VISUAL INSPECTION FOR INDUSTRIES

A PROJECT REPORT

Submitted to University of Mumbai in partial fulfillment
of the requirement for the degree of

BACHELOR OF ENGINEERING

In

COMPUTER ENGINEERING

Submitted By

CHINMAY GAIKAWAD
SUYASH SALVI
PRATIK MAURYA

Under the guidance of

Prof. PRAJAKTA KHELKAR

DEPARTMENT OF COMPUTER ENGINEERING

VASANTDADA PATIL PRATISHTHAN'S COLLEGE OF ENGINEERING & VISUAL ARTS, SION, MUMBAI-400022

UNIVERSITY OF MUMBAI 2021-2022

VISUAL INSPECTION FOR INDUSTRIES

A PROJECT REPORT

Submitted to University of Mumbai in partial fulfillment
of the requirement for the degree of

BACHELOR OF ENGINEERING

In

COMPUTER ENGINEERING

Submitted By

CHINMAY GAIKAWAD
SUYASH SALVI
PRATIK MAURYA

Under the guidance of

Prof. PRAJAKTA KHELKAR

DEPARTMENT OF
COMPUTER ENGINEERING

VASANTDADA PATIL PRATISHTHAN'S COLLEGE OF ENGINEERING
& VISUAL ARTS, SION, MUMBAI- 400022

UNIVERSITY OF MUMBAI 2021-2022

VASANTDADA PATIL PRATISHTHAN'S COLLEGE OF ENGINEERING & VISUAL ARTS SION,MUMBAI-400022

CERTIFICATE

This is to certify that the project work, entitled **VISUAL INSPECTION FOR INDUSTRIES** has completed successfully for the award of the Bachelor of Engineering in Computer Engineering by:

CHINMAY GAIKAWAD
SUYASH SALVI
PRATIK MAURYA

Prof. PRAJAKTA KHELKAR
Project Guide
Department of Computer Engineering

VASANTDADA PATIL PRATISHTHAN'S COLLEGE OF ENGINEERING & VISUAL ARTS SION,MUMBAI-400022

PROJECT APPROVAL CERTIFICATE

This is to certify that the project work, entitled **VISUAL INSPECTION FOR INDUSTRIES** has completed successfully for the award of the Bachelor of Engineering in Computer Engineering by:

	Group Members:	Mob No.
1	Chinmay Gaikawad	9422820783
2	Suyash Salvi	9969300248
3	Pratik Maurya	9137796515

Internal Examiner

Name _____

Date _____

External Examiner

Name _____

Date _____

Project Convener

HOD (COMPUTER)

Principal (PVPPCOE)

ABSTRACT

In product-based industries the quality of the final products dictates its value in the eyes of customers. Nobody wants a subpar product, and ever since the Industrial revolution each and every industry wants to make the best product with the best margin. And maintaining the quality of every single product, where hundreds if not thousands of products are being built, packaged every minute even with 1% fault rate. At the end of the day, it makes a good chunk of shipped product will contain low quality product if not managed properly.

To deal with maintaining the quality of the product to be shipped, Quality Assurance (QA) of the products is done. Quality assurance is a multistage process which begins with checking the raw products and ends with the final product. And maintaining the quality of product is very expensive and intensive task, which require constant attention to each and every stage of production.

In order to overcome the difficulties, the bottlenecks in the Quality Control, many big industries have started to include robots and machines to maintain the quality and reduces the errors. But these technologies are very expensive and hard to maintain since they're different for each industry. Hence there is a very high requirement for the affordable and high-quality systems for QA. Our project is to provide the small industries who don't have such high budget nor the connection to get high-quality QA tech. In our system, we are using Machine Learning and High-Quality cameras to inspect the products that are being built by providing constant surveillance. The pros of our system is that it requires minimal human input while still maintaining high accuracy while being cheap and easy to maintain since the system is scalable and easily changeable.

ACKNOWLEGEMENT

The project report on Visual Inspection for Industries is the outcome of the guidance, moral support and devotion bestowed on our group throughout our work. For this we acknowledge and express our profound sense of gratitude to everybody who has been the source of inspiration throughout project preparation. First and foremost, we offer our sincere phrases of thanks and innate humility to **Mrs. Prajakta Khelkar, Assistant Professor, Computer Department, VPPCOE&VA** and guide of our project for providing help whenever required.

We also would like to express our deepest gratitude to **Dr. M. A. Devmane, HOD, Computer Department, VPPCOE & VA** for providing the valuable inputs. The consistent guidance and support provided by **Dr. Alam Shaikh, Principal, VPPCOE & VA** is very thankfully acknowledged for key role played by him in providing us with precious ideas and suggestions and help that enabled in shaping the project work.

