CNN	VOC 2012 VOC 2010	MAP 68.4% MAP 68.8%	R. Girshick [33]
YOLOv1 neural network	Pascal VOC2007 Pascal VOC2012	detection rates 65.6% 58.7%	Tanvir et al. [37]
GANs	Their dataset under rain condition	PSNR 31.57 SSIM 0.9023	Rui Qian et al. [40]
YOLO	Roboflow datasets	Accuracy 72.3%	Teena Sharma et al. [42]

PART 3

THEORETICAL BACKGROUND

Object detection in an image is divided into two types: object localization and object classification by algorithm. As the DCNNs appear powerful feature representation strength [2], the standard object detection structures accord with the DCNNs, which may be classified into two groups: first, a two-stage object detector that distinguishes the task of object localization from the task of object categorization. The major benefit is excellent detection accuracy, whereas the main drawback is poor detection speed. R-CNN [44], SPP-Net[45], Quick R-CNN[46], Quicker R-CNN[47], Cover R-CNN [48] and RFCN [49] are all object detector architectures with two stages. Other variations include one-stage object detector architectures, which employ DCNNs for localization and classification of the item in the picture rather than splitting it into two divisions. The region suggestion approach, which is less sophisticated than the two-stage object identification method, is not required. Although the detection accuracy is frequently less than that of a two-stage object detection structure., the primary benefit is the quick detection time. Figure 3.1 depicts the object detection evolution milestones, for instance, YOLO series [44][47] Single Shot Multi-Box Detector (SSD)[50], DSSD[51], FSSD [52] and DSOD [53] are included in the stage object detection.

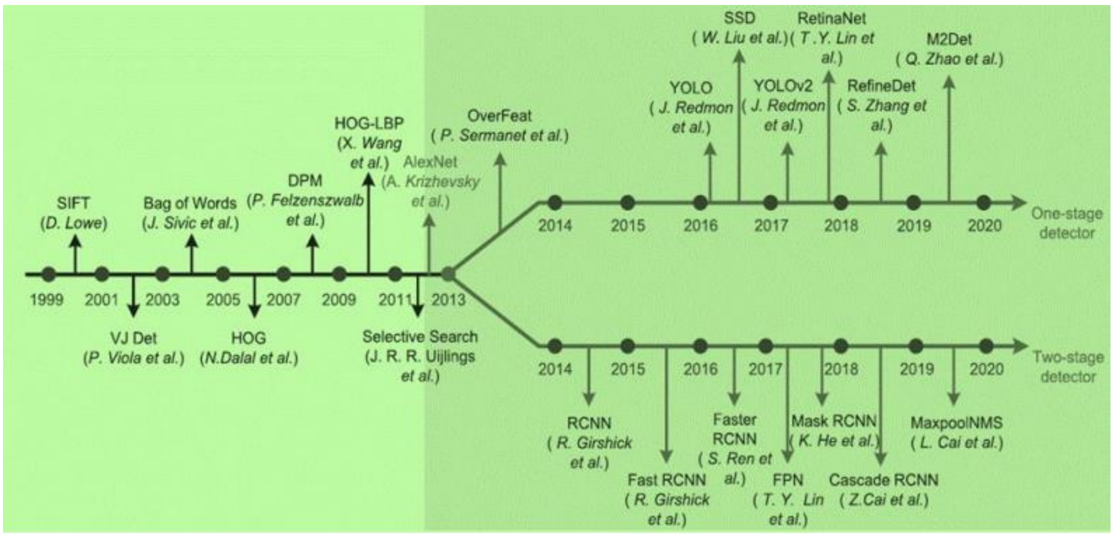

SIFT (D. Lowe)
Bag of Words (J. Sivic et al.)
DPM (P. Felzenszwalb et al.)
HOG-LBP (X. Wang et al.)
AlexNet (A. Krizhevsky et al.)
OverFeat (P. Sermanet et al.)
VJ Det (P. Viola et al.)
HOG (N.Dalal et al.)
Selective Search (J. R. R. Uijlings et al.)
1999 2001 2003 2005 2007 2009 2011 2013
YOLO (J. Redmon et al.)
SSD (W. Liu et al.)
YOLOv2 (J. Redmon et al.)
RetinaNet (T.Y. Lin et al.)
RefineDet (S. Zhang et al.)
M2Det (Q. Zhao et al.)
2014 2015 2016 2017 2018 2019 2020
One-stage detector
Two-stage detector
RCNN (R. Girshick et al.)
Fast RCNN (R. Girshick et al.)
Faster RCNN (S. Ren et al.)
FPN (T. Y. Lin et al.)
Mask RCNN (K. He et al.)
Cascade RCNN (Z.Cai et al.)
MaxpoolNMS (L. Cai et al.)

Figure 3.1. The signs to detect object evolution, where Alex-Net [54]works a division between traditional [55] and DCNNs-based methods. The advance in object detection according to DCNNs has one [56] and two-stage detection architectures [57][49].

3.1 ARCHITECTURE OF TWO-STAGE OBJECT DETECTION

3.1.1. R-CNN

R-CNN was inspired by Alex-Net's tremendous performance in extracting image features. HOG, SIFT, and other classic feature extraction methods were replaced [55], The backbone network for feature extraction was DCNN [54], and it was also used in conjunction with region proposal algorithms (example selective search [58], category-independent object proposals [59], MCG [6] and CPMC [33] to provide region suggestions for the R-CNN architecture (Regions with CNN characteristics) [60]. Figure 3.2 shows this. Also, the R-CNN architecture pipeline and the steps are as follows. The first process is “Around 2000 category-independent area recommendations are generated by the selective search method. Second, the region recommendations are sent into the DCNN, which produces a 4096-dimensional feature as a representation.”. Finally, the SVM algorithm is used to partition the attributes. To fine-tune the bounding-box, will be applied bounding-box regression and greedy non-maximum suppression (NMS) [61]. to adjust the bounding box, to sum up, RCNN adds 30%improvement to the performance than the traditional object detection algorithms [62].

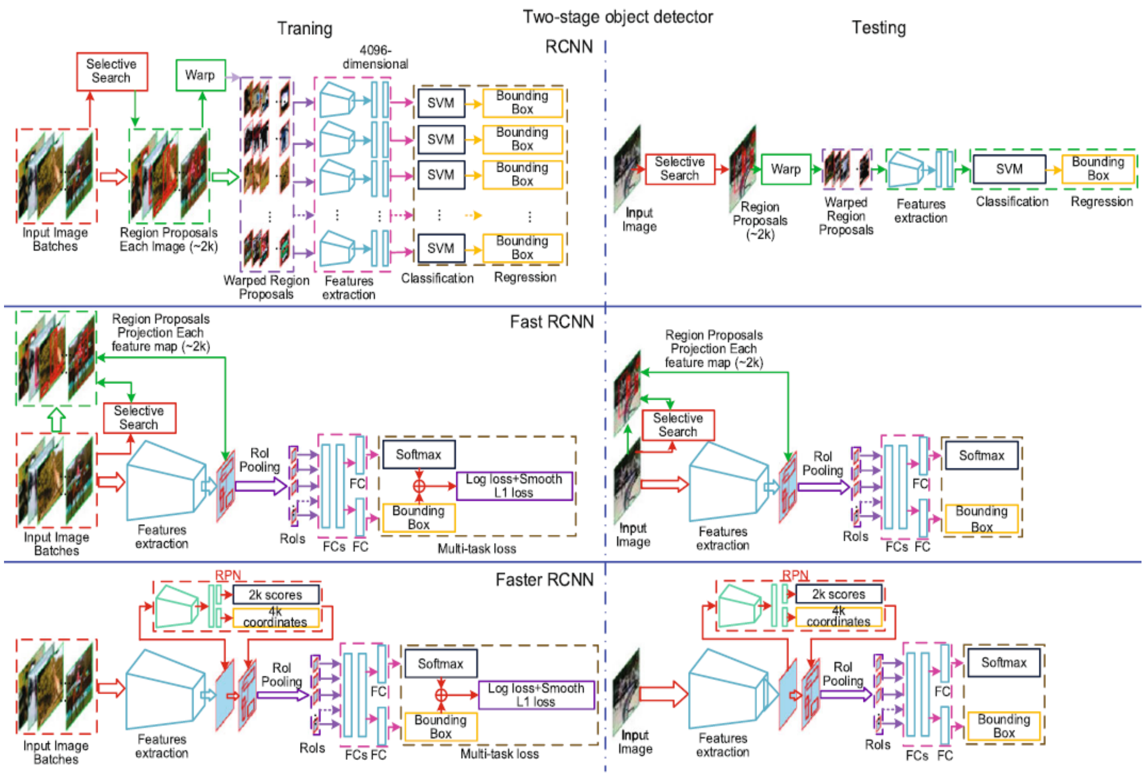


Figure 3.2. The traditional detectors of the two-stage object with ordinary, RCNNs that are fast and faster. The training has not changed, whereas the testing on the right reflects their relationship[20].

Although R-CNN has helped to improve object detection into neural networks, three disadvantages are preventing applying the instance scenario:

- Each picture requires around 2000 region proposals to be pre-fetched, which takes up plenty of storage and I/O resources.
- The cropped/warped area block gets distorted into a 227×227 RGB picture when Alex-Net [55] is used as the backbone network, this causes the object image to be truncated or stretched, as a result of which object information is lost.
- The extraction of each recommended feature from each region individually does not exploit the feature sharing features of DCNNs, resulting in a massive waste of computational resources.

3.1.2. SPP-Net.

Cropping and warping an image in the R-CNN is removable. Also, The spatial pyramid pooling (SPP) layer is comparable to the SPM layer [63] and can be considered as an addition following the layer of the last convolutional (Conv). So, the cropped/warped image-making object information missed is treatable. Therefore, an arbitrary size image could be inserted into the DCNNs for the calculation of a "Fixed-length 21-dimensional feature vector for fully-connected (FC) layer" [64]. The whole image feature sharing maps could make SPP-Net test speed 10 to 100× quicker than R-CNN. SPP-Net and R-CNN are identical with training that is not complete. The layer of convolutional (Conv) cannot continue the training during fine-tuning in SPP-Net limiting the network precision [52].

3.1.3. Fast R-CNN

The analysis of the problems of SPP-Net and R-CNN in speed and training process space consumption as "The ROI (Region of Interest) is a one-level SPP (spatial pyramid pooling)" [65]. "In the Fast R-CNN, DCNNs are used to compute the image's feature map. The selection search (SS) [49] technique is used to locate and map region suggestions in images". Now, the ROI transfers various properties to different features of the regions for fixed-size feature vectors. It then inserts them into the FC. The last stage is the softmax prediction of the object types to the bounding-box regression precisely locating the object, with its design as Figure 3.2 shows. The Fast R-CNN uses multi-task loss to aid classify and bound-box regression. As a result, the two jobs have comparable convolution features. Therefore, we can transform "stage-wise training of SVM + bounding-box regression (stage-wise training" [66]) into multi-task training [67]. The innovations make the Fast RCNN different from RCNN/SPP-Net as follows:

- The precision of Fast R-CNN is bigger than that of R-CNN/SPP-Net.
- The multi-task loss makes the training of the detector end-to-end.

- Every network layer can be updated by the Fast RCNN training while SPP-Net updates FC only.
- There is no need for Hard disk storage to feature caching.
- The speed of training and testing is bigger than that of R-CNN/SPP-Net.

3.1.4. Faster R-CNN

The Fast R-CNN has precision and speeds improved significantly. However, the generation of the selective search algorithm agrees with "approximately 2000 region proposals/ROIs." [49][58]. This algorithm is required to look for all these proposals. This happens writing in the picture. This is to map them into the feature, significantly time demanding. Fast R-CNN involved a 2.3s to predict, 2s for the generation of 2000 ROIs. Consequently, "the traditional region proposal algorithms [60] become the object detection architecture's bottleneck. Shaoqing Ren et al. [49] developed a regional proposal network (RPN) in the Faster R-CNN to address this problem." (As shown in Figure 3.2). To create region proposals, this approach is utilized instead of the selective search technique. The integration of “Extraction of RPN area suggestions into DCNNs " has image convolution features that are akin to the detection network. Because the feature mapping and RPN are implemented on the GPU, the procedure is nearly free.

3.1.5. Mask R-CNN

Mask R-CNN represents the R-CNN extension adding “Parallel to object classification and bounding-box regression, a mask network branch is used for ROI prediction segmentation” [60][34]“ It is capable of completing object detection and instance segmentation at the same time. The Mask network is a condensed form of the fully convolutional network that was used to” create a split mask for each ROI. “The feature map region and the original picture region are not aligned due to the integer quantization of the ROI pooling” [49][58]. “As a result, it generates a bias for properly predicting pixel-level masks”.

3.1.6. FPN

FPN may be thought of as a feature fusion and detection approach. Prior to FPN, detectors employed top-level feature detection or detection at many feature levels. Because poor semantic information is limited, while object location information is abundant, and there is plenty of high-level semantic information, but object placement information is sketchy, these systems cannot account for both categorization and position information. As a result, the FPN connects distinct feature layers via top-down and lateral linkages, detecting the fused multi-layer feature layers. This strategy significantly improves detection performance. Faster R-CNN with FPN offers cutting-edge performance on the MS COCO dataset [57].

3.2. ARCHITECTURE OF ONE-STAGE OBJECT DETECTION

3.2.1. Over Feat

The Over Feat main function is to use the multi-scale quick window to extract the patch of sliding [68] on the last DCNN pooling layer. For the prediction of the classification score all patches and merging them based on the scores. Thus, the multi-size problem and complex shape of the object image could be treated. Over Feat utilized the regression and classification of DCNN for the achievement of classifying and locating the object about R-CNN [33] and Over Feat shows some positive features in speed while lacking precision. According to (Yann LeCun et al.2013) [19] used a famous Over Feat architecture is the feature sharing of DCNNs for the integration of classifying and locating the object into an architecture of the network

3.2.2. YOLO

RPN is used by Faster RCNN to decrease the amount of region recommendations from around 2,000 (RCNN/Faster RCNN) to 300. Yet, some of these proposals overlap [69] inevitably causing repetitive computation. This makes it difficult for "the object identification architecture to overcome the performance barrier". YOLO was suggested in 2015, (You Only Look Once) as a neural network of end-to-end single [44] implementing bounding-boxes regression and class probabilities from a full-image directly. The full image is splitting by YOLO into $S \times S$ grids. The cell of every grid oversees object center detection from it. Each cell then can predict the B bounding-boxes, C class likelihoods, confidence ratings, as well as the whole picture encoded for the output of $S \times S \times (5B + C)$ tensor. Figure 3.3 shows the YOLO for training and testing.

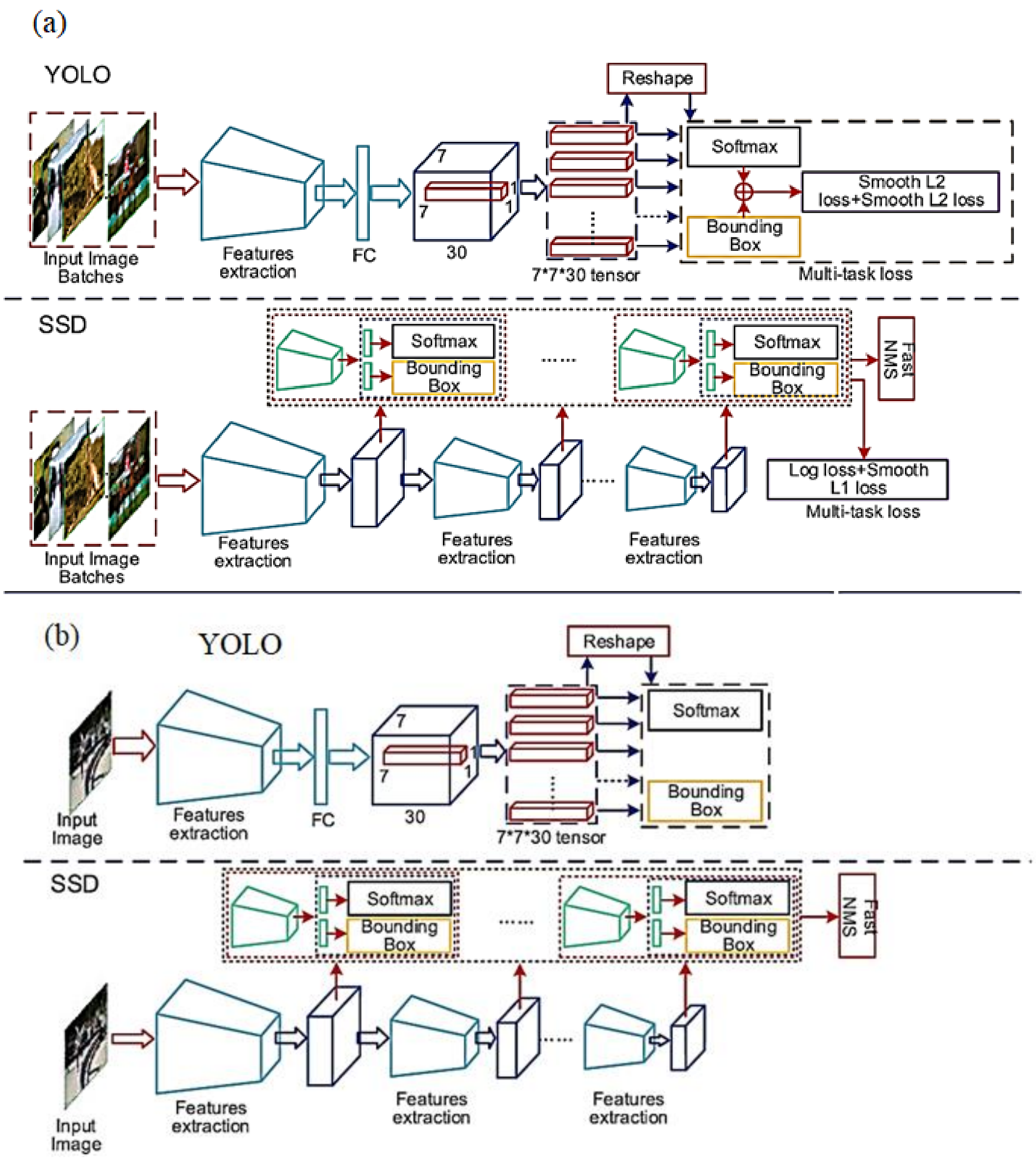


Figure 3.3. The classical one-stage object detectors with SSD and YOLO: (a) is the training process and (b) is the testing procedure that demonstrates the link between the two [20].

