IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

ENHANCING QUALITY CONTROL: A DEEP LEARNING APPROACH FOR VISUAL INSPECTION

A PROJECT REPORT

Submitted by

MANOJ KUMAR.S	**(815420104022)**
MARKANDAYAN.M	**(815420104023)**
YUVARAJ.P	**(815420104051)**
ASWINKANTH.B	**(815420104301)**

ABSTRACT

The production and distribution of bottled water have witnessed exponential growth globally, driven by factors such as convenience, health consciousness, and urbanization. With this surge in demand, ensuring the quality and integrity of bottled water products has become a top priority for manufacturers. Central to this endeavour is the need for effective inspection methods to detect and mitigate defects that may compromise product safety and consumer satisfaction. With the proliferation of bottled water consumption, ensuring the quality and safety of water bottles has become increasingly vital. Visual inspection methods provide a non-invasive and efficient means of identifying defects in water bottles during manufacturing processes.In this study, we propose a novel approach for the visual inspection of water bottles using YOLO, a deep learning architecture known for its effectiveness in

image classification tasks. The proposed system employs YOLO algorithm to analyse images of water bottles captured by cameras installed along the production line. By leveraging the hierarchical feature representations learned by YOLO algorithm, our method aims to accurately classify water bottles into categories such as "defective" or "acceptable" based on the presence of defects such as scratches, dents, or impurities. We also explore strategies for optimizing model hyperparameters and training parameters to improve classification performance.

CHAPTER 1

INTRODUCTION

1.1 DEEP LEARNING

Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers. These neural networks attempt to simulate the behaviour of the human brain—albeit far from matching its ability—allowing it to “learn” from large amounts of data. While a neural network with a single layer can still make approximate predictions, additional hidden layers can help to optimize and refine for accuracy. Deep learning drives many artificial intelligence (AI) applications and services that improve automation, performing analytical and physical tasks without human intervention. Deep learning technology lies behind everyday products and services (such as digital assistants, voice-enabled TV remotes, and credit card fraud detection) as well as emerging technologies (such as self-driving cars).

Deep learning is a subfield of machine learning that uses artificial neural networks to model and solve complex problems. It has emerged as one of the most promising areas of research in artificial intelligence and has been applied to a wide range of applications such as image and speech recognition, natural language processing, and robotics. Deep learning models are based on artificial neural networks that are inspired by the structure and function of the human brain. These networks consist of layers of interconnected nodes, each of which performs a mathematical operation on the input data. The output of each node is passed onto the next layer of nodes, where it is combined with the outputs of other nodes and further processed. This process continues until the output of the final layer isproduced, which represents the prediction or classification of the input data.

One of the key advantages of deep learning is its ability to learn complex patterns and relationships in the data. This is achieved by using multiple layers of nodes, each of which learns a different set of features from the input data. The first layer learns low-level features such as edges and corners, while subsequent layers learn higher-level features such as textures and shapes. This hierarchical learning process enables deep learning models to capture complex patterns and relationships in the data, making them highly effective in solving complex problems. Another advantage of deep learning is its ability to learn from large amounts of data. Deep learning models require large amounts of data to train effectively, but once trained, they can make accurate predictions on new, unseen data. This makes deep learning particularly well-suited for applications such as image and speech recognition, where large amounts of labeled data are available. Deep learning has also benefited from the availability of powerful hardware suchas GPUs and TPUs, which can accelerate the training and inference of deep learning models. This has enabled researchers and developers to train larger andmore complex models, leading to significant improvements in performance and accuracy. Despite its many advantages, deep learning also has some limitations and challenges. One of the main challenges is the need for large amounts of labeled data. Deep learning models require labeled data to learn from, which canbe difficult and expensive to obtain, especially for niche applications.

Another challenge is the interpretability of deep learning models. Deep learning models are often seen as black boxes, making it difficult to understand how they arrive at their predictions or classifications. This can be problematic in applications where interpretability is important, such as in healthcare or finance. In conclusion, deep learning has emerged as a powerful and versatile tool for solving complex problems in a wide range of applications. Its ability to learn complex patterns and relationships in the data, and its scalability to large datasets, make it particularly well-suited for applications such as image and speech recognition. As deep learning continues to evolve, it is likely that these challenges will be overcome, leading to even more sophisticated and accurate models.

1.1.1 DEEP LEARNING VS MACHINE LEARNING

If deep learning is a subset of machine learning, how do they differ? Deeplearning distinguishes itself from classical machine learning by the type of data that it works with and the methods in which it learns. Machine learning algorithms leverage structured, labelled data to make predictions—meaning thatspecific features are defined from the input data for the model and organized into tables.

Machine learning and deep learning models are capable of different types of learning as well, which are usually categorized as supervised learning, unsupervised learning, and reinforcement learning. Supervised learning utilizes labeled datasets to categorize or make predictions; this requires some kind of human intervention to label input data correctly. In contrast, unsupervised learning doesn't require labeled datasets, and instead, it detects patterns in the data, clustering them by any distinguishing characteristics.

Deep neural networks consist of multiple layers of interconnected nodes, each building upon the previous layer to refine and optimize the prediction or categorization. This progression of computations through the network is called forward propagation. The input and output layers of a deep neural network are called visible layers. For example,

Convolutional neural networks (CNNs), used primarily in computer vision and image classification applications, can detect features and patterns within an image, enabling tasks, like object detection or recognition. In 2015, a CNN besteda human in an object recognition challenge for the first time.

Recurrent neural network (RNNs) is typically used in natural language andspeech recognition applications as it leverages sequential or times series data. Asdeep learning continues to evolve, it is likely that these challenges will be overcome, leading to even more sophisticated and accurate models.

1.2 DEEP LEARNING APPLICATIONS

Real-world deep learning applications are a part of our daily lives, but in most cases, they are so well-integrated into products and services that users are unaware of the complex data processing that is taking place in the background. Some of these examples include the following:

- **Law enforcement**

Deep learning algorithms can analyze and learn from transactional data toidentify dangerous patterns that indicate possible fraudulent or criminal activity. Speech recognition, computer vision, and other deep learning applications can improve the efficiency and effectiveness of investigative analysis by extracting patterns and evidence from sound and video recordings, images, and documents, which helps law enforcement analyze large amounts of data more quickly and accurately.

- **Financial services**

Financial institutions regularly use predictive analytics to drive algorithmictrading of stocks, assess business risks for loan approvals, detect fraud, and help manage credit and investment portfolios for clients.

- **Customer service**

Many organizations incorporate deep learning technology into their customer service processes. Chatbots—used in a variety of applications, services, and customer service portals—are a straightforward form of AI. Traditional chatbots use natural language and even visual recognition, commonly found in call center-like menus. Virtual assistants like Apple's Siri, Amazon Alexa, or Google Assistant extends the idea of a chatbot by enabling speech recognition functionality. This creates a new method to engage users in a personalized way. In contrast, unsupervised learning doesn't require labeled datasets, and instead, it detects patterns in the data, clustering them by any distinguishing characteristics.

- **Healthcare**

International Research Journal
IJNRD
Research Through Innovation

The healthcare industry has benefited greatly from deep learning capabilities ever since the digitization of hospital records and images. Image recognition applications can support medical imaging specialists and radiologists,helping them analyze and assess more images in less time.

1.3 ADVANTAGES OF DEEP LEARNINGS

- **Feature Generation Automation**

Deep learning algorithms can generate new features from among a limited number located in the training dataset without additional human intervention. This means deep learning can perform complex tasks that often require extensive feature engineering. For businesses, this means faster application or technology rollouts that deliver superior accuracy.

- **Works Well With Unstructured Data**

One of the biggest draws of deep learning is its ability to work with unstructured data. In the business context, this becomes particularly relevant when you consider that the majority of business data is unstructured. Text, images, and voice are some of the most common data formats that businesses use. Classical ML algorithms are limited in their ability to analyze unstructured data, meaning this wealth of information often goes untapped. And here’s where deeplearning promises to make the most impact. Training deep learning networks with unstructured data and appropriate labeling can help businesses optimize virtuallyevery function from marketing and sales to finance.

- **Better Self-Learning Capabilities**

The multiple layers in deep neural networks allow models to become moreefficient at learning complex features and performing more intensive computational tasks, i.e., execute many complex operations simultaneously. It

outshines machine learning in machine perception tasks (aka the ability to make sense of inputs like images, sounds, and video like a human would) that involve unstructured datasets. This is due to deep learning algorithms' ability to eventually learn from its own errors. It can verify the accuracy of its predictions/outputs and make necessary adjustments. On the other hand, classical machine learning models require varying degrees of human intervention to determine the accuracy of output.

- **Supports Parallel and Distributed Algorithms**

A typical neural network or deep learning model takes days to learn the parameters that define the model. Parallel and distributed algorithms address this pain point by allowing deep learning models to be trained much faster. Models can be trained using local training (use one machine to train the model), with GPUs, or a combination of both. However, the sheer volume of the training datasets involved could mean that storing it in a single machine becomes impossible. And that’s where data parallelism comes in. With data or the model itself being distributed across multiple machines, training is more effective. Parallel and distributed algorithms allow deep learning models to be trained at scale. For instance, if you were to train a model on a single computer, it could take up to 10 days to run through all the data. On the other hand, parallel algorithms can be distributed across multiple systems/computers to complete thetraining in less than a day.

International Research Journal
IJNRD
Research Through Innovation

Depending on the volume of your training dataset and GPU computing power, you could use as few as two or three computers to over 20 computers to complete the training within a day.

- **Cost Effectiveness**

While training deep learning models can be cost-intensive, once trained, itcan help businesses cut down on unnecessary expenditure. In industries such as manufacturing, consulting, or even retail, the cost of an inaccurate prediction or product defect is massive. It often outweighs the costs of training deep learning models. Deep learning algorithms can factor in variation across learning features to reduce error margins dramatically across industries and verticals. This is particularly true when you compare the limitations of the classical machine learning model to deep learning algorithms.

- **Advanced Analytics**

Deep learning, when applied to data science, can offer better and more effective processing models. Its ability to learn unsupervised drives continuous improvement in accuracy and outcomes. It also offers data scientists with more reliable and concise analysis results. The technology powers most prediction software today with applications ranging from marketing to sales, HR, finance, and more. If you use a financial forecasting tool, chances are that it uses a deep neural network. Similarly, intelligent sales and marketing automation suites also leverage deep learning algorithms to make predictions based on historical data.

- **Scalability**

Deep learning is highly scalable due to its ability to process massive amounts of data and perform a lot of computations in a cost- and time-effective manner. This directly impacts productivity (faster deployment/rollouts) and modularity and portability (trained models can be used across a range of problems).

1.4 CHALLENGES IN DEEP LEARNING

While deep learning has many advantages, there are also some disadvantages to consider:

- High computational cost: Training deep learning models requires significant computational resources, including powerful GPUs and large amounts of memory. This can be costly and time-consuming.

- Overfitting: Overfitting occurs when a model is trained too well on the training data and performs poorly on new, unseen data. This is a common problem in deep learning, especially with large neural networks, and can be caused by a lack of data, a complex model, or a lack of regularization.
- Lack of interpretability: Deep learning models, especially those with many layers, can be complex and difficult to interpret. This can make it difficult to understand how the model is making predictions and to identify any errors orbiases in the model.
- Dependence on data quality: Deep learning algorithms rely on the quality of the data they are trained on. If the data is noisy, incomplete, or biased, the model's performance will be negatively affected.
- Data privacy and security concerns: As deep learning models often rely on large amounts of data, there are concerns about data privacy and security. Misuse of data by malicious actors can lead to serious consequences like identity theft, financial loss and invasion of privacy.
- Lack of domain expertise: Deep learning requires a good understanding of the domain and the problem you are trying to solve. If the domain expertise is lacking, it can be difficult to formulate the problem and select the appropriatealgorithm.
- Unforeseen consequences: Deep learning models can lead to unintended consequences, for example, a biased model can discriminate against certain groups of people, leading to ethical concerns.
- Black box models: some deep learning models are considered as "black-box" models, as it is difficult to understand how the model is making predictions and identifying the factors that influence the predictions. Its ability to learn unsupervised drives continuous improvement in accuracy and outcomes Similarly, intelligent sales and marketing automation suites also leverage deep learning algorithms to make predictions based on historical data

CHAPTER 2 SYSTEM ANALYSIS

2.1 LITERATURE SURVEY

2.1.1 TITLE: MACHINE-VISION-BASED PLASTIC BOTTLE INSPECTION FOR QUALITY ASSURANCE

AUTHOR: MAJIDA KAZMI , BASRA HAFEEZ , HASHIM RAZA KHAN ,SAAD AHMED QAZI

YEAR:2022

An automated defect inspection system for quality assurance is proposed for the plastic bottle manufacturing industry. This system inspects the most frequently occurring defects of the bottle by using machine vision. The proposedsystem is a portable and cost-efficient system which can be used in small industries without disturbing their production line. Quality assurance (QA) is very important in product development and thus it is necessary to prevent defects inmanufactured products. It requires the thorough inspection of a product before its dispatch. Currently, QA relies on manual inspection which is time consuming,error-prone and has integrity issues. Alternately, automated systems become a viable solution for inspecting different product defects in an error-free way. Many industries, such as fast-moving consumer goods (fmcg), textile, food and beverages, pharmaceuticals, etc., are now inclined to use such systems for qualityinspection. The highly sensitive part of the manufacturing industry is the packaging of their products where bottles are mainly used for liquid products **ADVANTAGES**

- Inspection system for quality assurance
- Accuracy is good

DISADVANTAGES

- Computation power is high
- Need Man power

2.1.2 TITLE: A REVIEW ON MODERN DEFECT DETECTION MODELS USING DCNNS – DEEP CONVOLUTIONAL NEURALNETWORKS

AUTHOR: ANDREI-ALEXANDRU TULBURE , ADRIAN-ALEXANDRUTULBURE , EVA-HENRIETTA DULF

YEAR:2022

Defect detection using computer vision models started to pick up popularity in the 21st century, as the object detection models became more and more popular. The general accepted idea is that the dataset, as well as the chosen model led to great performances, thus both need to receive attention from the developer. The training hardware doesn't 't need to be expensive, if the application does not mandate it. While general applications that have a target as to detect lots of defects, need very large and balanced datasets, a hardware setupwith lots of computational power and a specific detection model that is not just tweaked for defect detection, but built from the ground up for the specific action that we want it to perform. Thus, hardware setup and availability (that's why recommended is using virtual machines on a desktop machine learning station for general solutions) plays a role in the performance of the models. There is no rule of thumb when choosing which object detection model shall generalize the best on the particular dataset of interest.

ADVANTAGES

- Developed object detection models
- Modules are increased

DISADVANTAGES

- Need to improve the high accuracy
- Need more time

2.1.3 TITLE: FAST METHOD OF DETECTING PACKAGING BOTTLE DEFECTS BASED ON ECA EFFICIENTDET

AUTHOR: ZHENWEN SHENG , GUIYUN WANG

YEAR: 2022

This paper proposed a packaging defect detection method based on the ECA-EfficientDet object detection algorithm. We validated its effectiveness and advantages on a sample dataset of defects. Our results show the following: (1) The proposed method solves a challenging problem in conventional machine vision algorithms. It can simultaneously detect multiple defect objects, effectively reducing inspection costs on production lines. (2) Our design of ECA-Convblock in the backbone feature extraction network prevents dimension reduction in model channel importance prediction and enhances high-quality expressions of object features, effectively improving the defect detection accuracy. (3) The incorporation of Mosaic data augmentation and the Mish activation function into the model and the adoption of heterogeneous-data-based transfer learning for model training effectively enhance the model's generalization capability and improve its robustness in complex environments. It should be pointed out that the algorithm has obvious advantage in accuracy performance and algorithm stability, but it is slightly insufficient in detection speed. In addition, the object detection algorithm requires a large amount of data to build the model. However, it is difficult to obtain packaging defect samples.

ADVANTAGES

- Detect multiple defect objects
- More datasets can be added

DISADVANTAGES

- Detection speed is low
- Power consumption is high

2.1.4 TITLE: MACHINE VISION-BASED INTEGRATED INSPECTION SYSTEM FOR BEVERAGE BOTTLES WITH MULTIPLE DEFECTS AUTHOR: JIA WANG , KANG JINGXIN , LI FANGJUN , ZHANG AIJUN

YEAR: 2023

In this paper, a multi-functional beverage bottle defect detection system based on machine vision has been proposed, which mainly includes an image feature encoder and multiple parallel decoders. The encoder composed of convolution networks can generate shared feature layers, and the decoders designed according to the characteristics of labeling and spray coding processes can use the information from the shared feature layers to realize the corresponding defect detection. n addition, the system provides an expandable interface, through which detection decoders could be added according to the actual needs, so as to realize the simultaneous detection of different types of defects in multiple processes. Since the system designed in this paper depends on the computing platform's computing power, subsequent tests can be conducted on platforms with higher computing power in order to achieve higher detection efficiency. The experimental results show that the system has high accuracy and a short inspection time. In this paper, the detection of various types of defects onbeverage bottles can be realized synchronously by using only one system, so as to effectively reduce the waste of resources and space in the production line.

