Corn Classification Based on Quality Related to The Bug/Disease Using YOLOv8

MONTHER YOUSEF ABDULWASEA MOHAMMED AL-ASBAHI

BACHELOR OF MECHATRONICS ENGINEERING WITH HONOURS
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2024

Corn Classification Based on Quality Related to The Bug/Disease Using YOLOv8

MONTHER YOUSEF ABDULWASEA MOHAMMED AL-ASBAHI

A report submitted
in partial fulfillment of the requirements for the degree of
Bachelor of Mechatronics Engineering with Honors

Faculty of Electrical Technology and Engineering

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2024

DECLARATION

I declare that this thesis entitled "Corn Classification Based on Quality Related to The Bug/Disease Using YOLOv8 "is the result of my own research except as cited in the references. The thesis has not been accepted for any degree and is not concurrently submitted in the candidature of any other degree.

Signature	:	
Name	:	MONTHER YOUSEF ABDULWASEA MOHAMMED AL-ASBAHI
Date	:	17/06/2024

APPROVAL

I hereby declare that I have checked this report entitled " Corn Classification Based on Quality Related to The Bug/Disease Using Artificial Intelligent " and in my opinion, this thesis fulfills the partial requirement to be awarded the degree of Bachelor of Mechatronics Engineering with Honors

Signature	:	
Supervisor Name	:	IR DR ZAMANI BIN MD SANI
Date	:	17/06/2024

DEDICATIONS

This thesis is dedicated to my parents and family for their constant love and support. Thank you for believing in me and for making my success possible. Your sacrifices will always be appreciated.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor, IR. Dr. Zamani Bin Md Sani for his invaluable guidance, support, and encouragement throughout this project. His expertise and knowledge have been instrumental in shaping this project. I am truly grateful for his patience, and for providing me with the opportunity to work on such an interesting and challenging project.

I would also like to extend my gratitude to my friends and university colleagues for their constant support and encouragement throughout this project. They have been a source of inspiration and motivation, and I am thankful to have them in my life.

Lastly, I would like to thank my family for their love and support. Their unwavering belief in me has been a constant source of motivation, and I am grateful for their sacrifices and dedication to my education.

ABSTRACT

Artificial intelligence (AI) breakthroughs have transformed crop monitoring and quality control techniques in the agriculture industry. This study investigates the use of convolutional neural networks (CNNs) and the YOLOv8 algorithm to enhance maize quality monitoring. Traditional maize sorting techniques are labor-intensive, time-consuming, and prone to mistakes. The incorporation of AI technology provides a solution to these difficulties. The study's goal is to create an automated system for recognizing and categorizing pest and disease-related maize quality concerns using machine learning and image recognition techniques. A CNN model was created utilizing labeled information to properly identify and categorize maize quality concerns into three groups. The model's performance was assessed using the YOLOv8 method, which is noted for its quick and accurate object identification capabilities. The training was done in the Google Colab environment, with pre-trained weights to speed up model convergence. The findings indicated significant increases in detection accuracy and efficiency. The model's overall accuracy was 92.4%, with class-specific accuracies of 88% for "Healthy," 65.5% for "Water Rot," and 100% for "Bug." The average Precision (mAP) was 92.7%, with an Intersection over Union (IoU) of 52.3%. Visual and statistical studies, such as F1-Confidence and Recall-Confidence curves, offered information about the model's performance at different confidence levels. The findings emphasize the potential for AI-powered maize quality monitoring systems to improve agricultural practices, lower labor costs, and assure consistent and accurate quality evaluation. This study demonstrates the viability of implementing advanced deep-learning algorithms in real-world agricultural settings, opening the door for future crop monitoring and quality management advances.

ABSTRAK

Penemuan kecerdasan buatan (AI) telah mengubah pemantauan tanaman dan teknik kawalan kualiti dalam industri pertanian. Kajian ini menyiasat penggunaan rangkaian neural convolutional (CNN) dan algoritma YOLOv8 untuk meningkatkan pemantauan kualiti jagung. Teknik pengisihan jagung tradisional adalah intensif buruh, memakan masa, dan terdedah kepada kesilapan. Penggabungan teknologi AI menyediakan penyelesaian kepada kesukaran ini. Matlamat kajian adalah untuk mencipta sistem automatik untuk mengiktiraf dan mengkategorikan kebimbangan kualiti jagung berkaitan perosak dan penyakit menggunakan pembelajaran mesin dan teknik pengecaman imej. Model CNN dicipta menggunakan maklumat berlabel untuk mengenal pasti dan mengkategorikan kebimbangan kualiti jagung dengan betul kepada tiga kumpulan. Prestasi model dinilai menggunakan kaedah YOLOv8, yang terkenal dengan keupayaan pengenalan objek yang cepat dan tepat. Latihan telah dilakukan dalam persekitaran Google Colab, dengan pemberat pra-latihan untuk mempercepatkan penumpuan model. Penemuan menunjukkan peningkatan ketara dalam ketepatan dan kecekapan pengesanan. Ketepatan keseluruhan model ialah 92.4%, dengan ketepatan khusus kelas sebanyak 88% untuk "Sihat", 65.5% untuk "Reput Air" dan 100% untuk "Pepijat." Purata Ketepatan (mAP) ialah 92.7%, dengan Intersection over Union (IoU) sebanyak 52.3%. Kajian visual dan statistik, seperti lengkung F1-Confidence dan Recall-Confidence, menawarkan maklumat tentang prestasi model pada tahap keyakinan yang berbeza. Penemuan ini menekankan potensi sistem pemantauan kualiti jagung yang dikuasakan AI untuk menambah baik amalan pertanian, mengurangkan kos buruh dan memastikan penilaian kualiti yang konsisten dan tepat. Kajian ini menunjukkan daya maju melaksanakan algoritma pembelajaran mendalam lanjutan dalam tetapan pertanian dunia sebenar, membuka pintu untuk pemantauan tanaman masa depan dan kemajuan pengurusan kualiti.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

F1	-	F1 Score
P,p	-	Precision
R,r	-	Recall
MAP	-	Mean Average Precision
CNN	-	Convolutional Neural Network
YOLO	-	You Only Look Once
IOU	-	Intersection Over Union
SSD	-	Single Shot Detector
TP	-	True Positive
TN	-	True Negative
FP	-	False Positive
FN	-	False Negative
FPS	-	Frames Per Second
CIoU	-	Complete Intersection over Union
AI	-	Artificial Intelligence
BB	-	Bounding Box

CHAPTER 1

INTRODUCTION

1.1 Background

Zea Mays, the scientific name for Corn, is the popular name for the plant, which comes from Mexico or Central America. According to data from the US Department of Agriculture[1], as of 2020, the United States (US), China, Brazil, and Argentina accounted for most of the world's corn yield, or around 64.63% of total corn production worldwide.

Even though corn farming is a major economic sector in many countries, conventional sorting techniques—which primarily rely on human judgment—face difficulties related to higher labor costs, irregular time management, and grading errors. It is projected that the installation of automated sorting equipment will save labor expenses, speed up time management, and guarantee precise and consistent sorting of harvested corn. Utilizing developments in AI technology is one way to solve this problem. According to recent studies, corn quality indicators can be found by integrating AI using machine vision and image processing techniques. This method finds and isolates flaws in the corn product in addition to making quality inspection easier [2].

There have been significant advancements in the use of AI in corn quality monitoring in recent years. By applying machine learning and image recognition techniques, AI evaluates visual information such as corn kernels, ears, and plants to figure out their quality. In addition to helping identify potential problems like pests, illnesses, or abnormal growth patterns, this study gives farmers the tools they need to take initiative-taking measures to solve these problems and maximize crop output and quality.

Corn monitoring involves more than just evaluating quality. It entails a thorough assessment that considers things like the existence of pests or diseases, moisture content, maturity, and general state. To minimize crop loss and guarantee the security and integrity of the finished product, prompt identification and resolution of such issues are essential. Inadequate corn quality monitoring techniques could result in increased crop loss since pests and illnesses take longer to detect. This could also compromise food safety by distributing inferior produce that is contaminated with toxins or pathogens, endangering the health of consumers. Inadequate monitoring of corn quality can have negative economic effects as well. For example, it might lead to the rejection of subpar output that does not match industry requirements, which costs growers and the industry money. In addition, the selling of inferior corn could damage the industry's brand and cause it to lose market share and consumer confidence.

Native American communities in Mesoamerica have relied on corn as a basic food source for millennia. quality is influenced by several variables, such as post-harvest procedures, growth circumstances, and plant genetics. Among the factors influencing corn’s condition are moisture content, test weight, disease or insect damage, and the presence of foreign materials. New developments in genetic engineering (GE) and breeding have produced superior corn cultivars with higher nutrient density, improved yields, and insect resistance as shown in Figure 1.1 for the spoiled corn. Moreover, the quality of corn has increased dramatically because of sustainable farming methods that prioritize soil health and lessen dependency on chemical assistance.

Figure 1.1 Spoiled Corn.

To sum up, the integration of artificial intelligence (AI) technology with conventional agricultural practices has the potential to transform the monitoring of corn quality, guarantee agricultural sustainability, and satisfy the changing needs of the global market.

1.2 Motivation

Marketing fresh fruits and vegetables successfully requires maintaining quality through harvest and beyond. According to Cantwell, good visual, nutritional, and sensory attributes are closely linked to freshness [3], making this a crucial aspect of quality. Accelerated marketing is one way to achieve freshness, but there are other ways as well, such maintaining quality under controlled conditions for short periods of time (days). Data from MyAgri Consulting normally classifies post-harvest losses of fruits, like corn, into two categories: physiological losses and physical losses. Physical loss happens when the fruits are picked and either their physical structure is damaged, or they are attacked by microorganisms like fungus and bacteria. Changes in the color, taste, texture, and nutritional content of the harvested fruits result in physiological harm. Fruit sorting and grading are two post-harvest handling techniques that are intimately linked to both kinds of losses. A commodity with poor handling has a low market value.

A reasonable degree of freshness is required for agricultural goods, with a tolerance range of 3% to 10% classes for fruits and vegetables, according to records from the Federal Agricultural Marketing Board webpage [4]. It is forbidden to market items that do not meet the requirement of freshness since doing so would lower the quality of the other products in the lot. Freshness signs include the product's perfect skin, firm filling texture, softness, and absence of wrinkles.
A consistent size tolerance of 3% to 10% should be present in agricultural goods of a particular grade [4]. When every product in a packing unit weighs less than the maximum permitted by the size categorization, the packaging unit is deemed to be uniform.

Previous studies have shown that a lot of researchers are looking at ways to improve the quality of fruits, such as corn, by focusing on post-harvest handling and quality inspection, which involves grading and fresh fruit sorting. Lowering the rate of crop damage requires combining a good harvest with an unhealthy yield.

1.3 Problem Statement

The major issue that 'Corn Detection' seeks to address is the absence of precise and effective techniques for detecting, grading, and categorizing corn products in agricultural settings. Existing methods, which are largely based on manual evaluation approaches, typically result in labor-intensive procedures, wasteful use of time, and an increased chance of subjective mistakes.

Lack of accuracy in corn detection procedures makes it difficult to maximize crop production and impedes accurate quality assessment. This jeopardizes the financial stability of corn farming techniques since farmers are unable to accurately appraise and market their produce.

A dearth of industry-standard automated corn detection equipment makes it difficult for the sector to discover variances in corn quality promptly. This has an impact on resource allocation and causes delays in early intervention measures, which are required to ensure optimal crop utilization and compliance with industry requirements.

1.4 Research Objective

(a) Develop a CNN model using labeled datasets to accurately identify and classify corn quality issues caused by pests and diseases.

(b) Implement a classification system within the model to differentiate between three classes in the corn dataset, enhancing automated quality evaluation.

(c) To evaluate the accuracy and response time of the identification performance by using YOLOv8.

1.5 Scope

i. This system aims to systematically monitor corn quality, focusing on measurable outcomes such as accurate detection and classification.

ii. Research will be focusing on three classes (Healthy, Water Rot, and Bug).

iii. The procedure of detecting systems takes place in a bright area.

iv. The system's development will be conducted using the Python language, integrating TensorFlow, PyTorch, and OpenCV for measurable improvements in accuracy.

v. This system was trained for real-life detection by a microcomputer.

1.6 Research Outline

There are five primary chapters in the research structure. Sections and subsections are further separated into each of them.

Chapter 1 Provides a comprehensive overview of the issue before focusing on its specifics at a technical level. It also outlines the scope and objective of the research.

Chapter 2 A review of the literature on recent studies on crack detection and methods for identifying concrete cracks, together with benefits and drawbacks. It also goes over the definition of machine learning and its advantages. Finally, a literature summary is given.

Chapter 3 Explains and delivers the methodology, which includes the tools and implementation strategy needed to complete the project.

Chapter 4 Gives a summary of the project's initial results.

Chapter 5 Outlines the primary results and conclusions of the investigation, drawing on the issues covered in the earlier chapters, and offers suggestions for further research.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In terms of contemporary farming and agricultural technology, corn detection and identification in agricultural landscapes are highly significant. Conventional techniques for corn plant identification, which depend on visual evaluations and manual inspection, are frequently time-consuming, tedious, and subjective. However, integrating machine vision technologies presents a practical way to transform corn detection methods.

Machine vision is a branch of artificial intelligence that gives computers the ability to understand and interpret visual input, such as images. Corn detection applications can benefit from machine vision algorithms, which have enormous potential for processing and interpreting visual data when they are used with specialized software and hardware.

The purpose of this literature review is to examine the effectiveness and drawbacks of different machine vision algorithms used especially for corn detection. It also looks at the opportunities and difficulties that come with using machine vision to accurately identify corn plants, suggesting directions for further study and advancement.

Through a thorough examination and assessment of the wide range of machine vision techniques and their uses in corn detection, this review looks to provide a thorough analysis of the state of the field. By using AI-driven methods, it aims to supply light on new patterns and insights that will direct future developments in the field of corn detection.

2.2 Bug/Disease Affect to Corn Plant Appearance

Corn (Zea mays) is a global staple crop that serves as a key source of nourishment and raw material for a variety of industries. However, effective corn farming is often challenged by a slew of pests and diseases that may have a substantial influence on the plant's look, output, and general health. The purpose of this literature review is to investigate and consolidate existing research on the impact of insect and disease infestations on the visual properties of corn plants [5].

Insects constantly threaten corn fields, both above and below ground. Visual indicators of insect damage, such as leaf browning, wilting, and abnormalities, have been studied. Corn earworm (Helicoverpa zea), European corn borer (Ostrinia nubilalis), and corn rootworm (Diabrotica spp) are three notable insect pests. The effect of these insects on the look of corn plants varies, with some causing noticeable damage to foliage and others directly affecting corn's ears [6]

Fungal infections have a substantial impact on the health and appearance of corn plants. Common rust (Puccinia sorghi), northern corn leaf blight (Exserohilum turcicum), and southern corn leaf blight (Bipolaris maydis) are notable diseases. These diseases frequently present on the leaves as distinctive lesions, yellowing, and necrosis, affecting the overall visual quality of the corn plant. Furthermore, certain fungi harm corn ears, resulting in lower yield and less aesthetic appeal [7]

Bacterial and viral diseases are also important variables influencing corn plant appearance. Bacterial infections that generate visible streaks, lesions, and wilting include bacterial leaf streak (Xanthomonas vasicola) and Goss' wilt (Clavibacter michiganensis subsp, nebraskensis). Viral infections, such as corn dwarf mosaic virus (MDMV) and corn streak virus (MSV), contribute to mosaic patterns, stunting, and leaf deformity, affecting the overall appearance of corn plants as shown in Figure 2.1 [8]

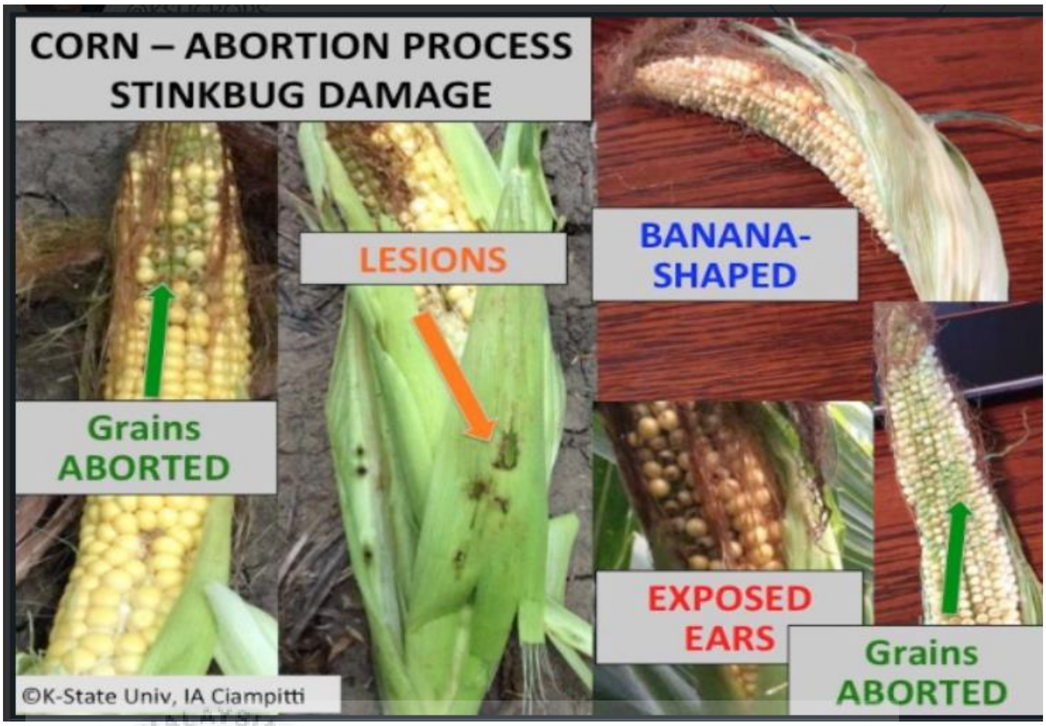

Figure 2.1 Illustrates the Bug/Disease in corn [6].