This architecture has 24 CON and 2 FCs with an entry of 448 × 448 picture network. This work shows the network training by Set S=7, B=2, C=20 in the PASCAL VOC dataset as the final prediction code is 7×7×7 tensor. About testing set 45 fps, the speed can fulfill the real-time image conditions. However, YOLO can be improved because of the following weaknesses:

- In dense small objects, YOLO is not operating very well because the grid cell prediction is restricted to two bounding boxes of the same class.
- There is a weakness in the generalization ability in which a new factor ratio happens in one object category in the image.
- The weakness of loss function influences the effect of the detection.

3.2.3. YOLOV2

YOLO performs real-time object detection, yet there are some localization errors with low recalls. For better precision, YOLOv2 [19] is better than YOLOv1 as follows:

- Implementing batch normalization [63] accelerates network convergence while improving network applicability.
- "Develop high-resolution classifiers" accommodates high accuracy images [70].
- Solving the weak generalization capability of different YOLO aspect ratio objects. YOLOv2's anchor idea is included in the Faster RCNN [49] and each grid cell is capable of predicting 3 scales in addition to 3 aspect ratios.
- In YOLOv2, there is a K-means [51] with a clustering algorithm. This helps to spontaneously locate the previous bounding boxes improving the detection and making it easier.
- YOLOv2 restricts the deviation of the ground truth for grid cell coordination between 0 and 1, hence resolving network model instability.

For faster speed, Darknet-19 [44] backbone network is designed according to the VGG-Net [71]. Here, "classification dataset's combined training process" is used (ImageNet [72]) with "object detection datasets" (COCO [[73]]) for neural network training. So, with the help of a trained network dubbed YOLO9000, the network discovers new object categories.

3.2.4. YOLOv3

YOLO-V3 [49] comes originally from YOLOv1 and YOLOv2/9000 with less weakness to perform a balance between speed and precision. Thus, the authors mix the residual block [74], binary cross-entropy loss, and feature pyramid network (FPN) [57], for upgrading YOLO to YOLOV3. This makes the detection network suit more intricate objects (more types and sizes of objects).

3.2.5. SSD

The RCNN and YOLO are both fast and accurate. The former has high detection precision with a slow speed. Yet, the YOLO shows a fast detection speed, the generalization capability with big dimensional change, and a weak influence of detection for small objects. These advantages helped to propose a Single Shot Multi-Box Detector (SSD) [50]. SSD usage of VGG16 may be viewed as a backbone network for extracting features in place of FC6/FC7 and Conv6/Conv7. After that Conv8, Conv9, Conv10, and Conv11 have been added. The SSD network scheme concept is hierarchical features extraction as in Figure 3.3. The one-stage network undergoes 6 stages: each extracting feature maps of varying semantic levels and conducting a bounding-box regression and object recognition. Faster RCNN enhances SSD adaption for multi-scale object recognition by combining "the multi-scale feature maps" and "the anchor mechanism". The speed of the precision of the SSD512 and VGG16 is 3 times bigger than R-CNN. The SSD300 works 59 fps faster than YOLO [50].

3.2.6. DSSD and FSSD

The improvement of the SSD capability for feature map low-level needs using DSSD [64] and ResNet101 working as a backbone network. Adding skip-connection and deconvolution modules [74] raises the low-level feature maps achieving a feature fusion certain degree. likewise, FSSD connects low-level into high-level features according to SSD with a significant improvement of the accuracy.

3.3. DETECTION OF SMALL OBJECT

Little object location is a challenge in object discovery. The change of little finding location advances the advancement of related uses, like programmed driving, farther detecting picture location, mechanical deformity location, and therapeutic picture location. There are some classic discovery strategies (for example Speedier RCNN, SSD, YOLO) that do not recognize little objects accurately. The analysis of the characteristics of small objects shows basic causes:

- Small objects require small pixel sizes in the original pictures, with minimum derived features for detection, referred to as object resolution less than 32 × 32.
- When the extraction of the original picture is completed by the backbone network down-sampling, the small object’s location details could be lost.
- In the original image, small objects are smaller which misbalancing small and medium or large entities.

Recently, various technical methods for tiny object detection have been developed by Mate Kisantal et al. [75]. One of the designers used the MS COCO dataset when the image is sampled with small objects through copy-pasting small object policies. Feature pyramid network improves multi-scale detection by using the identification and integration of multi-layer features to enhance small object detection [57]. They are followed by offering perceptual Generative Adversarial Networks (GANs) for the enhancement of tiny object detection and identification [76]. " To enhance the feature representation of tiny things, perceptual GANs exploit structural relationships between items of different sizes. The anchor sizes were redesigned and introduced in a Faster RCNN-based context model to improve tiny object recognition". H-CNN such detection “in synthetic aperture radar (SAR)” images in ships was also suggested [77] is multiscale “patch-based contrast” measuring infrared method. It successfully overpowers the interfering of background clutter [78]. There is a 7-layer DCNN aimed at achieving an automatic small object extraction of and clutter end-to-end suppression [79]. The problem in these programs could be treated by combining high- and low-level features, including DSSD [51], Feature-Fused SSD [80].

3.4. DATASETS

The MS COCO dataset (40GB) [73] seems huge in comparison to the above datasets, backed by Microsoft. The image annotation details can be categorized, its information located with the image semantic text description. Similar to the ImageNet in image detection and classification, the MS-COCO dataset turns out to be a yardstick when evaluating algorithm work in the area of visual semantic comprehension. The open source of Google is based on MS COCO dataset models targeting scene understanding and contains 91 classes, 2,500,000 labels, and 328,000 images.

The PASCAL VOC dataset [81] contains high picture quality and full labels, making it ideal for assessing algorithm performance. The training to testing sets ratio is around one-to-one. The image's category distribution is also consistent. The SUN dataset (7GB) [82], which comprises 908 scene categories and 4,479 item types, focuses on object recognition and scene recognition. 313,884 objects have the background associated to them.

The MS COCO cannot be compared to ImageNet and SUN. Yet each type has more images, this helps the model to easily acquire a bigger capability of analyzing a certain object in a training scene. “The MS COCO dataset has more pictures and classifications than the previously stated PASCAL VOC dataset”. For instance, the ImageNet dataset [72] “is at TB level and commonly used in the field of vision. ImageNet is used in a lot of computer vision research, such as picture categorization and object identification. It is cared after by a specialized crew”. Its data is having enough documentation and is accessible. Yet, there are certain Image-Net annotating issues that must be centrally dealt with once a year to be used again. In addition, the latest version is the best one. ImageNet seems to be an algorithm performance evaluation benchmark in its version for the computer. There are 14 million images covering about 20,000 types in it. The famous classification and detection of the ILSVRC challenge [83]depend on the ImageNet dataset.

Open Images is the release of the Google dataset in 2016[84]. This release includes 9 million images and about 6,000 types. It is a new data supporting computer vision developing models not used before. Simultaneously, these big data can guarantee deep network model complete training. Previously, Google unveiled Open Images V4, which has 15.4 million bounding boxes for 600 categories. The largest dataset with the most picture annotations is 1.9 million images. At the same time, there was the ECCV 2018 image challenge. There was much professional staff participating to guarantee high precision and constancy. Also, the sorts of images are diverse with complex scenes and scenarios of multiple objects. 1.5GB is a general size as the Open Images give the image URLs only. Figure 3.4 shows examples of some image.

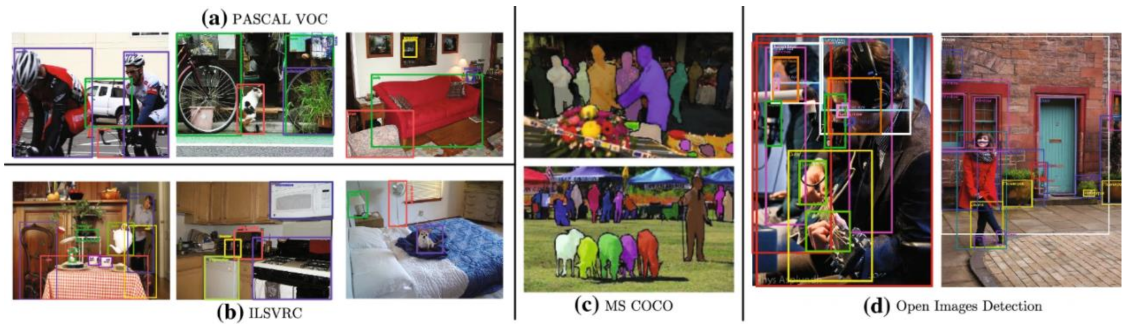


Figure 3.4. Examples on the types of datasets [8].

3.5. SUMMARY

Table 3.1 summarizes the advantages and disadvantages of the object detection architecture.

Table 3.1. The advantages and disadvantages of object detection architectures.

Method	Advantages	Disadvantages
R-CNN [7]	Using DCNNs for the extraction of the image features and search algorithm for choosing 2k region proposals; SVM is being used to classify the regions; Using bounding box regressed for refining regions.	Too slow training; Takes up too much space; The training is not end-to-end.
SPP-Net [85]	DCNNs are utilized for the extraction of the whole image properties with the search algorithm for the extraction of 2k region proposals. Yet, they are mapped to the feature maps; Spatial pyramid pooling is also used for inputting the multi-scale image to DCNNs.	Slow utilization of the selective search for the extraction of the region proposals whose training is not end-to-end

Fast R-CNN [33]	Utilize DCNNs to extricate the highlights of the whole picture; Utilize choice look calculation to extricate 2k locale picture proposition, yet outline them to the included maps; Utilize the ROI Pooling layer for testing the highlights of locale proposition for the inclusion of the fixed-size in the maps; Utilize Multi-task misfortune work	Slow utilization of the selective search for the extraction of the region proposals whose training is not end-to-end
Faster R-CNN [49]	Utilize Region Proposal Arrange (RPA) to supplant the selection search calculation; The RPA offers highlight maps with the spine organize; Can end-to-end training.	The detecting speed is insufficient for multi-scale and tiny objects, resulting in poor performance.
Mask R-CNN [34]	Utilize ROI Adjust pooling layer rather than ROI pooling layer, which makes strides location exactness; Combine preparing question discovery and division to move forward discovery precision; Conducive to a little target location.	no sufficient detection speed
FPN [57]	A suggested multi-level highlight combination Highlight Pyramid Organize is conducive to multi-scale question location and little question discovery.	no sufficient detection speed
YOLO [44]	Propose a novel single-stage discovery organize; Location speed is quick fulfilling real-time prerequisites.	There is no high detecting accuracy, especially for dense and tiny objects.
YOLOv2 [19]	Multi-dataset joint preparing: modern spine arranges (DarkNet19); utilize k-means clustering calculation to create grapple box	The training is complex.
YOLOv3 [47]	Utilize multi-level highlight combination to progress the exactness of multi-scale location; Unused spine arranges (DarkNet53).	As IoU increases, performance decreases.
SSD [50]	Multi-layer location instrument; multi-scale grapples component at distinctive layers.	The detection of a small object is not conducive.

PART 4

METHODOLOGY

In this part, we explain briefly the methods used first to enhance image quality by removing noise (rain), then we detect objects in the images with high accuracy using combined algorithms under different rain conditions. The proposed algorithm De-raining using Deep Detail Network (DDN) with Object Detection using (YOLOv3) is shown in Figure 4.1.

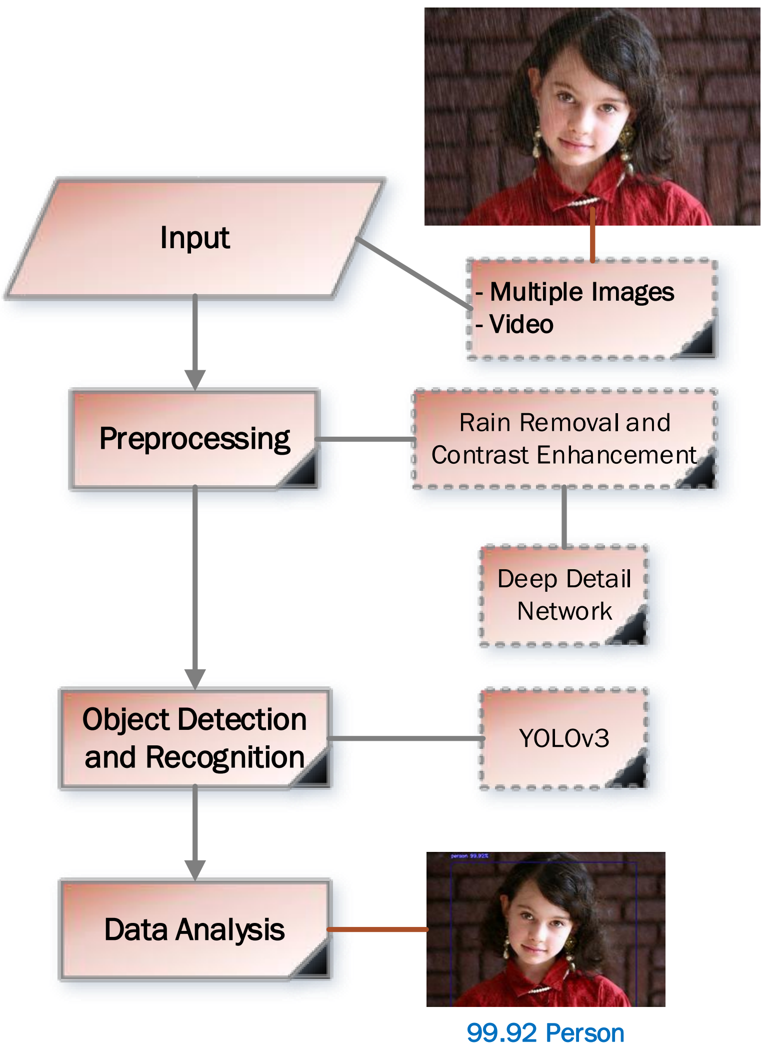


Figure 4.1. Proposed algorithmic scheme for our method (DDN+YOLOv3).

Images datasets were collected during this study in clear and rainy weather conditions or videos under rain conditions to obtain a high-resolution image of object detection using the rain removal algorithm and the object detection algorithm by (YOLOV3 and YOLOv3-tiny) and a comparison between them through the results.

Software requirements: The deep learning algorithms used in this study were developed with the Python programming language, the data has been processed with several libraries, including TensorFlow libraries (version 1.15), Sickit image (version 0.18.1), NumPy (version 1.20), and Imageio (version 1.2.0) that representing rain removal libraries. As well as object detection libraries, which are TensorFlow (version 2.4), Keras (version 1.4.3), pillow (version 7.0.0), matplotlib (version 3.3.2), h5py (version 2.10.0), NumPy (1.19.3), Scipy (version 1.4.1), Keras-resnet (version 0.2.0) and OpenCV-python. The primary platform used for this work is Spyder (python 3.8).

Hardware requirements: The proposed system was implemented using a personal computer that has a 64-bit operating system, x64-based processor, 16.00 GB of RAM, Windows10, on an Intel (R) Core (TM) i7-7500U CPU @ 2.70GHZ 2.90 GHz.

4.1. DATA COLLECTION

The data set that used in our study contains many images or videos of various conditions and accuracy as models for training and testing on conducting object detection and noise removal (rain). It represents the data used from the COCO (Common objects in context) dataset [73] is one the biggest datasets available for object detection and classification. COCO contains more than 330,000 images (200,000 labeled), and about 1.5 million object instances. Rainy images dataset [4] has 1,000 clean images each of which can generate 14 ones with changed rain streaks magnitudes and orientations. All are 14,000 images, 12,600 of them used can be used to train and 1,400 to test. YTVOS201dataset [86] which is considered one of the largest data used to detect the object under rain that it contain on 604 images for only testing and we also used real rain image [4]. The images used should be as diverse as possible and not only represent external scenes (such as surveillance cameras, traffic, and cars), but always include different data, rain conditions, and normal conditions.

4.2. FEATURE EXTRACTION

The model used is implemented as a convolutional neural network (CNN) to extract image features from each suggested region automatically when training and before we can calculate the proposed features a region, we must first transform the image data in that area into a format that the CNN can understand (the CNN's design needs inputs with a constant size such as 224 × 224 pixels). We choose the simplest of the numerous possible transformations for our arbitrary-shaped areas. In a tight bounding box, we warp all pixels around the candidate region to the desired size, regardless of its size or aspect ratio. Before warping, we dilate the tight bounding box such that the original box is surrounded by p pixels of distorted image context at the warped size.

4.3. DE-RAINING ALGORITHM

The performance of video/image processing operations may be substantially harmed by rain streaks. Removing rain streaks from photos and videos is receiving great research interest in computer vision and object recognition, and several approaches have lately been presented to tackle this problem.
Our main goal is to improve the accuracy of detecting objects by removing rain streaks, compared to previous works.

Following the enormous success of deep neural networks, researchers examined a deep architecture [4] for rain removal. The network is a deep convolutional neural network (DCNN). This network is mainly based on the deep residual network (Res-Net) [74]. The basic idea of Res-Net is simplifying by altering the mapping form during the learning phase. The same idea was implemented by using deep convolutional neural networks to a single picture with clear rain streaks.

Since the outcome of the de-rain process will be used in object detection, we needed a method that preserves the quality of the image. For this purpose, we studied an algorithm with the highest qualitative and quantitative measures named with Deep Detail Network (DDN) [4] considered approach inspired Residual Neural Network (Res-Net). The framework for de-raining from a single image in Figure 4.2.