ADVANTAGES

- Multi-functional beverage bottle defect detection system
- It has a high accuracy

DISADVANTAGES

- Time complexity is high
- Need high level configuration

2.1.5 TITLE: RESEARCH ON DEFECT DETECTION OF THE OUTER SIDE OF BOTTLE CAP BASED ON HIGH ANGLE AND MULTI-VIEWVISION SYSTEM

AUTHOR: CHENGHU HE , CHEN LI , BOWEN CHEN , BIN YUAN

YEAR: 2023

In this paper, the defect detection on the outer side of the bottle cap is the object of study. Since the surface to be inspected is a circular arc surface, uniform illumination cannot be achieved through the existing technologies. For the above problems, this paper proposes a high-angle multi-view inspection system (HAMV). By means of high-angle circular illumination, the chief ray of the light source is directly incident on circular arc surface of the cap, and a uniform illumination of 360◦ along the arc direction is achieved. In addition, complete imaging of the outer side of the bottle cap is accomplished at four imaging views. The use of line structure elements in grayscale image enables fast reconstruction of background texture and reduces the interference of non-skid bars texture on defect detection. At the same time, the adaptive segmentation of different sub- regions is completed using the bottom margin region in the reconstruction as well as the contour coordinates of the bottle cap in the image. Moreover, the defect detection algorithms are designed separately for different kinds of defects that appear in different sub-regions.

ADVANTAGES

- Inspected is a circular arc surface
- Detect the cap of the bottle

DISADVANTAGES

- False positive rate is high
- Less modules are used

2.1.6 TITLE: DEFECT DETECTION METHODS FOR INDUSTRIAL PRODUCTS USING DEEP LEARNING TECHNIQUES

AUTHOR: ALIREZA SABERIRONAGHI , JING REN , MOUSTAFAEL-GINDY

YEAR: 2023

Deep learning technology has revolutionized the field of defect detection in industrial products. However, finding a suitable deep learning model for solving the defect detection problem is very difficult due to the particularities ofindustrial scenarios. In the coming years, deep learning will encounter challenges and trends as it becomes more widely used in industrial fields. Non-destructive testing (NDT) is a method that uses radiography or ultrasound technologies to discover faults without causing damage to the detected objects. It is widely usedin engineering industries to detect and evaluate defects in materials of all types. An important technique in non-destructive testing is radiographic testing, which uses X-rays to identify and evaluate flaws or defects, such as cracks or porosities. Defects can appear in X-ray images in many shapes and sizes, making detection difficult. The traditional approach for identifying defects in industrial products is for human operators or experts to visually inspect radiographs. However, this method can be subjective and prone to errors. Additionally, the process of examining a large number of images can be time-consuming and may lead to misinterpretations.

ADVANTAGES

- Detecting defects in industrial products
- Developed object detection models

DISADVANTAGES

- Does not support multiple images
- large number of images can be time-consuming

2.1.7 TITLE: REAL-TIME PLASTIC SURFACE DEFECT DETECTIONUSING DEEP LEARNING

AUTHOR: MUHAMMAD IZZAT BIN ROSLAN , ZAIDAH IBRAHIM ,ZALILAH ABD AZIZ

YEAR: 2022

International Research Journal IJNRD Research Through Innovation

Plastic packaging products have been massively produced throughout the century. Almost 380 million tons of plastic products are produced every year. Global plastic production recorded 407 million tons of plastic packaging has been produced throughout the year 2015 which is nearly 200-fold of production compared to the 1950s. This shows how demanding plastic usage can be. Although plastic packaging is considered a major cause of toxic pollutants in theworld, it also brings great benefits to the distributor, retailer, and consumer. Industries such as food & beverage, health care, cosmetic and personal care, and consumer goods mostly rely on plastic packaging to contain their product mainly to help prevent contamination, prolong shelf life, prevent waste, display productinformation, and provide efficient transportation. Quality control is a process utilized in the plastic packaging industry to ensure that the products that are produced are high-quality. To avoid experiencing errors and minimize product defects, manual surface defect detection is typically performed by humans through the naked eyes. YOLO has shown excellent performance in object detection and this research applies YOLOv5.

ADVANTAGES

- Create a robust dataset
- Detect multiple defect objects

DISADVANTAGES

- Need high level configurations
- Error can be occur low level configuration

2.1.8 TITLE: FAULT DETECTION IN BOTTLE CAPS AND LABEL ALIGNMENT USING CONVOLUTIONAL NEURAL NETWORK AUTHOR: SAPANA INDARCHAND TATU , SHASHANK KUMAR SINGH , SAURABH BANSOD , PRASHANT PAL.

YEAR: 2023

This paper has successfully created a system based on CNN that accurately distinguishes between good and bad bottle caps and labels. We have referred various model which worked on detecting fault in bottle caps like VGG-16 gives 92% accuracy, Resnet 50 model gives 91% accuracy, Inception model gives accuracy of 93% and apart from this model our proposed model attains 98.55% accuracy, as shown in fig. 5. By removing human inspections from the process of inspecting for defective caps while still delivering quick inspections, increasing the sector's profitability. In this paper by using

Convolutional neural networks, image processing is used to detect fault in bottle caps which include loose caps, tilted caps, and missing caps and edge detection method is used to detect fault inlabel alignment which includes misplaced or tilted labels and inverted labels. The proposed model will improve precision and speed of fault detection. Due to the accuracy and time concession the adaptability of this machine learning method, the system can operate in a variety of lighting and backdrop conditions. There are several problems in quality control such as unequal level of liquid, accident immersion of foreign particles in bottles.

ADVANTAGES

- Inspecting for defective caps
- Developed object detection models

DISADVANTAGES

- Manual interventions are needed
- Detection speed is low

2.1.9 TITLE: CLASSIFICATION OF BEER BOTTLES USING OBJECT DETECTION AND TRANSFER LEARNING

AUTHOR: PHILIPP HOHLFELD , TOBIAS OSTERMEIER , DOMINIKBRANDL

YEAR: 2022

Deep learning has become a buzzword nowadays due to its increasing popularity in recent years. The high availability of large datasets as well as powerful graphics processing units (GPUs) and high efficiency algorithms madeit possible to achieve excellent results in many areas, such as image classification , object detection and natural language processing. Classification problems are common in Computer Vision. Despite this, there is no dedicated work for the classification of beer bottles. In this paper we present a deep learning model which classifies pictures of beer bottles in a twostep approach. As the first step,a Faster-R-CNN detects image sections relevant for classification independently of the brand. 1In the second step, the relevant image sections are classified by a ResNet-18. We were able to achieve 100 % accuracy after the challenge ended. We aim to build a model achieving the highest accuracy on the test dataset. Therefore, we solve the problem of image classification in two steps. In the first step, image sections relevant for classification are detected independently of the logo. In the second step, the relevant image sections are classified. The image section with the highest confidence is returned as class label.

ADVANTAGES

- Reach a high accuracy
- Detecting defects in industrial products

DISADVANTAGES

- Difficult to classify the multiple bottle images
- Computation power is high

2.1.10 TITLE: LPVIT: A TRANSFORMER BASED MODEL FOR PCBIMAGE CLASSIFICATION AND DEFECT DETECTION

AUTHOR: KANG AN , YANPING ZHANG

YEAR: 2022

PCB devices have greatly facilitated human life. The classification and defect detection of PCB image data can greatly accelerate downstream tasks such as production and sorting, effectively improve manufacturing and recycling efficiency, and meet the growing demand for PCBs. Among the massive PCBs, abig group of applications is using micro-PCBs manufactured by several prevailing companies. Once the makes and models of PCBs are identified, the sub-components could be easily retrieved through a bill of materials. In this research, we propose a transformer-based classification model, LPViT, which can be used to classify micro-PCBs based on their makes and models and detect defects on PCBs. Through comparative experiments, our proposed model has achieved SOTA performance on the micro-PCB dataset for classificaiton and on DeepPCB dataset for defect detection. They also contain valuable chemical elements (rare earth elements, gallium, etc) necessary for producing electronic products. The recent auto crisis during COVID-19 caused by the shortage of chips and small electronic goods makes those resources even more precious. Therefore,recycling and reuse of PCBs are getting more attention.

ADVANTAGES

- Improve manufacturing and recycling efficiency
- Inspecting for defective caps

DISADVANTAGES

- Does not support various image datasets
- Need to improve the high accuracy

2.2 EXISTING SYSTEM

Feature extraction is a pivotal component of computer vision systems, pivotal for capturing pertinent information from images to facilitate subsequent analysis and decision-making processes. Traditional methods for feature extraction, such as structural, statistical, filter-based, and model-based approaches, have long been employed across various computer vision applications. However, with the advent of deep learning, particularly convolutional neural networks (CNNs), end-to-end learning methodologies have gained prominence. Among these, the You Only Look Once (YOLO) algorithm stands out for its speed and accuracy in object detection tasks. YOLOv4, an enhanced iteration of the YOLO algorithm, integrates numerous improvements to achieve superior performance. It employs a single neural network to directly predict bounding boxes and class probabilities from entire images in a single evaluation, making it well-suited for real-time object detection applications. By leveraging feature reuse and promoting gradient flow, DenseNet facilitates more effective learning and utilization of parameters, leading to improved model performance even with smaller datasets. To leverage YOLOv4 for detecting scratches on PET bottle preforms, a typical workflow involves dataset collection, preprocessing, model training, evaluation, and deployment, culminating in a robust system capable of efficiently identifying defects in industrial settings. Among these, the You Only Look Once (YOLO) algorithm has garnered substantial attention due to its remarkable combination ofspeed and accuracy in object detection tasks.

DISADVANTAGES

- Manual interventions are needed
- Time complexity is high
- Error can be occurred
- Accuracy is less

2.3 PROPOSED SYSTEM

In this paper utilizing YOLOv5, a cutting-edge deep learning architecture, for the visual inspection of plastic bottles offers significant advantages in terms of accuracy, efficiency, and scalability. YOLOv5 builds upon the success of previous YOLO iterations while introducing several key improvements, making it an ideal candidate for such applications. Incorporating DenseNet architecture into YOLOv5 further enhances its capabilities for visual inspection tasks. DenseNet's unique dense connectivity pattern, where each layer is connected to every other layer in a feed-forward fashion, promotes feature reuse and facilitates gradient flow throughout the network. This dense connectivity ensures that information from earlier layers is efficiently propagated to subsequent layers, enabling the model to effectively learn intricate patterns and features present in the plastic bottles. Moreover, DenseNet's architecture enables efficient parameter usage, particularly beneficial when working with limited training data, which is often the case in industrial settings. By leveraging feature reuse and promoting gradient flow, DenseNet facilitates more effective learning and utilization of parameters, leading to improved model performance even with smaller datasets. By integrating YOLOv5 with DenseNet architecture, the visual inspection system gains the ability to accurately detect and classify defects, such as scratches, dents, or irregularities, on plastic bottles in real-time. This ensures stringent quality control measures, enhances production efficiency, and minimizes the risk of faulty products reaching consumers. Additionally, YOLOv5's superior accuracy minimizes false positives and false negatives, resulting in more reliable defect detection and reducing the need for manual intervention or rework.

ADVANTAGES

- Automated analysis
- Reduce time and computational complexity
- Accuracy can be improve

International Research Journal IJNRD Research Through Innovation

CHAPTER 3

SYSTEM ARCHITECTURE

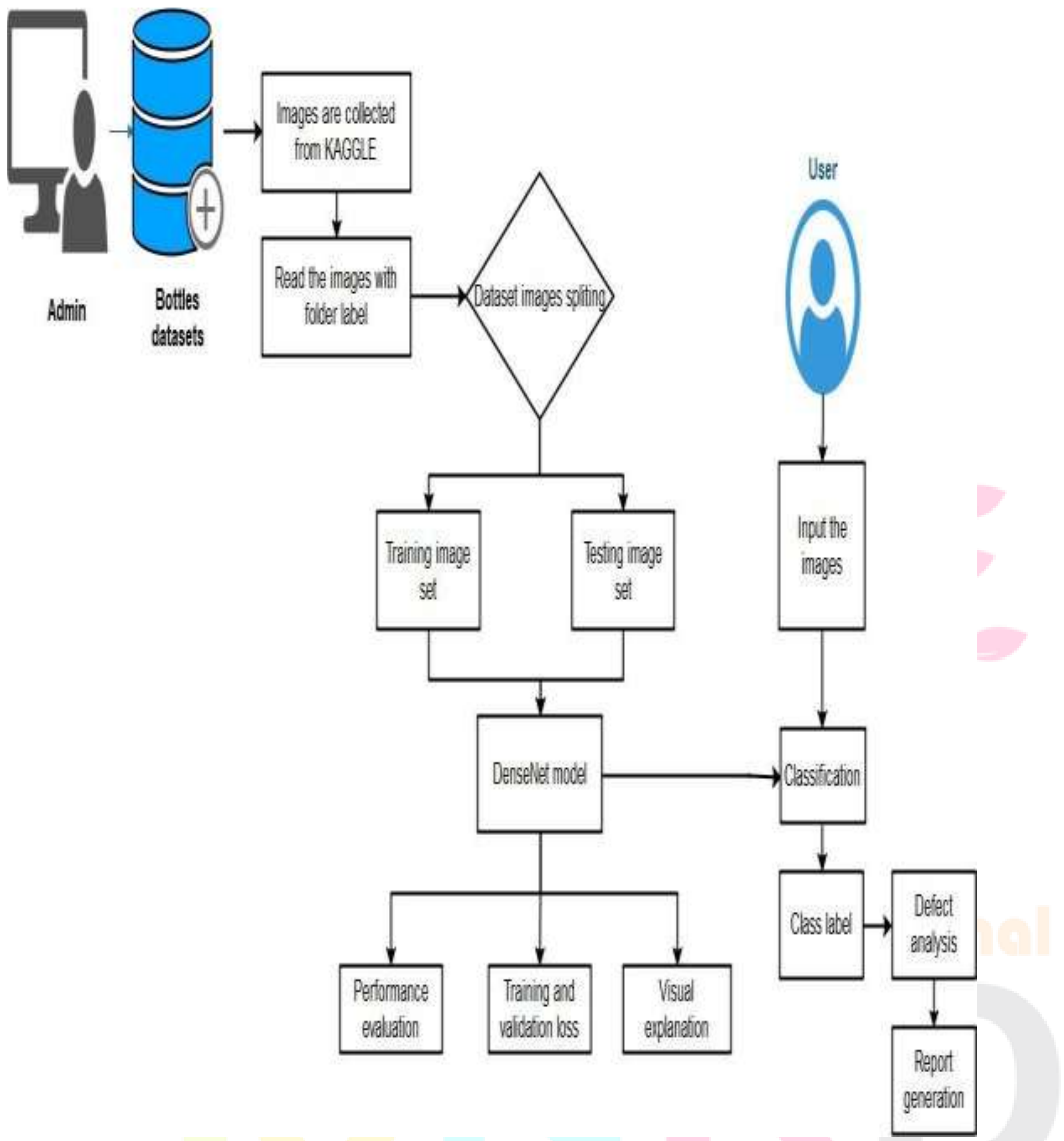


Fig. 3.1. System Architecture

Research Through Innovation

CHAPTER 4 SYSTEM SPECIFICATION

4.1 HARDWARE REQUIREMENTS

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design.

Processor	: Intel processor 2.6.0 GHZ
RAM	: 1GB
Hard disk	: 160 GB
Compact Disk	: 650 Mb
Keyboard	: Standard keyboard
Monitor	: 15 inch color monitor

4.2 SOFTWARE REQUIREMENTS

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is useful in estimating cost, planning team activities and performing tasks throughout the development activity.

Operating System	: Windows OS
Front End	: PYTHON
IDE	: PYCHARM
Back End	: MYSQL
Application	: WEB APPLICATION

CHAPTER 5

SOFTWARE DESCRIPTION

5.1 PYTHON

Python is a high-level, interpreted programming language that is widely used in various domains such as web development, scientific computing, data analysis, artificial intelligence, machine learning, and more. It was first released in 1991 by Guido van Rossum and has since become one of the most popular programming languages due to its simplicity, readability, and versatility. One of the key features of Python is its easy-to-learn syntax, which makes it accessible to both novice and experienced programmers. It has a large standard library that provides a wide range of modules for tasks such as file I/O, networking, regular expressions, and more.

Python also has a large and active community of developers who contribute to open-source libraries and packages that extend its capabilities. Python is an interpreted language, which means that it is executed line-by-line by an interpreter rather than compiled into machine code like C or C++. This allows for rapid development and testing, as well as easier debugging and maintenance of code. Python is used for a variety of applications, including web development frameworks such as Django and Flask, scientific computing libraries such as NumPy and Pandas, and machine learning libraries such as TensorFlow and PyTorch.

Python is an interpreted high-level programming language for general- purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales. In July 2018, Van Rossum stepped down as the leader in the language community. Python features a dynamic type system and automatic memory management.

Python and CPython are managed by the non-profit Python Software Foundation. Rather than having all of its functionality built into its core, Python was designed to be highly extensible. This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications.

Python's developers strive to avoid premature optimization, and reject patches to non-critical parts of CPython that would offer marginal increases in speed at thecost of clarity. [When speed is important, a Python programmer can move time- critical functions to extension modules written in languages such as C, or use PyPy, a just-in-time compiler.

There are two attributes that make development time in Python faster than in other programming languages:

- Python is an interpreted language, which precludes the need to compile code before executing a program because Python does the compilation in the background. Because Python is a high-level programming language, it abstracts many sophisticated details from the programming code. Python focuses so much on this abstraction that its code can be understood by most novice programmers.
- Python code tends to be shorter than comparable codes. Although Python offers fast development times, it lags slightly in terms of execution time. Compared to fully compiling languages like C and C++, Python programs execute slower. Of course, with the processing speeds of computers these days, the speed differences are usually only observed in benchmarking tests, not in real-world operations. In most cases,

International Research Journal
IJNRD
Research Through Innovation

Python is already included in Linux distributions and Mac OS X machines. Python is a versatile and powerful programming language that is well-suited for a wide range of applications.