2.3 Artificial Intelligence

Artificial Intelligence (AI) in agriculture has received a lot of interest in recent years, with academics looking for new ways to improve crop monitoring and management. Corn, being one of the world's most important staple commodities, is an important target for AI-based detection and classification systems. This literature review is to offer an overview of the current state of research in the area, highlighting significant approaches, problems, and advances in the field of artificial intelligence for corn identification and categorization [9]

One popular method for detecting corn is to use remote sensing techniques such as satellite photography and unmanned aerial vehicles (UAVs). Researchers have investigated the use of AI algorithms in conjunction with these technologies to extract important information from corn fields. Convolutional Neural Networks (CNNs) have proved to be successful in evaluating high-resolution satellite photos, allowing for accurate corn crop identification and mapping [10].

Corn identification and categorization rely heavily on image processing and computer vision. Numerous researchers have used deep learning approaches to interpret photos collected in the field, such as CNNs and recurrent neural networks (RNNs). These approaches enable the extraction of pertinent information such as leaf patterns, plant height, and disease signs, allowing for more exact corn categorization [11].

Deep learning and object identification technologies may also be used, as proven by the lightweight model LW-YOLOv7, which can recognize corn seedlings in the field in real-time [12] Furthermore, for identifying contaminants and breakage rates in harvested corn grains, a classification and identification approach based on a feature threshold and a backpropagation neural network has been presented [13].

Despite tremendous advances, obstacles remain in the development and deployment of artificial intelligence for Corn identification and categorization. Data quality issues, environmental variability, and the requirement for big, labeled datasets are all ongoing concerns. Furthermore, the interpretability of sophisticated AI models is still an issue, because comprehending the decision-making process is critical for establishing confidence and acceptability in agricultural operations.

2.4 Conventional Neural Network (CNN)

In agricultural activities, it is essential to recognize and evaluate the quality of corn with pests or diseases. It takes advanced techniques to use artificial intelligence (AI) for corn categorization. Artificial intelligence has been useful in analyzing corn quality about pests and illnesses. Specifically, it has done so by using machine learning techniques such as CNNs in conjunction with frameworks like You Only Look Once (YOLO) combined with Python and OpenCV [14].

CNNs are highly skilled at identifying patterns and features. They are praised for their abilities in a variety of computer vision applications. Their capacity to independently assimilate discrete attributes from input data renders them indispensable in the identification and classification of any problems that might impact corn quality. By using CNN architectures, problematic elements may be identified accurately and efficiently, which improves corn health evaluation and management [15].

Prominent for its real-life item identification, YOLO integrates object recognition and classification into one cohesive neural network with ease. Because of its accuracy and speed, it can identify and classify various corn quality problems, which is in line with the needs of agricultural settings for effective detection. YOLO-based solutions enable quick decision-making in the management of corn-related issues, such as diseases or pests, through real-life analysis [16].

Programming languages like Python, which are well-known for their large size of libraries and user-friendliness, are perfect for incorporating AI into agriculture. The versatility of this language as a bridge language in situations involving cross-language programming fits in nicely with the many requirements of AI-powered agricultural solutions. One of the most advanced image-processing packages for Python is OpenCV, which was first created by Microsoft and is a powerful image-processing program. In the context of evaluating corn health for pests or illnesses, its connection with Python simplifies the manipulation and analysis of visual data. Data scientists and agricultural specialists may effectively analyze and categorize corn quality by utilizing AI techniques, such as CNNs coupled with YOLO and Python backed by OpenCV. This will help with proactive pest or disease management for increased crop health and yield. like those shown below use Python and OpenCV as their primary programming languages [17].

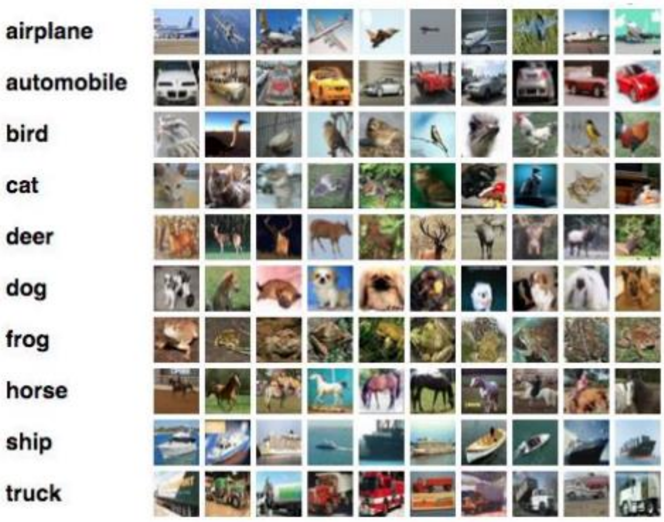

Figure 2.2 Image Classification [15].

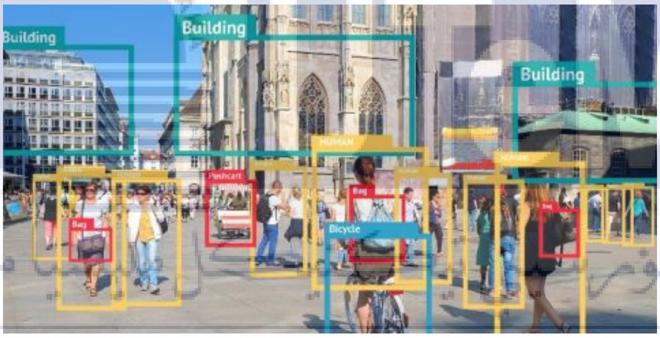

Figure 2.3 Object Detection[17].

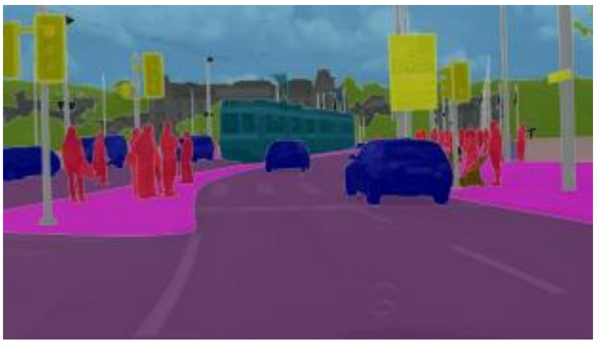

Figure 2.4 Segmentation[17].

YOLO's integration with OpenCV and Python for feature extraction from traffic signs has been thoroughly investigated. In conclusion, Yolo has become the main Python and OpenCV-based open-source tool for picture manipulation and detection. Furthermore, model testing is possible due to the magnitude of the plant dataset for the solutions shown in Figures 2.2, 2.3, and 2.4 [17].

2.5 Feature Extraction

Data reduction is the process of dividing a huge amount of raw data into smaller, more manageable chunks, of which feature extraction is a subset. Processing it afterward will be less of a headache [18]. The large number of variables included in these enormous datasets is one of their main characteristics. Handling these variables comes with a considerable computational cost. Feature extraction assists in identifying the optimal feature from such vast data sets by selecting and combining variables into features. Using feature extraction algorithms may lead to additional benefits including improved data visualization, faster training, reduced overfitting risk, and increased accuracy [19]. Rosanna C. et al. conducted a second investigation on the categorization of cavendish bananas [20].

Physical characteristics are the key criteria for determining a Cavendish banana's grade, banana's grade. Image conversion methods are employed to satisfy the requirement of a defect score of 0–2. The finger size requirement is meticulously defined and followed during the sample data prequalification procedure. You may measure the length of your finger by measuring the outer curve from the very tip of the fruit to the very tip of the stem using a measuring tape. Establishing a trustworthy sample size and a reference measurement is the aim.

2.5.1 Color Statistical Features

Megha's study [2] used ideas like color statistical characteristics and color texture features to perform trials on the quality of perfect and flawed tomatoes. Typically, an image is taken and then divided into its colors, such as red (R), green (G), and blue (B), also referred to as RGB color. The computation of an RGB color's mean, standard deviation, and skewness are shown in equations (2–1), (2–2), and

(2-3), respectively. These data can be used by authors [2] to calculate an image's average color intensity. When a color has a high mean value, it seems bright in the image, and when it has a low mean value, it appears dark.

$$\text{Color Mean }(\mu) = \frac{1}{N}\sum_{i=1}^{N} Pi \tag{2-1}$$

Customers may then calculate the standard deviation by utilizing the values of each pixel as well as the average. Users may be able to learn something about the image's color contrast by looking at its standard deviation. There is a lot of color contrast in the image if the standard deviation of a group of hues is high. Less fluctuation in the histogram levels will result in a less noticeable color contrast [21].

$$\text{Standard Deviation }(\sigma) = \left(\frac{1}{N-1}\sum_{i=1}^{N}(Pi-\mu)^2\right)^{\frac{1}{2}} \tag{2-2}$$

The degree of skewness measures the degree of dissymmetry. Researchers determine that a data set is symmetrical when there is little to no change in the data from left to right. Since a normal distribution has skewness near zero, any symmetric data should likewise have skewness close to zero.

$$\text{Skewness} = \frac{\sum_{i=1}^{N}(Pi-\mu)^3}{N\sigma^3} \tag{2-3}$$

2.5.2 Color Texture Features

The image's grey-level co-occurrence matrices (GLCM) are used to extract the four features from each color channel. The components of i, jLCM reflect the probability density function Pij, which counts the instances of pixel pairs with intensity values (i,j) being separated by a specified distance along the direction T. With an inter-pixel length of 1, the current study [2], [22] considers four angular directions: 0, 45, 90, and 135 degrees. The formulae for the contrast, correlation, energy, and homogeneity of the attributes extracted from the GLCM are shown in equations (2–4), (2–5), (2–6), and (2–7), respectively.

$$\text{Contrast} = \sum_{i,j=1}^{N} |i-j|^2 Pij \tag{2-4}$$

$$\text{Correlation} = \sum_{i,j=1}^{N} Pi,j \frac{(i-\mu)(j-\mu)}{\sigma^2} \tag{2-5}$$

$$\text{Energy} = \sum_{i,j=1}^{N} (Pij)^2 \tag{2-6}$$

$$\text{Homogeneity} = \sum_{i,j=1}^{N} \frac{Pij}{1+|i-j|} \tag{2-7}$$

Based on this experiment, M. P. Arakeri et al. [2] have concluded that a tomato's color is an excellent predictor of its maturity stage. As a result, to identify whether a tomato is ripe or not, the color feature is extracted from the image. These formulae are used to extract the R, G, and B values from the picture and average them. The threshold level is established by comparing the mean R with the criteria. Researchers can determine if a tomato is ripe if its level rises beyond a specific threshold; if not, it is not. Equation (2-8), (2-9), and (2-10) displays the formula for the mean value of the red layer, green layer, and blue layer, respectively.

$$\text{Mean R} = \frac{R}{N} \tag{2-8}$$

$$\text{Mean G} = \frac{G}{N} \tag{2-9}$$

$$\text{Mean B} = \frac{B}{N} \tag{2-10}$$

The tomato image is broken down into 12 color texture features and 9 color statistical characteristics, for a total of 21 attributes in total, according to [2]. To improve classification accuracy, a suitable feature set is selected from the original feature vector using the sequential forward selection (SFS) approach. Using an empty set as input, this greedy selection approach iteratively adds one characteristic from the subset to the subgroup if doing so improves the best possible answer.

2.6 Object Detection and Classification

Finding and categorizing things that are already in an image, along with marking them with rectangular bounding boxes that stand for the degree of certainty in their existence, is the process of object detection. Object-detecting methods can be used in two types of frameworks. One makes use of the conventional object detection pipeline, which starts with the creation of regional suggestions and continues with classifying each proposal based on the diverse kinds of objects. The second method takes a regression or classification approach to object identification, using a unified framework to swiftly arrive at the necessary results (locations and classifications) [23].

2.6.1 YOLO (You Only Look Once)

Researchers have been interested in pre-trained models like YOLO in recent years due to their ease of use and increased item identification accuracy. YOLO, an acronym for "You Only Look Once," is a CNN-based model that starts learning by transfer learning using pre-trained weights. In contrast to conventional CNN models, YOLO analyses the complete image in a single instance, which enables it to anticipate object class probability and bounding box locations. Compared to creating a CNN from scratch, this method requires a lot less work, which leads to faster learning and real-life detection capabilities as Figure 2.8 shows.

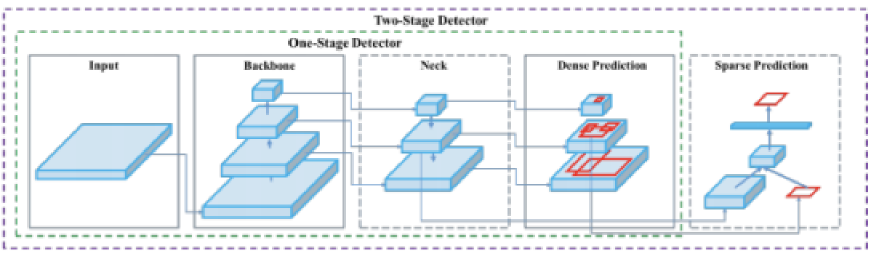

Figure 2.5 One-stage Detector Architecture [24].

Take a closer look at each element of the architecture, beginning with the backbone, which is a trained network that is intended to extract sophisticated picture information. In this technique, the image's spatial resolution is decreased but its feature channel resolution is improved. Subsequently, the neck component that extracts feature pyramids is incorporated into the model. This aids in the model's good generalization to objects with varying scales and sizes. Finally, by adding anchor boxes to the feature maps, the model head completes the last duties. It produces the output, which consists of bounding boxes, classes, and abjectness scores [24]

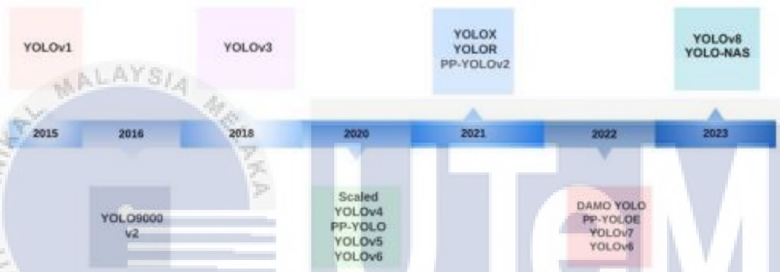

Figure 2.6 Is a Timeline of the YOLO Version [24]

The evolution of YOLO (You Only Look Once) models is shown in Figure 2.9, which shows the development from YOLOv1 to the most recent YOLOv8. Architecture, feature extraction, and performance optimization are improved with every iteration. This in-depth analysis seeks to disentangle the unique features of every YOLO iteration, clarifying the complex advancements and changes included in their designs.

1- YOLOv1 uses Darknet-19, a simple convolutional neural network (CNN) structure, as the foundation for its feature extraction system.

2- YOLOv2 improves performance by Darknet-19 (v2), a more resilient CNN architecture that uses residual connections to manage deep networks.

3- YOLOv3 uses a unique design known as Darknet-53 that makes use of shortcut connections as well as residual connections to increase performance.

4- YOLOv4 presents a new backbone architecture known as CSP-Darknet-53, which is an improvement on Darknet-53. Cross-Stage-Partial (CSP) connections are included to improve performance even more.

5- YOLOv5 has as its foundation the CSPDarknet-53 network design, which is an improved version of the YOLOv4 architecture. Interestingly, to enhance its capabilities, it integrates several feature extraction pathways with different resolutions and sizes.

6- YOLOv6 presents a PAN neck, an efficient decoupled head, and an efficient backbone known as EfficientRep. Label assignment, new losses, quantization techniques, and self-distillation are all incorporated. With an accuracy and performance improvement over earlier models, YOLOv6 achieved an AP of 57.2% on the MS COCO test-dev at 29 frames per second on an NVIDIA Tesla T4.

7- YOLOv7 outperformed other object detectors in terms of accuracy and speed. Changes to the architecture were added, including scaling for concatenation-based models and the creation of the Extended Efficient Layer Aggregation Network (E-ELAN). Several other "bag-of-freebies" were also used by YOLOv7, such as planned re-parameterized convolution, batch normalization in conv-bn-activation, coarse and fine label assignment, implicit knowledge from YOLOR, and exponential moving average for inference. Although they lengthened the training period, these improvements increased accuracy without compromising inference speed.