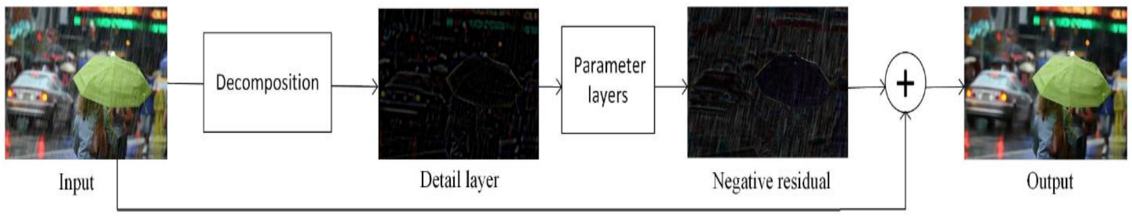


Figure 4.2. The framework for de-raining from a single image [4].

4.3.1. General Idea (negative residual mapping and deep detail structure)

This method uses the characteristic of rain streaks to enhance the network result and give some sort of bias. The authors first applied a standard deep convolutional neural network, but outcomes of the networks were not satisfying as shown Figure 4.3.

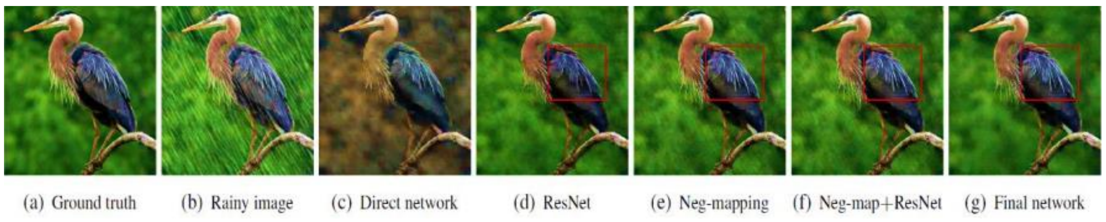


Figure 4.3. The effects of various network architectures on rain removal [4].

We can see that the direct network in Figure 4.3.c shows that image color space changed, so the network not only removed the rain but also changed the range of pixels (colors) in the image. The researcher briefly explains this phenomenon to make it simple, let us think that there is a normalization of using D pixels, the images X and Y to [0-1]. The regression function between these images' maps must be retrieved from $[0,1]^D$ to $[0,1]^D$ and must be obtained. This means to map a range covering the whole pixel values making it difficult to gain a thorough understanding of the regression function. Also, developing a deep neural network Straight photos have gradient issues that disappear when regularization approaches like batch normalization are utilized (Figure 4.4).

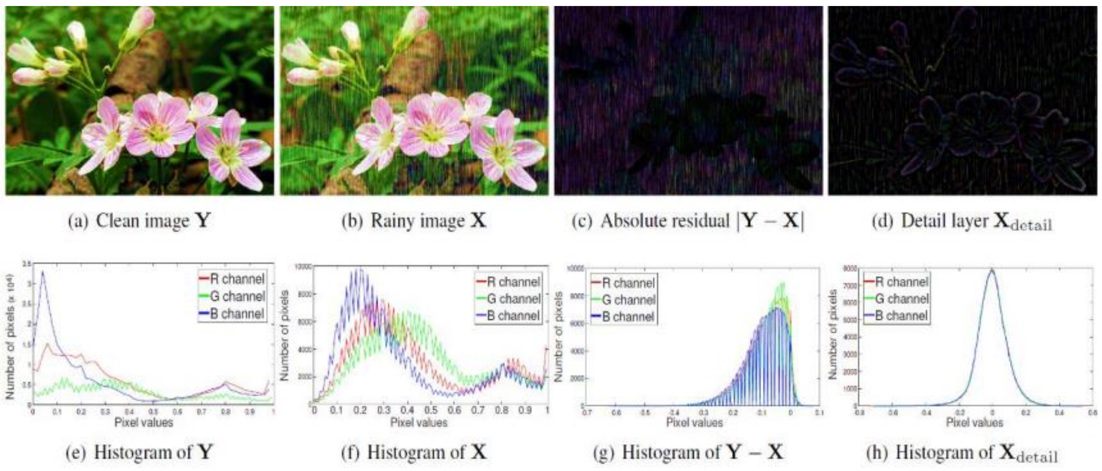


Figure 4.4. |Y-X| to improve visualization through range decrease, detail layer, and negative residual sparsity [4].

The improvement of the network knowing makes it critical to diminish the arrangement space the mapping run is compressed. Figure 4.4(e) and (g) show this and about the clean picture Y, the leftover of the blustery picture Y − X features a critical run lessening whose unit of measurement is the pixel. Thus, the residual can be presented into the network assisting to obtain the mapping. In this way, they utilized the remaining as a yield of the parameter layers.

This skip association too straightforwardly may engender lossless data by the complete network, valuable for evaluating the ultimate de-rained picture. Since rain seems in pictures as white streaks, Y − X could be negative in most cases, (Figure 4.4(g)). In this way, alluded "to this as negative leftover mapping" (neg-mapping for brief). An adjusted goal that consolidates the thought:

$$\mathcal{L} = ||h(Xi\,) + Xi - \ Yi\ ||_F^2 \tag{4.1}$$

In Figure 4.3(d), the de-raining produced from the common Res-Net model [74], is modified to suit the image regression weakness. In the original Res-Net structure, rain streaks are removed while blurring the bird's feathers. However, Figure 4.3(e) shows the details of both the color and object by setting in the neg-mapping with no use of the Res-Net structure. This is caused by the lossless information being in direct propagation and updating parameters with the help of the skip connection. In addition, neg-mapping has fewer testing errors and training than Res-Net.

In the long run, it can be seen within the produced pictures that not all rain streaks disappeared. This infers that rain streaks and protest points of interest are not completely recognized by the show. This spurs the taking after improvements of the neg-mapping thought.

Deeper architectures could grow the capacity and flexibility to explore and model the image features [71], a Res-Net [74] structure implemented with neg-mapping for better differences between the rain streaks from other details of objects. In this structure, the input information is guaranteed, and its propagation happens by all parameter layers, training the network. Yet, Figure 4.3(f) shows that even with the incorporation of both, slight rain streaks remain in the produced image. So, the source Res-Net [74] approach is different from the detail layer input to the parameter layers. Thus, the first model is:

$$X = X_{detail} + X_{base} \tag{4.2}$$

where 'detail' and 'base' are the detail and the base layer respectively. "The base layer can be obtained using low-pass filtering of X [87][88][89] after which the detail layer $Xdetail = X - Xbase$. After subtracting the base layer from the image, the interference of background is removed and only rain streaks and object structures remain in the detail layer" as Figure 4.4(d) and (h) show. Then, because most areas are near zero, the detail layer's sparsity is higher than the image. This sparsity is not used in the current de-rain methods [90][91], while the framework of deep learning is pure. Besides, (c), (d), (g), and (h) show in Figure 4.4 "both the detail layer and the neg-mapping residual display critical extend lessening". Hence, the viable mapping is produced by littler subsets ranging from $[0,1]^D$ to$[0,1]^D$. [74] Showing the network space contracted and thus network execution ought to be progressed.

This helps the combination of the detail layer $Xdetail$ with the suggested neg-mapping Y − X because there is an input to the Res-Net parameter layers. Scholars training the detail layer network name this "deep detail network." In Figure 4.3(g), the last output is produced from the combination of the designed deep detail network and neg-mapping. This output helps to achieve a good convergence rate and a good structure similarity index (SSIM), yet there is a cleaner visual de-raining influence in it.

4.3.2. Network Architecture

The de-rain outline input could be a blustery picture X and the yield is close to the clean picture Y. Thus, they characterized the objective work to be:

$$\mathcal{L} = \sum_{i=1}^{N} ||f(X_{i,detail}, W, b) + Xi - Yi\ ||_F^2 \tag{4.3}$$

where N is and $f(\cdot)$ they are Res-Net number of training images respectively. Also, W and b are two network factors that must be known. X_{detail}, guided filtering is utilized [87] as a low-pass filter for the division of X into detail and base and layers. The removal of the image indexing and basic network structure is:

$$\begin{aligned}
&X_{detail}^{0} = X - X_{base}, \\
&X_{detail}^{1} = \sigma(BN(\ W^{1} * X_{detail}^{0} - b^{1})), \\
&X_{detail}^{2l} = \sigma(BN(\ W^{2l} * X_{detail}^{2l-1} - b^{2l})) \\
&X_{detail}^{2l+1} = \sigma\left(BN(\ W^{2l+1} * X_{detail}^{2l} - b^{2l+1})\right) + \\
&X_{detail}^{2l-1}, \\
&Y_{approx} = BN(\ W^{L} * X_{detail}^{L-1} - b^{L}) + X,
\end{aligned} \tag{4.4}$$

Here $l = 1, \dots, \frac{L-2}{2}$ with all layers represented by L, $*$ is the operation of convolution, weights are the W and biases b. The batch normalization for the alleviation of the internal covariate shift that represents $BN(\cdot)$ [106]. $\sigma(\cdot)$ stands for the Rectified Linear Unit (Re-LU) for non-linearity. Here, all no pooling activities are detached for the preservation of spatial information.

Filters of size that exist in the first layer $c \times s_1 \times s_1 \times a_1$ are utilized for the generation of a_1 feature maps; The kernel size is s, while the image channels are c, for example, grayscale $c = 1$ and a color picture $c = 3$. The layer uses filters of size $a_2 \times s_3 \times s_3 \times c$ for the estimation of the negative residual. The direct addition of the estimated residual value of the rainy image X produces the de-rained image. Deep Detail Network (DDN) architecture is shown in Figure 4.5.

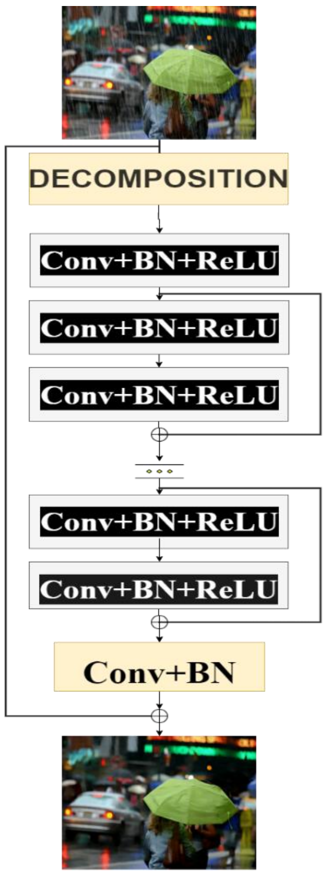


Figure 4.5. DDN architecture

4.4. OBJECT DETECTION ALGORITHM

In neural networks, the image can be classified as a current application of convolutional. In addition to simple categorization, there are many problems with the vision issue of the computer when an object is detected. It is known to be related to self-driving cars and more applications that are used with a blend of computer vision. The same is with video surveillance, when a crowd monitors the prohibition of terrorist attacks, counting people in common statistics or analyzing customers' experience with walking paths in shops. There are a set number of different object detection algorithms, divided into two categories:

- Classification-based algorithms: It is done in two steps. Firstly, identifying regions of interest in an image. Secondly, they use convolutional neural networks to categorize these areas, as illustrated in Figure 4.6. This solution could not be quick as we have to predict the chosen region. This type of algorithm is exemplified by the Region-based convolutional neural network (R-CNN) [92] and Fast R-CNN [86]. Other examples are also Faster-RCNN [49] Mask R-CNN and Retina-Net [56].

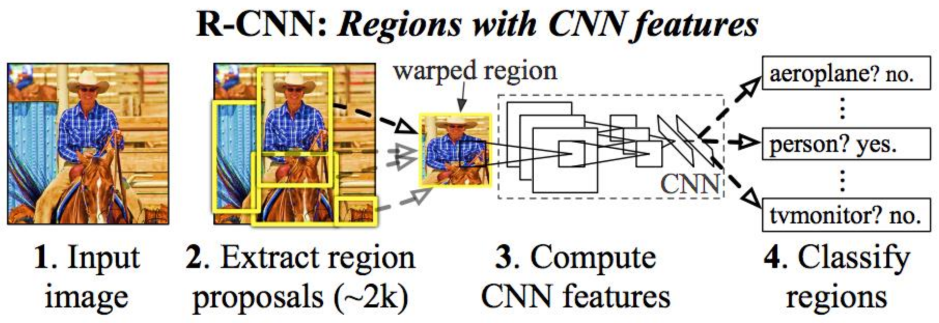


Figure 4.6. R-CNN Object detection system overview [92].

- Regression-based algorithms: They forecast classes and the entire bounding boxes of the picture in a single algorithm operation, rather than picking interesting areas of an image. The general framework could be seen in Figure 4.7. The YOLO (You Only Look Once) [44] and SSD (Single Shot Multi-Box Detector) [50] are instances of this usually utilized in real-time object detection trading some precision for a significant speed increase.

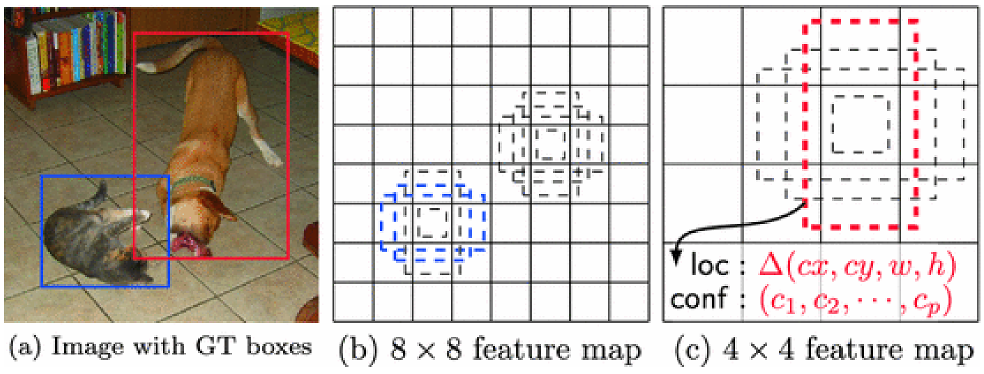


Figure 4.7. SSD framework [50].

In this work, we emphasized real-time object detection methods to get on high accuracy therefore YOLO [44] (and its different versions) was implemented.

Object detection is reframed as a single recurring issue in YOLO, from image pixels through bounding box coordination and course probability. You just look at a picture once (YOLO) to predict what items are presented and where they are using this framework. YOLO is a straightforward concept: As seen in Figure 4.8. At the same time, a single neural network predicts multiple bounding boxes and lesson probability for those boxes. YOLO improves location execution by training on full-frame images. This collection shows a few advantages over traditional object locating algorithms. The system resizes input image to 448 x 448, then applies the neural network to the entire image and applies the thresholds based on the model confidence.

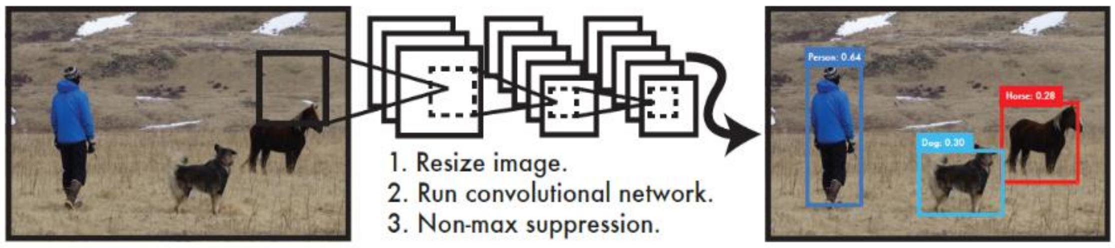


Figure 4.8. The YOLO Algorithm overview [44].

To begin with, YOLO is amazingly quick. The outline discovery as a relapse issue demonstrates that there is no need for a sophisticated workflow. They effectively execute the neural network on a current image at test time to predict location. Second, when proceeding predictions, YOLO looks at the whole image. YOLO displays full image details during the data training and testing process, in contrast to the methodologies and sliding windows based on the suggestion of the region, therefore Preserves all contextual and implicit knowledge about categories while also details of their shape. Fast R-CNN [33], because it is unable to understand the larger context, a top detection approach dismisses background regions in a picture as objects. When compared to Fast R-CNN, YOLO creates half as many backgrounds' mistakes.
Third, YOLO learns it object representations that are generalizable. YOLO significantly outperforms the best detection methods example R-CNN and DPM when using real picture data to train and evaluate models. Because YOLO is so generic, it is less likely to fail when applied immediately to new fields or unanticipated inputs.

4.4.1. The basic Idea of YOLO

YOLO is a neural network that combines several object detection components into a single neural network. Predict each bounding box, this grid to take on the features of full images. It also predicts all the bounding boxes of the image completely across all categories at the same time. This means that the network considers the full image as well as all of the objects included inside it. The YOLO architecture allows for end-to-end training at real-time speeds while keeping a high level of average precision.

The identification of system models as a regression issue. It splits the picture into a S x S grid and predicts B bounding boxes, confidence for those boxes, and C class probabilities for each grid cell. See Figure 4.9. If the center of the target object is located in the grid cell, this grid can detect the object.

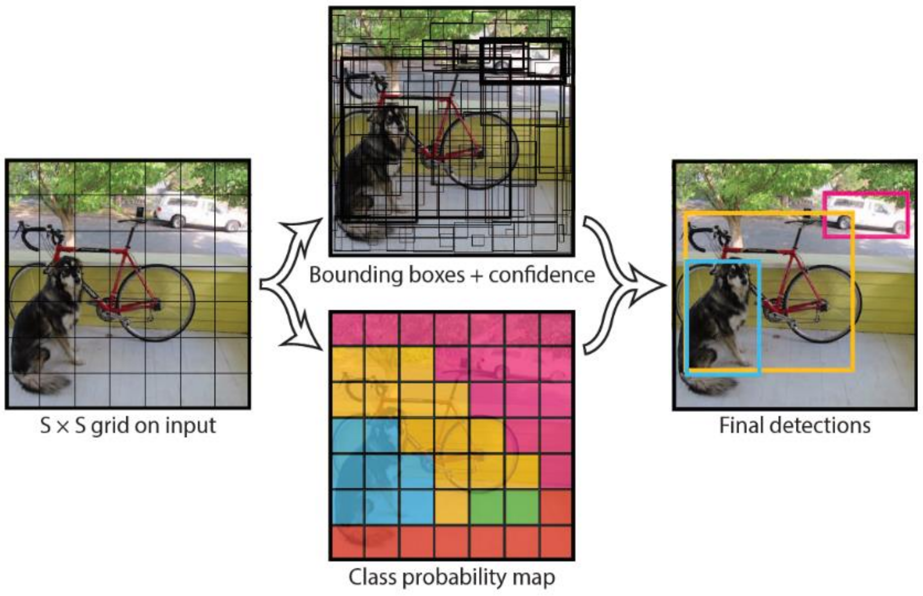


Figure 4.9. YOLO Object Detection Model [93].