Some popular Python libraries and frameworks include:

- NumPy: a library for numerical computing in Python, providing support for large, multi-dimensional arrays and matrices, along with a large collection of mathematical functions to operate on these arrays.
- Pandas: a library for data manipulation and analysis in Python, providing support for reading and writing data in a variety of formats, as well as powerful tools for manipulating and analyzing data.
- Matplotlib: a plotting library for Python that provides a variety of visualization tools, including line plots, scatter plots, bar plots, and more.
- TensorFlow: an open-source machine learning library for Python that provides a variety of tools and algorithms for building and training machine learning models.
- Django: a popular web framework for Python that provides a full-stack framework for building web applications, with support for everything from URL routing to user authentication and database integration.
- Python's popularity has also led to a large and active community of developers who contribute to open-source projects and share code and resources online. This community provides a wealth of resources for learning Python, including tutorials, online courses, and forums for asking and answering questions.

5.2 TENSORFLOW LIBARIES IN PYTHON

TensorFlow is an open-source machine learning framework developed by Google Brain Team. It is one of the most popular libraries for building and training machine learning models, especially deep neural networks. TensorFlow allows developers to build complex models with ease, including image and speech recognition, natural language processing, and more. One of the key features of TensorFlow is its ability to handle large-scale datasets and complex computations, making it suitable for training deep neural networks. It allows for parallelization of computations across multiple CPUs or GPUs, allowing for faster training times. TensorFlow also provides a high-level API called Keras thatsimplifies the process of building and training models. TensorFlow offers a widerange of tools and libraries that make it easy to integrate with other Python libraries and frameworks. It has built-in support for data preprocessing and visualization, making it easy to prepare data for training and analyze model performance. One of the major advantages of TensorFlow is its ability to deploy models to a variety of platforms, including mobile devices and the web.

- Graph-based computation: TensorFlow uses a graph-based computation model, which allows for efficient execution of computations across multiple devices and CPUs/GPUs.
- Automatic differentiation: TensorFlow provides automatic differentiation, which allows for efficient computation of gradients for use in backpropagation algorithms.
- High-level APIs: TensorFlow provides high-level APIs, such as Keras, that allow developers to quickly build and train complex models with minimal code.
- Preprocessing and data augmentation: TensorFlow provides a range of tools for preprocessing and data augmentation, including image and text preprocessing, data normalization, and more.
- Distributed training: TensorFlow supports distributed training across multiple devices, CPUs, and GPUs, allowing for faster training times and more efficient use of resources.
- Model deployment: TensorFlow allows for easy deployment of models to a variety of platforms, including mobile devices and the web.
- Visualization tools: TensorFlow provides a range of visualization tools for analyzing model performance, including TensorBoard, which allows for real-time visualization of model training and performance.

5.3 PYCHARM

PyCharm is an integrated development environment (IDE) for Python programming language, developed by JetBrains. PyCharm provides features such as code completion, debugging, code analysis, refactoring, version control integration, and more to help developers write, test, and debug their Python code efficiently. PyCharm is available in two editions: Community Edition (CE) and Professional Edition (PE). The Community Edition is a free, open-source version of the IDE that provides basic functionality for Python development. The Professional Edition is a paid version of the IDE that provides advanced features such as remote development, web development, scientific tools, database tools, and more. PyCharm is available for Windows, macOS, and Linux operating systems. It supports Python versions 2.7, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, and 3.10.

Features:

- Intelligent code completion
- Syntax highlighting
- Code inspection
- Code navigation and search
- Debugging
- Testing
- Version control integration
- Web development support
- Scientific tools support
- Database tools support

Integration with other JetBrains tools

PyCharm's code completion feature can help speed up development by automatically suggesting code based on context and previously written code. It also supports virtual environments, which allow developers to manage different Python installations and packages in isolated environments. It also includes toolsfor working with databases, such as PostgreSQL, MySQL, and Oracle.

Customization:

PyCharm allows developers to customize the IDE to their liking. Users can change the color scheme, fonts, and other settings to make the IDE more comfortable to use. PyCharm also supports plugins, which allow developers to extend the IDE with additional features.

Collaboration:

PyCharm makes it easy for developers to collaborate on projects. It supports integration with popular collaboration tools such as GitHub, Bitbucket,and GitLab. It also includes features for code reviews, task management, and team communication.

Education:

PyCharm provides a learning environment for Python programming language. PyCharm Edu is a free, open-source edition of PyCharm that includesinteractive courses and tutorials for learning Python.

Support:

PyCharm has an active community of users who provide support through forums and social media. JetBrains also provides comprehensive documentation, tutorials, and training courses for PyCharm.

Pricing:

PyCharm Community Edition is free and open-source. PyCharm Professional Edition requires a paid license, but offers a 30-day free trial. JetBrains also offers a subscription-based pricing model that includes access to all JetBrains IDEs and tools.

Integrations:

PyCharm integrates with a wide range of tools and technologies commonly used in Python development. It supports popular Python web frameworks like Flask, Django, Pyramid, and web2py. It also integrates with tools for scientific computing like NumPy, SciPy, and pandas. PyCharm also supports popular front - end technologies such as HTML, CSS, and JavaScript.

Performance:

PyCharm is known for its fast and reliable performance. It uses a combination of static analysis, incremental compilation, and intelligent caching to provide fast code completion and navigation. PyCharm also has a memory profiler that helps identify and optimize memory usage in Python applications.

Ease of Use:

PyCharm provides an intuitive and easy-to-use interface for developers. Ithas a well-organized menu structure, clear icons, and easy-to-navigate tabs.

Community:

PyCharm has a large and active community of developers who contribute to the development of the IDE. The PyCharm Community Edition is open -source, which means that anyone can contribute to its development. The PyCharm user community is also active in providing support, tips, and tutorials through forums,blogs, and social media.

5.4 MY SQL

MySQL is the world's most used open source relational database management system (RDBMS) as of 2008 that run as a server providing multi- user access to a number of databases. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements.

MySQL is a popular choice of database for use in web applications, and isa central component of the widely used LAMP open source web application software stack—LAMP is an acronym for "Linux, Apache, MySQL, Perl/PHP/Python." Free-software-open source projects that require a full-featured database management system often use MySQL.

Inter images

MySQL is primarily an RDBMS and ships with no GUI tools to administer MySQL databases or manage data contained within the databases. Users may use the included command line tools, or use MySQL "front-ends", desktop software and web applications that create and manage MySQL databases, build database structures, back up data, inspect status, and work with data records. The official set of MySQL front-end tools, MySQL Workbench is actively developed by Oracle, and is freely available for use.

Graphical

The official MySQL Workbench is a free integrated environment developed by MySQL AB that enables users to graphically administer MySQL databases and visually design database structures. MySQL Workbench replaces the previous package of software, MySQL GUI Tools. Similar to other third-party packages, but still considered the authoritative MySQL frontend, MySQL Workbench lets users manage database design & modeling, SQL development (replacing MySQL Query Browser) and Database administration (replacing MySQL Administrator).MySQL Workbench is available in two editions, the regular free and open source Community Edition which may be downloaded from the MySQL website, and the proprietary Standard Edition which extends and improves the feature set of the Community Edition. This process involves optimizing the model's weights to minimize the discrepancy between predicted bounding boxes and ground truth annotations, thereby enhancing its ability to accurately localize defects within the images.

CHAPTER 6

SYSTEM/ SUB SYSTEM SPECIFICATION

6.1 MODULES

There are five modules implemented in our system

- Datasets collection
- Model build
- Performance evaluation
- Bottle defect classification
- Reports

6.2 MODULE DESCRIPTION

6.2.1 DATASETS COLLECTION

Collecting datasets from the Roboflow website is an efficient approach foracquiring image data relevant to the visual inspection task of plastic bottles. The platform offers a curated collection of images categorized into various classes, including those depicting defects. With images available in JPEG format, compatibility and ease of integration into the training pipeline are ensured, facilitating seamless utilization within the chosen deep learning framework. By leveraging the datasets from Roboflow, the visual inspection system can access a diverse range of images encompassing different types of defects across multiple classes. Furthermore, the availability of datasets from Roboflow simplifies the data preprocessing stage, as the images are already organized and annotated, ready for use in model training. This streamlines the overall development process, allowing for quicker iterations and experimentation with different model architectures and hyperparameters. This augmentation helps to increase the variability of the dataset, improving the model's ability to generalize to unseen data and enhancing overall performance.

6.2.2 MODEL BUILD

In this module, we utilize the YOLO (You Only Look Once) framework toconstruct a robust model for defect detection in images of plastic bottles. YOLO is a state-of-the-art object detection algorithm that divides the input image into a grid and predicts bounding boxes and class probabilities for objects within each grid cell. We then select a suitable YOLO implementation, such as YOLOv3 or YOLOv4, and fine-tune the pre-trained model on our annotated dataset. Finally, we save the trained YOLO model, typically in a format such as ".h5", preserving its architecture and learned parameters for future use in defect detection tasks.

6.2.3 PERFORMANCE EVALUATION

In this module, the training process of the YOLOv5 model using a curateddataset is encapsulated, encompassing crucial steps for effective defect detectionin images of plastic bottles. Initially, the dataset, carefully curated to include various instances of defects, is prepared, ensuring comprehensive coverage of potential anomalies. Subsequently, the images are fed into the YOLOv5 network, which undergoes iterative adjustments of its parameters through backpropagation. This process involves optimizing the model's weights to minimize the discrepancy between predicted bounding boxes and ground truth annotations, thereby enhancing its ability to accurately localize defects within the images. To ascertain the model's generalization capability and robustness, it undergoes validation on a separate dataset distinct from the training data. This validation step serves to assess the model's performance on unseen data, ensuring that it can effectively detect defects in real-world scenarios beyond the training environment. By iteratively refining the model based on validation results and adjusting hyperparameters as necessary, the YOLOv5 model achieves a high level of accuracy and reliability in defect detection tasks. This training process culminates in the development of a sophisticated defect detection system.

6.2.4 WEB INTERFACE

In this module, a user-friendly web interface is developed to enable users to predict the defective status of uploaded bottles. The web application is createdusing frameworks like Flask, providing an intuitive platform for users to interact with. Upon uploading images of plastic bottles, preprocessing steps are applied to enhance image quality and eliminate noise, ensuring optimal input for defect detection. These preprocessing techniques may include resizing, normalization, and noise reduction, aimed at improving the accuracy of subsequent defect classification. The pre-trained YOLOv5 model is then employed to perform inference on the preprocessed images, predicting bounding boxes and classifying defects within the images. This iterative process of model training and validationhelps to enhance the accuracy and reliability of defect classification, ultimately providing users with more accurate insights into the quality of the plastic bottles.Through this web interface, users can efficiently assess the defective status of uploaded bottles, enabling timely interventions and quality control measures in industrial settings.

6.2.5 REPORTS

In this module, a reporting system is implemented to generate reports for each bottle, detailing the detected defects along with relevant information such asbottle ID and date. Upon detecting defects in the uploaded images through the web interface, the system automatically generates reports for each bottle. These reports contain comprehensive information about the bottle's ID, the types of defects detected, and the date and time of the inspection. Once the reports are generated, they are forwarded to the company administrator in the form of email alerts. This process is automated, ensuring that administrators receive timely notifications about any defects identified during the inspection process. The email alerts provide administrators with actionable insights, allowing them to take appropriate measures address the detected defects and maintain product quality.

CHAPTER 7

SYSTEM DESIGN

7.1 DATA FLOW DIAGRAM

A two-dimensional diagram explains how data is processed and transferred in a system. The graphical depiction identifies each source of data and how it interacts with other data sources to reach a common output. Individuals seeking to draft a data flow diagram must identify external inputs and outputs, determine how the inputs and outputs relate to each other, and explain with graphics how these connections relate and what they result in. This type of diagram helps business development and design teams visualize how data is processed and identify or improve certain aspects.

DFD LEVEL 0

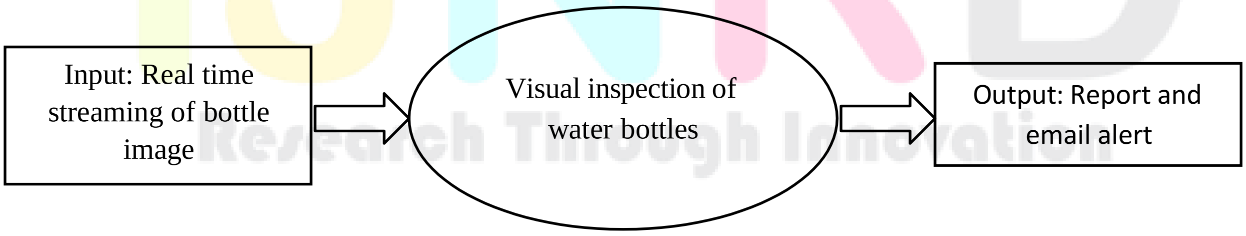

Input: Real time streaming of bottle image
Visual inspection of water bottles
Output: Report and email alert

Fig. 7.1. DFD Level 0

DFD LEVEL 1

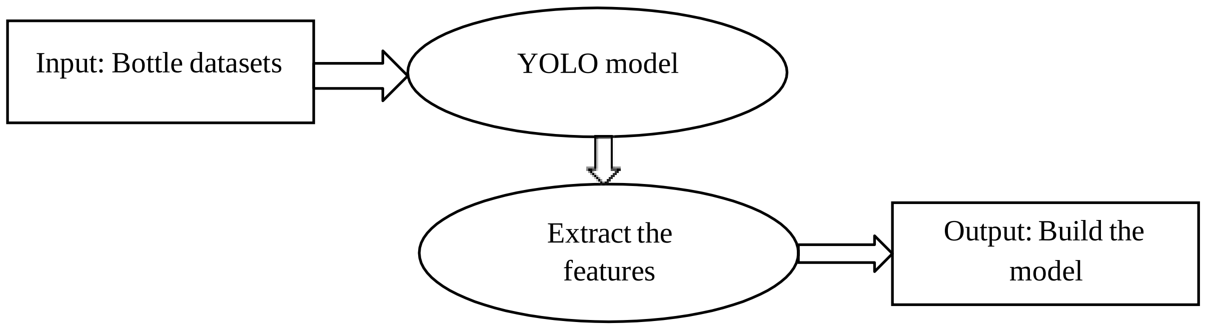


Fig. 7.2. DFD Level 1

DFD LEVEL 2

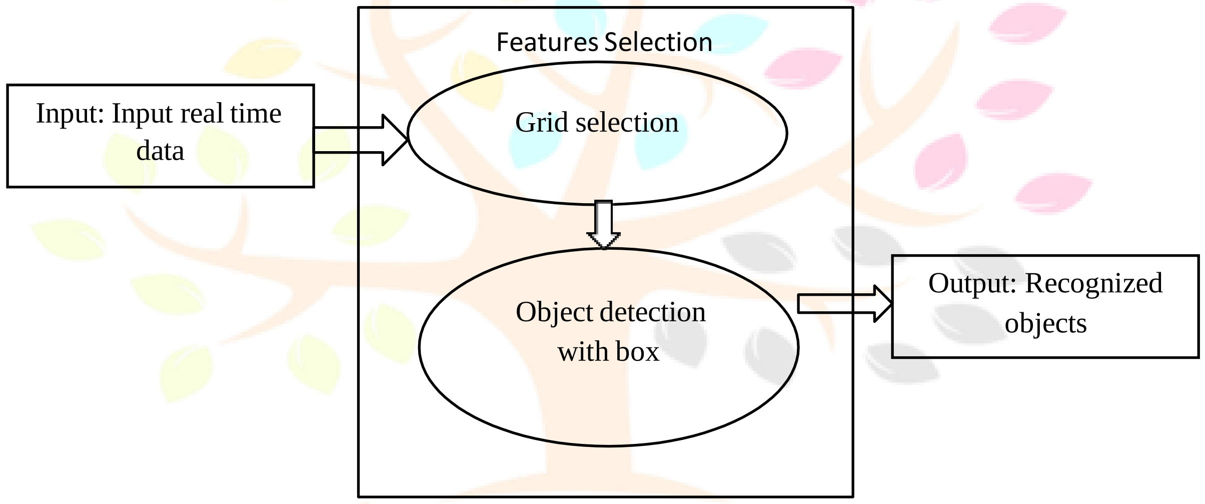

Features Selection
Input: Input real time data
Grid selection
Object detection with box
Output: Recognized objects

Fig. 7.3. DFD Level 2

DFD LEVEL 3

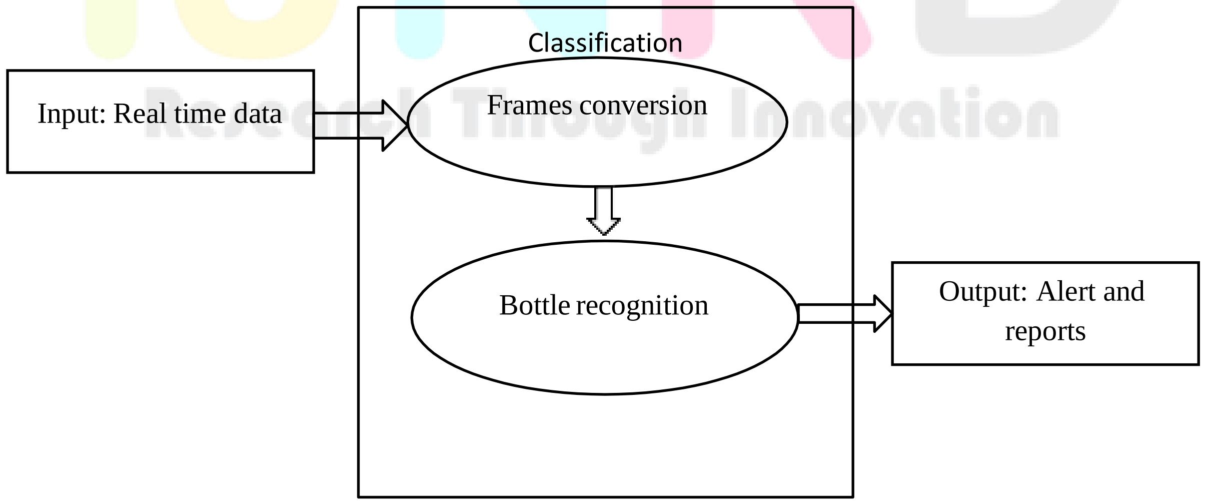


Fig. 7.4. DFD Level 3

7.2 USECASE DIAGRAM

A use case diagram is a type of UML diagram that represents the interactions between an actor (a user or system) and a system under various scenarios. The diagram provides a visual representation of the system's functionalities and the interactions between the actors and the system.