8- YOLOv8 supports a range of vision activities and provides five scaled variants. Although its structure is identical to that of YOLOv5, it adds modifications to the CSPLayer, which is now known as the C2f module. These modifications improve detection accuracy by merging contextual data with high-level features, with its anchor-free model and decoupled head, YOLOv8 improves accuracy by letting each branch concentrate on its specific task. For bounding box loss, it employs CIoU and DFL loss functions; for classification loss, binary cross-entropy is used. The semantic segmentation model, YOLOv8-Seg, maintains speed and efficiency while achieving state-of-the-art results. YOLOv8 offers deployment, training, and labeling integration options. Using an NVIDIA A100 and TensorRT, the largest version, YOLOv8x, obtains an AP of 53.9% at a 640-pixel input size at 280 frames per second.

The official YOLOv1–YOLOv4 models previously used The Darknet framework, an open-source convolutional neural network based on C language. But in 2020, Ultralytics released YOLOv5, which made use of the PyTorch framework. PyTorch is an open-source machine learning library that can be used with Python or C++ to create and train deep learning models. Using PyTorch, YOLOv5 beat YOLOv4 and YOLOv3 in terms of accuracy and training and inference speeds, according to comparative analyses conducted on the MSCOCO dataset [18] It is important to keep in mind that YOLO may be utilized with TensorFlow, Keras, and Caffe—three additional popular deep-learning frameworks. The object detection models YOLOv6, YOLOv7, and YOLOv8 are incremental improvements upon one another. An efficient decoupled head, PAN neck, and backbone are introduced by YOLOv6 and YOLOv7 uses scaling approaches, and E-ELAN architecture modifications to increase training efficiency and accuracy. YOLOv8 achieves state-of-the-art performance in a variety of vision tasks by introducing an anchor-free model with a decoupled head, while still using a comparable backbone to YOLOv5. Every version offers options for speed, accuracy, and particular use cases in object detection, while also bringing advancements in performance, loss functions, and training methodologies [19]

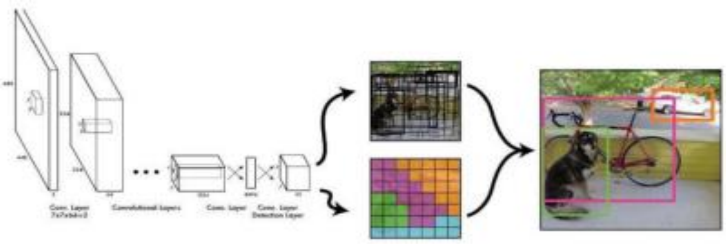

Figure 2.7 YOLO Architecture [25].

The effectiveness of YOLO models can be evaluated by comparing the Intersection over Union (IoU) of the ground truth bounding box (red in Figure 2.10) with the predicted bounding box (blue in Figure 2.11). An IoU number greater than 1 (which ranges from 0 to 1) shows an exact prognosis. To evaluate the quality of forecasts, a threshold is set to classify predictions as good or bad based on this IoU value [25].

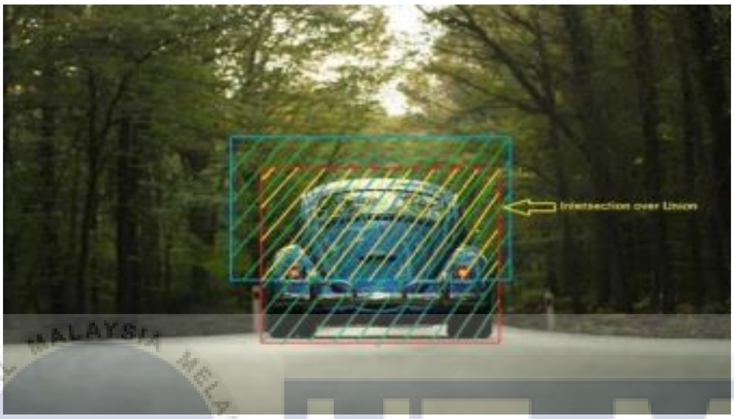

Figure 2.8 Intersection Over Union Concept [25].

2.6.2 You Only Look Once (YOLO) Framework

Guojin Li et al. created Lemon-YOLO (L-YOLO), a sophisticated object recognition system designed exclusively for recognising lemons in natural settings. This method includes numerous significant enhancements over typical YOLO models. The main change is the addition of a new backbone network, SE_ResGNet34, which replaces the DarkNet53 seen in YOLOv3. This improvement dramatically increases feature extraction capabilities, allowing the algorithm to better locate lemons in complicated backdrops. Additionally, the addition of SE_ResNet modules improves the quality of convolutional features, resulting in more accurate detections.

These enhancements have yielded exceptional performance figures on the lemon test set, with an average detection accuracy of 96.28% and a processing speed of 106 frames per second (FPS). The algorithm's excellent accuracy assures that it can correctly recognise lemons, while its quick processing speed makes it suited for real-time applications. Lemon-YOLO was developed and tested using PaddlePaddle

1.7.0 and Python 3.7.3 on a Windows 10 environment. These software tools offered a stable environment for developing and testing the algorithm, proving its usefulness and efficiency in real-world scenarios. Lemon-YOLO is a significant advancement in the specialised field of agricultural object recognition, providing a strong tool for increasing the efficiency and accuracy of lemon harvesting and quality control procedures.

Table 2-1 compares the performance of several object identification models on the Lemon dataset. It focuses on the backbone architecture, input size, average precision (AP) %, and frames per second (FPS). Among the models, L-YOLO has the greatest AP (96.28%) and the quickest processing speed (106 FPS). Other models, such as YOLOv4 and EfficientDet-D0, demonstrate good accuracy but vary in speed. SSD is known for its speed, whereas RetinaNet and Faster R-CNN provide great accuracy but are slower. Overall, L-YOLO has the most accuracy and quickness[26].

Table 2-1 Performance of Different Models on Lemon Dataset[26].

Model	Backbone	Input size [pixels]	AP [%]	FPS
SSD	Vgg16	512 × 512	90.15	87
RetinaNet	ResNet101-FPN	–	93.56	20
EfficientDet-D0	Efficient-B0	512 × 512	92.30	73
YOLOv4	CSPDarkNet53	704 × 704	93.97	41
Faster R-CNN	ResNet50	–	93.12	6
Cascade R-CNN	ResNet50	512 × 512	90.27	21
FCOS	ResNeXt-64 × 4d-101-FPN	–	92.03	9
L-YOLO	SE_ResGNet34	704 × 704	96.28	106

Wei Chen et al. looked at real-time apple detection in natural settings to improve the efficiency of autonomous harvesting robots and orchard management. They suggested Des-YOLO v4, an upgraded version of YOLO v4 with increased detection speed and accuracy. Key innovations include the use of an AP-Loss-based class loss algorithm to control sample imbalance and the replacement of traditional

NMS with Soft-NMS to better handle overlapping apples. The software tools used included a modified YOLO v4, Matlab for preprocessing, and Labelling for annotation. The hardware arrangement included a small OV2640 camera installed in an eye-in-hand position, allowing the robot to continually choose fruit. When tested on a custom dataset, Des-YOLO v4 has a mean average precision (mAP) of 97.13% and a recall of 90% at 51 frames per second. Practical testing revealed that the robot could select apples in 8.7 seconds per fruit, with a 92.9%

. Table 2-2 and Figure 2.9 compare the performance of Faster R-CNN, YOLO v4, and Des-YOLO v4 detection algorithms, emphasizing their backbone networks, mean Average Precision (mAP), and detection times. Des-YOLO v4 is the most accurate, with a mAP of 93.1% and a respectable detection rate. Faster R-CNN and YOLO v4 had mAPs of 88.1% and 87.9%, respectively. The accompanying photos show the apple identification process using YOLO v4, which includes partitioning the image into grids, forecasting class probabilities, regressing bounding boxes, and fine-tuning the final result. This visualization demonstrates YOLO v4's ability to effectively recognize and localize apples inside images[27].

Table 2-2 Performance Comparison of Different Algorithms[27].

Algorithm	Backbone network	mAP (%)	Detection speed ($f{\cdot}s^{-1}$)
Faster R-CNN	ResNet50	88.1	9
YOLO v4	CSPDarknet53	87.9	53
Des-YOLO v4	Des-Darknet	93.1	51

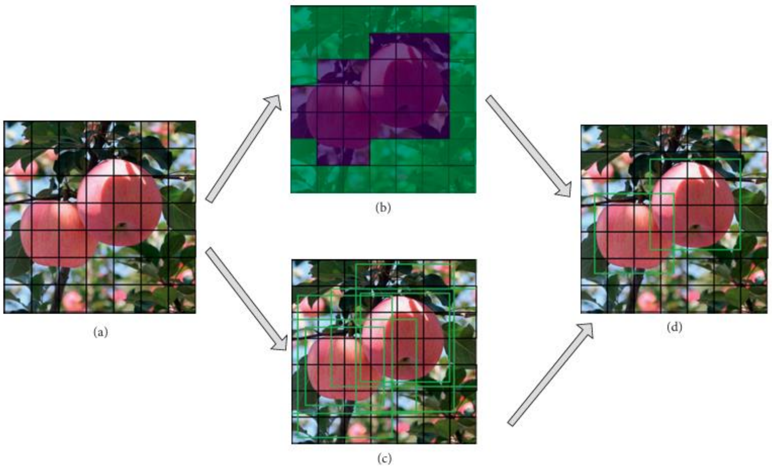

Figure 2.9 Apple detection based on YOLO v4. (a) Dividing image into S∗S grids. (b) Predicted class probability. (c) Regression bounding[27].

Baoju Wang et al. suggested an upgraded lightweight YOLOv5s model with the goal of enhancing maize and weed detection precision for optimal spraying applications. The model's achievements are focused on tackling critical difficulties in agricultural computer vision, including data imbalance and model size efficiency. To address data imbalance, the researchers used advanced data augmentation techniques, which were critical in improving the model's capacity to reliably discriminate between maize and weeds. Furthermore, the SENet (Squeeze-and-Excitation Network) module improves the model's capacity to focus on essential characteristics, resulting in higher overall detection accuracy. Furthermore, the C3-Ghost-bottleneck module plays an important role in lowering the model's size while preserving or even improving its performance metricsThe improved YOLOv5s model was trained and evaluated on a high-performance computer environment designed for deep learning. Windows 10, CUDA 10.2, Python 3.7, and PyTorch 1.10 were installed to ensure software framework compatibility and efficiency.

Empirical findings showed considerable improvements over the baseline model, including a 3.2% increase in mAP%0.5. The average accuracy (AP) for corn detection went from 93.2% to 96.3%, while weed AP rose from 85.6% to 88.9%. These advances demonstrate the usefulness of model adjustments in attaining more exact crop and weed identification, which is critical for optimizing spraying operations in precision agricultural applications. Weeds were recognized in 418 of the 465 occurrences, with 386 TP, 32 FN, 60 false positives (FP), 380 precise, and 126 wrong sprays[28].

Table 2-3 Test results of field precision spraying operation[28].

Category	Number	Successful detection	TP	FN	FP	Precise spraying	Incorrect spraying
Corns	600	598	556	42	–	–	68
Weeds	465	418	386	32	60	380	126

Table 2-4 compares the performance of the YOLOv5s with its upgraded version: for maize, the AP (average precision) increased from 96% to 97%, the mAP@0.5 (mean average precision at 0.5 IoU) from 91.7% to 93.4%, and the speed increased from 6ms/frame to 3.2ms/frame. For weed detection, AP increased from 87.4% to 89.8%, while mAP@0.5 increased from 91.7% to 93.4%, with comparable speed gains. This shows that the improved model has higher precision and faster processing[28].

Table 2-4 Evaluation results of test set under different conditions[28].

	Category	AP	mAP@0.5	Speed
yolov5s	corn	96%	91.7%	6ms/f
	weed	87.4%		
Improved yolov5s	corn	97%	93.4%	3.2ms/f
	weed	89.8%		

Chen Jiyao et al. created an improved algorithm for corn cob defect recognition based on the YOLOv7 architecture, which included novel components to boost detection accuracy. The Explicit Visual Centre Block (EVCBlock), which was created to improve the identification of minute and detailed flaws on maize cobs, is central to their strategy. This feature solves a typical issue in agricultural imaging: small objects necessitate precise localization and categorization. Furthermore, the Receptive Field Enhancement Module (RFEM) was created to extract complete characteristics from defective maize cobs, boosting the model's capacity to detect small alterations that indicate a defect.

The improved YOLOv7 model was experimentally validated under controlled settings using Python 3.9, using four batches and 200 epochs of training. The findings showed significant performance improvements, with the model reaching an average accuracy (mAP) of 88.1%. This is a significant 12.2 percentage point increase over the original YOLOv7 baseline. Such developments are critical in agricultural quality control applications, where precise defect identification is essential for assuring product quality and optimizing production operations. By combining EVCBlock and RFEM, the improved model not only improves detection sensitivity but also provides a solid framework for automating inspection duties in agricultural contexts, thereby lowering operational costs and increasing overall efficiency. [29]

2.7 Application and Algorithm in Fruit Classification System

2.7.1 Region Proposal-Based Model

The region proposal-based model is a basic method that is like the attentional mechanism of the human brain and is applicable in the field of corn detection. This model works in two phases: first, it performs an exhaustive scan of the entire image, and then it focuses on analyzing those areas that are important to the overall visual environment. When it comes to corn detection, this approach is effective at finding areas that are most likely to have corn plants. Through close examination of the innate traits and attributes of these chosen areas, the model skillfully distinguishes between areas that are corn-producing and areas that are not, so enhancing the precision and efficacy of corn plant identification [30]

By integrating Convolutional Neural Networks (CNNs) with the sliding window technique, this model allows bounding boxes (BBs) to be directly predicted from positions on the post-confidence evaluation of the highest-level feature map of underlying object categories. The exact location of corn plants inside the image is made possible by the prediction approach, which involves inferring BBs directly from specified coordinates on the feature map [31].

2.7.1.1 R-CNN

In the Fruit Classification System, R-CNN plays a crucial role in enhancing the quality and accuracy of candidate bounding boxes (BBs) by leveraging a deep architecture to extract high-level features from the input data [32]. suggested R-CNN as a remedy for these problems. They surpassed the previous best result by more than 30% with a mean average precision (MAP) of 53.3% in their study [33]. There are three main phases in the R-CNN flowchart, each of which has a specialized function in the detection process as shown in Figure 2.5.

Region Proposal Generation: For each image, the Fruit Classification System generates approximately two thousand region proposals using a technique called selective search [26]. This method of selective search effectively generates accurate candidate bounding boxes of varied sizes by combining saliency signals with bottom-up grouping. This technique allows for faster and more exact detection results by reducing the search area needed for item detection [24].

CNN-Based Deep Feature Extraction: To obtain a consistent resolution, each region proposal is scaled or cropped at this stage. The final representation of the area is then obtained by extracting a 4096-dimensional feature using the CNN module. For every proposed region, a robust, semantically rich, and high-level feature representation can be obtained because of CNN's large learning capacity, expressive skills, and hierarchical structure [34].

Classification and Localization: Using pre-trained category specific linear SVMs for several classes, each region proposal is given a score that considers both positive and background (negative) regions. The scored regions are filtered using a greedy non-maximum suppression (NMS) algorithm to produce the final bounding boxes for the detected object locations. Bounding box regression is then used for refining.

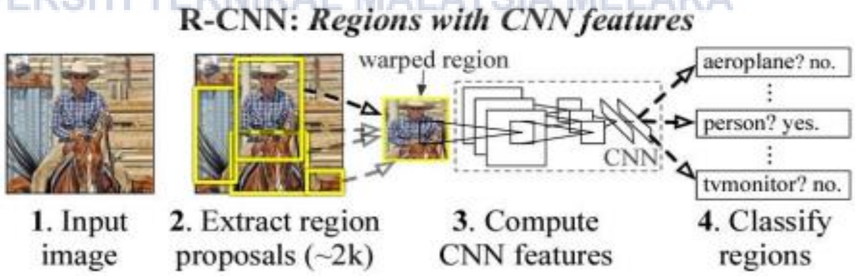

Figure 2.10 Flowchart Of R-CNN [33].

2.7.1.2 Fast R-CNN

The independent CNN forward propagation for every area proposal, which does not use shared computing, is the main bottleneck in an R-CNN performance. These regions often overlap; therefore, a significant amount of computation is duplicated during independent feature extractions. The fact that CNN forward propagation is only conducted on the entire image is among the most important improvements provided by the fast R-CNN over the R-CNN as in Figure 2.6 [33].