Both the bounding boxes B and the confidence ratings in these boxes are predicted. The way of the confidence operates in the model in the box with the object and the accuracy of the box prediction. Formally confidence = Pr *(Object)* $* IOU_{pred}^{truth}$. If a cell contains an object, then Pr(object)=1, or Pr (object) = 0 this means no object in the cell. otherwise, the confidence degree = the intersection over union (IOU) between the predicted box and the truth of the ground.

Each bounding box predicts 5 times: $(x, y, w, h,$ confidence). The (x, y) coordinates are the prediction box relative to the grid center. The (w, h) coordinates of height and width is made based on the image total size. Finally, The IOU between any piece of ground truth and the predicted box is the expected confidence.

Additionally, each grid cell makes a prediction of the *C* conditional class likelihoods, $Pr(Class_i|Object)$ which relies on the grid cell in which there is an object. The algorithm helps to predict only some group class likelihoods in every grid cell, no matter how many boxes *B* are there.

During tests, these probabilities and every box confidence prediction are multiplied,

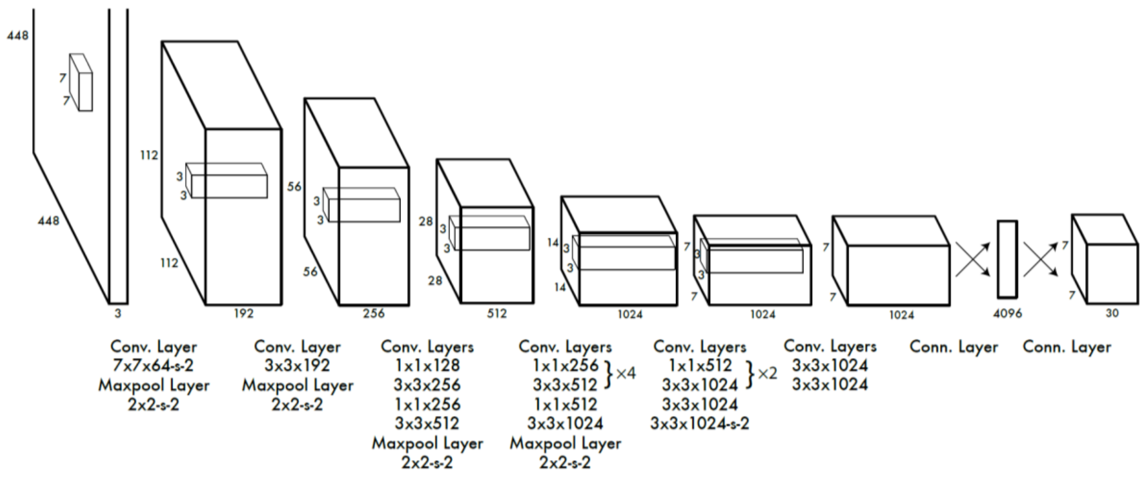
$$\begin{aligned} Pr(Class_i|Object) * \Pr(object) * IOU_{pred}^{truth} \\ = \Pr(Class_i) * IOU_{pred}^{truth} \end{aligned} \tag{4.5}$$

Producing scores of class-specific confidence to every box. The probability degrees that represent the presence of the required category in the bounding box in addition to the accuracy with which the element matches the projected box.

4.4.2. YOLO Network Architecture

The used model is implemented based on the convolutional neural network, in which the initial convolutional neural layers of the network extract all the properties from the used image, while the associated layers predict the throughput probabilities and coordinate values.
YOLO network design is propelled by the Google-Net demonstration for image classification [94]. 24 convolutional neural networks and two fully connected layers make up a convolutional network. Instead of the first units that utilized by Google-Net, YOLO essentially utilizes 1×1 decrease layers taken after by 3×3 CONV. The last

output of the network is the 7 × 7 × 30 prediction tensors. Figure 4.10 shows the total network [44]. The network is trained by half the size 448 x 448.



Conv. Layer 7x7x64-s-2 Maxpool Layer 2x2-s-2
Conv. Layer 3x3x192 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x128 3x3x256 1x1x256 3x3x512 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x256 3x3x512 }×4 1x1x512 3x3x1024 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x512 3x3x1024 }×2 3x3x1024 3x3x1024-s-2
Conv. Layers 3x3x1024 3x3x1024
Conn. Layer
Conn. Layer

Figure 4.10. Architecture of YOLO.

In the experimental results, we can see that YOLO is after the state-of-the-art object detecting algorithms in precision but it’s faster than all of them. To outcome this issue a bunch of changes has been introduced and the new version is called YOLOv3 [47]. In this work, we focused on YOLOv3 and tried to implement it for object detection.

4.4.3. YOLOv3

YOLOv3 [47] is an updated version of the YOLO object detection algorithm, in this version some changes have been made to the design of the model and the network to make the system more accurate but maintain the high speed of the original algorithm. The algorithm predicts bounding boxes based on YOLO9000 by employing dimension clusters as anchor boxes [44]. The grid used predicts four important coordinates for each bounding box, which are: tx, ty, tw, and th. The possible predictions correspond to if the cell is shifted from the upper left corner of the image used by (cx; cy) and the previous bounding box specifies the coordinates, which are width and height pw, ph:

$$\begin{aligned} bx &= \sigma(tx) + cx \\ by &= \sigma(ty) + cy \end{aligned} \tag{4.6}$$

$bw = pw\, e^{tw}$

$bh = ph\ e^{th}$

In the training time, the squared error loss sum is employed. When the ground truth coordinate predictions are $\hat{t}\ *$, our gradient turns the value of ground truth (calculated from the ground truth box) minus the prediction: $\hat{t} * - t *$. This value is calculated by overturning the equations above.

YOLOv3 expects an object-ness score for each bounding box utilizing calculated relapse. If the bounding box before encompasses a ground truth object by more than any previous bounding box before, this should be 1. When the bounding box isn't the finest, It covers a ground truth object by several boundaries, ignoring the expectation, taking after [49]. A limit of .5. utilized, different [49] from this framework allows one bounding box earlier for each ground truth protest. If a bounding box previously is not given to a ground truth protest, it brings about no misfortune for facilitate or course expectations, as it were object-ness.

Based on multilevel classifications, each box predicts which categories the bounding box uses. This framework uses no soft-max because it has been found it is superfluous for great execution, instep makes use of independently computed classifiers. Amid preparing, parallel cross-entropy misfortune was utilized for the lesson forecasts.

YOLOv3 predicts boxes on three separate scales, and using a feature pyramid network-like concept, the program collects features from the scales. To the base feature extractor, many convolutional layers were added. Finally, a three-dimensional tensor encompassing bounding box, object-ness, and class predictions is predicted.
In the testing with COCO [19] YOLOv3 predict three boxes at each scale so the tensor is $N\ x\ N\ x\ [3\ x\ (4 + 1 + 80)]$ for the four offsets of the bounding boxes, one object-ness prediction, and 80 class predictions.

Next, the feature map from two layers was previously obtained and up-sampled by 2×. A feature map from earlier in the network was also obtained and concatenated with up-sampled features. Researchers can extract more significant information from the features of the higher sample and accurate information from the previous feature map using this unique approach. Then, to process this merged feature map, a couple additional convolutional layers were added, which finally predicted a tensor that was double the size.

Boxes to expect for the ultimate scale, the authors repeat the design process. As a result, all the previous computations, as well as fine-grained data from earlier in the network, go into the 3rd scale forecasts.

In YOLOv3-tiny k-means clustering is still used to determine the bounding box priors. They basically picked nine clusters and three scales at random, then divided the clusters evenly over the scales. Using the COCO database [73] the 9 clusters were:

$$(10 \times 13);\ (16 \times 30);\ (33 \times 23);\ (30 \times 61);\ (62 \times 45);\ (59 \times 119);\ (116 \times 90);\ (156 \times 198);\ (373 \times 326)$$

There has been some change in the feature extraction method too, a novel network that does feature extraction. A network is created consisting of a combination of the remaining network models and the networks used in YOLOv2 and Darknet-19 [95]. The network employs a series of 3x3 and 1x1 convolutional layers, although it now has several shortcut links and is somewhat larger. We name it Darknet-53 since it includes 53 convolutional layers.

4.4.4. YOLOv3-tiny

Tiny-YOLOv3 is the version of YOLOv3 which is simple with fewer convolution layers than YOLOv3. This means tiny-YOLOv3 need not occupy much memory. Thus, it reduces the hardware requirement. Also, it speeds up detection significantly, yet loses some of the detection precision.

The backbone network of YOLOv3 uses the Darknet-53 network [96] which is too complex demanding a very high hardware computational power. The complex network structure (Figure 4.9) affects detection speed. The tiny-YOLOv3 backbone has only 7 convolutional and 6 pooling layers as in Figure 4.11. The simplified network helps in the improvement of the detection speed yet loses part of detection precision as a trade-off.

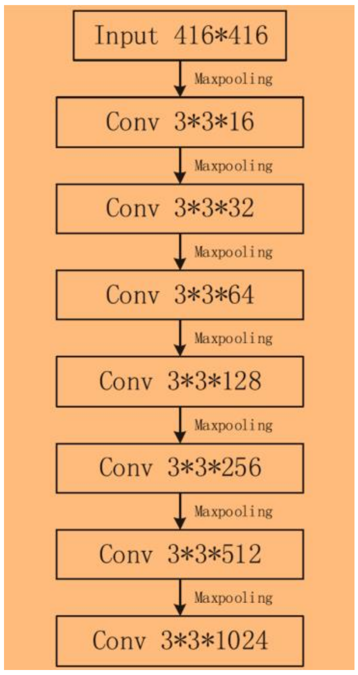


Figure 4.11. The network structure of YOLOv3-tiny [97].

4.5. TRAINING AND TESTING

Since the algorithm is two different models, each of them is trained separately using various datasets based on their purpose (removing rain or detect of objects).
The dataset utilized for training object detection models (YOLOv3 and YOLOv3-tiny) is COCO dataset [73]. The objects belong to 80 categories which are as follows: person, bicycle, car, motorcycle, airplane, bus, train, truck, boat, traffic light, fire hydrant, stop sign, parking meter, bench, bird, cat, dog, horse, sheep, cow, elephant, bear, zebra, giraffe, backpack, umbrella, handbag, tie, suitcase, Frisbee, skis, snowboard, sports ball, kite, baseball bat, baseball glove, skateboard, surfboard, tennis racket, bottle, wine glass, cup, fork, knife, spoon, bowl, banana, apple, sandwich, orange, broccoli, carrot, hot dog, pizza, donut, cake, chair, couch, potted plant, bed, dining table, toilet, TV, laptop, mouse, remote, keyboard, cell phone, microwave,

oven, toaster, sink, refrigerator, book, clock, vase, scissors, teddy bear, hairdryer, toothbrush.

Image samples from the COCO dataset [73] are shown in Figure 4.12. The dataset is available online at (https://cocodataset.org/#home).

Figure 4.12. Image samples from the COCO dataset.

In the de-raining model, because of lack of accessing the ground truth for real-world rainy images. The synthesized dataset of such pictures helps use clean ones for training the network, 70% rainy images in input file with same number images ground truth images (clean image) in label file for training and 30% image for testing from Rainy image dataset [4] and YTVOS201 [86] that use to test only. Following training, the network is utilized for the sake of output clear image whether rain real or synthesis, same dataset that testing for de-raining using for object detection. Some of the image samples from the Rainy image dataset are shown in Figure 4.13.

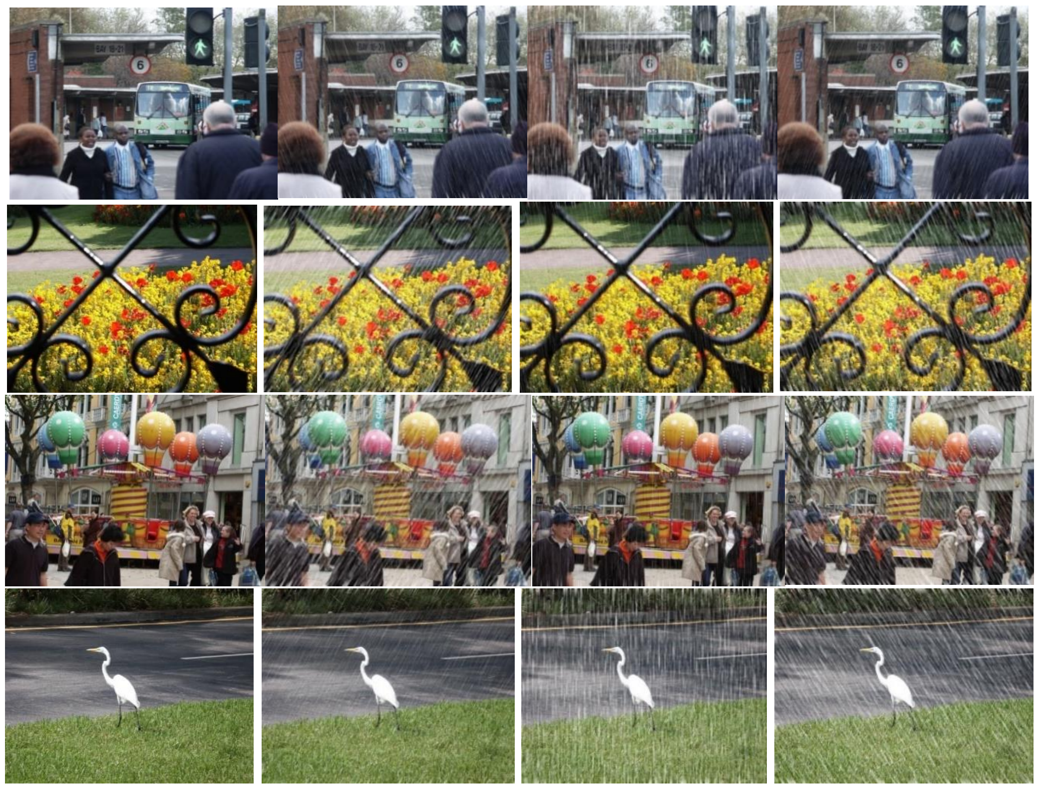

(a) (b) (c) (d)

Figure 4.13. Image samples from the rainy image dataset [4]. (a) Ground truth. (b), (c) and (d) rainy images with different rain streaks (size and orientation).

4.6. METRICS AND EVALUATION

In this stage, all models built by (YOLOv3, YOLOv3-tiny, and rain removal algorithms) are evaluated based on the test dataset with some algorithms in the same task, which is an approach that has not been followed before in any earlier research. Explanation of classification, which are more than 20 categories for evaluation, and they represent important components in any scientific experiment (Accuracy, precision, recall, f1-score, Confusion matrix ...etc.).

The following is the meaning of each frame for evaluation purposes:

True Positive (TP): The model correctly predicts the presence of an object.
False Positive (FP): Refers to a model that predicts the presence of an object but is inaccurate.

True Negative (TN): This is the model's prediction of no object, which is right.
False Negative (FN): This is the model's prediction of no object, which is erroneous.

The accuracy of an object identification model is influenced by the quality and quantity of training samples, the input images, model parameters, and the accuracy requirement threshold.

The total correct prediction (True Positives + True Negatives) is divided by the total number of predictions obtained from the ground truth. Which will result in one value is known as accuracy.

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{Total No of Objects}} \tag{4.7}$$

Precision is calculated by dividing the number of true positives by the total number of positive predictions.

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \tag{4.8}$$

Recall is calculated as the quantity of true positives divided by the total number of true (relevant) items.

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \tag{4.9}$$

The accuracy and recall are weighted in the F1 score. The values range from 0 to 1, with 1 indicating the greatest level of precision.

$$\text{F-score} = \frac{2*\text{Precision}*\text{Recall}}{\text{Recall+Precision}} \tag{4.10}$$

PART 5

RESULTS AND DISCUSSION

In this section, we present the implementation details of our two-stage approach, then the experimental results for each model are presented first separately than as a single model. Finally, we test our model on some real rain images for rain removable and object detection.

YOLO is an open-source and publicly available network for research purposes and could be implemented in different ways like (Rain conditions, Object detection API, and Image-AI). Now we will review three frameworks/libraries that can be used for real-time object detection (De-raining with YOLOv3) including what we used for this project.

The original algorithm of YOLO by its author is presented in Darknet [74]. Darknet is an open-source framework that contains a bunch of state-of-the-art neural network approaches. It contains models for classification (ImageNet classification), object detection (YOLO), and many other algorithms. It could be used for training and testing.

The object detection algorithm presented by Huang et al. [21] is available within Tensor-flow Object Detection API [98], an open-source library that contains dozens of other approaches. The object detection API for python needs to be downloaded [98] and installed. Object detection API could be used for training and testing also.

In this research Image-AI [99] was used. Image-AI is also an open-source Python library for deep learning and computer vision. This library contains several approaches for image object detection, video object detection, Custom image recognition, and inference, and custom object detection training and inference (YOLOv3 and YOLOv3-

tiny models not available with de-raining (DDN)). Image AI was used for testing using a pre-trained model only because it does not have a training option for YOLO or YOLO-tiny but since we had no intention to train the algorithm all over again this was not a problem for us.

All the mentioned implementation requires Tensor-flow and OpenCV, although OpenCV is an option for some models, it will be a good addition for visualization. Also, all the frameworks/libraries are having CPU and GPU options.

To evaluate our combined image enhancement and object detection model (DDN+YOLOv3), we tested the approach on images chosen from two datasets. Both of them are rain removal datasets, the first rainy image dataset [4], the second YTVOS201 dataset [86]. Since both datasets are synthesized and designed for rain removal application not all the images are suitable or even contain objects to be detected, therefore, the criteria of choosing images are based on the existence or not of objects to be detected by the object detection algorithm. Also, some images contain an object which does not exist in COCO, therefore, cannot be detected or classified, so we remove them too. Some examples of selected and some examples of removed images from both datasets are shown in Figures 5.1 and 5.2.