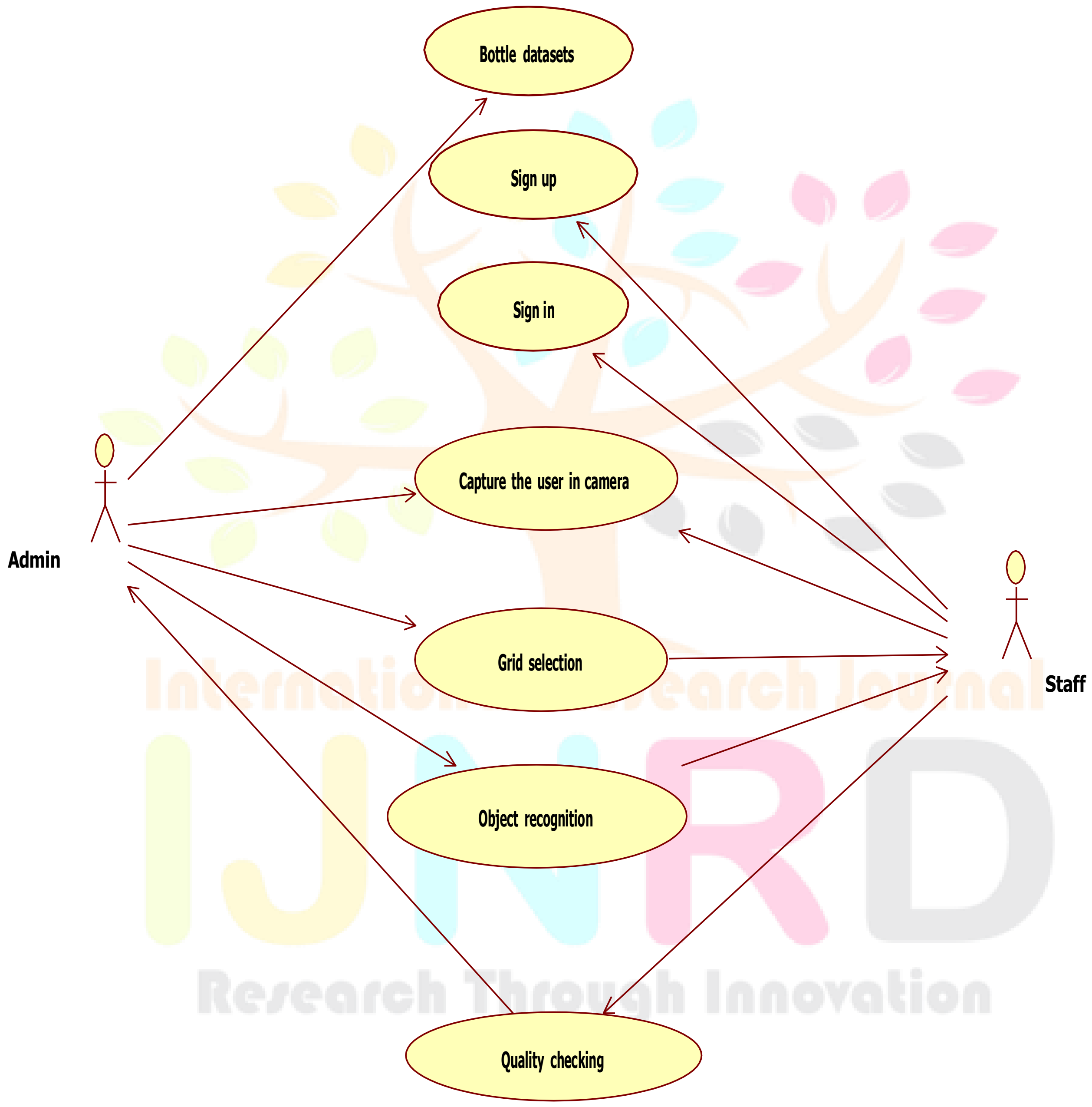


Fig. 7.5. Use Case Diagram

7.3 CLASS DIAGRAM

A class diagram is a type of visual representation used in software development to depict the classes, attributes, and methods of a system, as well as the relationships that exist between them. In this type of diagram, each class is represented as a box that includes its name, along with its attributes and methods. The attributes of a class are represented as variables that describe the characteristics of the class, while methods are represented as functions that define the behavior of the class.

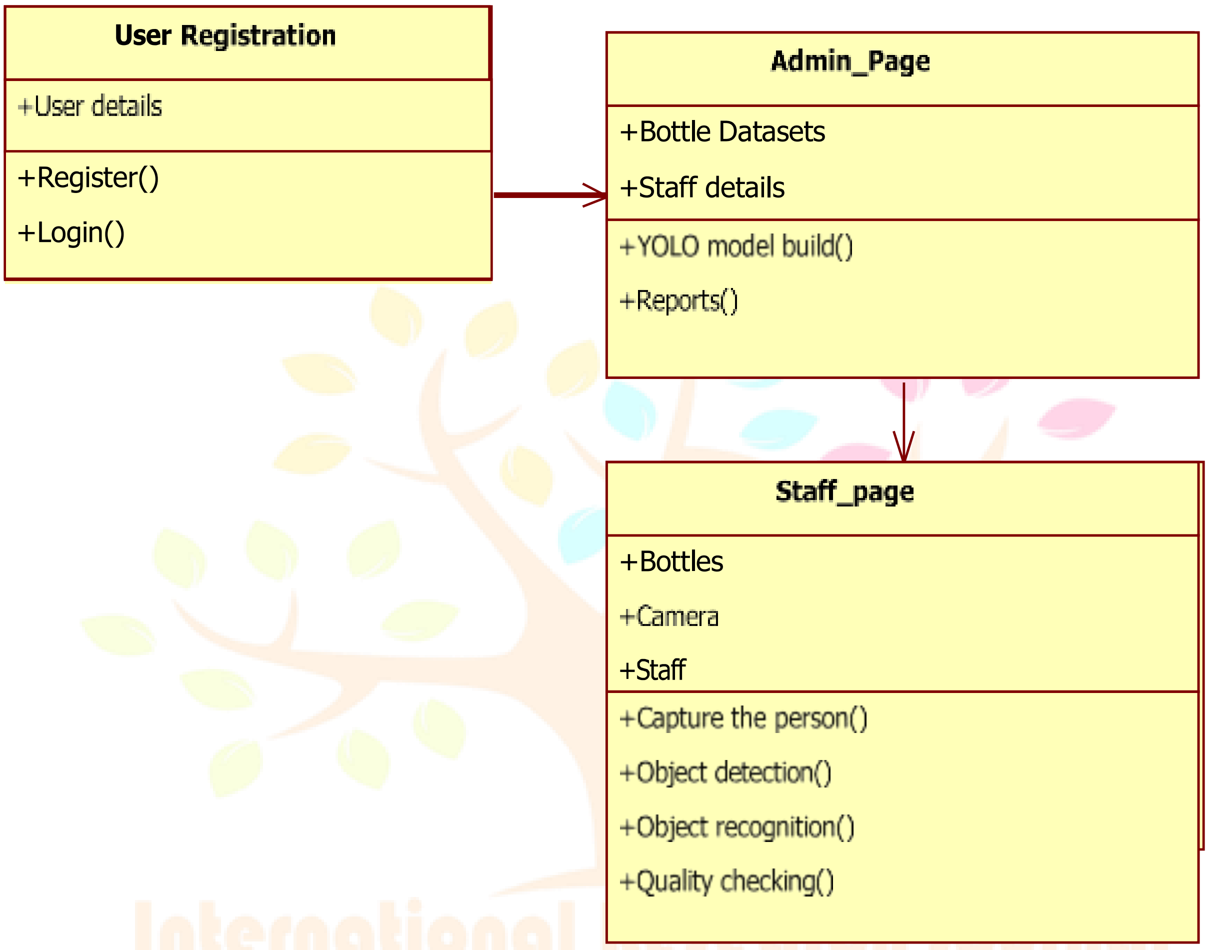


Fig. 7.6. Class diagram

7.4 ACTIVITY DIAGRAM

Activity diagrams show the workflow from a start point to the finish point detailing the many decision paths that exist in the progression of events contained in the activity. They may be used to detail situations where parallel processing may occur in the execution of some activities. This type of diagram helps business development and design teams visualize how data is processed and identify or improve certain aspects. the interactions between different components of a system, and the sequence of events that occur during a particular process. the interactions between objects in a system.

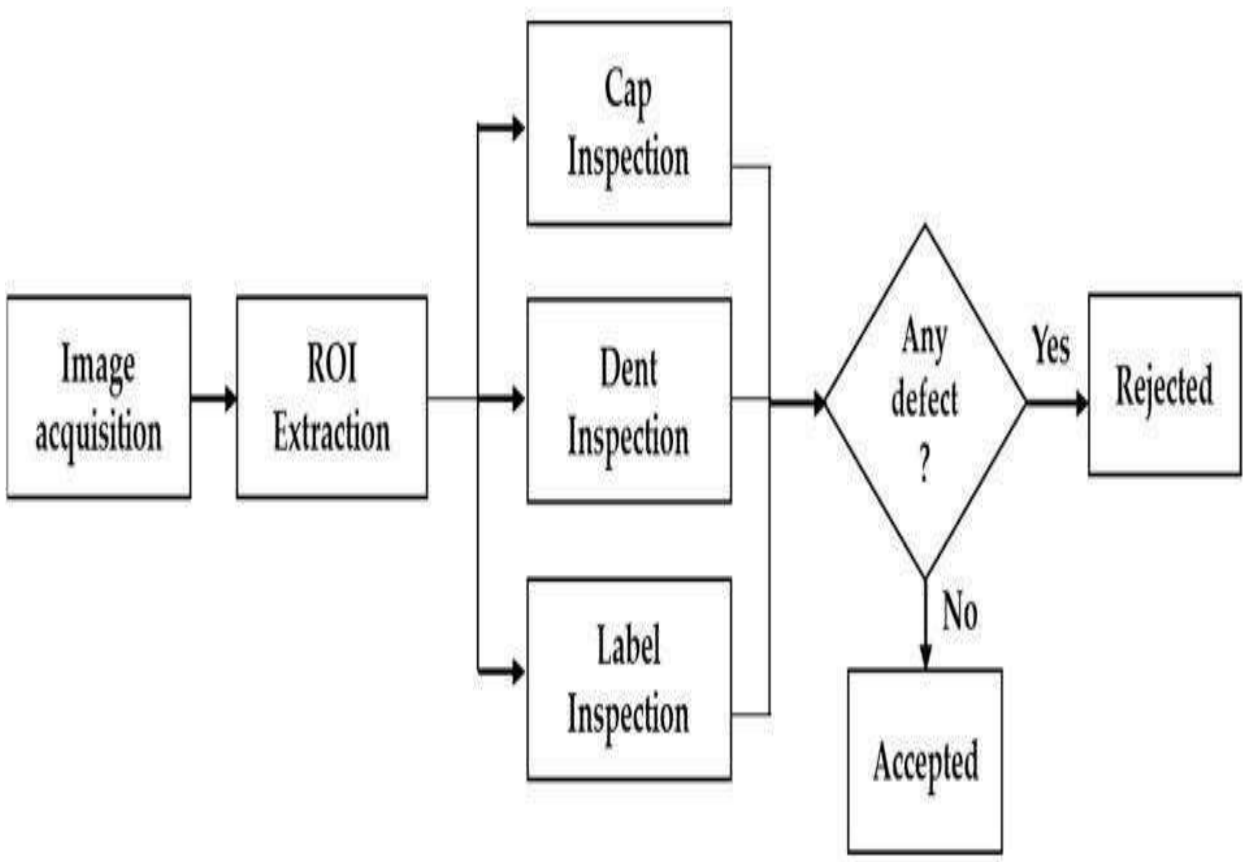


Fig. 7.7. Activity Diagram

7.5 SEQUENCE DIAGRAM

A sequence diagram is a type of UML (Unified Modeling Language) diagram that illustrates the interactions between objects in a system in a sequential order. It shows the sequence of messages exchanged between objects over time, and can be used to represent a variety of scenarios, such as the flow ofcontrol between objects, the interactions between different components of a system, and the sequence of events that occur during a particular process.

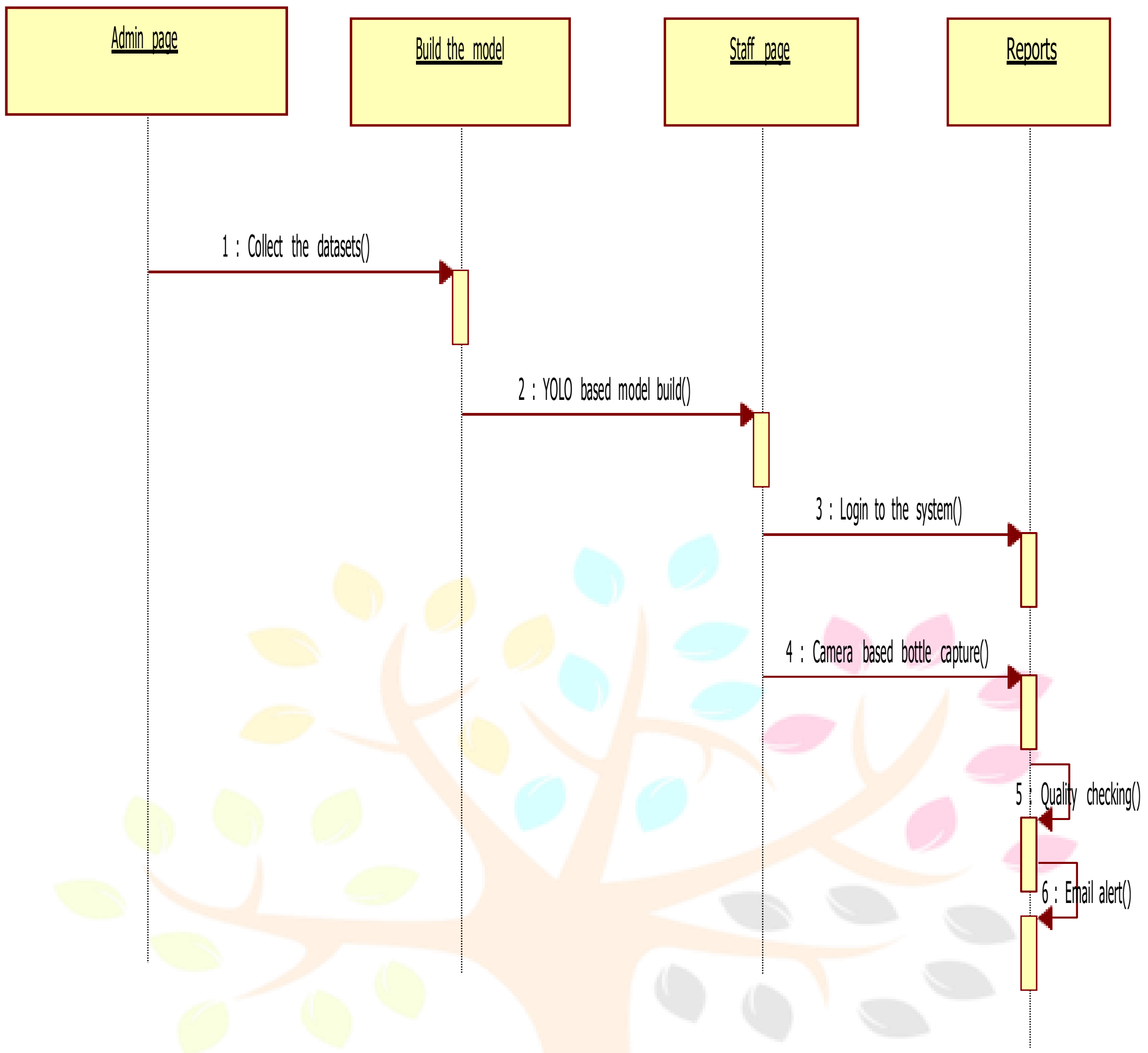

Admin page
Build the model
Staff page
Reports
1 : Collect the datasets()
2 : YOLO based model build()
3 : Login to the system()
4 : Camera based bottle capture()
5 : Quality checking()
6 : Email alert()

Fig. 7.8. Sequence Diagram

CHAPTER 8 SOFTWARE TESTING

8.1 TYPES OF TESTING

Software testing is a method of assessing the functionality of a software program. There are many different types of software testing but the two main categories are dynamic testing and static testing. Dynamic testing is an assessment that is conducted while the program is executed; static testing, on the other hand, is an examination of the program's code and associated documentation. Dynamic and static methods are often used together.

8.1.1 UNIT TESTING

The first test in the development process is the unit test. The source code is normally divided into modules, which in turn are divided into smaller units called units. These units have specific behavior. The test done on these units of code is called unit test.

Unit test depends upon the language on which the projectis developed. Unit tests ensure that each unique path of the project performs accurately to the documented specifications and contains clearly defined inputs and expected results. Functional and reliability testing in an Engineering environment. Producing tests for the behavior of components (nodes and vertices)of a product to ensure their correct behavior prior to system integration.