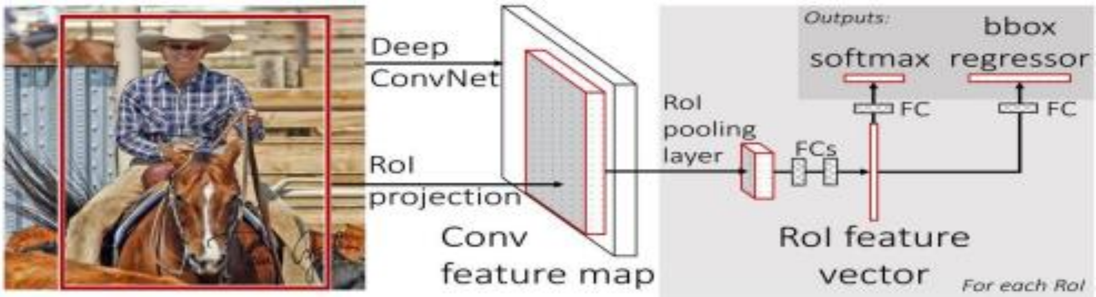

Figure 2.11 Architecture of Fast R-CNN [33].

Fast R-CNN major computations are as follows:

In contrast to the R-CNN, the rapid R-CNN more effectively uses the complete image as input for feature extraction. As a result, considering each area suggestion independently is no longer necessary Figure 2-2 Architecture of Fast R-CNN [34]. Moreover, training is possible for the CNN that powers the fast R-CNN. When a picture is fed into this CNN, the resultant form is $1 \times c \times h\_1 \times w\_1$, where c stands for the number of channels and h_1 and w_1 stand for the output's height and width, respectively.

Assume that distinct region suggestions of varying forms are produced by a selective search method. These region suggestions pinpoint interesting areas in CNN’s output, which also come in a variety of forms. Additional features of the same shape—specified as height h_2 and width w_2—are taken from these regions of interest to make concatenation easier. To do this, the fast R-CNN adds a layer known as the area of interest (ROI) pooling layer. This layer receives the region suggestions and CNN output as inputs, producing concatenated features with the shape $n \times c \times h\_2 \times w\_2$. For every region suggestion, these concatenated features are then extracted. Concatenate the features and create an output with the form $n \times d$, where d is a model-designed variable. This can be done by using a fully connected layer.

The bounding box and class are determined using the fast R-CNN for each of the proposed regions. To anticipate the classes, the fully connected layer's output is specifically changed into the shape of n×q, where q is the number of classes. To forecast the bounding boxes, the output is additionally further converted into an n×4 shape. For class prediction, the SoftMax regression approach is used.

2.7.1.3 Faster R-CNN

The fast R-CNN model usually uses selective search to provide many region recommendations to increase object detection accuracy. In place of selective search, the quicker RCNNN model presents a region proposal network. By using this method, fewer regions are suggested while maintaining the same degree of accuracy [35]. The faster R-CNN uses a region proposal network in place of selective search, which is the main way in which it differs from the fast R-CNN in terms of how it proposes regions. The model's remaining elements are unaltered as table 2.1 below explains the differences.

Table 2-5 Difference between R-CNN, Faster R-CNN, and Faster R-CNN [28].

	R-CNN	Fast R-CNN	Faster R-CNN
Test time per image	50 seconds	2 seconds	0.2 seconds
Speed up	1x	25x	250x

Yunling Liu et al. used sophisticated object identification algorithms to recognise maize tassels, combining the Faster R-CNN framework with ResNet and VGGNet as feature extraction networks. The process entailed taking photographs of maize tassels with a UAV and a cell phone, then annotating bounding boxes around the tassels. These annotated photos were then used to train Faster R-CNN models to correctly identify maize tassels. An important adjustment to their technique was altering the anchor sizes in the Region Proposal Network (RPN) to improve the identification of little tassels, which is a critical step in boosting recognition accuracy.

The investigation provided substantial results, confirming ResNet models' higher performance over VGGNet. The ResNet101 model obtained a remarkable average accuracy (AP) of 94.99% on 600x600 UAV pictures and 89.96% on 5280x2970 UAV images with the upgraded anchors. Furthermore, when used with mobile phone photographs, the method proved to be quite successful. After resizing the photos, the approach had an AP of 95.95%. These findings demonstrate the effectiveness of deploying UAVs and mobile devices for agricultural surveillance, giving a dependable and precise method of recognising maize tassels. The study emphasises the possibility of combining sophisticated neural networks with novel imaging techniques to improve precision agriculture operations[36].

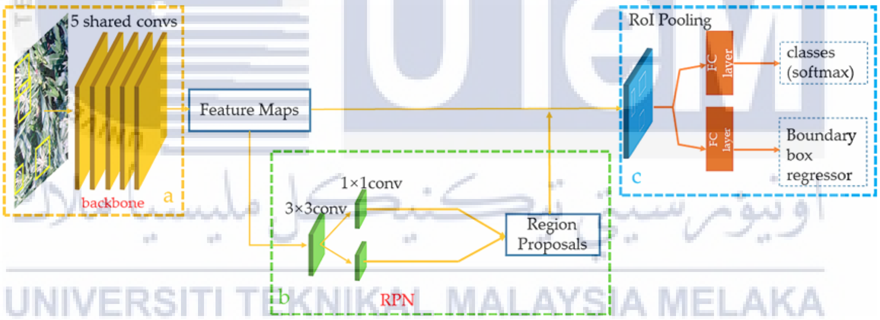

Figure 2.12 Faster R-CNN: Feature Extraction, Region Proposal, and Classification[36].

Figure 2.12 depicts the architecture of a region-based convolutional neural network (R-CNN) for object identification. The method begins with a backbone network (a), which is made up of five convolutional layers that create feature mappings from the input picture. The Region Proposal Network (RPN) later processes these feature maps. The RPN uses 3×3 and 1×1 convolutions to detect areas with objects, discriminate between foreground and background, and perform initial bounding box regression. The RPN-proposed areas are next subjected to Region of Interest (RoI) Pooling (c), which resizes them for further processing using fully linked layers. During this stage, each region is classified into particular item categories using a softmax function, and the bounding box coordinates are refined for exact object localization[36].

In terms of image recognition of figure 2.13 , the graphic depicts a VGG19 convolutional neural network (CNN) architecture. This deep learning model is excellent at recognising and categorising items in photos. VGG19 does this by methodically processing the image using a number of layered layers. The early convolutional layers function as feature detectors, distinguishing edges, shapes, and other basic visual components. Subsequent pooling layers compress this data, making it easier to handle on the network. Finally, fully-connected layers analyse the retrieved characteristics and assign probabilities, effectively identifying which category the picture belongs to (for example, cat, automobile). The complicated interplay between these layers enables VGG19 to understand complex patterns and perform outstanding picture recognition[36].

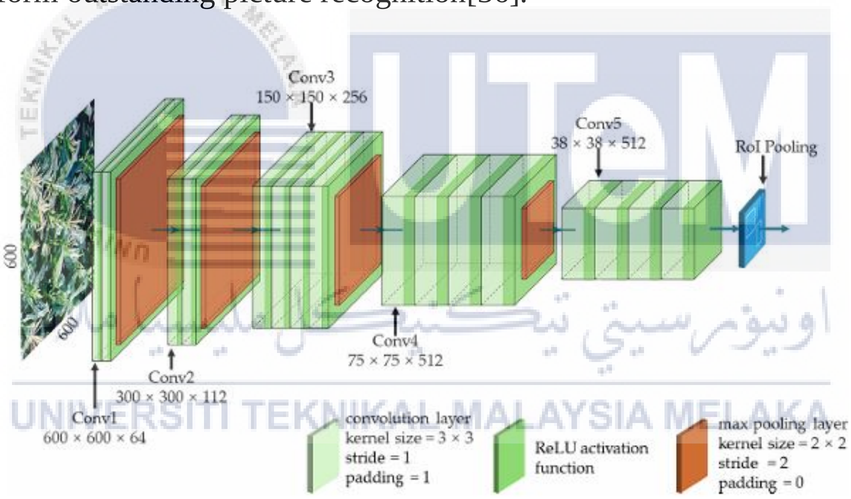

Figure 2.13 Extracted feature map using VGG19 with 600*600 resolution image. Conv represents convolutional neural network[36].

2.7.1.4 Mask R-CNN

For precise object instance segmentation, Mask R-CNN is a potent deep learning system that combines object identification with pixel-level mask creation. For our project, used the Mask R-CNN model that Matterport Mask_RCNN offered. Pretrained on a variety of datasets, the model provides a solid basis for use. By adding more training and manually annotating the model, improved its performance in a particular target domain. This research goals are facilitated by the exact object recognition and segmentation made possible by the incorporation of Mask R-CNN into our system as shown in Figure 2.7 [37].

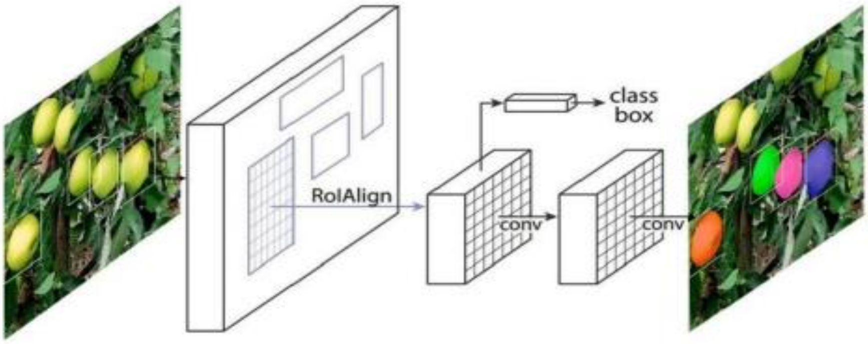

Figure 2.14 The Mask R-CNN Framework, for Instance, and Segmentation [37].

Mask R-CNN's capacity to manage many items in an image at once is one of its key features. This is accomplished by adding, alongside the branches for object recognition and classification, a new branch to the network that oversees the production of binary masks. Additionally, Mask R-CNN can manage several object classes in a single image. The network uses discrete, fully connected layers for each object class to achieve this, which enables the network to notice and recognize the unique characteristics of each class.

The Mask R-CNN technique has become a potent object instance segmentation solution in a variety of fields, including image editing, autonomous driving, and medical picture analysis. Due to its adaptability and efficiency, it has gained popularity in various areas where it allows for the exact and accurate delineation of object boundaries for advanced driving systems, improved visual comprehension, and medical diagnostic applications.

Henry O. Velesaca et al. used the Mask R-CNN architecture with a ResNet-101 backbone for maize kernel instance segmentation. For classification, they created the CK-CNN, a one-of-a-kind lightweight network made up of three convolutional layers and two fully linked layers. The researchers assembled a comprehensive collection of maize kernel photos, which included shots of both clusters of kernels and single kernels arranged in a grid. The Mask R-CNN was trained on cluster pictures to separate distinct kernel instances. The CK-CNN divided these segmented cases into three categories: great corn, flawed corn, and contaminants as shown in figure 2.15, and 2.16[38].

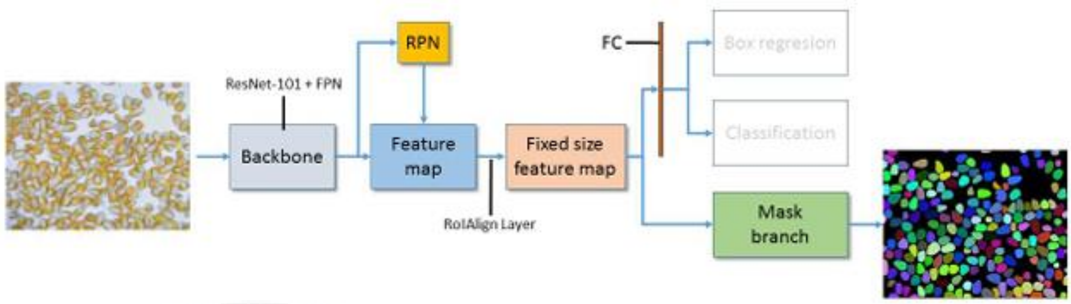

Figure 2.15 Mask R-CNN architecture used for corn kernel instance segmentation[38].

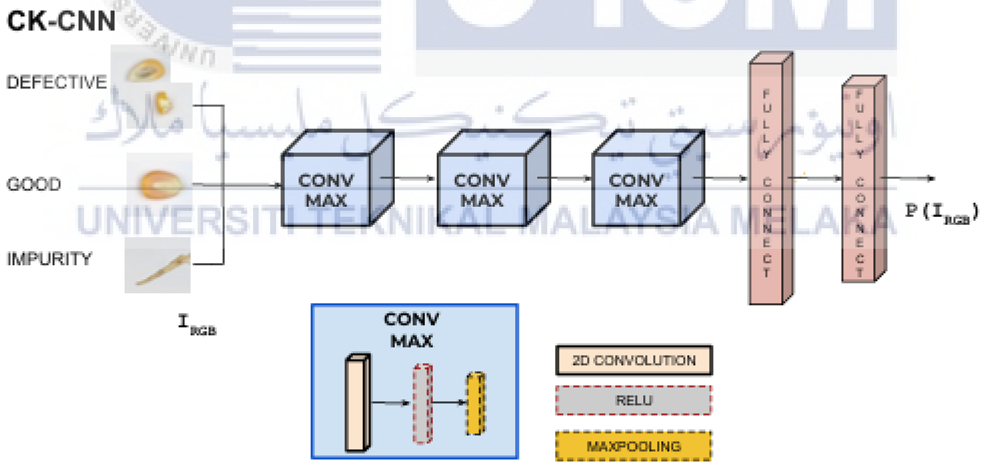

Figure 2.16 Proposed corn kernel classification network (CK-CNN)[38].

The suggested pipeline displayed excellent performance. The Mask R-CNN has an average Intersection over Union (IoU) of 0.903 for binary mask segmentation and 0.890 for instance segmentation. The CK-CNN has a classification accuracy of 97.9% for good maize, 90.0% for defective maize, and 97.3% for pollutants. This performance oumethods butther models such as VGG16, ResNet50, and Mask R-CNN classification methods, but utilizing substantially less parameters[38].

Furthermore, the segmentation and classification pipeline stood out for its efficiency and accuracy. The Mask R-CNN's strong IoU scores show precise segmentation capabilities, which are essential for correctly detecting and separating individual kernels from clusters. Following segmentation, the CK-CNN's excellent classification accuracy guarantees that each kernel is accurately recognised, increasing the system's dependability. The findings indicate that combining Mask R-CNN for segmentation and CK-CNN for classification may successfully meet the issues of maize kernel quality evaluation, resulting in a robust solution for agricultural applications. The lightweight CK-CNN's efficiency benefits emphasise the potential for real-time applications, making this method suitable for implementation in realistic situations[38].

Table 2-6 Results of classification stage for 3-classes[38].

Network	Good corn	Def. corn	Impurity	Avg. Acc	# of Net. Param.
Mask R-CNN	0.960	0.695	0.286	0.647	63738 K
VGG16	0.974	0.876	0.819	0.890	134272 k
ResNet50	0.986	0.860	0.931	0.925	23593 K
CK-CNN	**0.979**	**0.900**	**0.973**	**0.956**	**3306 K**

2.7.2 Other Machine Vision Approach

Various machine vision approaches, such as Support Vector Machines (SVM) and Artificial Neural Networks (ANN), can be very important in the field of corn detection. Using a hyperplane that is defined in a multi-dimensional space, SVM, for example, may successfully separate various corn varieties or attributes within the image. By drawing a border that separates different kinds of corn, this approach makes it easier to classify new data points as positive (corn) or negative (non-corn) depending on how they are positioned about this hyperplane. This technique makes it easier to classify different types of corn, similar to the strategy used to distinguish plastic products in Figures 2.17, and 2.18 [39].

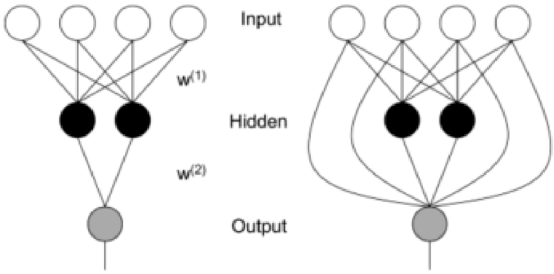

Figure 2.17 Hyperplane of SVM [39].

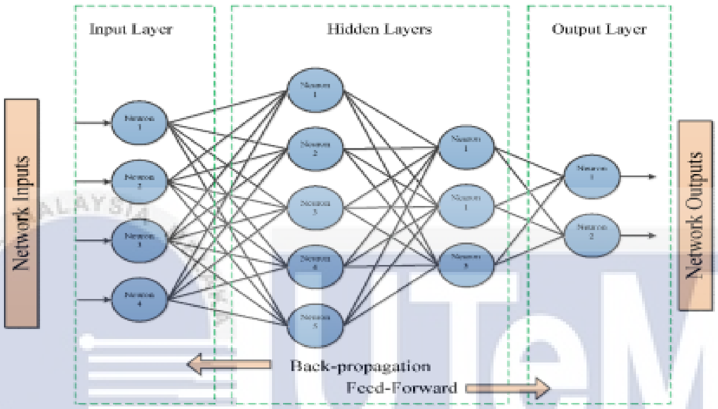

Figure 2.18 Sample Artificial Neural Network Architecture [39].