Figure 5.1 Example of unsuitable images from rainy image dataset [4].

Figure 5.2 Example of unsuitable images from YTVOS201 image dataset [86].

5.1. OBJECT DETECTION RESULTS IN CLEAR WEATHER

The YOLOv3 and YOLOv3-tiny are a network to detect over 80 diverse object classes. The detection is the fastest in terms of the algorithm in comparison with low precision to different algorithms. When comparing YOLOv3-tiny and YOLOv3, the first is faster than the second but the detection accuracy is lower. But since real-time application mostly prefers fast algorithm on the cost of detection accuracy, YOLOv3-tiny is used frequently. The detection accuracy of YOLOv3-tiny and YOLOv3 presented in Table 5.1 tested on 70 objects. We can see that YOLOv3 with a detection accuracy of 91.4% is far beyond YOLOv3-tiny with a detection accuracy of 55.7%. Generally, there is no precise object detection for complex scenarios. In a road scene, for example, the pedestrian object is simply undetectable [97]. Therefore, the algorithm did not work very well on the rainy image or the other dataset because most of the image existing is for pedestrian. The results of the testing dataset are shown in Figures 5.3 and Table 5.1. The original images appear in the first column, the second column is the detection objects using YOLOv3, and the third column is the results of YOLO-tiny, these are performed on images in clear weather.

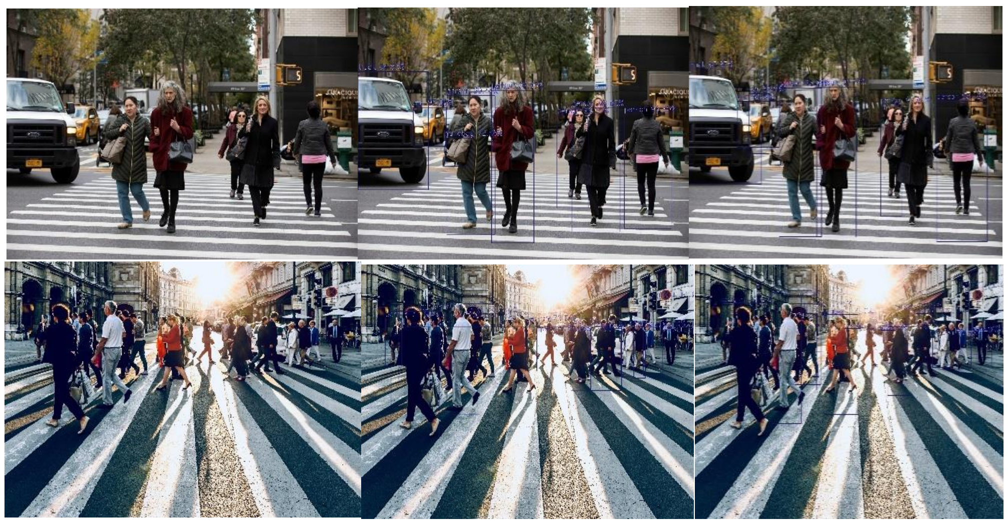


Figure 5.3 Results of YOLOv3 and YOLOv3-tiny.

Table 5.1. Accuracy evaluation of YOLOv3 and YOLOv3-tiny.

Dataset	YOLO	YOLO-tiny
Dataset 1	91.4%	55.7%

Table 5.1 and Figure 5.3 shows that the evaluation of performance using YOLOv3 is more accurate and effective than using YOLOv3-tiny.

5.2. DE-RAINING (DDN) RESULTS

The results of the modified method of de-raining image and real video are tested on two datasets [4][86], on the YTVOS201 dataset the experimental results are shown in Figure 5.4, and on the rainy image, dataset results are shown in Figure 5.5.

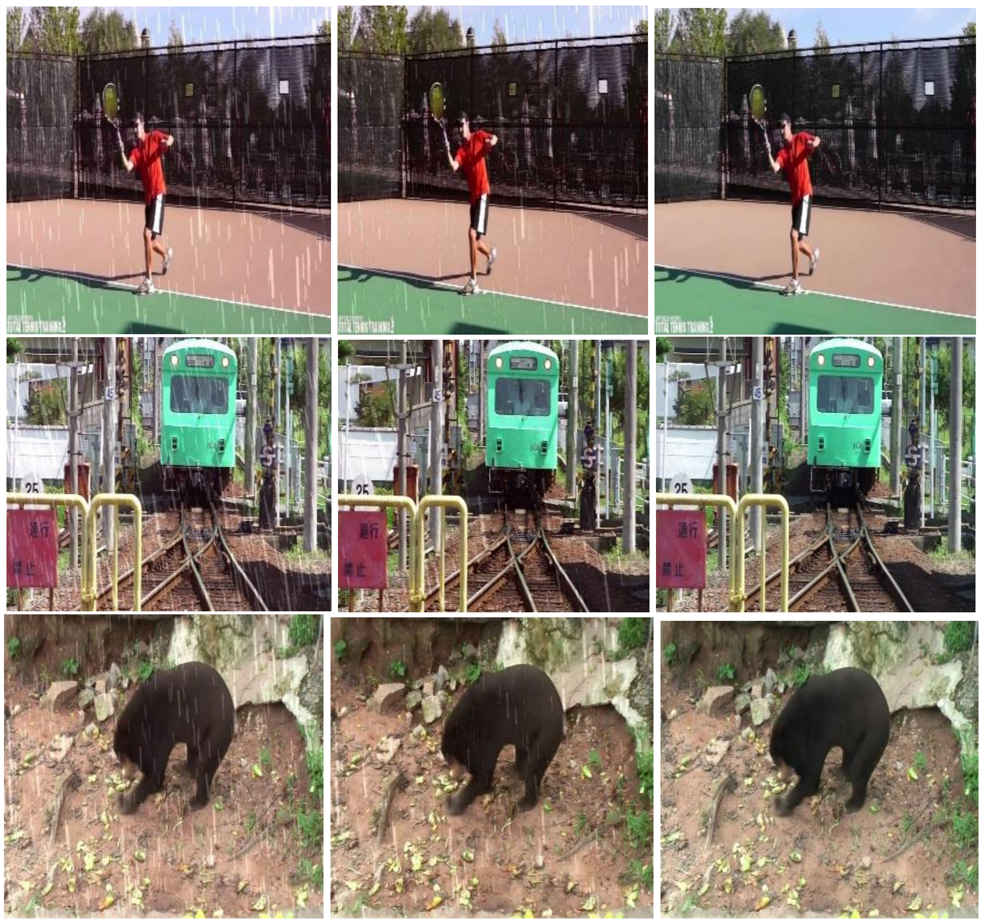


Figure 5.4. Results of the de-raining algorithm on the YTVOS201 dataset.

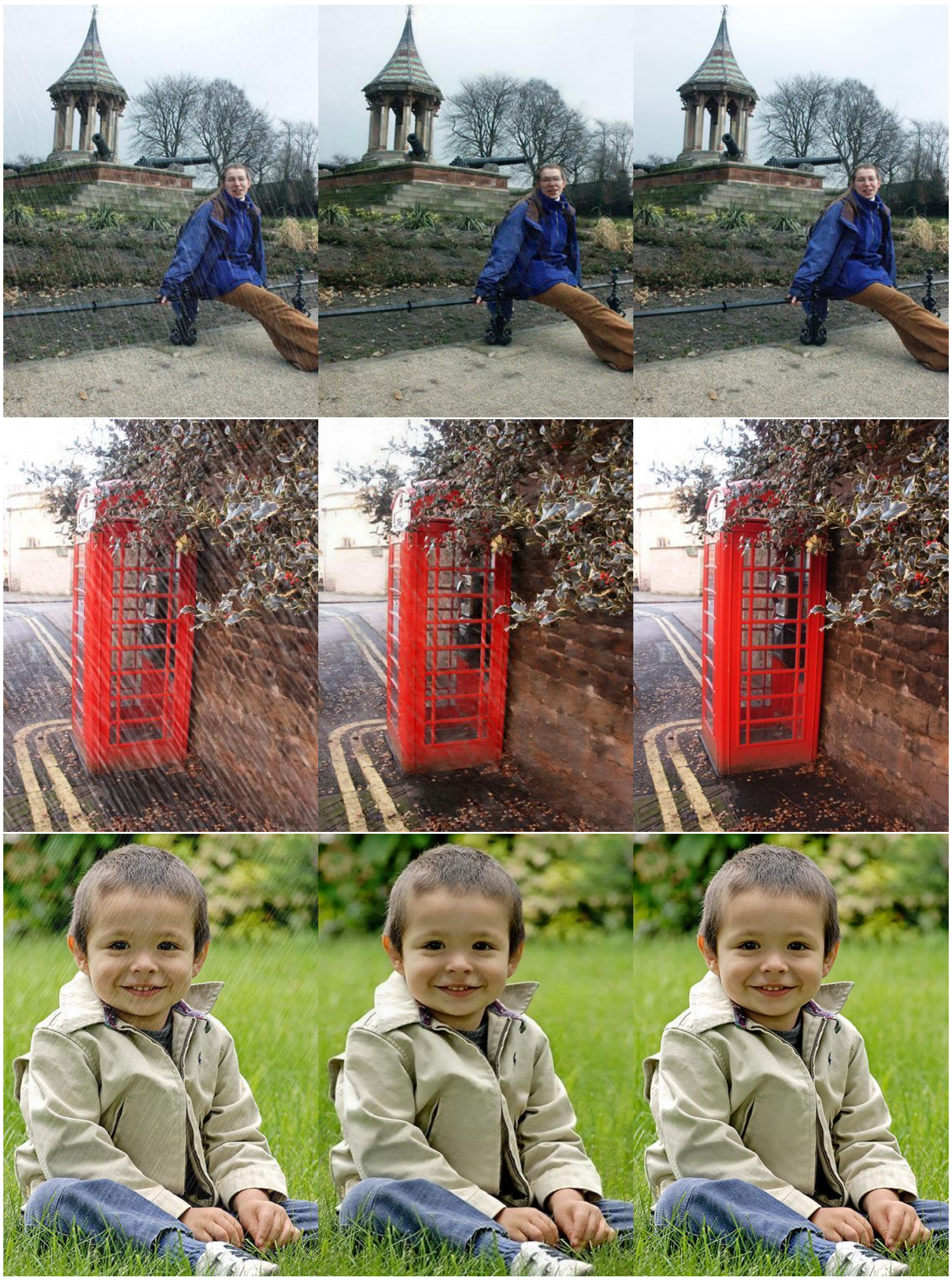


Figure 5.5. Results of the de-raining algorithm on a rainy image dataset.

Standard criteria for evaluating image quality are PSNR and SSIM. PSNR represents the noise ratio to measure the quality of the image after modification. SSIM represents an indicator for assessing image quality from three aspects: brightness, contrast and structure. SSIM values are between [0,1] the higher their value to 1 the less distortion[100]. Table 5.2 demonstrates on synthetic images, quantitative results of all competing approaches from Figures 5.4 and 5.5. From the table, we can conclude that the method that we used for rain removal is more effective, each dataset represents a set of the images.

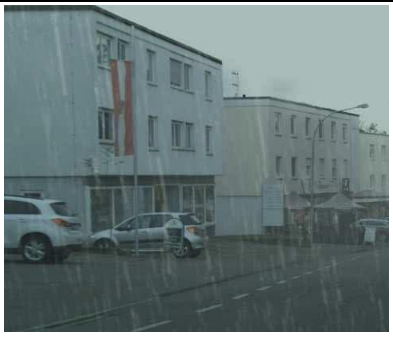
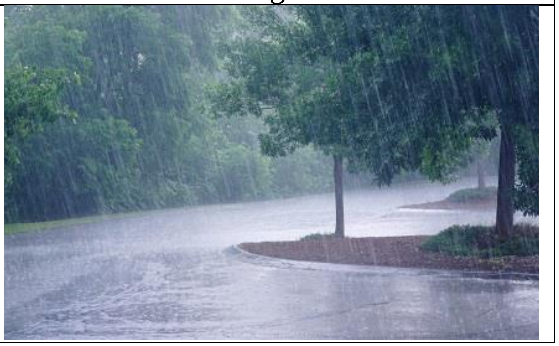
Table 5.2. Evaluation of the performance of PSNR and SSIM on synthetic images.

Images	PSNR	SSIM	RAIN REMOVED %
Dataset 1	**16.25**	**0.912**	**95 %**
Dataset 2	**15.68**	**0.959**	**96.6 %**

The results of rain removal on both datasets, it is obvious that the model succeeds in removing the rain streaks from the rainy image dataset while some rain streaks are not removed completely or at all YTVOS201 datasets. The main reason could be that the rain streaks are bigger and not realistic in YTVOS201 dataset images, and this led to leaving some rain or sometimes shadows or marks in the images.
The comparative findings with alternative rain removal technologies from rainy image are shown in Table 5.3 and Figure 5.6.

Table 5.3. Comparisons with other methods de-rain using PSNR and SSIM.

Method	PSNR	SSIM
DCPDN [101]	18.588	0.8329
DAF-Net [102]	21.063	0.8713
DID-MDN [103]	18.863	0.8331
RESCAN [104]	18.238	0.8288
De-rain (DDN)	31.374	0.9437

Methods	Image 1	Image 2
Input Image		

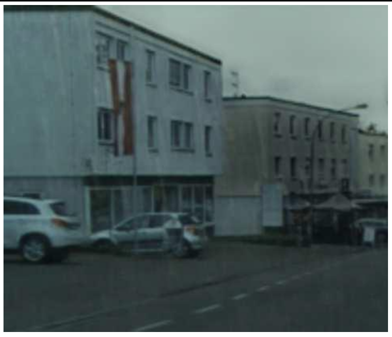
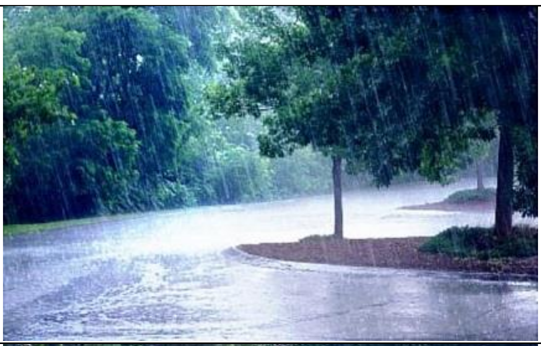
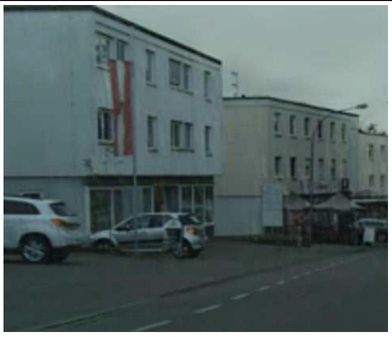
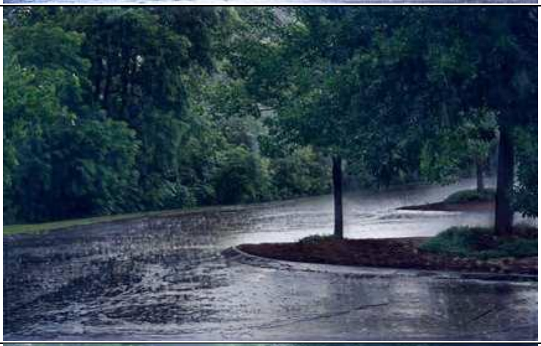
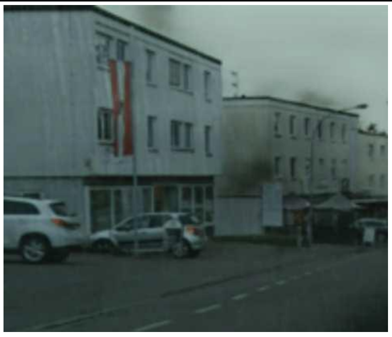
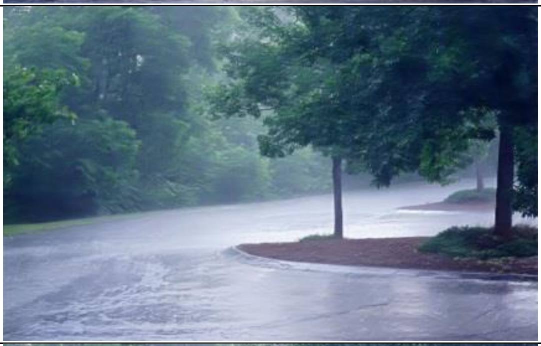
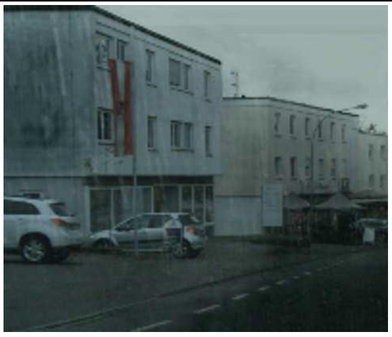
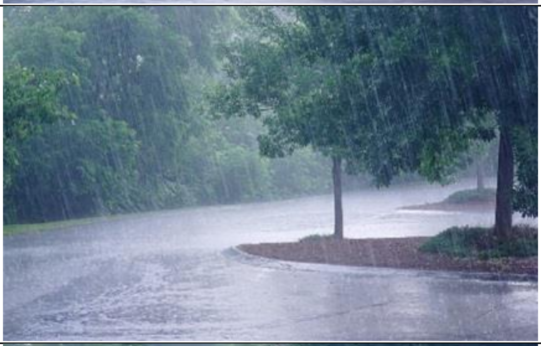
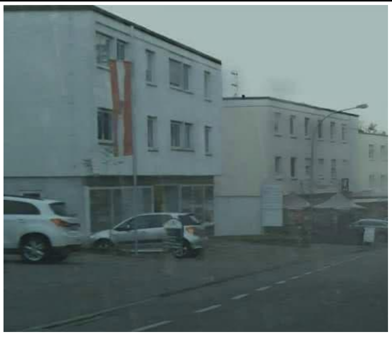
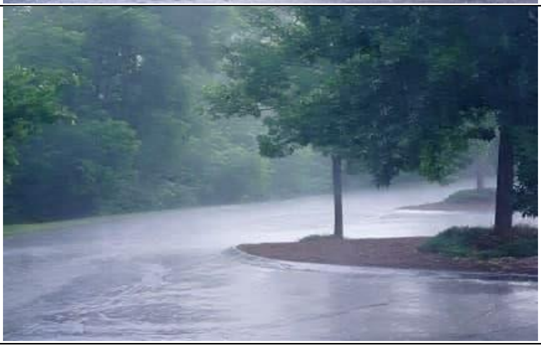
DCPDN [101]		
DAF-Net [102]		
DID-MDN [103]		
RESCAN [104]		
DE-RAINING (DDN)		

Figure 5.6. Results of removing rain using different methods and DDN.