8.1.2 INTEGRATION TESTING

Integration testing is a type of software testing that focuses on verifying that the individual components of a software system work together as expected. The objective of integration testing is to ensure that the software system as a whole function correctly, and that the individual components interact with each other as intended. Integration testing is typically performed after the individual components of the software system have been tested and verified to work correctly.

8.1.3 FUNCTIONAL TESTING

Functional test can be defined as testing two or more modules together with the intent of finding defects, demonstrating that defects are not present, verifying that the module performs its intended functions as stated in the specification and establishing confidence that a program does what it is supposed to do.

8.1.4 SYSTEM TESTING

Testing is a set activity that can be planned and conducted systematically. Testing begins at the module level and works towards the integration of the entire computers-based system. Nothing is complete without testing, as it is a vital success of the system.

8.1.5 BLACK BOX TESTING

Testing without knowledge of the internal workings of the item being tested. Tests are usually functional. This testing can be done by the user who hasno knowledge of how the shortest path is found.

8.1.6 WHITE BOX TESTING

Testing based on an analysis of internal workings and structure of a piece of software. This testing can be done using the percentage value of load and energy. The tester should know what exactly is done in the internal program. Includes techniques such as Branch Testing and Path Testing. Also known as Structural Testing and Glass Box Testing.

8.2 ACCEPTANCE TESTING

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements. Functional and reliability testing in an Engineering environment. Producing tests for the behavior of components (nodes and vertices)of a product to ensure their correct behavior prior to system integration.

CHAPTER 9

SYSTEM IMPLEMENTATION

9.1 USER TRAINING

Implementation of software refers to the final installation of the package in its real environment, to the satisfaction of the intended users and the operation of the systems. The people are not sure that the software is meant to make their job easier.

- The active user must be aware of the benefits of using the systems.
- Their confidence in the software built up.
- Proper guidance is impaired to the user so that he is comfortable in using the application.

Before going ahead and viewing the systems, the user must know that for viewing the results, the server program should be running in the server. If the server object is not running on the server, the actual processes will not take place.

9.2 TRAINING ON THE APPLICATION SOFTWARE

To achieve the objective and benefits expected from the proposed system it is essential for the people who will be involved to be confident of their role in the new system. As system becomes more complex, the need for education and training is more and more important. Education is complementary to training. It brings life to formal training by explaining the back ground to the resources for them. Education involves creating the right atmosphere and motivating user staff. Education information can make training more interesting and more understandable.

9.3 OPERATIONAL DOCUMENTATION

After providing the necessary basic training on the computer awareness, the users will have to be trained on the new application software. This will give the underlying philosophy of the use of the new system such as the screen flow, screen design, type of help on the screen, type of errors while entering the data, the corresponding validation check at each entry and the ways to correct the dataentered. This training may be different across different user groups and across different levels of hierarchy.

Operational maintenance is the care and minor maintenance of equipment using procedures that do not require detailed technical knowledge of the equipment 's or system 's function and design. This category of operational maintenance normally consists of inspecting, cleaning, servicing, preserving, lubricating, and adjusting, as required. Such maintenance may also include minor parts replacement that does not require the person performing the work to have highly technical skills or to perform internal alignment.

9.4 SYSTEM MAINTENANCE

Once the implementation plan is decided, it is essential that the user of thesystem is made familiar and comfortable with the environment. A documentation providing the whole operations of the system is being developed. Useful tips andguidance is given inside the application itself to the user. The system is developed user friendly so that the user can work the system from the tips given in the application itself. The results obtained from the evaluation process help the organization to determine whether its information systems are effective and efficient or otherwise. the process of monitoring, evaluating, and modifying of existing information systems to make required or desirable improvements may betermed as system maintenance. System maintenance is an ongoing activity, which covers a wide variety of activities, including removing program and design errors, updating documentation and test data and updating user support.

9.5 CORRECTIVE MAINTENANCE

The maintenance phase of the software cycle is the time in which softwareperforms useful work. After a system is successfully implemented, it should be maintained in a proper manner. System maintenance is the important aspect in The software development life cycle.The need for system maintenance is to make adaptable to the changes in the system environment. There may be social, technical and other environment changes, which affect a system which is being implemented.

Software product enhancements may involve providing new functional capabilities, improving user displays and mode of interaction, upgrading the performance characteristics of the system. So only thru proper system maintenance procedures, the system can be adapted to cope up with these changes. Software maintenance is of course, far more than finding mistakes. This type of maintenance implies removing errors in a program, which might have crept in the system due to faulty design or wrong assumptions. Thus, in corrective maintenance, processing or performance failures are repaired.

9.6 ADAPTIVE MAINTENANCE

The first maintenance activity occurs because it is unreasonable to assume that software testing will uncover all latent errors in a large software system. During the use of any large program, errors will occur and be reported to the developer. The process that includes the diagnosis and correction of one or more errors is called Corrective Maintenance. Adaptive maintenance includes changes to the functionality of the system developed for specific customer needs. Adaptive maintenance also implies the need for modifications of certain functionalities, although the system works as expected and in this sense that there is no fault or error in the system. It usually occurs when there comes to a changein legal norms or a shift in the political business users.

Such changes usually cause divergence in originally set system and its parameters, and therefore the need for harmonization and implementation of new functionalities based on user requests is required. System maintenance is the important aspect in The software development life cycle.The need for system maintenance is to make adaptable to the changes in the system environment.

9.7 PERCEPTIVE MAINTENANCE

The second activity that contributes to a definition of maintenance occurs because of the rapid change that is encountered in every aspect of computing. Therefore, adaptive maintenance termed as an activity that modifies software to properly interfere with a changing environment is both necessary and commonplace. Perceptive Process Design is a user-friendly business process modelling environment that allows business users to diagram, model, display, and document and publishes business process maps and meets process-specific compliance requirements.

Perceptive Process Enterprise is a case-based business process management tool that supports complex case-handling and work process execution and automation for a variety of industries and organizations.

9.8 PREVENTIVE MAINTENANCE

The third activity that may be applied to a definition of maintenance occurs when a software package is successful. As the software is used, recommendations for new

International Research Journal
IJNRD
Research Through Innovation

capabilities, modifications to existing functions, and general enhancement are received from users. To satisfy requests in this category,perceptive maintenance is performed. This activity accounts for the majority of all efforts expended on software maintenance.

The care and servicing by personnel for the purpose of maintaining equipment in satisfactory operating condition by providing for systematic inspection, detection, and correction of incipient failures either before they occur or before they develop into major defects. The work carried out on equipment in order to avoid its breakdown or malfunction. It is a regular and routine action taken on equipment in order to prevent its breakdown. Maintenance, including tests, measurements, adjustments, parts replacement, and cleaning, performed specifically to prevent faults from occurring.

CHAPTER 10

CONCLUSION AND FUTURE ENHANCEMENT

10.1 CONCLUSION

In conclusion, the YOLO (You Only Look Once) algorithm provides a robust and efficient solution for object detection tasks, including the detection of defects in images of plastic bottles. By dividing the input image into a grid and predicting bounding boxes and class probabilities directly, YOLO achieves real-time detection with impressive accuracy. In the context of plastic bottle inspection, YOLO enables automated identification and classification of defects, facilitating quality control processes in industrial settings. Overall, YOLO's speed, accuracy, and ease of integration make it a valuable tool for defect detection and quality assurance across various industries. In addition to its technical capabilities, the YOLO algorithm offers several advantages that make it particularly well-suited for defect detection in industrial applications. One notable advantage is its efficiency in processing images in real-time, allowing for rapid analysis of large datasets and enabling timely decision-making in production environments. Moreover, YOLO's single-pass architecture simplifies model deployment and integration, reducing the computational overhead associated with complex object detection systems.

10.2 FUTURE ENHANCEMENT

Explore dynamic thresholding techniques and adaptive learning algorithms that can automatically adjust detection thresholds and adapt to changes in bottle designs, manufacturing processes, or environmental conditions over time. Integrate advanced sensor technologies such as hyperspectral imaging or thermal imaging alongside computer vision to capture additional information about the physical properties and composition of the water bottles, facilitating more comprehensive inspection.

APPENDIX - 1

SOURCE CODE

```
# YOLOv5 🚀 by Ultralytics, AGPL-3.0 license"""

Train a YOLOv5 model on a custom dataset.

Models and datasets download automatically from the latest YOLOv5 release.


Usage - Single-GPU training:

    $ python train.py --data coco128.yaml --weights yolov5s.pt --img 640 # from
pretrained (recommended)

    $ python train.py --data coco128.yaml --weights '' --cfg yolov5s.yaml --img640 # from
scratch


Usage - Multi-GPU DDP training:

    $ python -m torch.distributed.run --nproc_per_node 4 --master_port 1 train.py --data
coco128.yaml --weights yolov5s.pt --img 640 --device 0,1,2,3


Models:     https://github.com/ultralytics/yolov5/tree/master/models Datasets:

            https://github.com/ultralytics/yolov5/tree/master/data Tutorial:

            https://docs.ultralytics.com/yolov5/tutorials/train_custom_data """


import argparseimport

math import os import

random

import subprocessimport

sys

import time

from copy import deepcopy

from datetime import datetime, timedeltafrom

pathlib import Path
```

International Research Journal
IJNRD
Research Through Innovation

```
try:
    import comet_ml # must be imported before torch (if installed)except
ImportError:
    comet_ml = None

import numpy as npimport
torch
import torch.distributed as distimport
torch.nn as nn
import yaml
from torch.optim import lr_schedulerfrom
tqdm import tqdm

FILE = Path(__file__).resolve()
ROOT = FILE.parents[0]  # YOLOv5 root directoryif
str(ROOT) not in sys.path:
    sys.path.append(str(ROOT)) # add ROOT to PATH ROOT =
Path(os.path.relpath(ROOT, Path.cwd()))  # relative

import  val  as  validate  #  for  end-of-epoch  mAP  from
models.experimental      import      attempt_load      from
models.yolo import Model
from utils.autoanchor import check_anchors
from utils.autobatch import check_train_batch_sizefrom
utils.callbacks import Callbacks
from utils.dataloaders import create_dataloader
from utils.downloads import attempt_download, is_url
from utils.general import (LOGGER, TQDM_BAR_FORMAT, check_amp,check_dataset,
check_file, check_git_info,
                         check_git_status, check_img_size, check_requirements,check_suffix,
check_yaml, colorstr,
                         get_latest_run, increment_path, init_seeds, intersect_dicts,
```

International Research Journal
IJNRD
Research Through Innovation

```
labels_to_class_weights,
                        labels_to_image_weights, methods, one_cycle, print_args,print_mutation,
strip_optimizer,
                        yaml_save)
from utils.loggers import LOGGERS, Loggers
from utils.loggers.comet.comet_utils import check_comet_resumefrom
utils.loss import ComputeLoss
from utils.metrics import fitness from
utils.plots import plot_evolve
from utils.torch_utils import (EarlyStopping, ModelEMA, de_parallel,select_device,
smart_DDP, smart_optimizer,
                            smart_resume, torch_distributed_zero_first)


LOCAL_RANK = int(os.getenv('LOCAL_RANK', -1)) #
https://pytorch.org/docs/stable/elastic/run.html
RANK = int(os.getenv('RANK',  -1)) WORLD_SIZE =
int(os.getenv('WORLD_SIZE', 1))GIT_INFO =
check_git_info()


def train(hyp, opt, device, callbacks):  # hyp is path/to/hyp.yaml or hypdictionary
    save_dir, epochs, batch_size, weights, single_cls, evolve, data, cfg, resume,noval,
nosave, workers, freeze = \
        Path(opt.save_dir), opt.epochs, opt.batch_size, opt.weights, opt.single_cls,opt.evolve,
opt.data, opt.cfg, \
        opt.resume, opt.noval, opt.nosave, opt.workers, opt.freeze
    callbacks.run('on_pretrain_routine_start')


    # Directories
    w = save_dir / 'weights'  # weights dir
    (w.parent if evolve else w).mkdir(parents=True, exist_ok=True) # make dirlast, best = w
    / 'last.pt', w / 'best.pt'
    # Hyperparameters
    if isinstance(hyp, str):
        with open(hyp, errors='ignore') as f:
            hyp = yaml.safe_load(f) # load hyps dict LOGGER.info(colorstr('hyperparameters: ')
    + ', '.join(f'{k}={v}' for k, v in
hyp.items()))
    opt.hyp = hyp.copy()  # for saving hyps to checkpoints


    # Save run settingsif not
    evolve:
        yaml_save(save_dir / 'hyp.yaml', hyp)
        yaml_save(save_dir / 'opt.yaml', vars(opt))


    # Loggers
    data_dict = None
    if RANK in {-1, 0}:
        include_loggers = list(LOGGERS)
        if getattr(opt, 'ndjson_console', False):
            include_loggers.append('ndjson_console')
        if getattr(opt, 'ndjson_file', False):
            include_loggers.append('ndjson_file')


        loggers = Loggers(
            save_dir=save_dir,
            weights=weights, opt=opt,
            hyp=hyp, logger=LOGGER,
            include=tuple(include_loggers),
        )

        # Register actions
        for k in methods(loggers):
            callbacks.register_action(k, callback=getattr(loggers, k))

        # Process custom dataset artifact linkdata_dict =
        loggers.remote_dataset
        if resume:  # If resuming runs from remote artifact
            weights, epochs, hyp, batch_size = opt.weights, opt.epochs, opt.hyp,opt.batch_size


    # Config
    plots = not evolve and not opt.noplots  # create plotscuda =
    device.type != 'cpu'
    init_seeds(opt.seed + 1 + RANK, deterministic=True)with
    torch_distributed_zero_first(LOCAL_RANK):
        data_dict = data_dict or check_dataset(data)  # check if Nonetrain_path,
    val_path = data_dict['train'], data_dict['val']
    nc = 1 if single_cls else int(data_dict['nc'])  # number of classes
    names = {0: 'item'} if single_cls and len(data_dict['names']) != 1 elsedata_dict['names'] #
class names
    is_coco = isinstance(val_path, str) and val_path.endswith('coco/val2017.txt')# COCO
dataset


    # Model
    check_suffix(weights, '.pt')  # check weights
    pretrained = weights.endswith('.pt')
    if pretrained:
        with   torch_distributed_zero_first(LOCAL_RANK):
            weights = attempt_download(weights) # download if not found locally ckpt =
        torch.load(weights, map_location='cpu')  # load checkpoint to CPU
to avoid CUDA memory leak
        model = Model(cfg or ckpt['model'].yaml, ch=3, nc=nc,
anchors=hyp.get('anchors')).to(device)  # create
        exclude = ['anchor'] if (cfg or hyp.get('anchors')) and not resume else [] #exclude keys
        csd = ckpt['model'].float().state_dict()  # checkpoint state_dict as FP32 csd =
        intersect_dicts(csd, model.state_dict(), exclude=exclude) # intersect
        model.load_state_dict(csd, strict=False) # load
        LOGGER.info(f'Transferred {len(csd)}/{len(model.state_dict())} itemsfrom
{weights}') # report
    else:
        model = Model(cfg, ch=3, nc=nc, anchors=hyp.get('anchors')).to(device) #create

    amp = check_amp(model)  # check AMP


    # Freeze

    freeze = [f'model.{x}.' for x in (freeze if len(freeze) > 1 elserange(freeze[0]))] #
layers to freeze

    for k, v in model.named_parameters():

        v.requires_grad = True  # train all layers

        # v.register_hook(lambda x: torch.nan_to_num(x))  # NaN to 0(commented
for erratic training results)

        if any(x in k for x in freeze):

            LOGGER.info(f'freezing {k}')

            v.requires_grad = False


    # Image size

    gs = max(int(model.stride.max()), 32)  # grid size (max stride)

    imgsz = check_img_size(opt.imgsz, gs, floor=gs * 2) # verify imgsz is gs-multiple


    # Batch size

    if RANK == -1 and batch_size == -1: # single-GPU only, estimate best batchsize

        batch_size = check_train_batch_size(model, imgsz, amp)

        loggers.on_params_update({'batch_size': batch_size})


    # Optimizer

    nbs = 64  # nominal batch size
    accumulate = max(round(nbs / batch_size), 1) # accumulate loss beforeoptimizing

    hyp['weight_decay']  *=  batch_size  * accumulate  /  nbs #  scale  weight_decayoptimizer =

    smart_optimizer(model, opt.optimizer, hyp['lr0'],
hyp['momentum'], hyp['weight_decay'])


    # Scheduler if

    opt.cos_lr:

        lf = one_cycle(1, hyp['lrf'], epochs)  # cosine 1->hyp['lrf']else:
```

```
        lf = lambda x: (1 - x / epochs) * (1.0 - hyp['lrf']) + hyp['lrf']  # linear
    scheduler = lr_scheduler.LambdaLR(optimizer, lr_lambda=lf)  #
plot_lr_scheduler(optimizer, scheduler, epochs)


    # EMA
    ema = ModelEMA(model) if RANK in {-1, 0} else None


    # Resume
    best_fitness, start_epoch = 0.0, 0if
    pretrained:
        if resume:
            best_fitness, start_epoch, epochs = smart_resume(ckpt, optimizer, ema,weights,
epochs, resume)
        del ckpt, csd


    # DP mode
    if cuda and RANK == -1 and torch.cuda.device_count() > 1:LOGGER.warning(
            'WARNING ⚠ DP not recommended, use torch.distributed.run for best DDP Multi-
GPU results.\n'
            'See Multi-GPU Tutorial at
https://docs.ultralytics.com/yolov5/tutorials/multi_gpu_training to get started.'
        )
        model = torch.nn.DataParallel(model)


    # SyncBatchNorm
    if opt.sync_bn and cuda and RANK != -1:model =
torch.nn.SyncBatchNorm.convert_sync_batchnorm(model).to(device) LOGGER.info('Using
        SyncBatchNorm()')
    # Trainloader
    train_loader, dataset = create_dataloader(train_path,imgsz,
batch_size // WORLD_SIZE,gs,
single_cls, hyp=hyp,
augment=True,
cache=None if opt.cache == 'val' else opt.cache,rect=opt.rect,
rank=LOCAL_RANK, workers=workers,
image_weights=opt.image_weights,
quad=opt.quad, prefix=colorstr('train: '),
shuffle=True,
   seed=opt.seed)
  labels = np.concatenate(dataset.labels, 0)
  mlc = int(labels[:, 0].max())  # max label class
  assert mlc < nc, f'Label class {mlc} exceeds nc={nc} in {data}. Possible classlabels are
0-{nc - 1}'

  # Process 0
  if RANK in {-1, 0}:
    val_loader = create_dataloader(val_path,imgsz,
batch_size // WORLD_SIZE * 2,gs,
single_cls,hyp=hyp,
cache=None if noval else opt.cache,
rect=True,
rank=-1, workers=workers * 2,
pad=0.5, prefix=colorstr('val: '))[0]
    if not resume:
      if not opt.noautoanchor:
        check_anchors(dataset, model=model, thr=hyp['anchor_t'],
imgsz=imgsz) # run AutoAnchor
      model.half().float() # pre-reduce anchor precision
    callbacks.run('on_pretrain_routine_end', labels, names)

  # DDP mode
  if cuda and RANK != -1: model =
    smart_DDP(model)
  # Model attributes
  nl = de_parallel(model).model[-1].nl  # number of detection layers (to scalehyps)
  hyp['box'] *= 3 / nl  # scale to layers
  hyp['cls'] *= nc / 80 * 3 / nl  # scale to classes and layers
  hyp['obj'] *= (imgsz / 640) ** 2 * 3 / nl  # scale to image size and layers
  hyp['label_smoothing'] = opt.label_smoothing
  model.nc = nc # attach number of classes to model model.hyp = hyp  #
  attach hyperparameters to model
  model.class_weights = labels_to_class_weights(dataset.labels, nc).to(device)
* nc  # attach class weights
  model.names = names

  # Start training t0 =
  time.time()
  nb = len(train_loader)  # number of batches
  nw = max(round(hyp['warmup_epochs'] * nb), 100)  # number of warmupiterations, max(3
epochs, 100 iterations)
  # nw = min(nw, (epochs - start_epoch) / 2 * nb) # limit warmup to < 1/2 oftraining
  last_opt_step = -1
  maps = np.zeros(nc)  # mAP per class
  results = (0, 0, 0, 0, 0, 0, 0)  # P, R, mAP@.5, mAP@.5-.95, val_loss(box,obj, cls)
  scheduler.last_epoch = start_epoch - 1  # do not movescaler =
  torch.cuda.amp.GradScaler(enabled=amp)
        scaler.update()
        optimizer.zero_grad() if ema:
            ema.update(model)
        last_opt_step = ni

      # Log
      if RANK in {-1, 0}:
        mloss = (mloss * i + loss_items) / (i + 1)  # update mean lossesmem =
        f'{torch.cuda.memory_reserved() / 1E9 if
```

```
torch.cuda.is_available() else 0:.3g}G' # (GB)
        pbar.set_description(('%11s' * 2 + '%11.4g' * 5) %
                    (f'{epoch}/{epochs - 1}', mem, *mloss, targets.shape[0],
imgs.shape[-1]))
        callbacks.run('on_train_batch_end', model, ni, imgs, targets, paths,list(mloss))
        if callbacks.stop_training:return
      # end batch___________________-_____________________________-
_______________-


    # Scheduler
    lr = [x['lr'] for x in optimizer.param_groups]  # for loggersscheduler.step()

    if RANK in {-1, 0}:# mAP
      callbacks.run('on_train_epoch_end', epoch=epoch)
      ema.update_attr(model, include=['yaml', 'nc', 'hyp', 'names', 'stride','class_weights'])
      final_epoch = (epoch + 1 == epochs) or stopper.possible_stopif not
      noval or final_epoch:  # Calculate mAP
        results, maps, _ = validate.run(data_dict,
batch_size=batch_size // WORLD_SIZE * 2, imgsz=imgsz,
half=amp, model=ema.ema,
single_cls=single_cls,
dataloader=val_loader,
save_dir=save_dir, plots=False,
callbacks=callbacks,
compute_loss=compute_loss)


      # Update best mAP
      fi = fitness(np.array(results).reshape(1, -1)) # weighted combination of[P, R,
mAP@.5, mAP@.5-.95]
      stop = stopper(epoch=epoch, fitness=fi)  # early stop checkif fi >
      best_fitness:
        best_fitness = fi
      log_vals = list(mloss) + list(results) + lr callbacks.run('on_fit_epoch_end',
```