Conversely, although drawing inspiration from biological neural networks, Artificial Neural Networks (ANNs) may lack the specialized capabilities needed to tackle computer vision problems related to corn detection. In this domain, Convolutional Neural Networks (CNNs) are often more effective than classic ANNs, demonstrating higher skills to solve intricate visual tasks related to distinguishing various corn plant kinds or attributes in farm scenes [40].

The primary distinction between YOLO and SSD is that the former uses priors (anchor boxes) while the latter deals with numerous bounding boxes for the same instance of an object. Priors are pre-calculated, fixed-size boxes with a score greater than 0.5 that use the IOU methodology. They bear similarities to the original ground-truth boxes. They laid the groundwork for bounding-box regression right away. Following that, there will be less accuracy loss and more precision as the convolution model starts to regress towards the ground-truth bounding boxes [39].

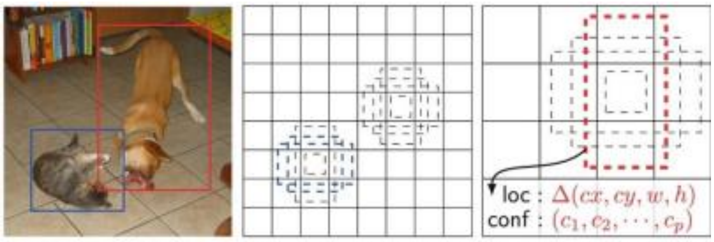

Figure 2.19 SSD framework[40].

2.8 Comparison Between Object Detection Models

In this part, Yolo, SSD, and Faster-RCNN will be evaluated for real-time vehicle type recognition [41]. However, it's important to remember that reliable comparisons across various object detectors might be difficult. There is no simple solution to the debate over which model is superior. Decisions are made in the context of real-world applications to strike a compromise between speed and accuracy. In addition to the different types of detectors, additional factors that could affect performance must also be considered. The dataset for the experiment is shown in Table 2-7.

Table 2-7 Data Set for Vehicle Type Recognition [41].

Labelling Name	TRAIN set	TEST set
car	1447	276
mini_van	244	55
big_van	33	8
mini_truck	516	163
truck	94	27
compact	286	39

Three models were trained on the same set of data: YOLO, SSD, and Faster-RCNN. YOLO v4 [42], the most recent version, to apply YOLO, and the performance was improved. Mobile Net v1 is used for (SSD), and Inception v2 is used for faster-RCNN. After comparing MAP and FPS, concluded that YOLOv4 is the most suitable model among the examined object identification techniques. As illustrated in Tables 2-8, and 2-9.

Table 2-8 Performance of YOLO v4 Model [41].

Label	Average Precision	True Positive	False Positive
car	98.08%	273	25
mini_van	94.93%	52	5
big_van	100.0%	8	0
mini_truck	99.04%	162	4
truck	98.52%	27	5
compact	98.59%	36	1

Table 2-9 Performance of Faster R-CNN Model [41].

Label	Average Precision	True Positive	False Positive
car	93.2%	262	17
mini_van	87.2%	52	41
big_van	100.0%	8	1
mini_truck	99.7%	164	16
truck	100.0%	27	2
compact	80.3%	25	10

Table 2-10 Performance of SSD Model [41]

Label	Average Precision	True Positive	False Positive
car	92.7%	257	34
mini_van	84.3%	29	1
big_van	87.5%	7	0
mini_truck	97.9%	151	0
truck	94.9%	21	5
compact	85.8%	32	15

Table 2-11 FPS of Deep Learning Models [41].

	YOLO v4	SSD	Faster-RCNN
FPS	82.1	105.14	36.32

Table 2-12 Evaluation of Deep Learning Models [41].

Models	F1score	Precision	Recall	MAP
Yolo	0.96	0.93	0.98	98.19
SSD	0.88	0.90	0.87	90.56
Faster-Rcnn	0.90	0.86	0.94	93.40

Figure 2.20 Performance of Deep Learning Model

2.9 Related Work

This section discusses the different sophisticated strategies and algorithms described in the literature for agriculture and item identification difficulties. Felzenszwalb and colleagues created a cutting-edge object detection system by combining multiscale deformable component models and latent SVM for discriminative training. He et al. proposed Region Proposal Networks (RPNs), which share convolutional characteristics with the downstream object identification network and improve efficiency by offering cost-free region proposals. Girshick and colleagues created R-CNN, which uses high-capacity convolutional neural networks to area suggestions, allowing for reliable object recognition and semantic segmentation.

In agricultural contexts, recent studies used J48 decision trees and region-based CNNs combining ResNet and Faster R-CNN to classify maize seed lots by physiological quality attributes, particularly vigour, and detect maize leaves in images with significant weed occlusion, outperforming VGG16 in mean average precision. Further research examined the YOLOv3, YOLOv4, and YOLOv5l algorithms for locating safe emergency landing places for UAVs using the DOTA dataset, evaluating both accuracy and speed. Furthermore, a technique based on form (eccentricity) and color (HSV) parameters identified by an artificial neural network was presented for determining maize seed quality, efficiently differentiating between good and poor seeds. Examining these various approaches provides insights into the strengths and limits of present tactics, as well as revealing unsolved difficulties and prospective paths for future study in agricultural technology and general object identification.

Table 2-13 Overview of Related Work

Citation	Authors	Research Content	Hardware Used	Accuracy/mAP	Algorithm Used
[36]	Yunling Liu, Chaojun Cen	Detection of maize tassels from UAV RGB imagery	UAV DJI Inspires 2 with ZENMUSE X5S camera, Mobile phone	mAP up to 94.99% for UAV images, 95.95% for mobile phone images	Faster R-CNN with ResNet and VGGNet as feature extraction networks
[38]	Henry O. Velesaca, Raúl Mira, Patricia L. Suárez	Proposed a pipeline for segmenting and classifying corn kernels into good, defective, and impurity categories. Created CORN-KERNEL dataset with annotated contours of corn kernels.	Basler ACE acA645-100gc camera, LED lamps	2-class: Avg. Acc. 0.945 (CK-CNN), 3-class: Avg. Acc. 0.956 (CK-CNN) 2-class: Avg. Acc. 0.933 (VGG16), 0.911 (ResNet50), 0.803 (Mask R-CNN); 3-class: Avg. Acc. 0.890 (VGG16), 0.925 (ResNet50), 0.647 (Mask R-CNN).	CK-CNN (custom lightweight CNN) VGG16, ResNet50, Mask R-CNN
[26]	Guojin Li, Xiaojie Huang, Jiaoyan Ai, Zeren Yi, Wei Xie	Efficient object detection method for lemons in natural environment	Inteli3-8100CPU, NVIDIA Tesla V100 GPU	96.28% AP	Lemon-YOLO (L-YOLO)
[9]	Gizele I. Gadotti et al.	Classification of corn seed lots	Not specified	Not reported	J48, RandomForest, CVR, lBk, MLP, NaiveBayes
[10]	Mohammad Ibrahim Sarker et al.	Corn leaf detection in images	Nvidia Titan X GPU	70.36% mAP (Faster R-CNN with ResNet-101)	Faster R-CNN with VGG16 and ResNet-101

Citation	Authors	Research Content	Hardware Used	Accuracy/mAP	Algorithm Used
[34]	Haddad Alwi Yafie et al.	Corn seed identification based on shape and color features	Not specified	89% (BIMA-20 Good vs Bad), 97% (BIMA-20 Good vs NASA-29 Good)	Artificial Neural Network
[33]	Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik	Object detection using R-CNN	Nvidia GPU, CPU	53.7% on VOC 2010	R-CNN with CNN feature
[32]	Pedro Felzenszwalb, Ross Girshick, David McAllester and Deva Ramanan	Object detection system using mixtures of multiscale deformable part models	Desktop computer (2.8Ghz 8-core Intel Xeon)	Not reported	Mixtures of multiscale deformable part models, Latent SVM
[35]	Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun	Faster R-CNN for real-time object detection with region proposal networks	GPU (unspecified)	73.2% on PASCAL VOC 2007, 70.4% on VOC 2012	59.9% on VOC 2007 (with ZF net)
[18]	Nepal, U.; Eslamiat, H.	Comparing YOLOv3, YOLOv4, and YOLOv5 for autonomous landing spot detection in faulty UAVs	Personal Computer: NVIDIA GeForce RTX 2070 GPU; Companion Computer: Nvidia Jetson Xavier NX	Not reported	YOLOv3: 0.46; YOLOv4: 0.607; YOLOv5l: 0.633

2.10 Research gap

Beyond yet, little emphasis has been paid to comparing the effectiveness of YOLO v8 and Mask R-CNN in the context of corn categorization based on pest or disease detection quality. Furthermore, there are few research that outline ways for corn categorization based on quality influenced by bugs or illnesses, which is critical for evaluating the improvement of contemporary agriculture. This work attempts to bridge these gaps by conducting a thorough evaluation of corn detection and classification algorithms based on their quality in terms of insect and disease presence.

2.11 Summary

The literature review chapter covers several topics related to object identification and machine vision. Begin by introducing the concept of machine vision and its many components, including informative region selection, feature extraction, and classification. Next, the chapter covers a variety of object identification models, including region proposal-based models like R-CNN, Fast R-CNN, Faster R-CNN, and Mask R-CNN, and regression/classification-based models like YOLO and SDD.

The chapter also covers terms that are often used in the evaluation of object identification models, as well as assessment metrics including accuracy, recall, and mean average precision (MAP). A comparison of several object detection models and a section on related work describing the state-of-the-art in the field are also included in this chapter. The chapter concludes with a synopsis of the major concepts discussed and identifies the need for more investigation.

CHAPTER 3

METHODOLOGY

3.1 Introduction

This part will go over the strategy and techniques used to finish this final project. First, a flowchart of the entire project procedure and an overview of the graduation outcome in an improved. The hardware of the designated microcomputer will be covered next. The system illustration must be explained and shown next. This chapter's first section will provide the reader with an overview, and the second section will go deeper into the processes that this thesis will subsequently address and assess. Section three will offer an overview of the data-gathering technique and some extra useful details regarding the provided dataset, while Section four will concentrate on streamlining and optimizing the modeling process.

3.2 Project Overview

The Project Overview in Figure 3.1 illustrates its general process. A flowchart outlining the procedures required to effectively finish this project is provided below. In this study, the YOLOv8 model will be utilized to detect corn quality. YOLOv8, which stands for "you only look once," is one of the algorithms that recognize objects faster and more accurately. This project aims to achieve excellent classification.

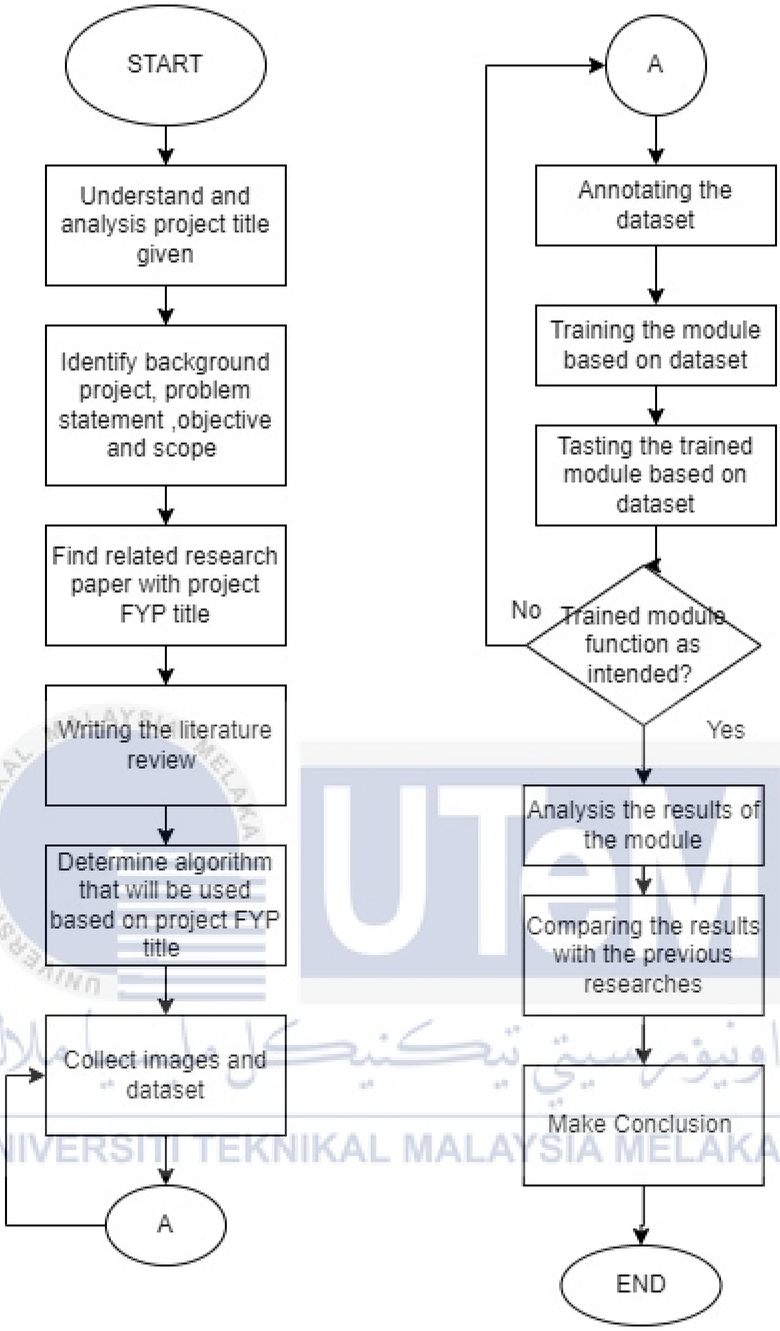

Figure 3.1 Project Overview Flowchart.

3.3 Corn Quality Detection and Recognition

Corn Quality Detection and Recognition is the process of evaluating and classifying corn crop quality via use of cutting-edge technology, mainly computer vision and artificial intelligence. To identify and categorize the several elements affecting corn quality, such as the presence of pests, illnesses, or other quality-related concerns, machine learning models and image processing techniques are used. To optimize crop output and promote better agricultural practices, the objective is to give farmers and other agricultural experts a methodical and automated way to effectively monitor, assess, and manage the quality of corn harvests.

3.3.1 Training Process YOLOv8 Using Google Colab

Figure 3.2 below provides a broad overview of the architecture and training module YOLOv8 for corn quality detection and identification utilizing the Darknet format. This flowchart is used continuously throughout the project, from its beginning to the testing of the final dataset.

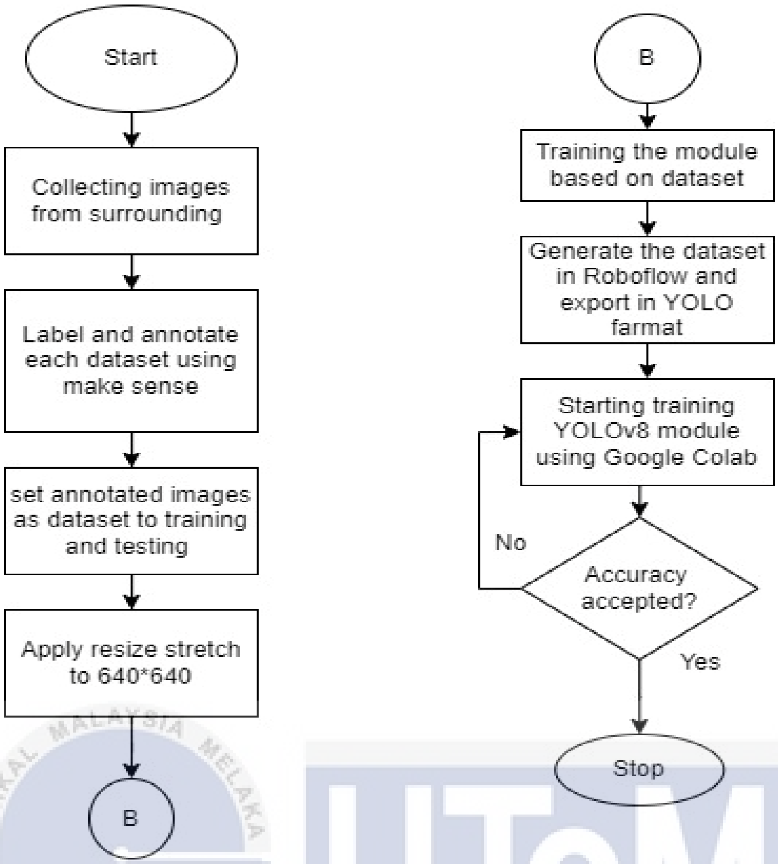

Figure 3.2 YOLOv8 Training Module.