We can say in words that we must at outset tender our intimacy for receipt of affectionate care to **VPPCOE & VA** for providing such a stimulating atmosphere and conducive work environment.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHPATER 1

INTRODUCTION

1.1 Introduction

Over the past two decades, automation in manufacturing has been transforming factory floors, the nature of manufacturing employment, and the economics of many manufacturing sectors. Today, we are on the cusp of a new automation era: rapid advances in robotics, artificial intelligence, and machine learning are enabling machines to match or outperform humans in a range of work activities, including ones requiring cognitive capabilities. Industry executives those whose companies have already embraced automation, those who are just getting started, and those who have not yet begun fully reckoning with the implications of this new automation age need to consider the following three fundamental perspectives: what automation is making possible with current technology and is likely to make possible as the technology continues to evolve; what factors besides technical feasibility to consider when making decisions about automation; and how to begin thinking about where and how much to automate in order to best capture value from automation over the long term.

1.2 Aim and Objectives

The main objectives of the development of our project -VISUAL INSPECTION FOR INDUSTRIES are following approaches:

- To use Machine Learning for Visual Inspection of assembly line
- Make an accurate model for standard product

1.3 Motivation

- The visual supervision systems available by the Indian manufacturers are priced at around 60K - 5 lakhs.
- This makes a large investment for small-scale industry. Moreover, the systems are not dynamic if at all the industry wants to scale its production.
- There is a need for a cheaper yet accurate visual supervision system that would look after one of the most important aspects of the product i.e. QUALITY ASSURANCE!!!

1.4 Scope

The usage of this system will help the assembly line to inspect products whether they are defective products or perfect products. This system will constantly monitoring important parameters of the product.

Our project can be used in various small scale as we as large scale industries, such as food, health care, airline, automotive, computer equipment manufacturing, textile, solar panel etc.

1.5 Contribution

The contributions to this project are the tensorflow library and a high speed camera to detect objects moving at high speed. Tensorflow enabled us to make the models and implement the software whereas a camera plays the same role for hardware. The mobilenetv2 model helped in making a light weight software for the minimum software requirements. Jupyter notebook helped with documenting the errors and implementations

1.6 Organization of the report

The organization of the report is as follows: Chapter 2 presents the Literature Survey and a brief insight into related work. Chapter 3 is a technical chapter in which we have briefly discussed about the Design and Implementation. Chapter 4 is the Result and Discussions chapter finally in chapter 5 we present the conclusions.

CHAPTER 2

LITERATURE SURVEY

2.1 Introduction

To understand the scope of possible automation in the manufacturing sector as a whole, we conducted a study of manufacturing work in 46 countries in both the developed and developing worlds, covering about 80 percent of the global workforce. Our data and analysis show that as of 2015, 478 billion of the 749 billion working hours (64 percent) spent on manufacturing-related activities globally were automatable with currently demonstrated technology.1 These 478 billion working hours represent the labour equivalent of 236 million out of 372 million full-time employees $2.7 trillion out of $5.1 trillion of labour that could be eliminated or repurposed, assuming that demonstrated technologies are adapted for use in individual cases and then adopted. These figures suggest that, even though manufacturing is one of the most highly automated industries globally, there is still significant automation potential within the four walls of manufacturing sites, as well as in related functional areas such as supply chain and procurement.

2.2 Existing System

In the current world existing system, the solutions to have automated QA System is either proprietary solutions by big MNCs or there are generally very expensive. To name a few:

Table 2.1: Existing System on Visual Inspection

Infinity Automation Systems PVT LTD	5,00,000 INR
Rabro Systems	2,00,000 INR
Analte Control	3,00,000 INR
Nexgen Robotic Automation PVT LTD	4,00,000 INR
Radhe Krishna Technology	3,00,000 INR
Mayura Automation & Robotic Systems PVT. LTD	60,000 INR

The existing system are either very expensive or they are not dynamic and scalable and aren't fully automated. Being static system makes it harder for the growing industries for scaling their QA solutions. Cause the system need to be re-evaluated for the new production line.

2.3 Need of New System

The existing systems available in the market are quite expensive for an entry level business. Moreover, the scope for scaling the same system for increasing production in difficult. As the systems are concealed with no scope for up gradation. There was a need of a system that would be easily upgradable and also the entry cost should be less. The system shouldn't compromise on quality.

It is observed that there is increase in the number of industries that need mass production and to inspect that visual detection systems can be more efficient than anything else.

This project is an effort for achieving the same.

2.4 Problems Definition

Problem: The expensive costs of visual inspection systems used in assembly lines.

Also, the places where manual inspection is still carried out, the accuracy is not guaranteed. There is danger involved while working at risky places. The problem is need to diminish it by replacing manual humans to machines.

Problem Statement: Making systems extremely affordable and scalable according to the needs of the industry.

Providing the same quality of detection at high speeds. The system needs to be feed suitable data so as to differentiate between the perfect and faulty product and point out the faulty one.

CHAPTER 3

DESIGN AND IMPLEMENTATION

3.1 Proposed system

The Visual inspection System is expected to be trained for a specific data. The specific product is classified into the perfect and the defective product. As the product/ object passes through the camera coverage in the assembly line, the system should alert about the defects in the product.

Also the system should be able to cope with the increasing/decreasing speed of the production line.

The system should be modifiable according to various products.