International Research Journal IJNRD Research Through Innovation

```
        log_vals, epoch, best_fitness, fi)


        # Save model
        if (not nosave) or (final_epoch and not evolve):  # if saveckpt = {
            'epoch': epoch, 'best_fitness':
            best_fitness,
            'model': deepcopy(de_parallel(model)).half(),'ema':
            deepcopy(ema.ema).half(), 'updates': ema.updates,
            'optimizer': optimizer.state_dict(),'opt':
            vars(opt),
            'git': GIT_INFO,  # {remote, branch, commit} if a git repo'date':
            datetime.now().isoformat()}


fi)
      # Save last, best and delete
      torch.save(ckpt, last)
      if best_fitness == fi: torch.save(ckpt,
        best)
      if opt.save_period > 0 and epoch % opt.save_period == 0:torch.save(ckpt, w /
        f'epoch{epoch}.pt')
      del ckpt
      callbacks.run('on_model_save', last, epoch, final_epoch, best_fitness,
# EarlyStopping
if RANK != -1:  # if DDP training
   broadcast_list = [stop if RANK == 0 else None] dist.broadcast_object_list(broadcast_list,
   0)  # broadcast 'stop' to all
ranks

if RANK != 0:
   stop = broadcast_list[0]
      if stop:
        break  # must break all DDP ranks
        # end epoch -________________________________________________-
________________-

    # end training -_________________________________________________-
________________- if RANK

    in {-1, 0}:

        LOGGER.info(f'\n{epoch - start_epoch + 1} epochs completed in
{(time.time() - t0) / 3600:.3f} hours.')for f in

        last, best:

            if f.exists():

                strip_optimizer(f)  # strip optimizersif f is

                best:

                    LOGGER.info(f'\nValidating {f}...')results, _, _ =

                    validate.run(

                        data_dict,

                        batch_size=batch_size // WORLD_SIZE * 2,

                        imgsz=imgsz,

                        model=attempt_load(f, device).half(),

                        iou_thres=0.65 if is_coco else 0.60,  # best pycocotools at iou
0.65

                        single_cls=single_cls,

                        dataloader=val_loader,

                        save_dir=save_dir,

                        save_json=is_coco, verbose=True,

                        plots=plots, callbacks=callbacks,

                        compute_loss=compute_loss)  # val best model with plots
```

International Research Journal IJNRD Research Through Innovation

```
    # NDJSON logging if
        is_url(opt_data):
            opt.data = check_file(opt_data)  # avoid HUB resume auth timeout
    else:
        opt.data, opt.cfg, opt.hyp, opt.weights, opt.project = \
            check_file(opt.data), check_yaml(opt.cfg), check_yaml(opt.hyp),
str(opt.weights), str(opt.project)  # checks
        assert len(opt.cfg) or len(opt.weights), 'either --cfg or --weights must bespecified'
        if opt.evolve:
            if opt.project == str(ROOT / 'runs/train'):  # if default project name,rename
to runs/evolve
                opt.project = str(ROOT / 'runs/evolve')
            opt.exist_ok, opt.resume = opt.resume, False  # pass resume to exist_okand
disable resume
        if opt.name == 'cfg':
            opt.name = Path(opt.cfg).stem # use model.yaml as name
        opt.save_dir = str(increment_path(Path(opt.project) / opt.name,
exist_ok=opt.exist_ok))


    # DDP mode
    device = select_device(opt.device, batch_size=opt.batch_size)if
    LOCAL_RANK != -1:
        msg = 'is not compatible with YOLOv5 Multi-GPU DDP training'
        assert not opt.image_weights, f'--image-weights {msg}'
        assert not opt.evolve, f'--evolve {msg}'
        assert opt.batch_size != -1, f'AutoBatch with --batch-size -1 {msg}, please
pass a valid --batch-size'
        assert opt.batch_size % WORLD_SIZE == 0, f'--batch-size
{opt.batch_size} must be multiple of WORLD_SIZE'

@app.route("/")def
homepage():
    return  render_template('index.html')
```

```
@app.route("/Home")
def Home():
    return  render_template('index.html')

@app.route("/AdminLogin")
def AdminLogin():
    return  render_template('AdminLogin.html')

@app.route("/UserLogin")
def UserLogin():
    return  render_template('UserLogin.html')

@app.route("/NewStaff")
def NewStaff():
    return  render_template('NewStaff.html')

@app.route("/adminlogin", methods=['GET', 'POST'])
def adminlogin():
    error = None
    if request.method == 'POST':
        if request.form['uname'] == 'admin' and request.form['password'] =='admin':
            return render_template('AdminHome.html')else:
            return render_template('index.html', error=error)

@app.route("/AdminHome")
def AdminHome():
    return  render_template('AdminHome.html')

@app.route("/newstaff", methods=['GET', 'POST'])
def newstaff():
    if request.method == 'POST': name
        = request.form['name'] gender =
```

International Research Journal
IJNRD
Research Through Innovation

```
        request.form['gender']


        mobile = request.form['pnumber']

        email = request.form['email'] address

        = request.form['address'] username =

        request.form['uname'] password =

        request.form['password']


        conn = mysql.connector.connect(user='root', password='', host='localhost',
database='2bottledefectdb')

        cursor = conn.cursor()

        cursor.execute(

            "insert into stafftb values('','" + name + "','" + gender + "','" + mobile +"','"
+ email + "','" + address + "','" + username + "','" + password + "')")

        conn.commit()

        conn.close()

        flash("Record Saved!")

    return render_template('NewStaff.html')

@app.route("/Remove")

def Remove():

    did = request.args.get('id')


    conn = mysql.connector.connect(user='root', password='', host='localhost',
database='2bottledefectdb')

    cursor = conn.cursor()

    cursor.execute("delete from stafftb where Id='" + did + "' ")

    conn.commit()

    conn.close()


    conn = mysql.connector.connect(user='root', password='', host='localhost',
database='2bottledefectdb')

    # cursor = conn.cursor()

    cur = conn.cursor()
    cur.execute("SELECT * FROM stafftb ")
    data = cur.fetchall()
    flash('Staff Remove successfully..!')

    return  render_template('StaffInfo.html',   data=data)

@app.route("/StaffInfo")
def StaffInfo():
    conn  = mysql.connector.connect(user='root',  password='',  host='localhost',
database='2bottledefectdb')
    cur = conn.cursor()
    msg['Subject'] = "Alert"
    # string to store the body of the mail
    body = message
    # attach the body with the msg instance
    msg.attach(MIMEText(body, 'plain'))

    # creates SMTP session
    s = smtplib.SMTP('smtp.gmail.com', 587)

    # start TLS for security
    s.starttls()

    # Authentication
    s.login(fromaddr, "qmgn xecl bkqv musr")

    # Converts the Multipart msg into a string
    text = msg.as_string()

    # sending the mail
    s.sendmail(fromaddr, toaddr, text)
    # terminating the session
    s.quit()
if__name__== '__main__':
    app.run(debug=True, use_reloader=True)
```

APPENDIX - 2

SCREENSHOTS

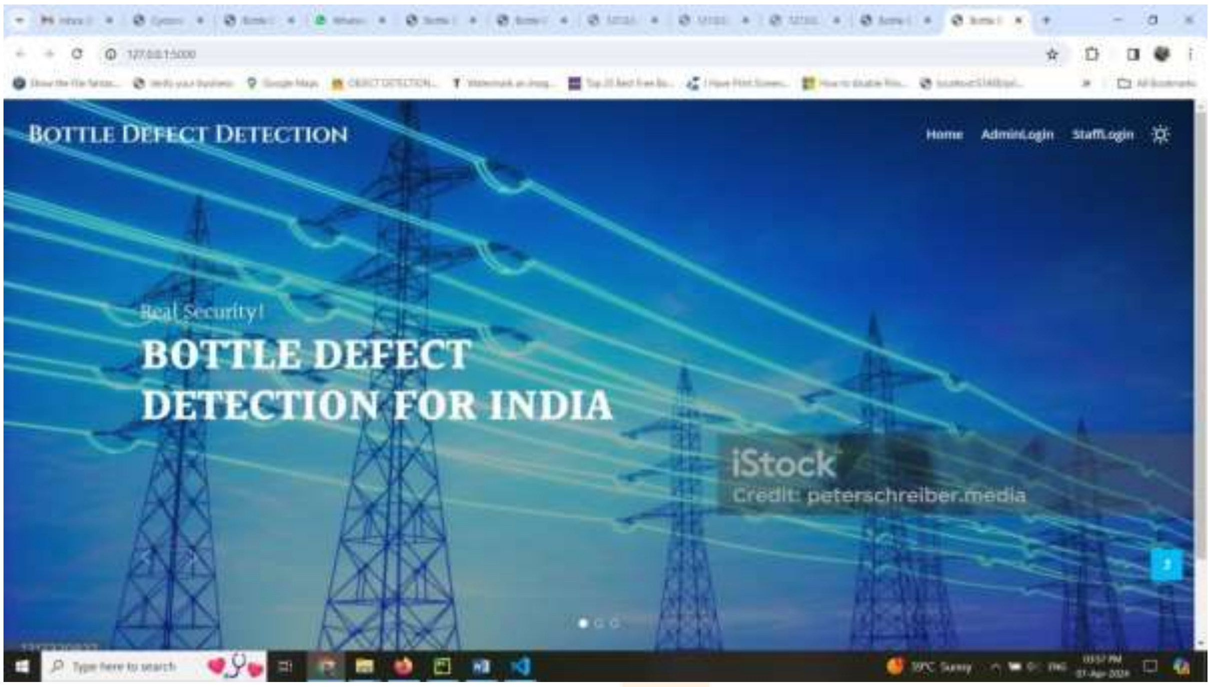


Fig.1. HOME PAGE

International Research Journal

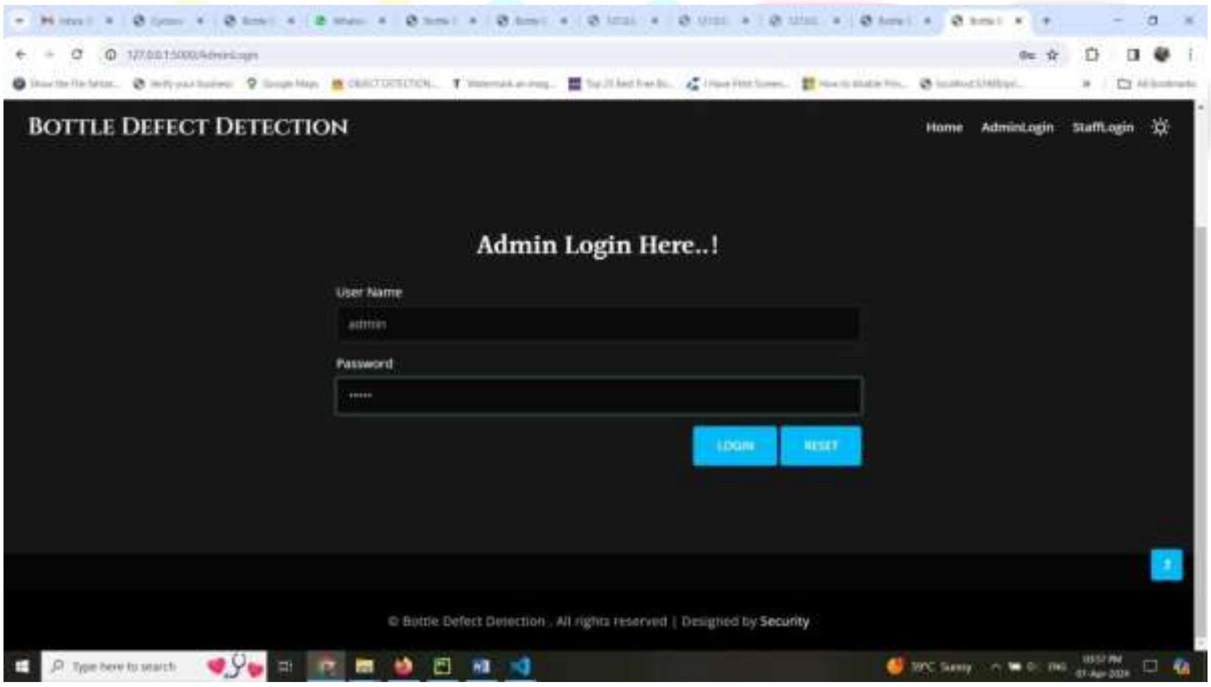
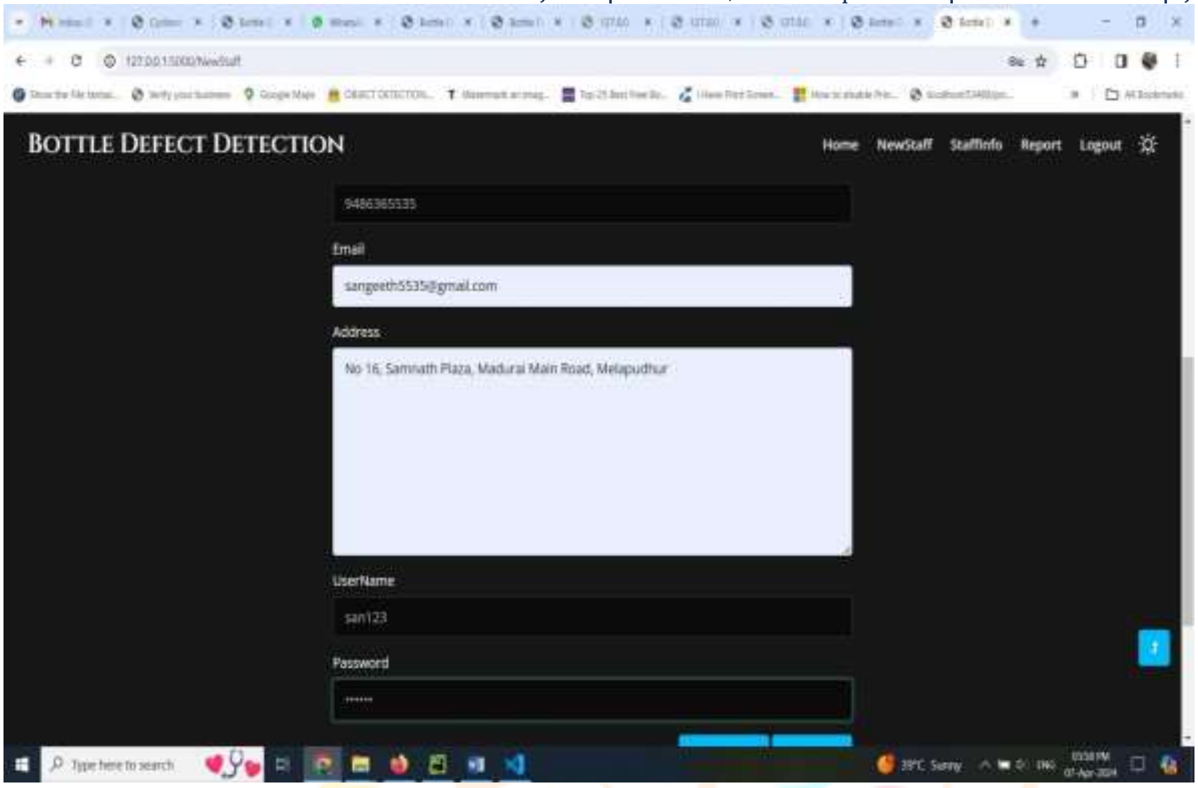