3.4 Collecting Dataset

The 771 images in the picture collection for this project were taken using a cell phone camera on the farm and show a variety of viewpoints, including angled, above, and ground-level views. These photos are real case data taken straight from the farm, giving viewers a correct portrayal of the conditions associated with growing corn. This collection, in contrast to pictures seen online or in grocery shops, concentrates on showing real corn items in their natural environments. Eighty percent of the total photos will be used to train the detection system, and the remaining twenty percent will be used for validation and testing. Figures 3.3 and 3.4 supply selections from the gathered images. Figure 3.3 displays corn with inadequate quality, while Figure 3.4 represents high-quality corn, illustrating the diversity of conditions observed within the farm setting, and Table 3.1 the test and learning process for the collected images after being Augmented from 771 to become 1771 images.

Table 3-1 Illustrates Testing and Learning Process of a Total of 1771 Images.

Train Set	Valid Set	Test Set
85%	8%	7%

Figure 3.3 Bad Quality of Corn.

Figure 3.4 Good Quality of Corn.

3.4.1 Image Annotation

The photos in a dataset must be tagged and annotated for the data to be used. To enable the module to understand the intended result in this example, the identification of corn based on its size the technique aims to assign a class to each type of detected item. For labeling reasons, this study divided corn varieties into three categories: Healthy, Water Rot, and Bug. While there are other kinds of corn available, sweet corn is the one that is most commonly used nowadays.

3.4.2 Data Augmentation

Rotation, flipping, and scaling were some of the techniques used to increase the dataset to improve the model's performance and prevent overfitting. Following augmentation, the total number of photos was 1771. The augmentation approaches applied to the dataset contributed to the improvement of the model's robustness and variety by adding changes in the photographs.

3.4.3 Roboflow

Roboflow is a system that facilitates the creation of machine learning datasets. It's an online program that facilitates the organization, annotation, and enhancement of picture collections. In this study, a unique collection of concrete surfaces with various defects and fissures was collected using Roboflow. Because the dataset was already tagged, a significant amount of time and effort were saved. Furthermore, the Roboflow platform included data augmentation capabilities that expanded the dataset and enhanced the functionality of the model. All things considered, Roboflow greatly simplified the process of obtaining and preparing the dataset for this project. Figure 3.5 User interface For Roboflow.

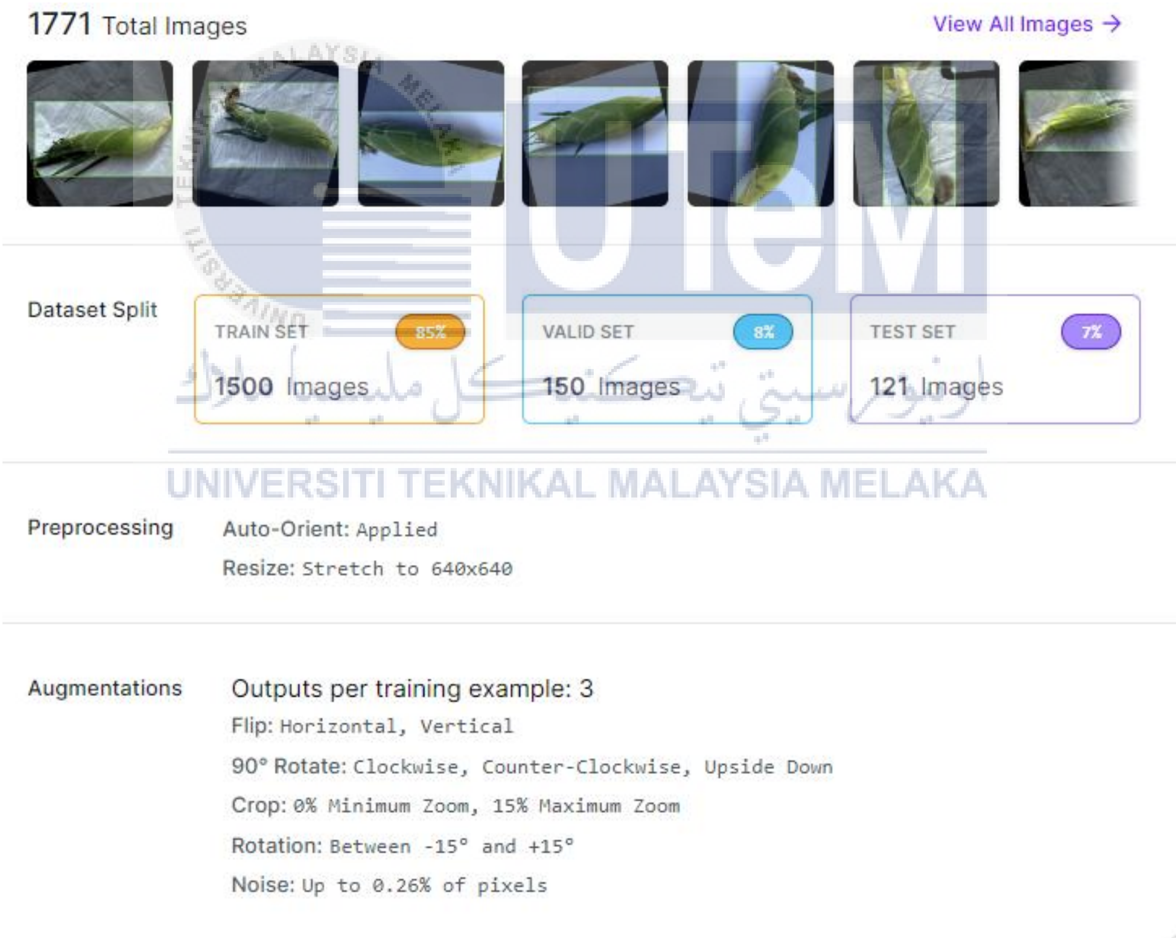

Figure 3.5 User Interface for Roboflow.

3.5 Building Machine Learning Model

To identify and separate corn, convolutional neural networks are created as part of the machine learning model-building process. The YOLO v8 method was chosen as the foundation model for this project's implementation since it is a real-life object recognition system with several applications, including autonomous driving, photo editing, and medical image analysis.

The following actions were taken to construct the model:

1. First, the Ultralytics dependencies were installed to prepare the environment for training as shown in Figure 3.6.

```
%pip install ultralytics
import ultralytics
ultralytics.checks()
```

```
Ultralytics YOLOv8.2.27 🚀 Python-3.10.12 torch-2.3.0+cu121 CUDA:0 (Tesla T4, 15102MiB)
Setup complete ✅ (2 CPUs, 12.7 GB RAM, 30.3/201.2 GB disk)
```

Figure 3.6 Installing the Dependencies.

2. Second, this code snippet imports the necessary modules, clears the output, and executes a YOLO-specific command to check the mode or configuration of the YOLO object detection system as shown in Figure 3.7.

```
from ultralytics import YOLO
import os
from IPython.display import display, Image
from IPython import display
display.clear_output()
!yolo mode=checks
```

Figure 3.7 Importing YOLOv8 From the Library.

3. Next, the custom dataset obtained from Roboflow was loaded in the YOLO v8 format as shown in Figure 3.8.

```
!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="FQIPEpQOenHCfa5oK7RG")
project = rf.workspace("monther-alasbahi").project("corn-detection-o7hls")
version = project.version(7)
dataset = version.download("yolov8")
```

Figure 3.8 Downloading the Correctly Formatted Data.

4. The model was then evaluated using the validation dataset to ensure that it generalizes well on unseen data. This process is done by measuring the performance metrics such as precision, recall, and mean average precision (MAP).

5. Finally, the model's performance was evaluated on test images by running the YOLO v8 inference, this process involves applying the trained model on test images and measuring the performance metrics as shown in Figure 3.9.

```
!yolo train model=yolov8n.pt data=/content/Corn-Detection-7/data.yaml epochs=100 imgsz=640
```

Figure 3.9 Testing the Module with 100 Epochs Using Test Images.

3.6 Applications, Libraries, and Tools

With the use of GPU assistance for training, this work proposes a real-life corn identification and segmentation approach based on the YOLO v8 deep learning model. It allows for the quick and hardware-free assessment of corn quality using a regular laptop camera.

3.6.1 Python Language

Web development, machine learning, data science, and other diverse fields all heavily rely on high-level, open-source Python programming. Python was the primary programming language used in this project, and it was used to implement the suggested method for using a convolutional neural network to detect and segment concrete fractures in real life.

One of the main advantages of using Python is the huge ecosystem of libraries and tools available for different jobs. In this project, the recommended technique was implemented utilizing a variety of well-known Python packages, including**:**

- **TensorFlow:** Google developed TensorFlow, an open-source machine learning library. It is widely employed in the creation and training of deep learning models. TensorFlow was used in this project to build the YOLO v7 approach, and the dataset was used to enhance the model.
- **OpenCV**: The open-source computer vision library known as OpenCV provides a range of image processing and computer vision functionalities. OpenCV was used in this project for both data augmentation and picture preparation.
- **NumPy:** The robust numerical computation toolkit NumPy supports multidimensional arrays and matrices. It handled picture data and performed mathematical calculations on the data in this project.

These libraries were essential to the project's success. They supplied the resources and expertise required to implement the recommended method, train, and evaluate the model. TensorFlow was used to build the model, OpenCV was used for image preprocessing and data augmentation, and NumPy was used for mathematical operations on the data. Overall, using Python and these dependable modules greatly expedited the development and implementation of the recommended solution.

```python
import os
import cv2
import secrets
import string
from ultralytics import YOLO
import datetime

def generate_random_string(length):
    alphabet = string.ascii_uppercase + string.digits
    return ''.join(secrets.choice(alphabet) for i in range(length))

cap = cv2.VideoCapture(0)

if not cap.isOpened():
    print("Error opening video file.")
    exit()

H, W = None, None
model = YOLO('best12.pt')
threshold = 0.6
capture_interval = 1  # Interval for capturing images in seconds
last_capture_time = datetime.datetime.now()  # Initialize last_capture_time as a datetime object

while True:
    ret, frame = cap.read()
    if not ret:
        break

    current_time = datetime.datetime.now()
    results = model(frame)[0]
    capture_flag = False
```

Figure 3.10 Code Used to Test the Process.

To verify the performance of our corn condition detection algorithm, we used PyCharm, a powerful integrated development environment (IDE), in a real situation. Our technique relied on the YOLO (You Only Look Once) object identification algorithm, which is known for its speed and accuracy in real-time applications. The model was fine-tuned with a pre-trained weights file called best12.pt, which was particularly designed to categorize corn into three categories: healthy, water rot, and bug.

In our trial setup, we used the default camera to collect a live video stream that served as input for our detection algorithm. The object detection confidence level was set at 0.6. This number was carefully determined to strike a compromise between precision and recall, ensuring that the model accurately recognizes corn conditions while minimizing false positives.

The live camera stream was analyzed continually, allowing for real-time detection and categorization of the corn's health state. This approach enabled us to monitor the model's performance under a variety of scenarios and tweak settings as needed to improve accuracy. The dynamic nature of live testing revealed important information about the model's robustness and dependability in actual circumstances.

Overall, the usage of PyCharm for testing, together with the robust YOLO detection framework, revealed the model's capacity to correctly and effectively recognize various corn situations. This technique not only tested the model's performance, but also demonstrated its viability in real-world agricultural applications.

3.6.2 Google Colab

The free online application Google Colab provides an easy-to-use interface for developing and refining machine learning models. Users may use it to run and execute code as well as develop machine learning models. This study used Google Colab as the training environment to build the proposed convolutional neural network model for real-life corn detection and segmentation. It made it feasible to train the deep learning model using the YOLO v8 method using high-performance computer resources, such as GPU support.

3.6.3 Microcomputer Camera

The approach comprised taking pictures and classifying them using a typical MSI laptop's built-in camera and its specification GF63 Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz -64-bit ,x64-based processor. . Although there wasn't a dedicated camera module like the Arducam Auto Focus Camera made for Raspberry Pi, the laptop's camera performed the crucial job of taking the pictures needed to classify the corn's quality. This method allowed for real-life detection by using code to turn on the laptop's camera, and it made it possible to assess the performance of the trained model right away without requiring any additional hardware.

3.7 Common Terms in Object Detection Models Evaluation

To evaluate the outcomes of the various deep learning-based models, we adopted a range of standard measures that are commonly used in the evaluation process of machine learning models.

3.7.1 Intersection Over Union (IOU)

The Intersection over Union metric is often used to assess the accuracy of localization and identify the faults that localization adds to object identification models. To compute the IOU using the predictions and the ground truth, we must first find the region that sits at the intersection of the bounding boxes about a particular prediction and the bounding boxes relating to the ground truth for the same region. The Union, or the total area included by the two bounding boxes, is then computed.

As we can see in Figures 3.11 and 3.12, the ratio of the overlap to the total area which is derived by dividing the intersection by the union gives us a reliable indication of how closely the bounding box resembles the original forecast.

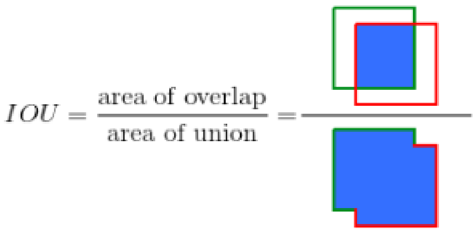

Figure 3.11 Equation For IOU.

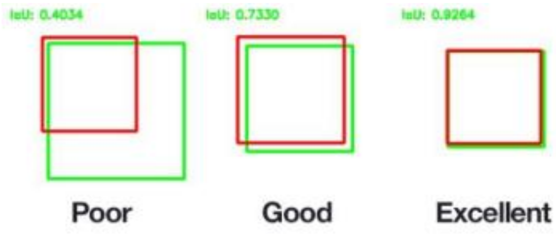

Figure 3.12 Evaluating the Performance of the Predicted BBs.

3.7.2 True Positive, False Positive, False Negative, and True Negative

When the output categories of the test samples are compared to the categories of the real labels, four outcomes are possible. These results are known as **true positives (TP), false positives (FP), false negatives (FN), and true negatives (TN)** [43].

Can set an IOU threshold value to indicate whether the object detection was successful. In this instance, if the IOU is more than or equal to 0.5, the object detection will be considered a TP. Suppose for the moment that the IOU is set to 0.5. If the IOU is less than 0.5, which suggests that the detection was made wrongly, we will refer to it as FP. A result should be labeled as false positive (FN) if there is a ground truth in the picture, but the model is unable to locate the object. TN refers to any region of the image where we did not expect an item to be present. Considering that this measure does not.

3.7.3 Precision, Recall, and Mean Average Precision (MAP)

The observed statistics are used to compute common measures such as recall, precision, mean average precision (MAP), F1 score (F1), and frames per second (FPS). Recall is the ratio of identified targets to all targets in the sample set when it comes to the target detection process, while precision is the ratio of accurately detected targets to all targets that have been detected [43]. Equations 1 and 2 provide definitions of precision and recall, respectively:

$$\text{Precision} = \frac{TP}{TP + FP} \qquad (1)$$

$$\text{Recall} = \frac{TP}{TP + FN} \qquad (2)$$

The harmonic weighted average of recall and precision is represented by the letter F1. Equation 3[43] provides the following description of the F1 score, which is computed using the precision and recall rates:

$$F1 = \frac{2PR}{P + R} \qquad (3)$$

Equation 4 defines "average precision" (AP) as the level of accuracy reached across all elements of a specific category [43]:

$$\text{AP} = \int_0^1 P(r)\,dr \qquad (4)$$

The overall performance of the model is evaluated using MAP, which is computed as the average value of the AP total across all categories [43]. Equation 5 illustrates the definition, which is as follows:

$$\text{MAP} = \frac{1}{N}\sum_{i=1}^{N} APi \qquad (5)$$

CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 Introduction

The first results of the proposed method for real-life corn detection are presented in this chapter. The system's output is covered in great length in this section. Training and validation loss metrics are among the assessment criteria that are used to determine how successful the proposed model is. In addition, the system's ability to recognize and categorize corn will be tested using a set of test images.

4.2 Performance Evaluation of the Proposed Model

```
      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
    100/100      2.26G     0.5799     0.3462      1.143         10        640: 100% 66/66 [00:22<00:00,  2.90it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 8/8 [00:04<00:00,  1.69it/s]
                   all        243        353      0.553      0.713       0.63      0.417

100 epochs completed in 0.829 hours.
Optimizer stripped from runs/detect/train2/weights/last.pt, 6.3MB
Optimizer stripped from runs/detect/train2/weights/best.pt, 6.3MB

Validating runs/detect/train2/weights/best.pt...
Ultralytics YOLOv8.2.27 🚀 Python-3.10.12 torch-2.3.0+cu121 CUDA:0 (Tesla T4, 15102MiB)
Model summary (fused): 168 layers, 3006233 parameters, 0 gradients, 8.1 GFLOPs
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 8/8 [00:06<00:00,  1.17it/s]
                   all        243        353      0.578      0.665       0.65      0.447
                   Bug         83        193      0.416       0.56      0.448      0.242
               healthy         84         84      0.811      0.976      0.942      0.746
             water rot         76         76      0.507       0.46       0.56      0.352
Speed: 0.6ms preprocess, 3.5ms inference, 0.0ms loss, 3.6ms postprocess per image
Results saved to runs/detect/train2
```

Figure 4.1 Results Training has been Completed for YOLOv8.