From Table 5.3 and Figure 5.6, we can recognize that our results are clearer than other methods (the rate of error is less than all methods that are compared with), where PSNR is 31.374 and SSIM is 0.9437. The result of other methods collecting some of the rain and losing the image quality and the contrast of the image.

5.3. DE-RAINING AND OBJECT DETECTION RESULTS

We have used two different datasets of test images that have different objects in size and location. The first dataset is Rainy image [4] which having images for testing, 53 images were selected to display the calculated values (116 objects), and second datasets is called YTVOS201, which having 13 images (33 objects), this images that we used to find required comparison values. After we evaluated each section separately (object detection before and after de-raining). In Table 5.4, On the test dataset, the number of true positives, true negatives, false positives, and false negatives achieved by the models has been provided.

Table 5.4. The Calculated Values of the Object Detection Algorithms before De-raining and our method.

DATASET	**No. of Objects**	**YOLOv3- tiny**			**YOLOv3**			**Our method (DDN+YOLOv3)**		
		True Positives	**False Positives**	**False Negatives**	**True Positives**	**False Positives**	**False Negatives**	**True Positives**	**False Positives**	**False Negatives**
Rainy Images Dataset	**116**	46	45	15	90	11	15	**105**	**8**	**3**
YTVOS201 Dataset	**33**	14	13	6	20	10	3	**27**	**4**	**2**

From the values presented in Table 5.4, we get the evaluation results on each dataset presented in form of the Precision, Recall, and F1 scores. We can see that the average object detection results of our methodology (DDN+YOLOV3) are more accurate than other methods such as YOLOv3 and YOLOv3-tiny before de-raining as expected. Therefore, the results after preprocessing are more accurate in detecting objects than before preprocessing. The results on each dataset are presented in form of the

Precision, Recall, and F1 scores before de-raining and (DDN+YOLOv3) as in Figure 5.7.

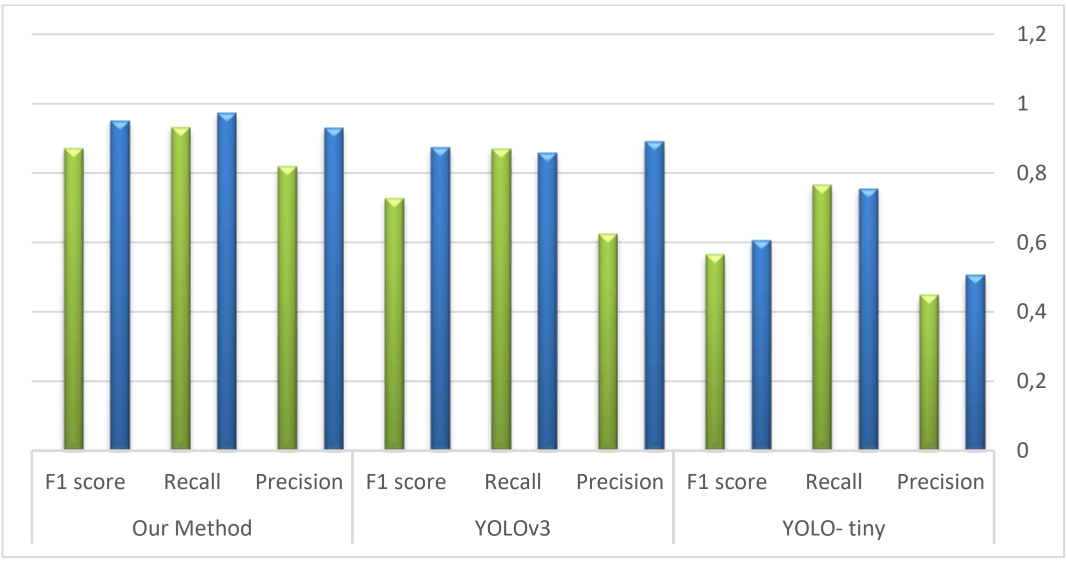


Figure 5.7. The evaluation results for the performance of object detection algorithms on Rainy Images and YTVOS201 datasets.

From the values presented in Figure 5.7, the Precision, Recall, and F1 score of the models have been calculated to be 0.5054, 0.7540, and 0.6051 on Rainy Images dataset, and on YTVOS201 Dataset 0.4482, 0.7647, and 0.5651 respectively for YOLO-tiny, and YOLO has been calculated to be 0.8910, 0.8571, and 0.8737 on Rainy Images dataset, and on YTVOS201 Dataset 0.6250, 0.8695, and 0.7272 respectively, for our methodology (DDN+YOLOv3) has been calculated on Rainy Images dataset 0.9292, 0.9722, and 0.9502, and on YTVOS201 Dataset 0.8181, 0.9310, and 0.8709 respectively. We can recognize that the object detection of the Rainy Image dataset is better than the YTVOS201 Dataset, and as much as the results of our methodology are more accurate than others, which indicates the proposed methodology is more useful to use.

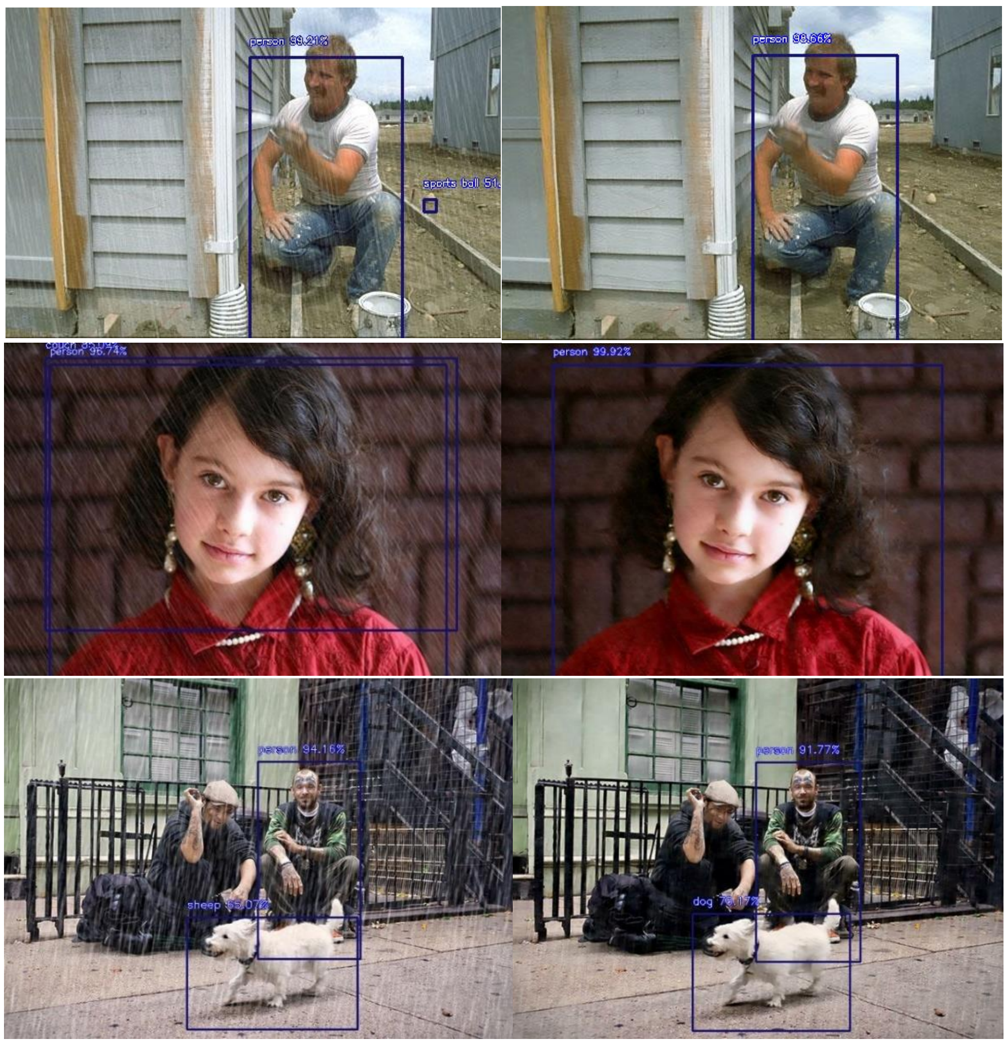
The reason for our method's (DDN+YOLOv3) higher score of object detection in rainy conditions is due to its architecture, identifies objects at three scales, making (DDN+YOLOv3) more efficient in recognizing tiny objects or objects in tough settings such as partially visible objects in a frame. Because the items in some instances are on the far side of the scaled site, they look smaller. As a result, the (DDN

+ YOLOv3) model, as shown in Figure 5.7, is very accurate in real-time detection after de-raining images and video.

5.3.1. Qualitative Comparison of the proposed method

The Rainy Images dataset is more suitable for object detection than YTVOS201dataset in terms of the variety of objects that exist in images, but the synthesized rain streaks are extremely and abnormally big. Therefore, the result of the de-raining algorithm is not successful as the results of the rainy image dataset or even as real rain images as shown in Figure 5.7. As we can see from images the residual rain streaks in images lead to disorder in the borders of the object or even blur some objects, eventually the object detection algorithm fails to detect some objects. Also, this dataset was constructed by extracting frames from YouTube videos, because of motion these images (frames) are not always clear which affects the enhancement and detection accuracy.

The Rainy image dataset although has a very good performance in quality enhancement, the general concept of the dataset is not about object detection but since we could not find any joint dataset for rainy image and object detection, we choose this one. In Figure 5.8 some examples of motion effects on images and objects.



(a) Rainy images (b) Rain removed

Figure 5.8. Examples from datasets [4][86] show the movement of false detected objects after the rain removal algorithm applied to the images.

The results are very promising for the optimization algorithm. Although the true detection rates are not very satisfying the improvement in the true and false detection rates before and after rain is remarkable.

Eventually, the enhancement (de-raining) algorithm's most important property is removing uncertainty from detection, which means quite a high number of false detected or false classified objects after the rain removal algorithm become true detected or classified as seen in Figure 5.8. Even though it is not affected or seen in Table 5.4, there was a high number of objects were detected with high detection rates (more certainty). This means the de-raining algorithm not only removed uncertainty from false detections but also emphasized or detected and classified the true objects with higher confidence as shown in Figure 5.9, the true detection on the rainy image is 97.33 and on removed rain is 99.92.

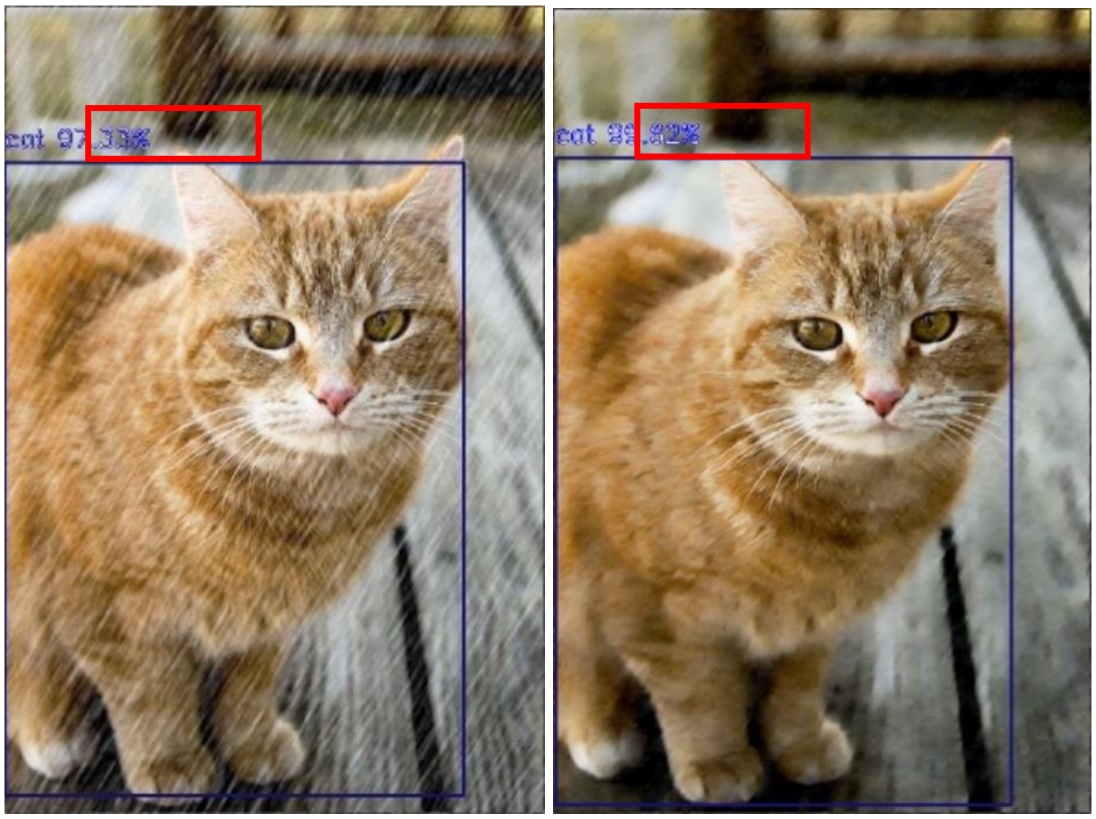


Figure 5.9. The image sample from the rainy image dataset shows the improvement in the detection rate after the rain removal algorithm was applied.

Finally, because we don't have a real rainy image dataset and want to see the results on real rainy images, we tested our model on some real rainy images, and the results are shown in Figure 5.10, where the first column shows the original images and the second column shows the images using object detection without rain removal (using only YOLOv3 model), and Figure 5.11, which the first column represents the image after preprocessing(removing rain to increase the accuracy of object detection) and the second column indicates to processed images using the modified methodology (DDN+YOLOV3) for object detection.

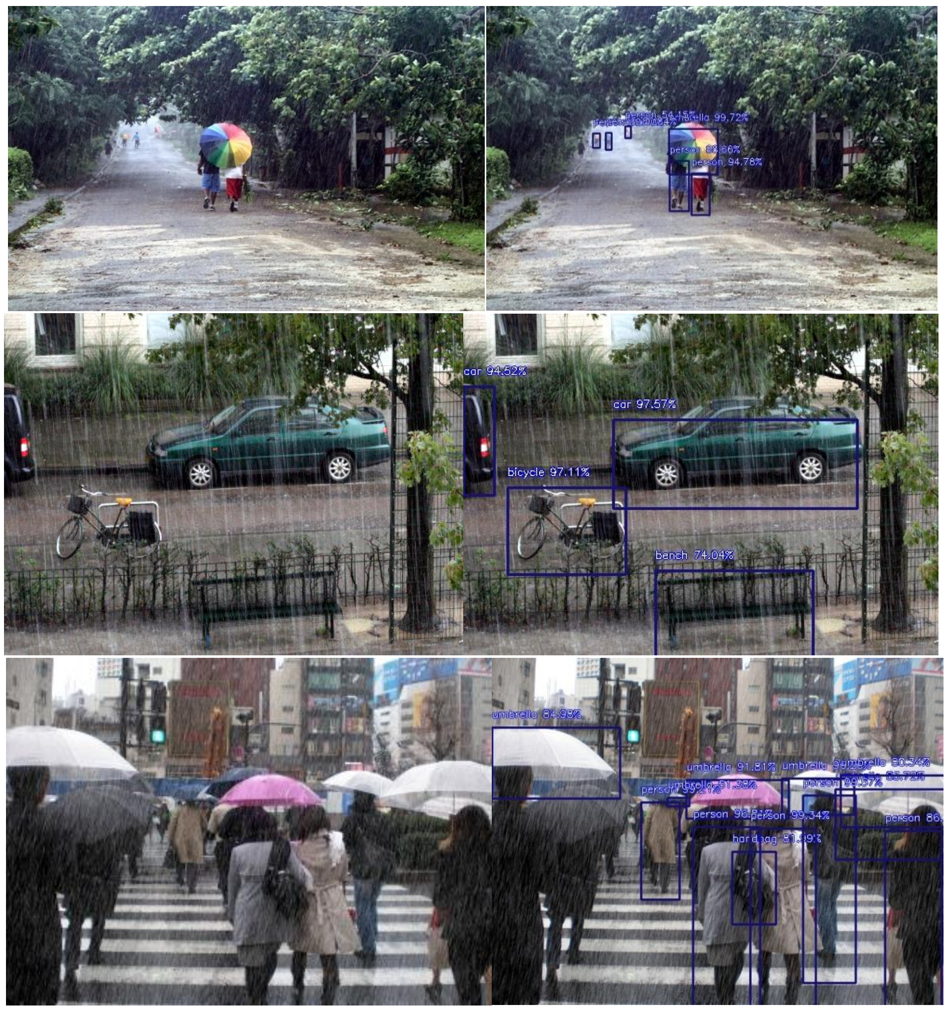


Figure 5.10. Experimental results of YOLOv3 model on real rainy images.