Fig.2. ADMIN LOGIN



9486365535

Email

sangeeth5535@gmail.com

Address

No 16, Samnath Plaza, Madurai Main Road, Melapudhur

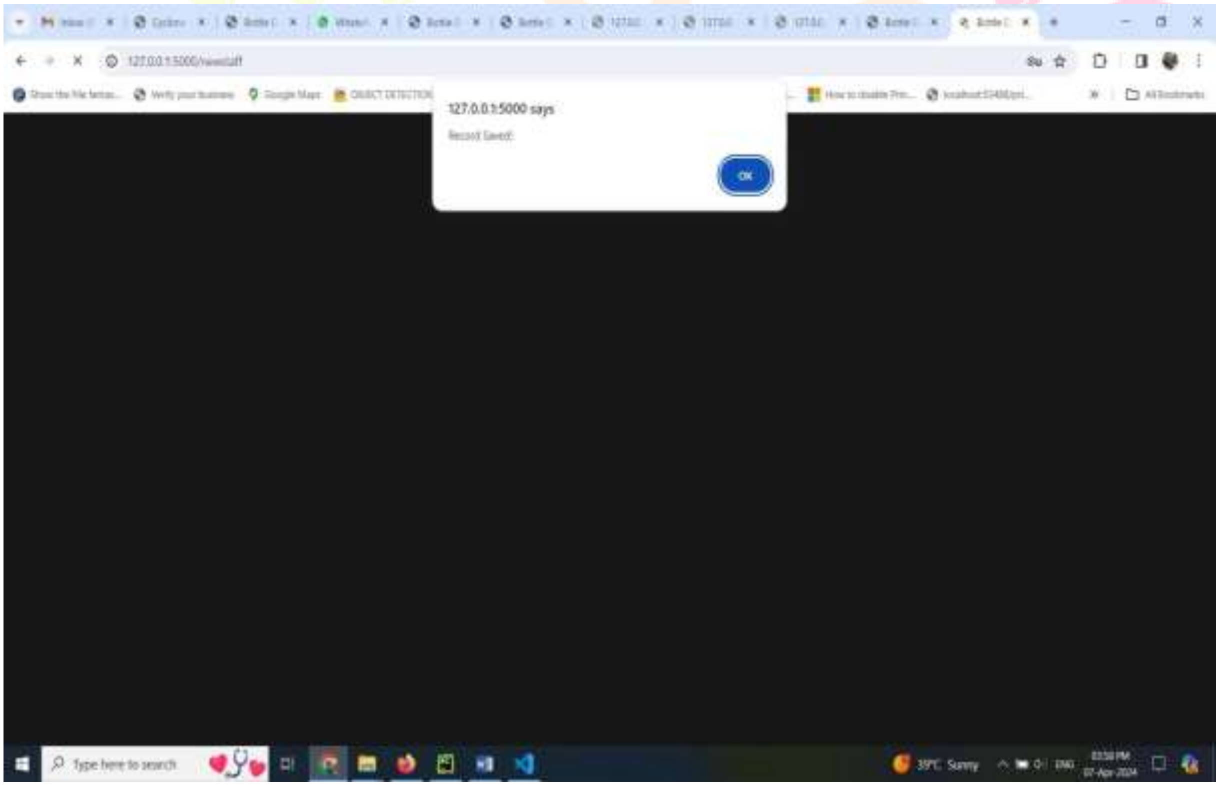
UserName

san123

Password

••••••

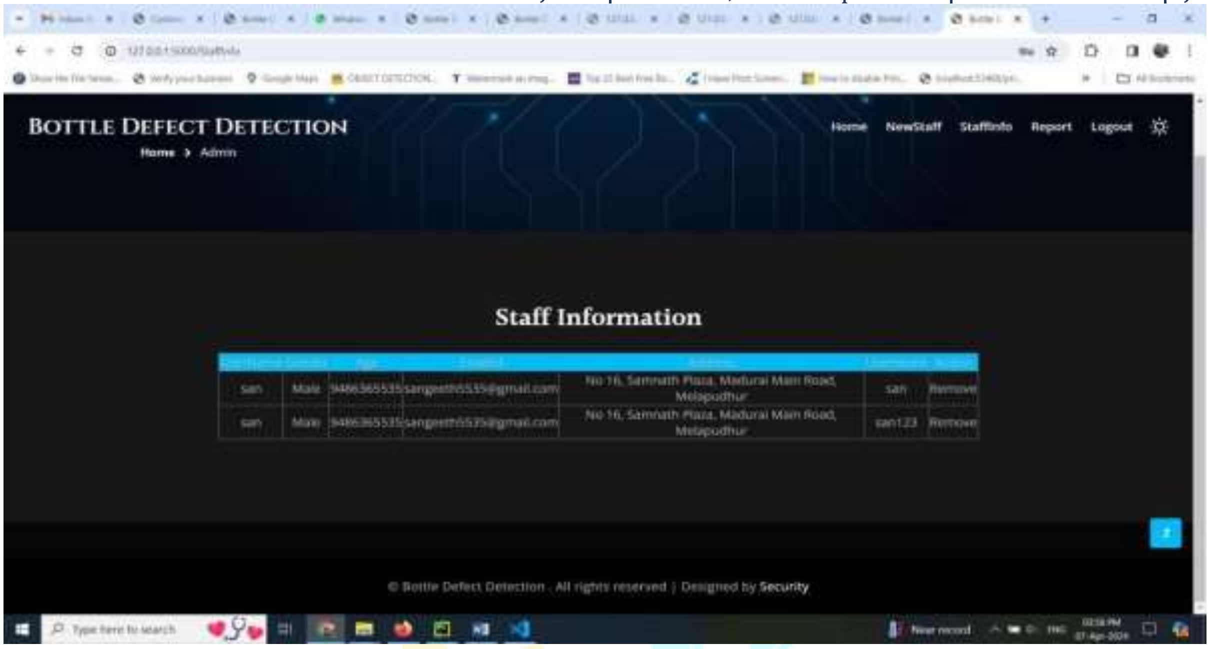
Fig.3. NEW STAFF REGISTRATION



127.0.0.1:5000 says

Record Saved!

Fig.4. USER STATUS

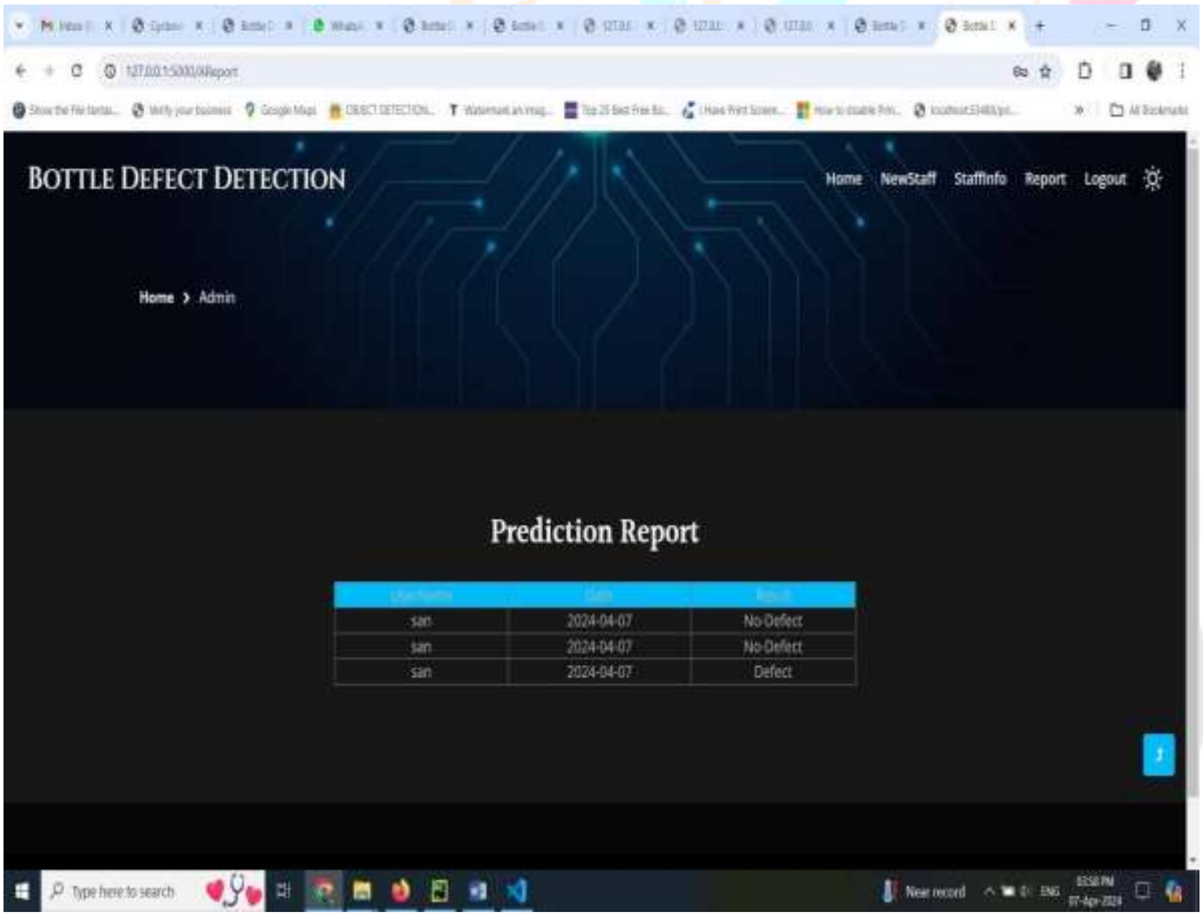


Staff Information

StaffName	Gender	[illegible]	EmailId	Address	Username	Action
san	Male	9486365535	sangeeth5535@gmail.com	No 16, Samnath Plaza, Madurai Main Road, Melapudhur	san	Remove
san	Male	9486365535	sangeeth5535@gmail.com	No 16, Samnath Plaza, Madurai Main Road, Melapudhur	san123	Remove

© Bottle Defect Detection . All rights reserved | Designed by Security

Fig.5. STAFF INFORMATION



Prediction Report

UserName	Date	Result
san	2024-04-07	No-Defect
san	2024-04-07	No-Defect
san	2024-04-07	Defect