Figure 4.1 shows in detail the training results obtained by using the YOLOv8 model in the Google Colab environment. YOLOv8 is well-known for its ability to use advanced deep learning algorithms to identify objects in photos quickly and correctly. Researchers may benefit from a smooth training experience across a wide range of datasets by leveraging the capabilities of Google Colab, an accessible cloud computing platform. The execution of the command "python train.py" with meticulously tailored arguments such as "data=/content/Corn-Detection-7/data.yaml" and "--weights best12.pt" signals the start of the machine learning training process, emphasizing the incorporation of pre-trained weights to facilitate accelerated model convergence.

Furthermore, the defined training regimen, which has a batch size of 16 and runs for 100 epochs, represents a concerted effort towards complete model optimization and rigorous performance evaluation. This collaborative approach to training incorporates critical parts of the machine learning model training paradigm, enabling not just effective resource utilization but also robust performance improvement. Researchers successfully optimize model convergence and improve detection accuracy by judiciously using transfer learning approaches and rigorous parameter adjustments. This rigorous approach, along with the versatility of Google Colab as an instrumental teaching platform, is a huge step towards promoting creativity and growth in the field of computer vision research.

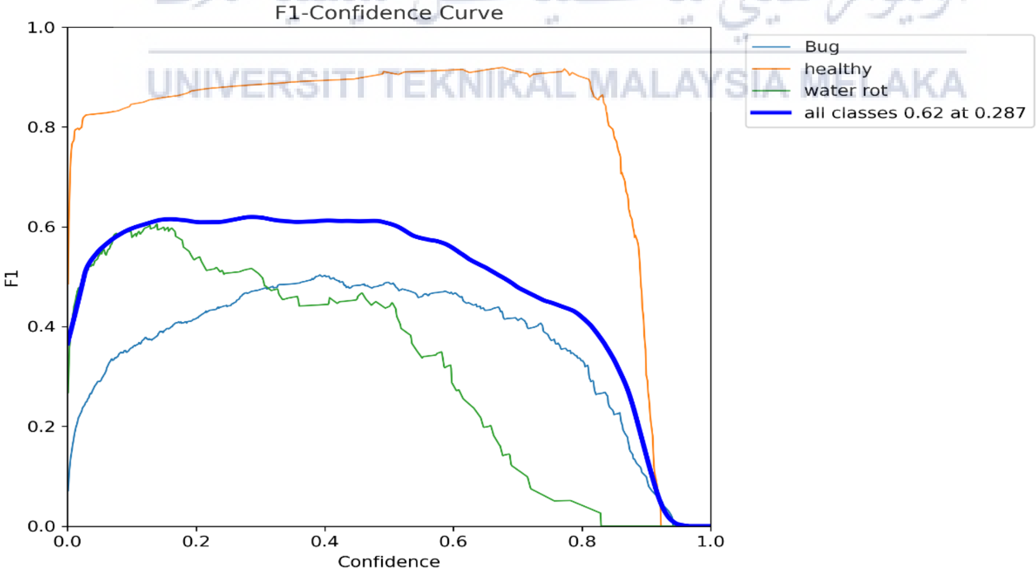

Figure 4.2 F1-Confidence Curve.

Figure 4.2 Shows graphical depiction of a classifier's performance at various confidence levels, which aids in determining how effectively the model discriminates between classes. In our study, we examined the efficacy of our corn condition identification algorithm using this curve, which displays the F1 score against various confidence levels. The F1 score strikes a compromise between precision (accurate positive predictions) and recall (ability to recognise all positive cases). Our findings revealed that the "Healthy" class performed consistently well across several thresholds, demonstrating trustworthy identification. In contrast, the "Bug" and "Water Rot" classes peaked at specified levels, demonstrating their sensitivity to the selected threshold.

Overall, the model performed best, with an F1 score of 0.62 and a confidence threshold of 0.287. This threshold indicates the ideal equilibrium, with the model achieving the best-combined accuracy and recall across all classes. Identifying this ideal threshold is critical for adjusting the model and maximizing its efficacy. The F1-Confidence Curve gives a complete perspective of the model's performance and directs the fine-tuning process to achieve optimal detection accuracy, as well as assisting in the establishment of confidence thresholds that improve the model's dependability in real-world applications.

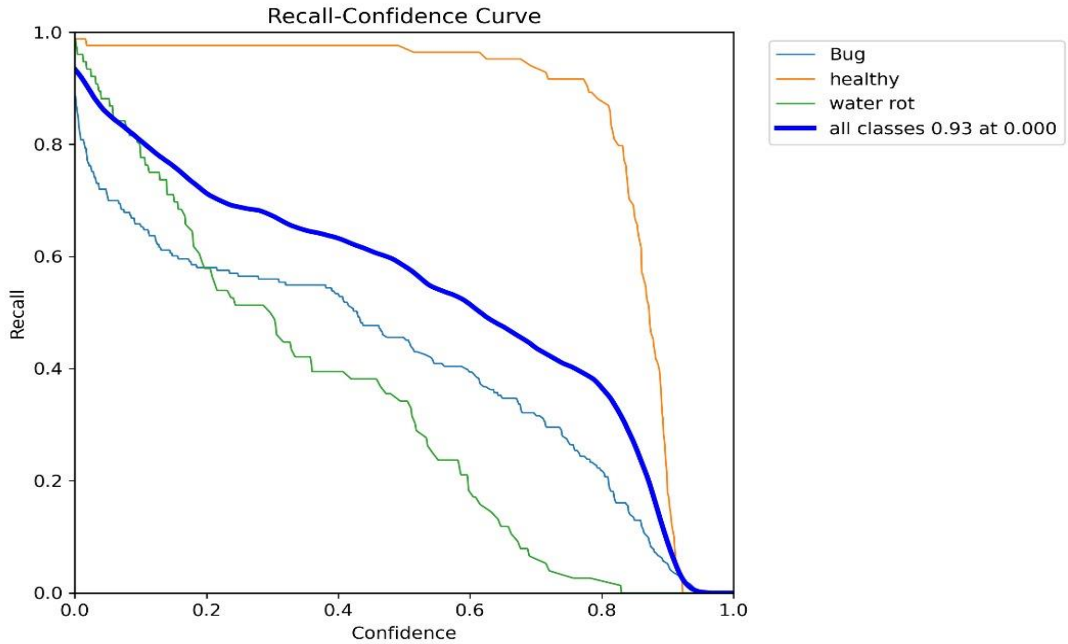

Figure 4.3 Recall-Confidence Curve

Figure 4.3 depicts the classifier's recall at different confidence levels, providing information on its sensitivity to spotting positive examples. In our investigation, the curve for the "Healthy" class (orange) had consistently high recall across all thresholds, demonstrating that the model can reliably detect healthy corn. Conversely, recall for the "Bug" (blue) and "Water Rot" (green) classes decreased consistently as the confidence threshold increased, indicating a loss in sensitivity for these situations at higher thresholds.

The model's overall performance, as depicted by the average recall (thick blue line), was most impressive at a confidence level of 0.000 when it attained an average recall of 0.93. This high recall at the lowest threshold indicates maximal sensitivity, implying that the model correctly recognizes virtually all occurrences of the target classes when the threshold is low. The Recall-Confidence Curve therefore gives a thorough perspective of how recall varies with confidence levels, assisting in the selection of acceptable thresholds to balance sensitivity and specificity for various application requirements.

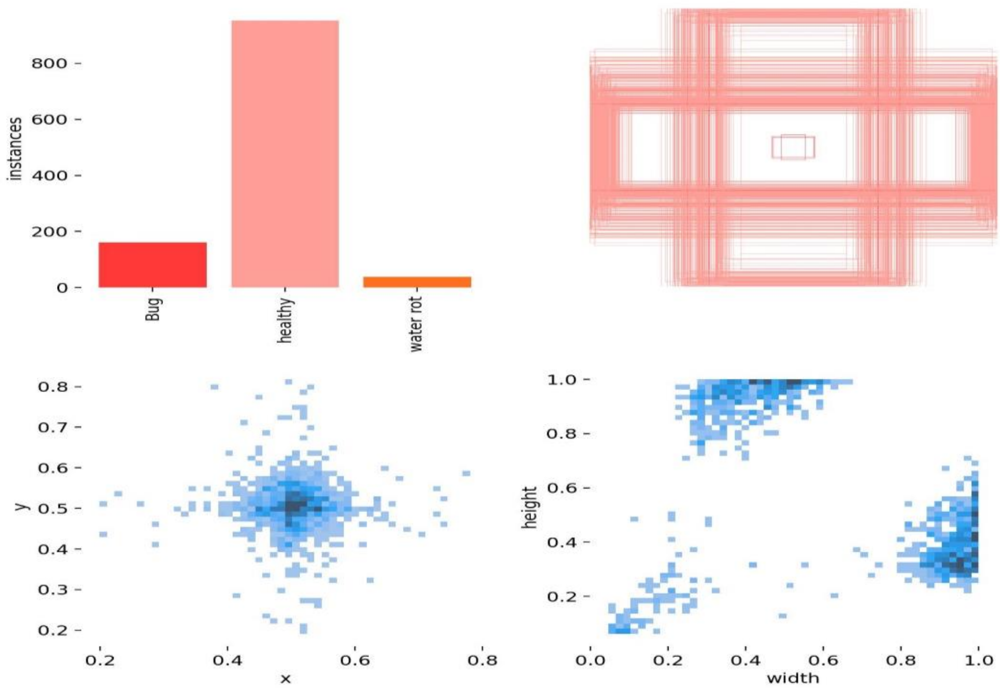

Figure 4.4 Visualizations Illustrate Width, Height, and instances.

The visualizations provide a concise yet thorough summary of class distribution and detection outcomes. The bar chart graphically depicts the frequency of "Healthy," "Bug," and "Water Rot" events, directing model training and evaluation procedures. In addition, bounding box distributions and scatter plots give information on item placements, sizes, and localization accuracy, which is important for fine-tuning detection algorithms and enhancing overall model performance. Together, these visualizations provide data scientists with meaningful information for iteratively improving object detection algorithms effectively and efficiently.

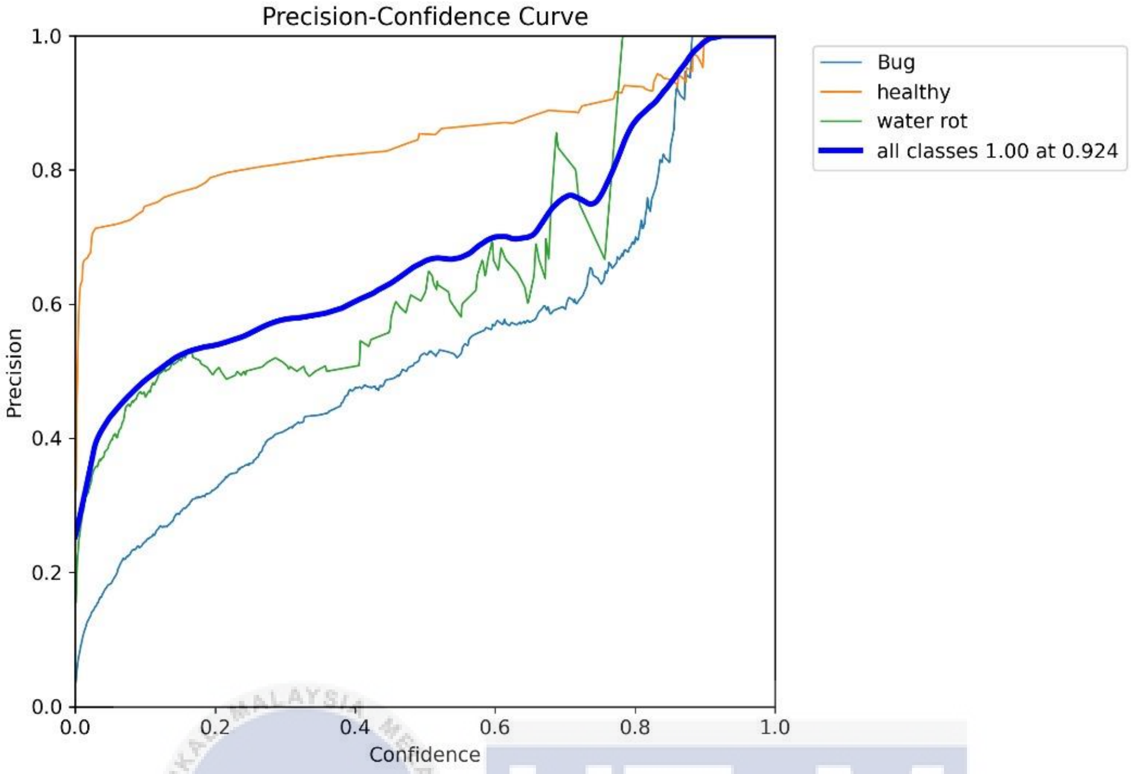

Figure 4.5 Precision-Confidence Curve.

Figure 4.5 gives extensive information on the model's precision at different confidence levels, particularly when all classes are considered. The model achieves perfect accuracy (1.00) at a confidence level of 0.924, indicating strong confidence in its predictions. Within individual classes, the healthy class constantly maintains a high accuracy rate, as seen by the orange curve. However, the Bug (blue) and Water Rot (green) classes show higher fluctuation in precision across different confidence levels. This variability indicates that the model's dependability in predicting these classes varies, with greater uncertainty than in the healthy class. Overall, the curve demonstrates the model's ability to forecast the healthy class with high precision, as well as areas for possible development in predicting Bug and water rot occurrences.

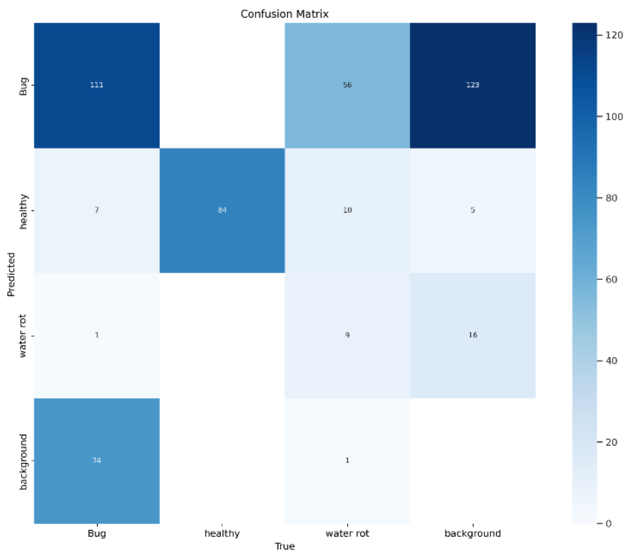

Figure 4.6 Confusion matrix

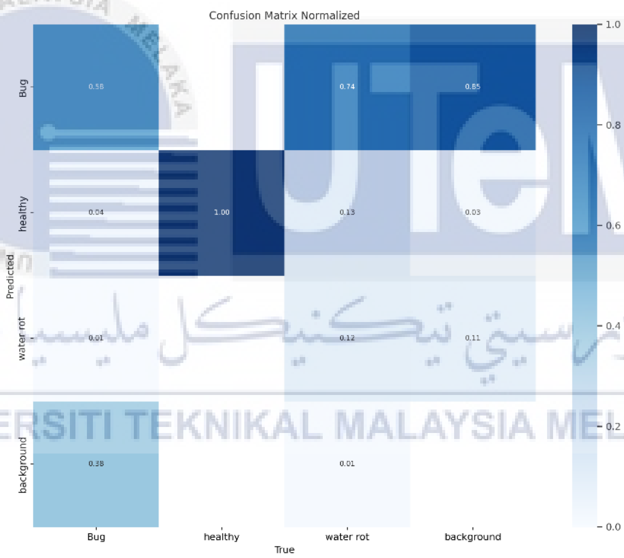

Figure 4.7 Confusion Matrix Normalized

Figures 4.6 and 4.7 depict a confusion matrix and its normalized counterpart, which are used to assess the performance of a classifier model. The first graphic shows the raw counts of true and anticipated classifications across four categories: Bug, Healthy, Water Rot, and Background. For example, 111 cases in the Bug category were accurately identified as Bug, whereas 56 were incorrectly labelled as Healthy, 1 as Water Rot, and 74 as Background. The second figure is the normalized confusion matrix, in which the counts are transformed into proportions, reflecting relative performance. In this matrix, the Bug class has a 58% accurate classification rate, with considerable misclassifications to Healthy (4%), Water Rot (1%), and Background (38%). The Healthy class gets flawless 100% accuracy. The Water Rot and Background classes had lower accurate classification percentages (12% and 11%, respectively), resulting in significant misclassifications into other categories. Normalisation allows us to assess the model's accuracy and mistakes in relative terms, regardless of the number of examples in each category.

Finally, the model was put through a thorough assessment procedure where it was tested against a carefully selected collection of test photos. The exhaustive results and conclusions that arise from this assessment stage are carefully recorded and illustrated in Figure 4.8 and Table 4.1 shows the summary of data. The report's latter sections elaborate on the finer points and perceptive insights that emerged from this critical assessment, offering a thorough examination of the model's functionality and effectiveness in corn quality identification and detection.

Table 4-1 Results Training YOLOv8.