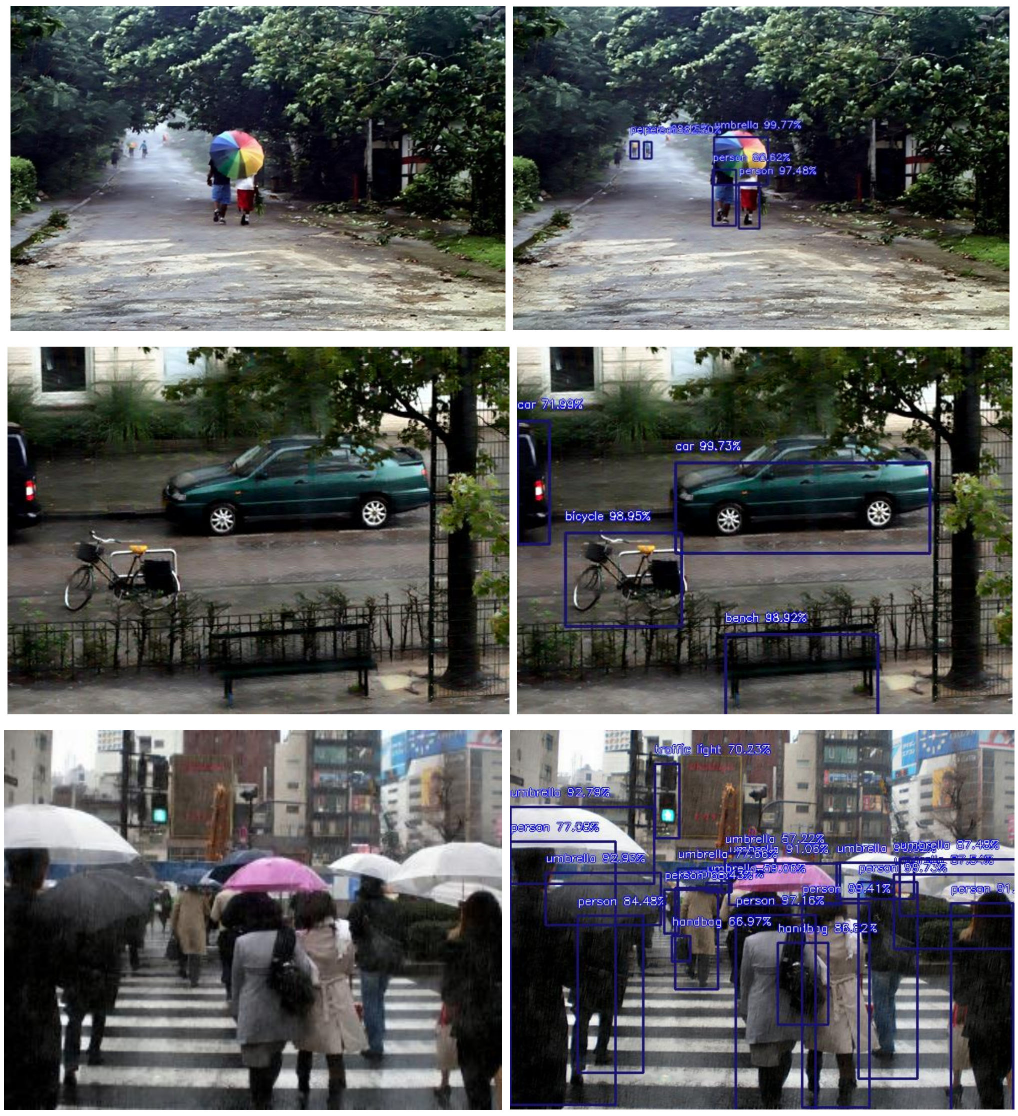


Figure 5.11. Experimental results of our method (DDN+YOLOv3) on real rainy images.

From Figures 5.10 and 5.11 can be seen the developed methodology has more accuracy for object features, reducing the false object detection, detecting objects even under rain conditions.

Figure 5.12 and Figure 5.13, we have applied our methodology (DDN+YOLOV3) on both synthetic and real-world rain dataset, which compete with other modern methods and with various levels of rain (Light, Medium and Heavy rain).

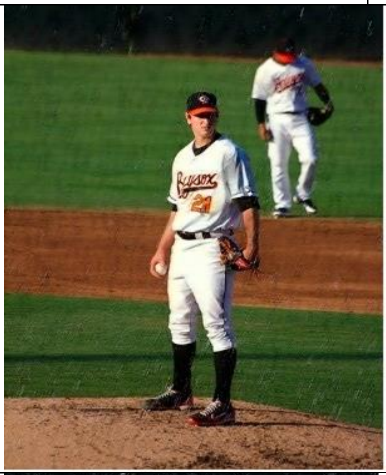
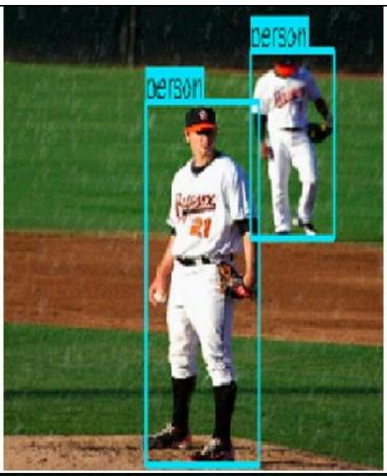
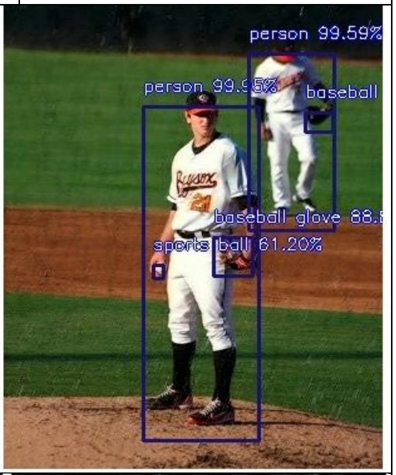
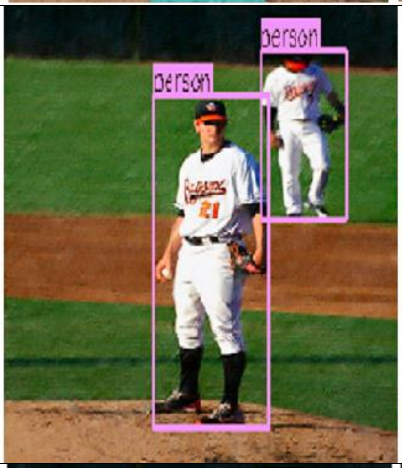
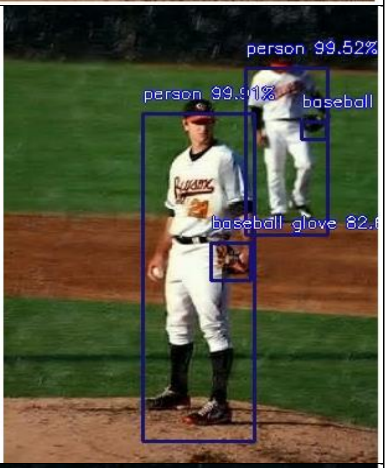
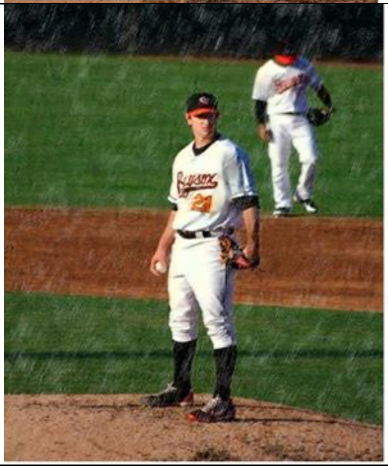
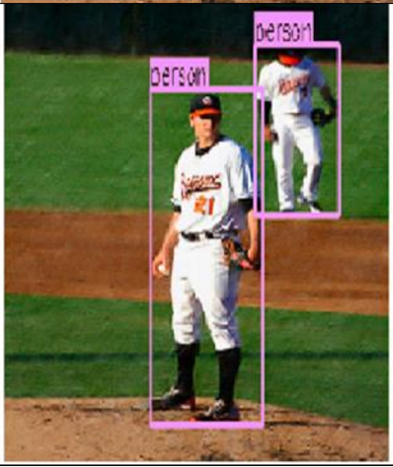
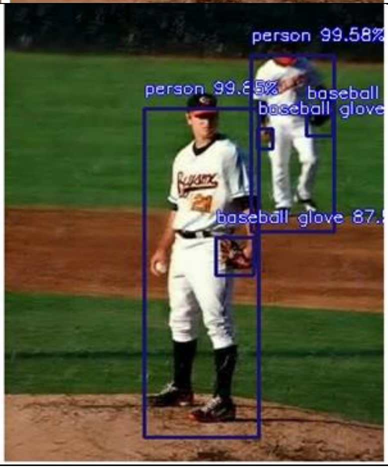
	Input Images	**Pedestrian detection mode** [43]	**Our Method**
Light Rain			 person 99.59% person 99.95% baseball baseball glove 88. sports ball 61.20%
Medium Rain			
Heavy Rain			 person 99.58% person 99.85% baseball baseball glove baseball glove 87.

Figure. 5.12. Qualitative comparison of our method with other modern method on the synthetic rain images dataset.

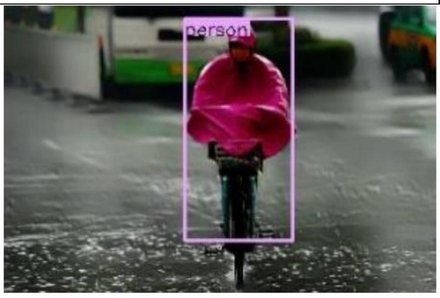
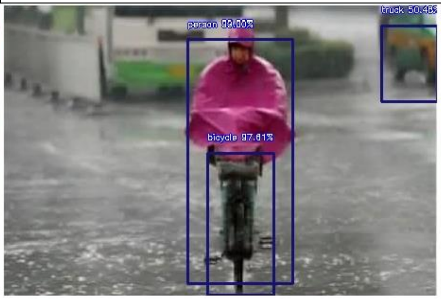
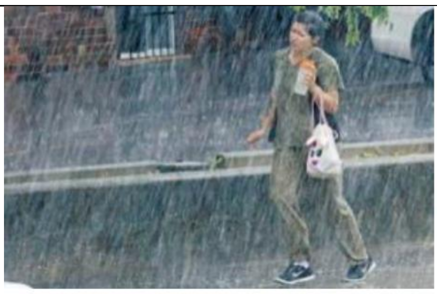
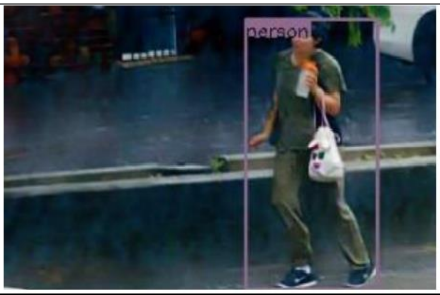
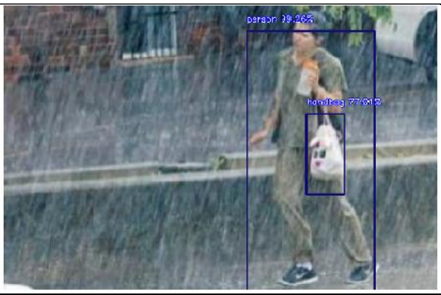
Input Images	**Pedestrian detection mode** [43]	**Our Method**
		
		

Figure. 5.13. Comparison of our method with other modern method on real-world rainy images.

Figures 5.12 and 5.13 demonstrate that the proposed methodology recovers the original image's identification effectiveness. The presented approach improves average object identification in light, medium, and severe rain conditions. The findings further demonstrate the importance of the de-raining module. Even after testing pedestrian datasets on rainfall events, previous systems were unable to properly recognize pedestrians. On rainy days, on the other hand, our approach can correctly identify pedestrians.

5.3.2. Accuracy

The accuracy of the models for the predicated objects has been calculated to be 77.58% for YOLOv3 and 90.51% for the presented model (DDN with YOLOv3) on rainy datasets, while on the YTVOS201 Dataset the models' accuracy has been determined to be 60.60% for YOLOv3 and 81.81% for the presented model (DDN with YOLOv3). Shawn in Figure.5.14.

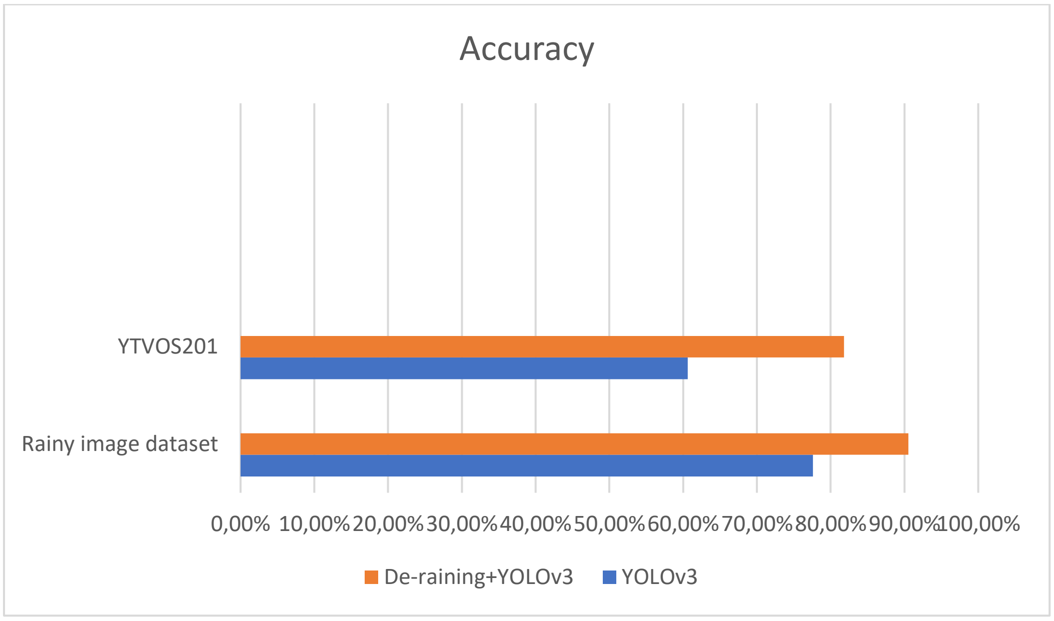


Figure 5.14. Diagram of the results of the accuracy of object detection on two datasets before de-raining and (DDN+YOLOv3).

5.4. DISCUSSION

The object detection system has achieved promising results in several different fields. In terms of the difference in performance accuracy high accuracy and performance in detecting large and small objects in different locations under normal conditions and different rain conditions (light, medium, and heavy) and using a diverse and common data set with previous works [103][104][43] and in all prescribed conditions.

The standard of accuracy and image quality was compared in clear weather and rain condition, also object detection when de-rain, this our system (DDN+YOLOv3) is more accurate and has higher quality than comparable works. YOLOv3 and tiny YOLOv3 were also compared, YOLOv3 is more accurate in detecting and YOLO-tiny is less accurate in object detecting in clear and rainy weather.

PART 6

CONCLUSION AND FUTURE WORK

The suggested model based on deep learning for rain removal in images or video to increase the quality of image, object detection with high accuracy and reduce the rate of false under rain conditions. the accuracy 90.51% for our method (DDN+YOLOv3) on rainy datasets, while on the YTVOS201 Dataset the accuracy 81.81% for (DDN+YOLOv3). According to the findings, the combined methodology De-raining (DDN) with object detection (YOLOv3) showed the best performance to detect objects under rain conditions and suit of many applications such as the safety camera and self-drive cars. This work mainly deals with all sizes objects, locations of objects and different rain conditions. The types of objects are limited to a great extent. This method is based on the training of the algorithm to detect the objects to be detected. So, increase training objects categories is very critical in this method. The training on many different classifications of objects provides extra objects types for detection and recognition.

We suggest the following in the future work:

- Training the model with a traffic sign and traffic light dataset beside the COCO dataset. Since the existing model has only three classes for traffic (traffic light, stop sign, and parking meter) and for visually impaired persons traffic is a big issue so we need to recognize more traffic signs and the color of the traffic light (red, green, and yellow). For this purpose, there are traffic sign and traffic light datasets that could be used like the LISA traffic sign dataset which contains 47 sign types and 6610 frames with 7855 annotations, or Traffic Signs Dataset which contains There are almost 20,000 images, however only 20% of them are labeled.

- Developing our method to be used in several important applications in our daily lives, such as security cameras, self-driving cars, and other obstacle detection systems.
- Designing a voice system that will convert program outcomes (object location and class) voice orders and direct the person through the way to avoid the obstacles.
- Connect the system to a weather application where we can know if it’s raining or not if it is raining activate the de-raining algorithm before object recognition.

REFERENCES

1. C. Peng *et al.*, “Megdet: A large mini-batch object detector,” ***Proceedings of the IEEE conference on Computer Vision and Pattern Recognition***. pp. 6181–6189, 2018.

2. G. Lewis, “Object detection for autonomous vehicles.” ***Stanford University, Stanford, CA***, 2014.

3. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection.” [Online]. Available: ***http://pjreddie.com/yolo***

4. X. Fu, J. Huang, D. Zeng, Y. Huang, X. Ding, and J. Paisley, “Removing rain from single images via a deep detail network,” ***in Proceedings of the IEEE conference on computer vision and pattern recognition***, 2017, pp. 3855–3863.

5. H. Wang, Q. Xie, Q. Zhao, and D. Meng, “A model-driven deep neural network for single image rain removal,” ***in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition***, 2020, pp. 3103–3112.

6. Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, “Deepface: Closing the gap to human-level performance in face verification,” ***in Proceedings of the IEEE conference on computer vision and pattern recognition***, 2014, pp. 1701–1708.

7. R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” ***in Proceedings of the IEEE conference on computer vision and pattern recognition***, 2014, pp. 580–587.

8. L. Liu et al., “Deep learning for generic object detection: A survey,” ***Int. J. Comput. Vis***., vol. 128, no. 2, pp. 261–318, 2020.

9 W. Ouyang and X. Wang, “Joint deep learning for pedestrian detection,” ***in Proceedings of the IEEE international conference on computer vision***, 2013, pp. 2056–2063.