Fig.6. PREDICTION REPORT

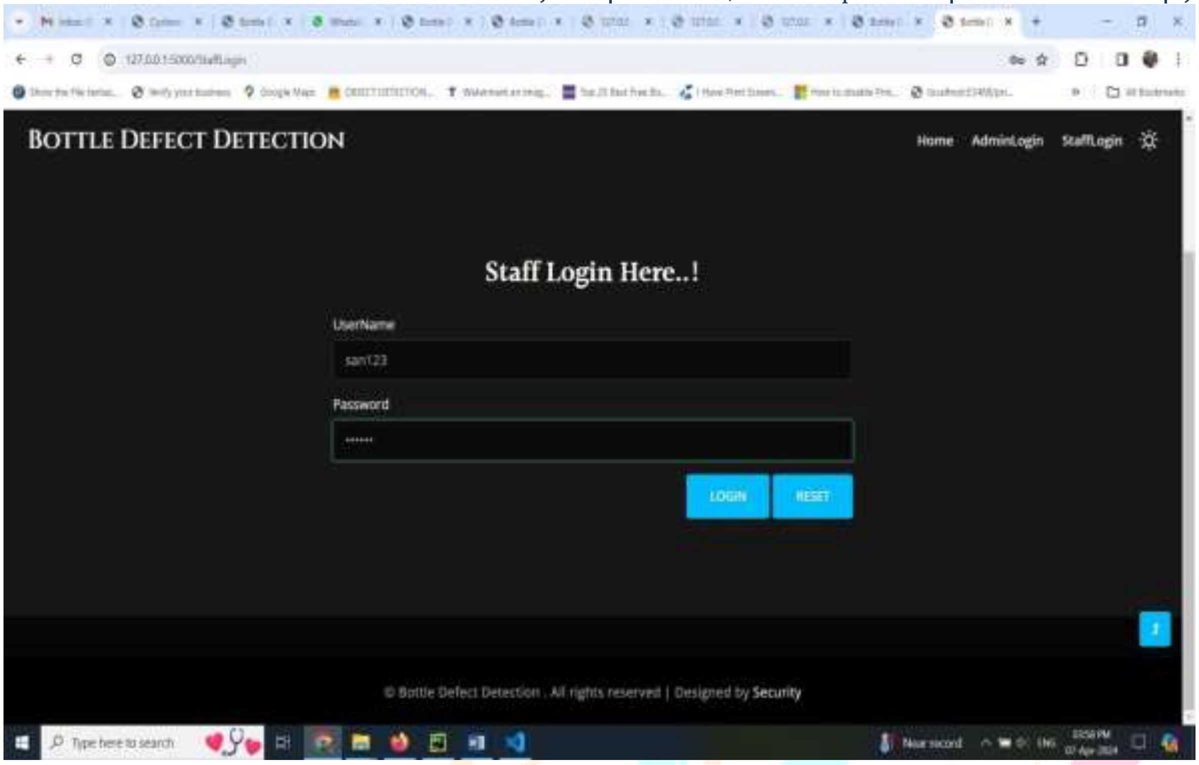


Fig.7. STAFF LOGIN

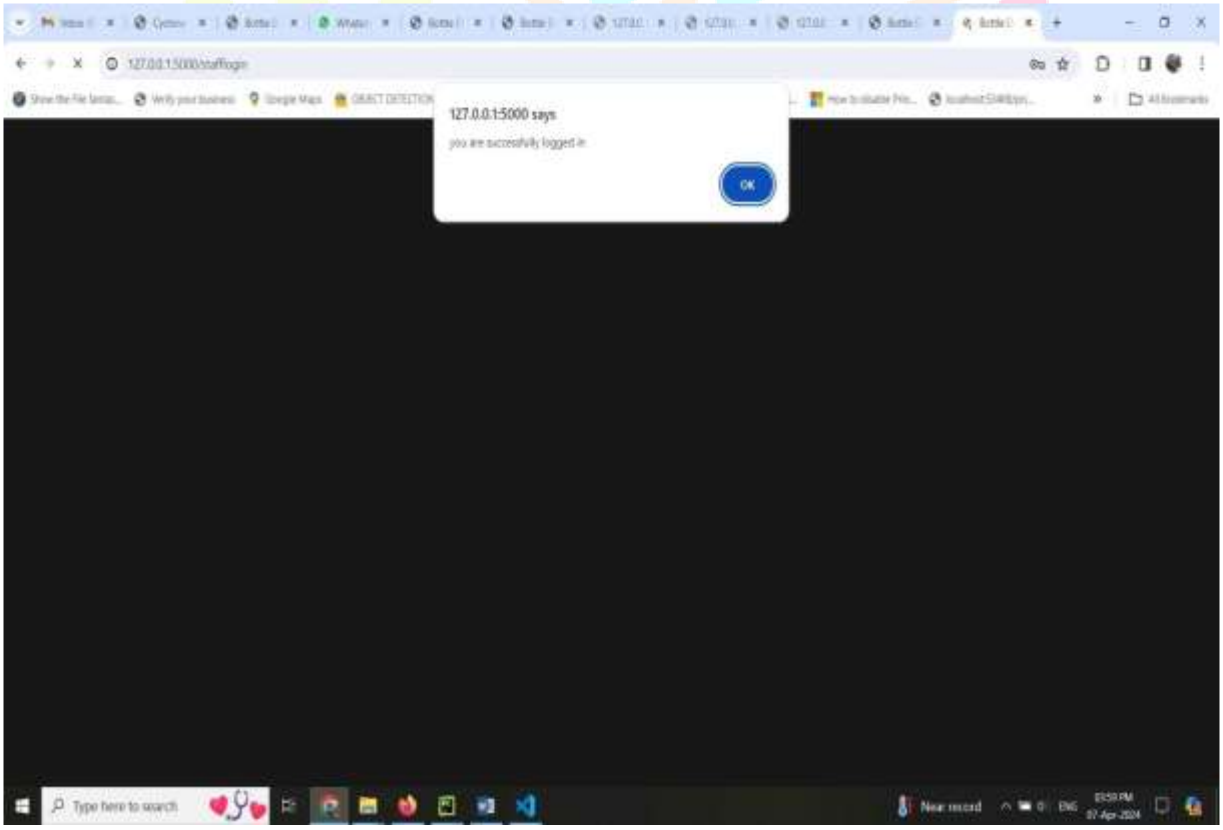
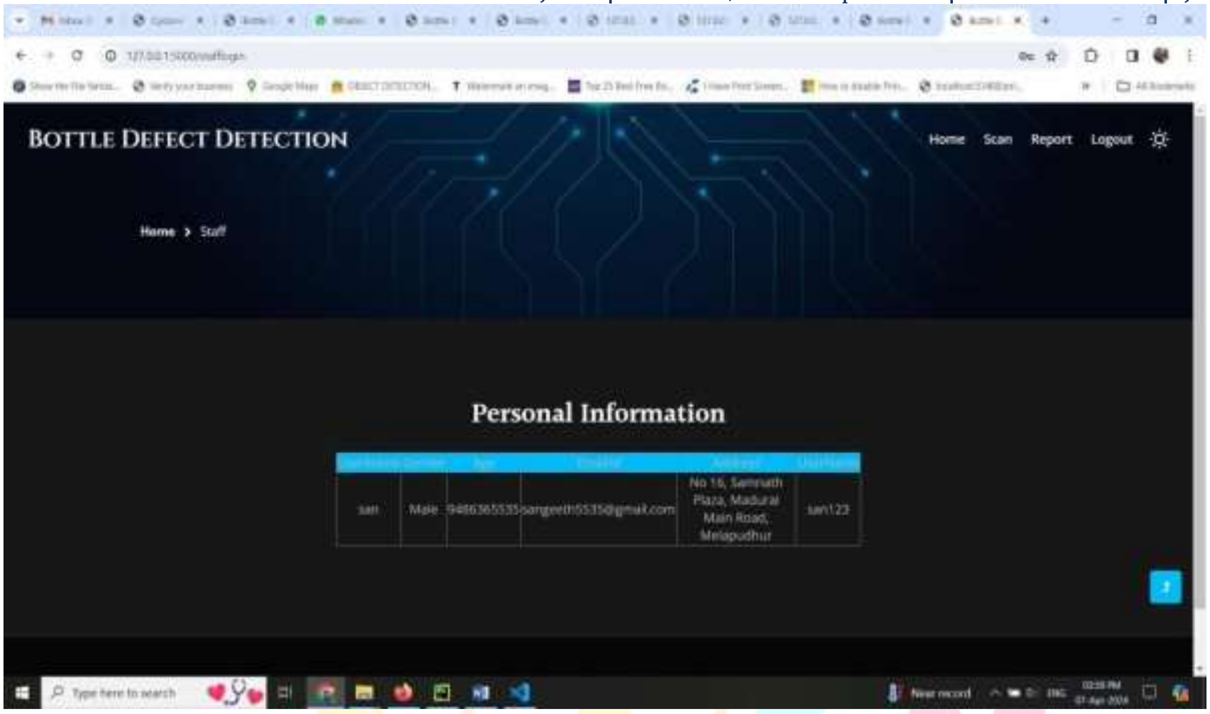


Fig.8. LOGIN STATUS

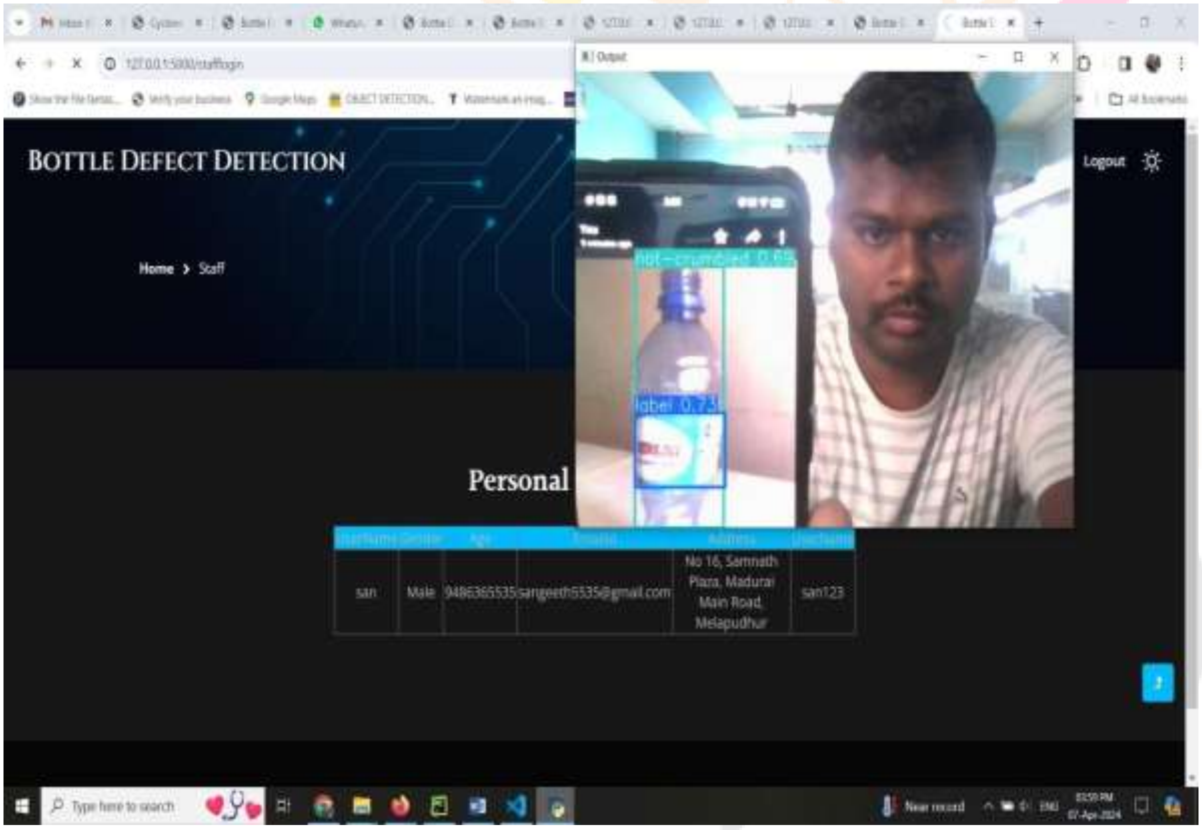


BOTTLE DEFECT DETECTION

Personal Information

StaffName	Gender	Age	EmailId	Address	UserName
san	Male	9486365535	sangeeth5535@gmail.com	No 16, Samnath Plaza, Madurai Main Road, Melapudhur	san123

Fig.9. PERSONAL INFORMATION

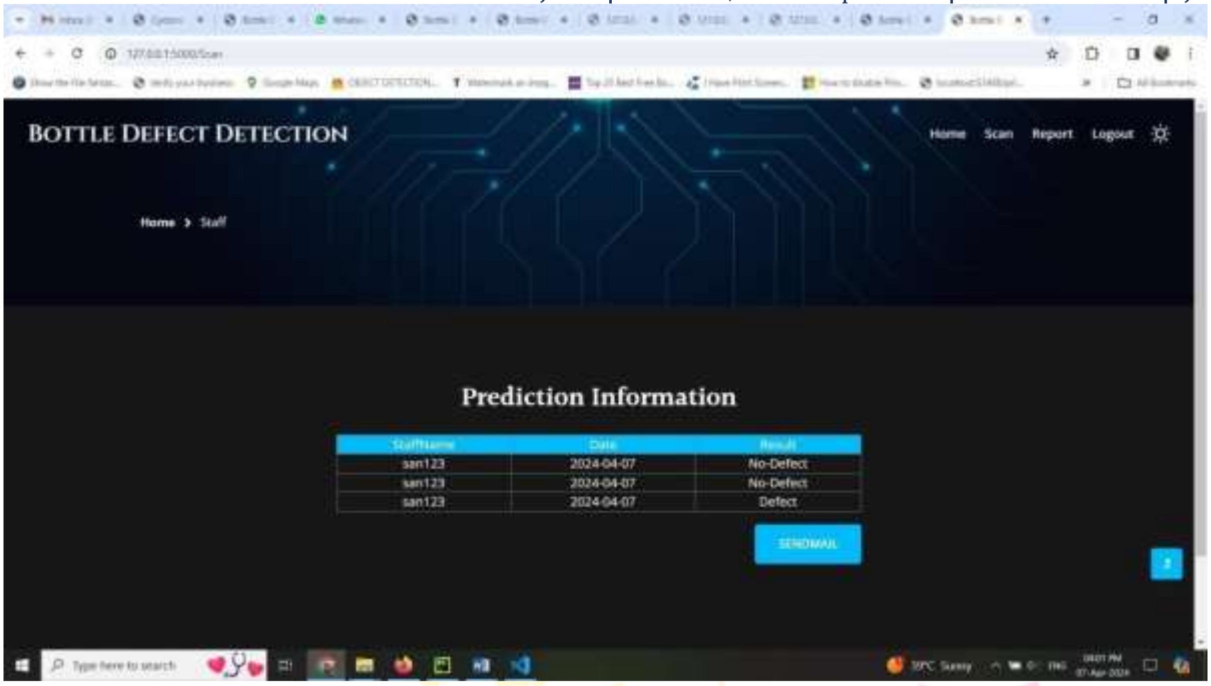


BOTTLE DEFECT DETECTION

Personal

StaffName	Gender	Age	EmailId	Address	UserName
san	Male	9486365535	sangeeth5535@gmail.com	No 16, Samnath Plaza, Madurai Main Road, Melapudhur	san123

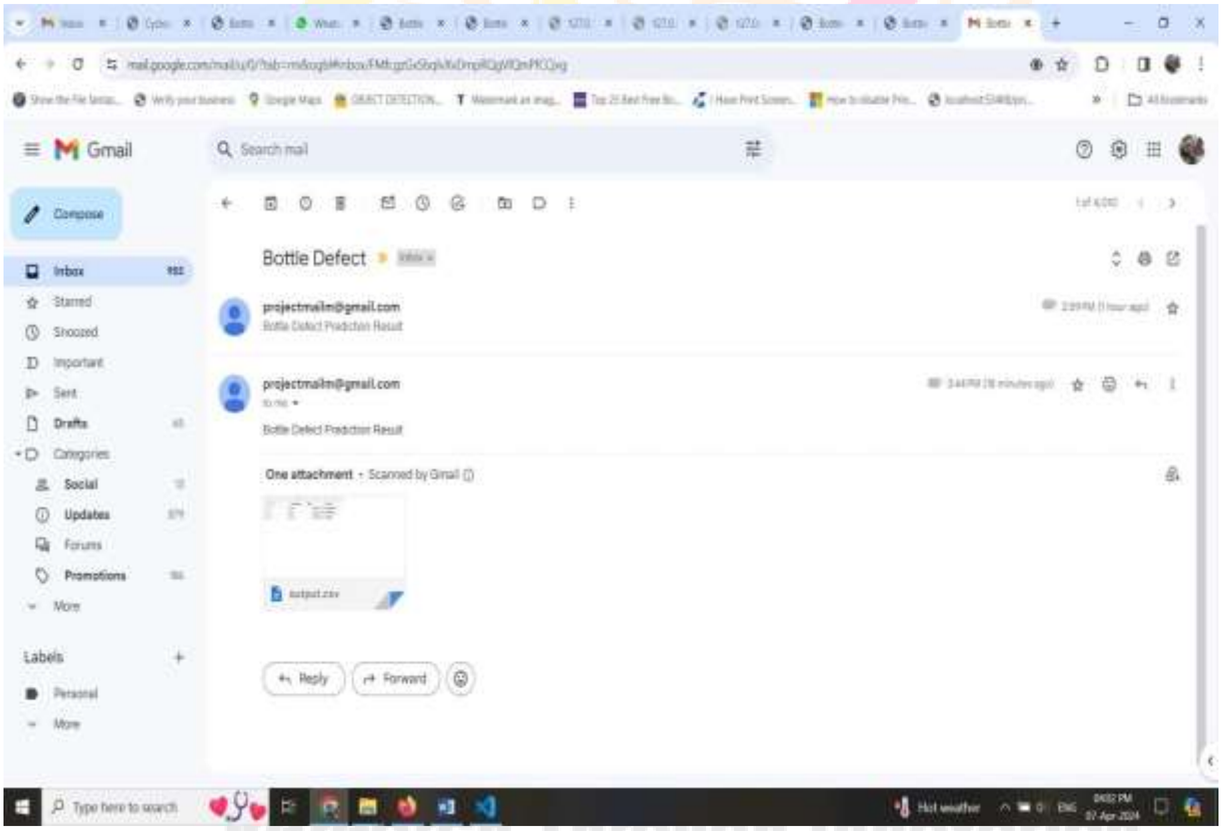
Fig.10. BOTTLE SCANNING STATUS



Prediction Information

StaffName	Date	Result
san123	2024-04-07	No-Defect
san123	2024-04-07	No-Defect
san123	2024-04-07	Defect

Fig.11. REPORT GENERATION



Bottle Defect

projectmailm@gmail.com
Bottle Defect Prediction Result

projectmailm@gmail.com
to me
Bottle Defect Prediction Result

One attachment • Scanned by Gmail
output.csv

Fig.12. MAIL ALERT

REFERENCES

[1] An, Kang, and Yanping Zhang. "LPViT: a transformer-based model for PCB image classification and defect detection." IEEE Access 10 (2022): 42542- 42553.

[2] A. C. Bovik, ‘‘Basic gray level image processing,’’ in The Essential Guide to Image Processing. Amsterdam, The Netherlands: Elsevier, 2009, pp. 43–68.

[3] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, ''MobileNets: Efficient convolutional neural networks for mobile vision applications,'' 2017, arXiv:1704.04861.

[4] B. Zoph and Q. V. Le, ''Neural architecture search with reinforcement learning,'' 2017, arXiv:1611.01578. [37] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, ''Learning transferable architectures for scalable image recognition,'' in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 8697–8710.

[5] C. Solomon and T. Breckon, Fundamentals of Digital Image Processing: A Practical Approach With Examples in MATLAB, 1st ed. Hoboken, NJ, USA: Wiley, 2010.

[6] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, ''Going deeper with convolutions,'' in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2015, pp. 1–9.

[7] He, Chenghu, et al. "Research on defect detection of the outer side of bottle cap based on high angle and multi-view vision system." IEEE Access (2023).

[8] Hohlfeld, Philipp, Tobias Ostermeier, and Dominik Brandl. "Classification of beer bottles using object detection and transfer learning." arXiv preprint arXiv:2201.03791 (2022).

[9] Kazmi, Majida, et al. "Machine-vision-based plastic bottle inspection for quality assurance." Engineering Proceedings 20.1 (2022): 9.

[10] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, ''MobileNetV2: Inverted residuals and linear bottlenecks,'' in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 4510–4520.

[11] M. Tan and Q. V. Le, ''EfficientNet: Rethinking model scaling for convolutional neural networks,'' 2019, arXiv:1905.11946.

[12] Q. Liang, W. Zhu, W. Sun, Z. Yu, Y. Wang, and D. Zhang, ''In-line inspection solution for codes on complex backgrounds for the plastic container industry,'' Measurement, vol. 148, Dec. 2019, Art. no. 106965.

International Research Journal
IJNRD
Research Through Innovation

[13] Roslan, Muhammad Izzat Bin, Zaidah Ibrahim, and Zalilah Abd Aziz. "Real -Time Plastic Surface Defect Detection Using Deep Learning." 2022 IEEE 12th Symposium on Computer Applications & Industrial Electronics (ISCAIE). IEEE, 2022.

[14] Saberironaghi, Alireza, Jing Ren, and Moustafa El-Gindy. "Defect detection methods for industrial products using deep learning techniques: A review." Algorithms 16.2 (2023): 95.

[15] Sheng, Zhenwen, and Guiyun Wang. "Fast Method of Detecting Packaging Bottle Defects Based on ECA-EfficientDet." Journal of Sensors 2022 (2022).

[16] Tulbure, Andrei-Alexandru, Adrian-Alexandru Tulbure, and Eva-Henrietta Dulf. "A review on modern defect detection models using DCNNs–Deep convolutional neural networks." Journal of Advanced Research 35 (2022): 33-48.

[17] Wang, Jia, et al. "Machine Vision-based Integrated Inspection System for Beverage Bottles with Multiple Defects." Journal of Physics: Conference Series. Vol. 2577. No. 1. IOP Publishing, 2023.

[18] Tatu, Sapana Indarchand, et al. "Fault Detection In Bottle Caps And Label Alignment Using Convolutional Neural Network." 2023 Third International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT). IEEE, 2023.

[19] Z. Sheng and G. Wang, ‘‘Fast method of detecting packaging bottle defects based on ECA-EfficientDet,’’ J. Sensors, vol. 2022, pp. 1–9, Feb. 2022.

[20] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, ‘‘A survey of convolutional neural networks: Analysis, applications, and prospects,’’ IEEE Trans. Neural Netw. Learn. Syst., vol. 33, no. 12, pp. 6999–7019, Dec. 2022.

International Research Journal
IJNRD
Research Through Innovation