Type of quality								
Healthy			**Water Rot**			**BUG**		
True positive	False positive	Precision %	True positive	False positive	Precision %	True positive	False positive	Precision %
3	2	88%	1	3	65.5%	40	11	100%
Total Precision %			Recall %			F1 score %		
82%			91.6%			80%		
IoU %			mAP %			Epochs		
52.3%			92.7%			100		

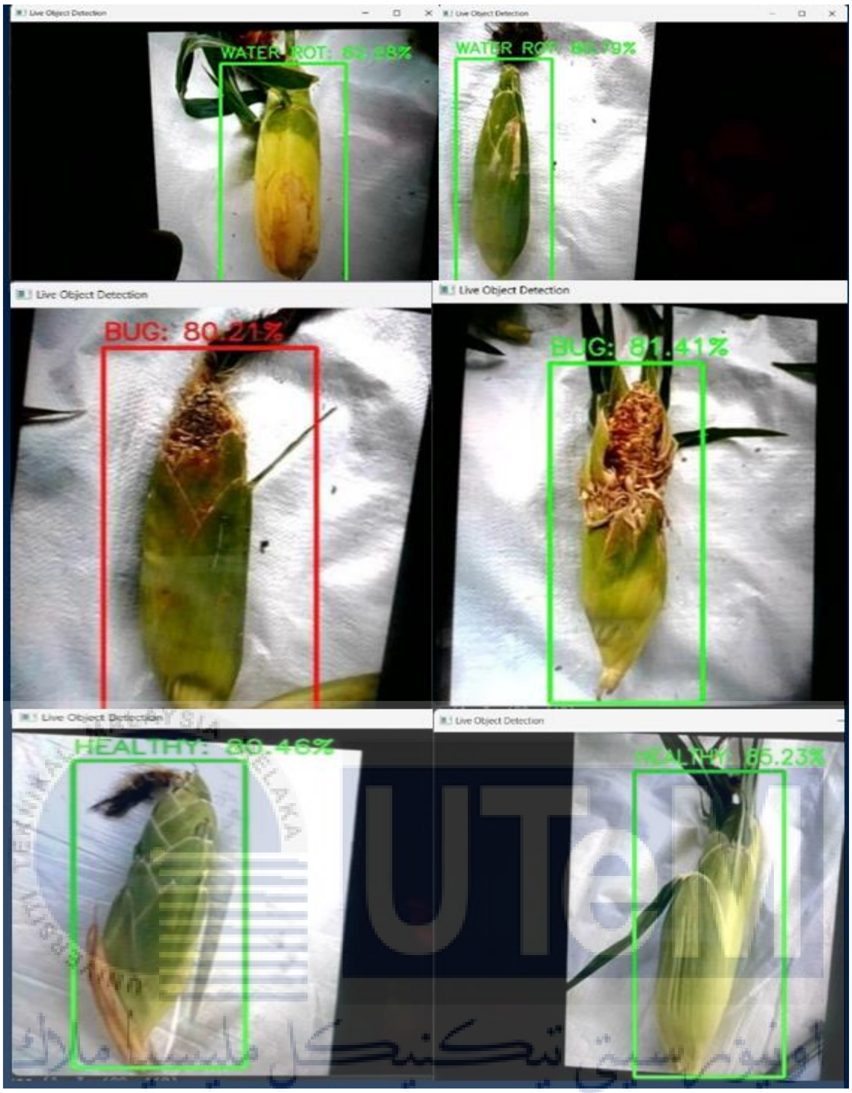

Figure 4.8 Examples of the Tested Images.

4.3 Frames Per Second on Different GPU

Graphics Processing Units (GPUs), such as the Tesla T4 that has been used in this project, improve YOLOv8's object recognition capabilities by providing parallel computing capacity, which is critical for processing huge datasets in real-time applications. YOLOv8, which uses convolutional neural networks (CNNs), depends on GPUs to quickly scan and analyze photos and videos, resulting in fast object recognition. The Tesla T4 GPU, a popular choice for inference workloads, strikes a compromise between performance and cost, resulting in competitive FPS for YOLOv8. Specifically, the Tesla T4 enables high-speed inference, making YOLOv8 appropriate for real-time video surveillance, smart city infrastructure, and industrial automation. YOLOv8's FPS performance on the Tesla T4 demonstrates its capacity to manage large amounts of data effectively, even on mid-range GPUs, providing flexibility deployment across many circumstances while retaining rapid.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

Finally, the suggested study successfully showed the use of convolutional neural networks and the YOLOv8 algorithm to detect corn quality in real-world circumstances. The YOLOv8 model was trained and evaluated in the Google Colab environment, and the results were substantial across a variety of criteria. The dataset included 771 annotated photos, with 85% set out for training, 8% for validation, and 75 for testing, guaranteeing a thorough and robust training procedure.

The findings show that the model has an overall accuracy of 92.4%. Specifically, the model had an accuracy of 88% for the "Healthy" class, 65.5% for the "Water Rot" class, and 100% for the "Bug" class. The overall accuracy for all classes was 82%, with a recall of 91.6% and an F1 score of 80%. Over the duration of 100 epochs, the mean Average Precision (mAP) was determined to be 92.7%, with an Intersection over Union (IoU) of 52.3%. These measures demonstrate the model's ability to effectively detect and categorize corn quality.

The YOLOv8 model's outstanding performance may be due to its sophisticated design, which expertly catches complex characteristics inside photos. Despite some heterogeneity in precision among classes, the model's overall excellent precision and recall rates demonstrate its dependability and durability. The findings demonstrate the potential of this technique to considerably improve agricultural monitoring operations by delivering faster, more accurate, and cost-effective solutions.

In summary, the research successfully demonstrated the practical application of CNNs with the YOLOv8 algorithm for real-world corn identification, hence meeting the project's objectives. This result not only supports the use of sophisticated deep learning algorithms for object recognition, but also paves the way for future advancements in domains that require accurate and efficient detection skills.

5.2 Future Work

A number of enhancements are expected to increase the YOLOv8 model's performance and accuracy in detecting corn. Increasing the dataset size is a top objective for improving robustness and accuracy. Using an external high-resolution camera will allow you to capture finer details during detection, resulting in better outcomes. Additionally, creating a more advanced training environment will improve accuracy and efficiency.

Improving data augmentation techniques is also important. The model's capacity to handle data changes may be enhanced by artificially increasing the dataset via cropping, flipping, rotating, and introducing noise. Collecting datasets under a variety of situations, such as variable object orientations and low-light settings, will assure the model's adaptability and dependability in a wide range of scenarios.

These tactics attempt to greatly enhance the performance of the YOLOv8 model, making it a more accurate and reliable tool for detecting corn quality in the real world.

REFERENCES

[1] J. Wang and X. Hu, "Research on corn production efficiency and influencing factors of typical farms: Based on data from 12 corn-producing countries from 2012 to 2019," *PLoS One*, vol. 16, no. 7 July, Jul. 2021, doi: 10.1371/journal.pone.0254423.

[2] M. P. Arakeri and Lakshmana, "Computer Vision Based Fruit Grading System for Quality Evaluation of Tomato in Agriculture industry," in *Procedia Computer Science*, Elsevier B.V., 2016, pp. 426–433. doi: 10.1016/j.procs.2016.03.055.

[3] T. Reda, P. Thavarajah, R. Polomski, W. Bridges, E. Shipe, and D. Thavarajah, "Reaching the highest shelf: A review of organic production, nutritional quality, and shelf life of kale (Brassica oleracea var. acephala)," *Plants People Planet*, vol. 3, no. 4. Blackwell Publishing Ltd, pp. 308–318, Jul. 01, 2021. doi: 10.1002/ppp3.10183.

[4] M. Mondal and L. N. Satpati, "Changing character of pool-riffle sequence: a quantitative representation of long profile of Ichamati, India by." [Online]. Available: https://www.researchgate.net/publication/305774779

[5] M. K. Dhillon, V. K. Kalia, and G. T. Gujar, "Insect-pests and their management: Current status and future need of research in quality maize," in *Maize: Nutrition Dynamics and Novel Uses*, Springer India, 2013, pp. 95–103. doi: 10.1007/978-81-322-1623-0_8.

[6] G. A. Silva *et al.*, "Yield losses in transgenic Cry1Ab and non-bt corn as assessed using a crop-life-table approach," *J Econ Entomol*, vol. 111, no. 1, pp. 218–226, Feb. 2018, doi: 10.1093/jee/tox346.

[7] D. Czarnecka, A. Czubacka, M. Agacka-Mołdoch, A. Trojak-Goluch, and J. Księżak, "The Occurrence of Fungal Diseases in Maize in Organic Farming Versus an Integrated Management System," *Agronomy*, vol. 12, no. 3, Mar. 2022, doi: 10.3390/agronomy12030558.

[8] R. A. C. Jones and M. J. Barbetti, "Influence of climate change on plant disease infections and epidemics caused by viruses and bacteria," *CAB Reviews: Perspectives in Agriculture, Veterinary Science, Nutrition and Natural Resources*, vol. 7. 2012. doi: 10.1079/PAVSNNR20127022.

[9] G. I. Gadotti, N. A. B. Moraes, J. G. da Silva, R. de M. Pinheiro, and R. de C. M. Monteiro, "PREDICTION OF RANKING OF LOTS OF CORN SEEDS BY ARTIFICIAL INTELLIGENCE," *Engenharia Agricola*, vol. 42, no. 4, 2022, doi: 10.1590/1809-4430-Eng.Agric.v42n4e20210005/2022.

[10] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, International Conference on Learning Representations, ICLR, 2015.

[11] H. A. Yafie, E. Rachmawati, E. Prakasa, and A. Nur, "Corn Seeds Identification Based on Shape and Colour Features Assessment Institutes for Agricultural Technology of Gorontalo Ministry of Agriculture of the Republic of Indonesia Gorontalo," 2020.

[12] K. Zhao, L. Zhao, Y. Zhao, and H. Deng, "Study on Lightweight Model of Maize Seedling Object Detection Based on YOLOv7," *Applied Sciences (Switzerland)*, vol. 13, no. 13, Jul. 2023, doi: 10.3390/app13137731.

[13] K. Wu, M. Zhang, G. Wang, X. Chen, and J. Wu, "A Continuous Single-Layer Discrete Tiling System for Online Detection of Corn Impurities and Breakage

Rates," *Agriculture (Switzerland)*, vol. 12, no. 7, Jul. 2022, doi: 10.3390/agriculture12070948.
[14] D. I. Patrício and R. Rieder, "Computer vision and artificial intelligence in precision agriculture for grain crops: A systematic review," *Computers and Electronics in Agriculture*, vol. 153. Elsevier B.V., pp. 69–81, Oct. 01, 2018. doi: 10.1016/j.compag.2018.08.001.
[15] A. F. Gad, *Practical Computer Vision Applications Using Deep Learning with CNNs: With Detailed Examples in Python Using TensorFlow and Kivy*. Apress Media LLC, 2018. doi: 10.1007/978-1-4842-4167-7.
[16] B. Naveen *et al.*, "Detection Brain Tumor Using CNN and YOLO," 2023, doi: 10.32628/IJSRST.
[17] Q. Zhang, J. Lu, and Y. Jin, "Artificial intelligence in recommender systems," *Complex and Intelligent Systems*, vol. 7, no. 1, pp. 439–457, Feb. 2021, doi: 10.1007/s40747-020-00212-w.
[18] U. Nepal and H. Eslamiat, "Comparing YOLOv3, YOLOv4 and YOLOv5 for Autonomous Landing Spot Detection in Faulty UAVs," *Sensors*, vol. 22, no. 2, Jan. 2022, doi: 10.3390/s22020464.
[19] J. R. Terven and D. M. Cordova-Esparza, "A COMPREHENSIVE REVIEW OF YOLO ARCHITECTURES IN COMPUTER VISION: FROM YOLOV1 TO YOLOV8 AND YOLO-NAS A PREPRINT," 2024.
[20] R. C. Ucat and J. C. Dela Cruz, "POSTHARVEST GRADING CLASSIFICATION OF CAVENDISH BANANA USING DEEP LEARNING AND TENSORFLOW."
[21] F.-E.-M. Baharum and B. Baharudin, "MEAN AND STANDARD DEVIATION FEATURES OF COLOR HISTOGRAMUSING LAPLACIAN FILTER FOR CONTENT-BASED IMAGE RETRIEVAL," *J Theor Appl Inf Technol*, vol. 15, no. 1, 2011, [Online]. Available: www.jatit.org
[22] D. Ireri, E. Belal, C. Okinda, N. Makange, and C. Ji, "A computer vision system for defect discrimination and grading in tomatoes using machine learning and image processing," *Artificial Intelligence in Agriculture*, vol. 2, pp. 28–37, Jun. 2019, doi: 10.1016/j.aiia.2019.06.001.
[23] Z. Q. Zhao, P. Zheng, S. T. Xu, and X. Wu, "Object Detection with Deep Learning: A Review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11. Institute of Electrical and Electronics Engineers Inc., pp. 3212–3232, Nov. 01, 2019. doi: 10.1109/TNNLS.2018.2876865.
[24] H. Loghashankar and H. Nguyen, "Real-Time Traffic Sign Detection: A Case Study in a Santa Clara Suburban Neighborhood."
[25] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection." [Online]. Available: http://pjreddie.com/yolo/
[26] G. Li, X. Huang, J. Ai, Z. Yi, and W. Xie, "Lemon-YOLO: An efficient object detection method for lemons in the natural environment," *IET Image Process*, vol. 15, no. 9, pp. 1998–2009, Jul. 2021, doi: 10.1049/ipr2.12171.
[27] W. Chen, J. Zhang, B. Guo, Q. Wei, and Z. Zhu, "An Apple Detection Method Based on Des-YOLO v4 Algorithm for Harvesting Robots in Complex Environment," *Math Probl Eng*, vol. 2021, 2021, doi: 10.1155/2021/7351470.
[28] B. Wang, Y. Yan, Y. Lan, M. Wang, and Z. Bian, "Accurate Detection and Precision Spraying of Corn and Weeds Using the Improved YOLOv5 Model," *IEEE Access*, vol. 11, pp. 29868–29882, 2023, doi: 10.1109/ACCESS.2023.3258439.

[29] "Research on corn ears defect detection algorithm based on improved YOLOv7," *Academic Journal of Engineering and Technology Science*, vol. 7, no. 3, 2024, doi: 10.25236/AJETS.2024.070306.

[30] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. Lecun, "OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks," 2014.

[31] G. W. Taylor, I. Spiro, C. Bregler, and R. Fergus, "Learning Invariance through Imitation." [Online]. Available: http://www.oneframeoffame.com

[32] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans Pattern Anal Mach Intell*, vol. 32, no. 9, pp. 1627–1645, 2010, doi: 10.1109/TPAMI.2009.167.

[33] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation." [Online]. Available: http://arxiv.

[34] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun ACM*, vol. 60, no. 6, pp. 84–90, Jun. 2017, doi: 10.1145/3065386.

[35] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks." [Online]. Available: https://github.com/

[36] Y. Liu, C. Cen, Y. Che, R. Ke, Y. Ma, and Y. Ma, "Detection of Maize Tassels from UAV RGB imagery with faster R-CNN," *Remote Sens (Basel)*, vol. 12, no. 2, Jan. 2020, doi: 10.3390/rs12020338.

[37] E. Grilli, R. Battisti, and F. Remondino, "An advanced photogrammetric solution to measure apples," *Remote Sens (Basel)*, vol. 13, no. 19, Oct. 2021, doi: 10.3390/rs13193960.

[38] H. O. Velesaca, R. Mira, P. L. Suárez, C. X. Larrea, and A. D. Sappa, "Deep Learning based Corn Kernel Classification." [Online]. Available: http://www.fao.org/in-action/inpho/crop-compendium/cereals-

[39] E. Byvatov, U. Fechner, J. Sadowski, and G. Schneider, "Comparison of Support Vector Machine and Artificial Neural Network Systems for Drug/Nondrug Classification," *J Chem Inf Comput Sci*, vol. 43, no. 6, pp. 1882–1889, 2003, doi: 10.1021/ci0341161.

[40] M. G. M. Abdolrasol *et al.*, "Artificial neural networks based optimization techniques: A review," *Electronics (Switzerland)*, vol. 10, no. 21. MDPI, Nov. 01, 2021. doi: 10.3390/electronics10212689.

[41] J. A. Kim, J. Y. Sung, and S. H. Park, "Comparison of Faster-RCNN, YOLO, and SSD for Real-Time Vehicle Type Recognition," in *2020 IEEE International Conference on Consumer Electronics - Asia, ICCE-Asia 2020*, Institute of Electrical and Electronics Engineers Inc., Nov. 2020. doi: 10.1109/ICCE-Asia49877.2020.9277040.

[42] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," Apr. 2020, [Online]. Available: http://arxiv.org/abs/2004.10934

[43] L. Tan, T. Huangfu, and L. Wu, "Comparison of YOLO v3, Faster R-CNN, and SSD for Real-Time Pill Identiication," 2021, doi: 10.21203/rs.3.rs-668895/v1.