10. S. C. H. Hoi et al., “Logo-net: Large-scale deep logo detection and brand recognition with deep region-based convolutional networks,” ***arXiv Prepr. arXiv1511.02462, 2015.***

11. K. Kang et al., “T-cnn: Tubelets with convolutional neural networks for object detection from videos,” ***IEEE Trans. Circuits Syst. Video Technol.,*** vol. 28,

no. 10, pp. 2896–2907, 2017.

12. K. Kang, W. Ouyang, H. Li, and X. Wang, “Object detection from video tubelets with convolutional neural networks,” ***in Proceedings of the IEEE conference on computer vision and pattern recognition***, 2016, pp. 817–825.

13. X. Chen, S. Xiang, C.-L. Liu, and C.-H. Pan, “Vehicle detection in satellite images by parallel deep convolutional neural networks,” ***in 2013 2nd IAPR Asian conference on pattern recognition,*** 2013, pp. 181–185.

14. Q. Fan, L. Brown, and J. Smith, “A closer look at Faster R-CNN for vehicle detection,” ***in 2016 IEEE intelligent vehicles symposium (IV),*** 2016, pp. 124–129.

15. W. Lotter, G. Kreiman, and D. Cox, “Deep predictive coding networks for video prediction and unsupervised learning,” ***arXiv Prepr. arXiv1605.08104, 2016.***

16. Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” ***Nature, vol. 521, no. 7553, pp. 436–444, 2015.***

17. P. Felzenszwalb, D. McAllester, and D. Ramanan, “A discriminatively trained, multiscale, deformable part model,” ***in 2008 IEEE conference on computer vision and pattern recognition,*** 2008, pp. 1–8.

18. L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, “Fully-convolutional siamese networks for object tracking,” ***in European conference on computer vision,*** 2016, pp. 850–865.

19. J. Redmon and A. Farhadi, “YOLO9000: better, faster, stronger,” ***in Proceedings of the IEEE conference on computer vision and pattern recognition***, 2017, pp. 7263–7271.

20. Y. Xiao et al., “A review of object detection based on deep learning,” ***Multimed. Tools Appl.***, vol. 79, no. 33, pp. 23729–23791, 2020.

21. J. Huang et al., “Speed/accuracy trade-offs for modern convolutional object detectors,” ***in Proceedings of the IEEE conference on computer vision and pattern recognition,*** 2017, pp. 7310–7311.

22. Z. Li, C. Peng, G. Yu, X. Zhang, Y. Deng, and J. Sun, “Light-head r-cnn: In defense of two-stage object detector***,” arXiv Prepr. arXiv1711.07264, 2017.***

23. D. M. Montserrat, Q. Lin, J. Allebach, and E. J. Delp, “Training object detection and recognition CNN models using data augmentation,” ***Electron. Imaging***, vol. 2017, no. 10, pp. 27–36, 2017.

24. L. Yang, L. Wang, and S. Wu, “Real-time object recognition algorithm based on deep convolutional neural network,” ***in 2018 IEEE 3rd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)***, 2018,

pp. 331–335.

25. A. Arcos-Garcia, M. Soilan, J. A. Alvarez-Garcia, and B. Riveiro, "Exploiting synergies of mobile mapping sensors and deep learning for traffic sign recognition systems," ***Expert Syst. Appl.***, vol. 89, pp. 286–295, 2017.
26. K. Zhou, Y. Zhan, and D. Fu, "Learning region-based attention network for traffic sign recognition," ***Sensors,*** vol. 21, no. 3, p. 686, 2021.

27. S. K. Chadalawada, "Real time detection and recognition of construction vehicles: using deep learning methods." 2020.

28. J. Yim and K.-A. Sohn, "Enhancing the performance of convolutional neural networks on quality degraded datasets," ***in 2017 International Conference on Digital Image Computing: Techniques and Applications (DICTA)***, 2017, pp. 1–8.

29. Y. Tian, K. Pei, S. Jana, and B. Ray, "Deeptest: Automated testing of deep-neural-network-driven autonomous cars," ***in Proceedings of the 40th international conference on software engineering,*** 2018, pp. 303–314.

30. T. Dreossi, S. Ghosh, X. Yue, K. Keutzer, A. Sangiovanni-Vincentelli, and S. A. Seshia, "Counterexample-guided data augmentation," ***arXiv Prepr. arXiv1805.06962, 2018.***

31. D. Maturana and S. Scherer, "Voxnet: A 3d convolutional neural network for real-time object recognition," ***in 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),*** 2015, pp. 922–928.

32. S. Zheng, Y. Song, T. Leung, and I. Goodfellow, "Improving the robustness of deep neural networks via stability training," ***in Proceedings of the ieee conference on computer vision and pattern recognition***, 2016, pp. 4480–4488.

33. R. Girshick, "Fast r-cnn," ***in Proceedings of the IEEE international conference on computer vision***, 2015, pp. 1440–1448.

34. K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," i***n Proceedings of the IEEE international conference on computer vision,*** 2017, pp. 2961–2969.
35. Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei, "Fully convolutional instance-aware semantic segmentation," i***n Proceedings of the IEEE conference on computer vision and pattern recognition,*** 2017, pp. 2359–2367.

36. A. Shrivastava, R. Sukthankar, J. Malik, and A. Gupta, "Beyond skip connections: Top-down modulation for object detection," ***arXiv Prepr. arXiv1612.06851, 2016.***

37. T. Ahmad, Y. Ma, M. Yahya, B. Ahmad, and S. Nazir, "Object detection through modified YOLO neural network," ***Sci. Program.***, vol. 2020, 2020.

38. K. Pei, Y. Cao, J. Yang, and S. Jana, “Deepxplore: Automated whitebox testing of deep learning systems,” ***in proceedings of the 26th Symposium on Operating Systems Principles***, 2017, pp. 1–18.

39. M. Hnewa and H. Radha, “Object detection under rainy conditions for autonomous vehicles: a review of state-of-the-art and emerging techniques,” ***IEEE Signal Process. Mag.***, vol. 38, no. 1, pp. 53–67, 2020.

40. R. Qian, R. T. Tan, W. Yang, J. Su, and J. Liu, “Attentive generative adversarial network for raindrop removal from a single image,” ***in Proceedings of the IEEE conference on computer vision and pattern recognition***, 2018, pp. 2482–2491.

41. J. Peng, Y. Xu, T. Chen, and Y. Huang, “Single-image raindrop removal using concurrent channel-spatial attention and long-short skip connections,” ***Pattern Recognit. Lett.,*** vol. 131, pp. 121–127, 2020.

42. T. Sharma, B. Debaque, N. Duclos, A. Chehri, B. Kinder, and P. Fortier, “Deep Learning-Based Object Detection and Scene Perception under Bad Weather Conditions,” ***Electronics,*** vol. 11, no. 4, p. 563, 2022.

43. Y. Liu, J. Ma, Y. Wang, and C. Zong, “A novel algorithm for detecting pedestrians on rainy image,” **Sensors**, vol. 21, no. 1, p. 112, 2020.

44. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” ***in Proceedings of the IEEE conference on computer vision and pattern recognition***, 2016, pp. 779–788.

45. Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos, “A unified multi-scale deep convolutional neural network for fast object detection,” i***n European conference on computer vision,*** 2016, pp. 354–370.

46. “M. Anukrati, A Comprehensive Guide to Types of Neural Networks, 2019. [Online]. Available***: www.digitalvidya.com/blog/types-of-neural-networks/.”***

47. J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” ***arXiv Prepr. arXiv1804.02767, 2018.***

48. “K. Ayoosh, What’s New in YOLO v3?, 2018. [Online]. Available: ***www. towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b.”***

49. S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” ***Adv. Neural Inf. Process. Syst.***, vol. 28, 2015.

50. W. Liu et al., “Ssd: Single shot multibox detector,” ***in European conference on computer vision,*** 2016, pp. 21–37.

51. C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, “Dssd: Deconvolutional

single shot detector," ***arXiv Prepr. arXiv1701.06659, 2017.***

52. Z. Li and F. Zhou, "FSSD: feature fusion single shot multibox detector," ***arXiv Prepr. arXiv1712.00960, 2017.***

53. Z. Shen, Z. Liu, J. Li, Y.-G. Jiang, Y. Chen, and X. Xue, "Dsod: Learning deeply supervised object detectors from scratch," i***n Proceedings of the IEEE international conference on computer vision,*** 2017, pp. 1919–1927.

54. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," A***dv. Neural Inf. Process. Syst.,*** vol. 25, 2012.

55. D. G. Lowe, "Distinctive image features from scale-invariant keypoints," I***nt. J. Comput. Vis***., vol. 60, no. 2, pp. 91–110, 2004.

56. T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," ***in Proceedings of the IEEE international conference on computer vision,*** 2017, pp. 2980–2988.

57. T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," ***in Proceedings of the IEEE conference on computer vision and pattern recognition***, 2017, pp. 2117–2125.

58. J. R. R. Uijlings, K. E. A. Van De Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," I***nt. J. Comput. Vis.***, vol. 104, no. 2, pp. 154–171, 2013, doi: 10.1007/s11263-013-0620-5.

59. I. Endres and D. Hoiem, "Category independent object proposals," ***Lect. Notes Comput. Sci.*** (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 6315 LNCS, no. PART 5, pp. 575–588, 2010, doi: 10.1007/978-3-642-15555-0_42.

60. C. L. Zitnick and P. Dollár, "Edge boxes," Lect. Notes Comput. ***Sci.*** (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 8693 LNCS, no. PART 5, pp. 391–405, 2014.

61. J. Hu, "Squeeze-and-Excitation_Networks_CVPR_2018_paper.pdf," Cvpr, pp. 7132–7141, 2018, [Online]. Available: ***http://openaccess.thecvf.com/content_cvpr_2018/html/Hu_Squeeze-and-Excitation_Networks_CVPR_2018_paper.ht***ml

62. F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size," pp. 1–13, 2016, [Online]. A***vailable: http://arxiv.org/abs/1602.07360***

63. "Ioffe S, Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. In: I***nternational conference on***

machine learning (ICML), pp 448–456."

64. K. Janocha and W. M. Czarnecki, "On loss functions for deep neural networks in classification," ***Schedae Informaticae,*** vol. 25, pp. 49–59, 2016, doi: 10.4467/20838476SI.16.004.6185.

65. J. Kawahara and G. Hamarneh, "Multi-resolution-tract CNN with hybrid pretrained and skin-lesion trained layers," ***Lect. Notes Comput. Sci***. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 10019 LNCS, pp. 164–171, 2016, doi: 10.1007/978-3-319-47157-0_20.

66. B. Mladenov, L. Damiani, P. Giribone, and R. Revetria, "A short review of the SDKs and wearable devices to be used for AR application for industrial working environment," ***Lect. Notes Eng. Comput. Sci.***, vol. 2237, pp. 137–142, 2018.

67. A. Pentina, V. Sharmanska, and C. H. Lampert, "Curriculum learning of multiple tasks," ***Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.,*** vol. 07-12-June, pp. 5492–5500, 2015, doi: 10.1109/CVPR.2015.7299188.

68. H. Harzallah et al., "Combining efficient object localization and image classification To cite this version : HAL Id : inria-00439516 Combining efficient object localization and image classification," 2009.

69. D. Eigen, D. Krishnan, and R. Fergus, "Restoring an image taken through a window covered with dirt or rain," ***in Proceedings of the IEEE international conference on computer vision,*** 2013, pp. 633–640.

70. G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller, "Introduction to wordnet: An on-line lexical database," ***Int. J. Lexicogr.,*** vol. 3, no. 4, pp. 235–244, 1990, doi: 10.1093/ijl/3.4.235.

71. K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc., pp. 1–14, 2015.

72. Jia Deng, Wei Dong, R. Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," pp. 248–255, 2009, doi: 10.1109/cvprw.2009.5206848.

73. T.-Y. Lin et al., "Microsoft coco: Common objects in context," ***in European conference on computer vision***, 2014, pp. 740–755.

74. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," i***n Proceedings of the IEEE conference on computer vision and pattern recognition,*** 2016, pp. 770–778.

75. M. Kisantal, Z. Wojna, J. Murawski, J. Naruniec, and K. Cho, "Augmentation for small object detection," vol. 2017, pp. 119–133, 2019, doi:

10.5121/csit.2019.91713.

76. J. Li, X. Liang, Y. Wei, T. Xu, J. Feng, and S. Yan, "Perceptual generative adversarial networks for small object detection," P***roc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR*** 2017, vol. 2017-Janua, pp. 1951–1959, 2017, doi: 10.1109/CVPR.2017.211.

77. J. Wang, T. Zheng, P. Lei, and X. Bai, "A hierarchical convolution neural network (CNN)-based ship target detection method in spaceborne SAR imagery," ***Remote Sens***., vol. 11, no. 6, 2019, doi: 10.3390/rs11060620.

78. Y. Wei, X. You, and H. Li, "Multiscale patch-based contrast measure for small infrared target detection," Pattern Recognit., vol. 58, pp. 216–226, 2016, doi: 10.1016/j.patcog.2016.04.002.

79. L. Liangkui, W. Shaoyou, and T. Zhongxing, "Using deep learning to detect small targets in infrared oversampling images," ***J. Syst. Eng. Electron.***, vol. 29, no. 5, pp. 947–952, 2018, doi: 10.21629/JSEE.2018.05.07.

80. xuemei xie, G. Cao, W. Yang, Q. Liao, G. Shi, and J. Wu, "Feature-fused SSD: fast detection for small objects," no. April 2018, p. 236, 2018, doi: 10.1117/12.2304811.

81. M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge,***" Int. J. Comput. Vis***., vol. 88, no. 2, pp. 303–338, 2010, doi: 10.1007/s11263-009-0275-4.

82. J. Xiao, K. A. Ehinger, J. Hays, A. Torralba, and A. Oliva, "SUN Database: Exploring a Large Collection of Scene Categories,***" Int. J. Comput. Vis***., vol. 119, no. 1, pp. 3–22, 2016, doi: 10.1007/s11263-014-0748-y.

83. O. Russakovsky et al., "ImageNet Large Scale Visual Recognition Challenge," ***Int. J. Comput. Vis.***, vol. 115, no. 3, pp. 211–252, 2015, doi: 10.1007/s11263-015-0816-y.

84. G. (2016) O. a public dataset for large-scale multi-label and multi-class image classification. D. available from ***https://githubcom/openimages 2(6): 96.*** Krasin I, Duerig T, Alldrin N, Veit A, Abu-El-Haija S, Belongie S, Cai D, Feng Z, Ferrari V, "No Title."

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

86. M. Li, X. Cao, Q. Zhao, L. Zhang, and D. Meng, "Online rain/snow removal from surveillance videos," ***IEEE Trans. Image Process***., vol. 30, pp. 2029–2044, 2021.

87. K. He, J. Sun, and X. Tang, "Guided image filtering,***" IEEE Trans. Pattern***

Anal. Mach. Intell., vol. 35, no. 6, pp. 1397–1409, 2012.

88. C. Tomasi and R. Manduchi, “Bilateral filtering for gray and color images,” in Sixth international conference on computer vision ***(IEEE Cat.*** No. 98CH36271), 1998, pp. 839–846.

89. Q. Zhang, X. Shen, L. Xu, and J. Jia, “Rolling guidance filter,” i***n European conference on computer vision,*** 2014, pp. 815–830.

90. D.-A. Huang, L.-W. Kang, Y.-C. F. Wang, and C.-W. Lin, “Self-learning based image decomposition with applications to single image denoising,” ***IEEE Trans. Multimed.,*** vol. 16, no. 1, pp. 83–93, 2013.

91. L.-W. Kang, C.-W. Lin, and Y.-H. Fu, “Automatic single-image-based rain streaks removal via image decomposition,” ***IEEE Trans. image Process***., vol. 21, no. 4, pp. 1742–1755, 2011.

92. R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Region-based convolutional networks for accurate object detection and segmentation,” ***IEEE Trans. Pattern Anal. Mach. Intell.,*** vol. 38, no. 1, pp. 142–158, 2015.

93. “Introduction to YOLO Algorithm for Object Detection, ***https://www.section.io/engineering-education/introduction-to-yolo-algorithm-for-object-detection/, accessed on Aug/15/2021.”***

94. C. Szegedy et al., “Going deeper with convolutions,” ***in Proceedings of the IEEE conference on computer vision and pattern recognition***, 2015, pp. 1–9.

95. R. Lienhart and J. Maydt, “An extended set of haar-like features for rapid object detection,” i***n Proceedings. international conference on image processing***, 2002, vol. 1, pp. I–I.

96. “J. Redmon. Darknet: open-source neural networks in c. ***http://pjreddie.com/darknet***/. 2013–2016.”

97. Z. Yi, S. Yongliang, and Z. Jun, “An improved tiny-yolov3 pedestrian detection algorithm,” Optik (Stuttg)., vol. 183, pp. 17–23, 2019.

98. “Tensorflow Object Detection API. ***(https://github.com/tensorflow/models/tree/master/research/object_detection).***

99. “M. Olafenwa and J. Olafenwa. ImageAI: python library for deep learning and computer vision. ***https://imageai.readthedocs.io/en/latest/.”***

100. Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” ***IEEE Trans. image Process.,*** vol. 13, no. 4, pp. 600–612, 2004.

101. H. Zhang and V. M. Patel, “Densely connected pyramid dehazing network,” in Proceedings of the ***IEEE conference on computer vision and pattern recognition,*** 2018, pp. 3194–3203.

102. X. Hu, C.-W. Fu, L. Zhu, and P.-A. Heng, “Depth-attentional features for single-image rain removal,” i***n Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition,*** 2019, pp. 8022–8031.

103. H. Zhang and V. M. Patel, “Density-aware single image de-raining using a multi-stream dense network,” ***in Proceedings of the IEEE conference on computer vision and pattern recognition***, 2018, pp. 695–704.

104. Y. Li, R. T. Tan, X. Guo, J. Lu, and M. S. Brown, “Rain streak removal using layer priors,” ***in Proceedings of the IEEE conference on computer vision and pattern recognition,*** 2016, pp. 2736–2744.

RESUME

Faris Kareem HALYUT, completed high school education at (Othman bin Said) high school in Karbala/Iraq. then, He obtained a bachelor's degree from Iraqi university / Network Engineering in 2019.To complete their M.Sc., he moved to Karabuk/Turkey in 2020. He started his master education at the department of computer engineering in Karabuk University.