3.2 Requirement Gathering and Analysis

We have come up with a low-cost, visual inspection System, which requires the following hardware and software gathering:

Hardware Requirement

Speed motion camera:

Figure 3.1: Speed motion camera

A high-speed camera is a device capable of capturing moving images with exposures of less than 1/1,000 second or frame rates in excess of 250 fps. It is used for recording fast-moving objects as photographic images onto a storage medium. After recording, the images stored on the medium can be played back in slow motion. Early high-speed

cameras used film to record the high-speed events, but were superseded by entirely electronic devices using either a charge-coupled device (CCD) or a CMOS active pixel sensor, recording, typically, over 1,000 fps onto DRAM, to be played back slowly to study the motion for scientific study of transient phenomena.

Raspberry pi:

Figure 3.2: Raspberry pi

The Raspberry Pi is a very cheap computer that runs Linux, but it also provides a set of GPIO (general purpose input/output) pins, allowing you to control electronic components for physical computing and explore the Internet of Things (IoT).

Softaware Requirement

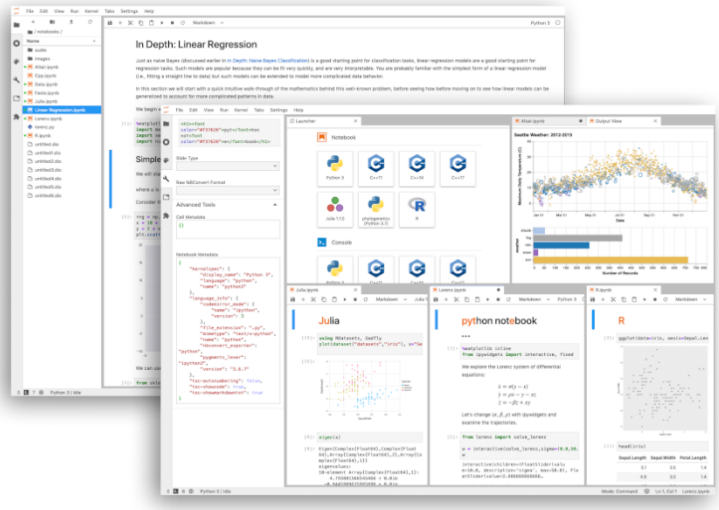
Python:

Figure 3.3: Python

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small- and large-scale projects.

Jupyter Notebook:

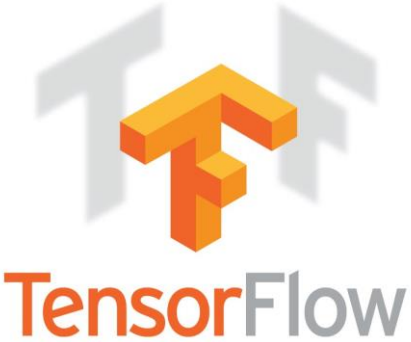
Figure 3.4: Jupyter Notebook



JupyterLab is the latest web-based interactive development environment for notebooks, code, and data. Its flexible interface allows users to configure and arrange workflows in data science, scientific computing, computational journalism, and machine learning. A modular design invites extensions to expand and enrich functionality.

Tensorflow:

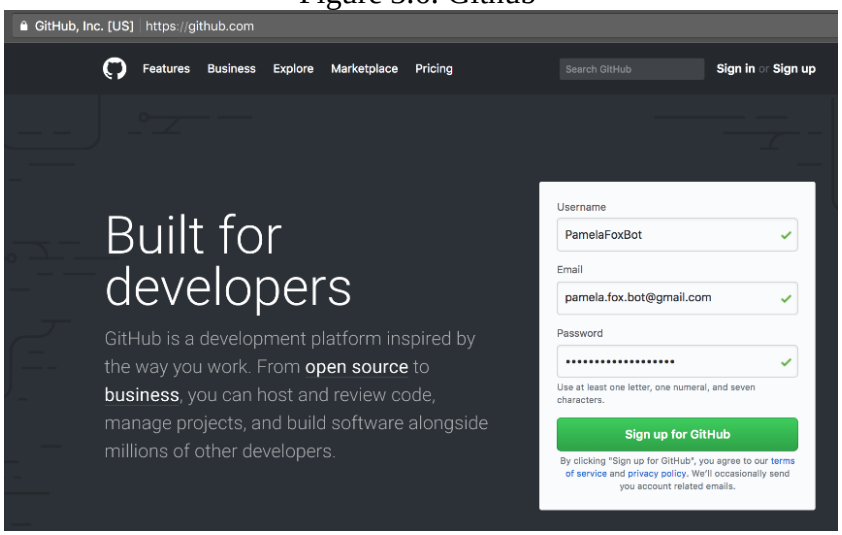
Figure 3.5: Tensorflow



TensorFlow is a free and open-source software library for machine learning and artificial intelligence. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks.

Github:

Figure 3.6: Github



GitHub, Inc. is a provider of Internet hosting for software development and version control using Git. It offers the distributed version control and source code management functionality of Git, plus its own features.

3.3 Design

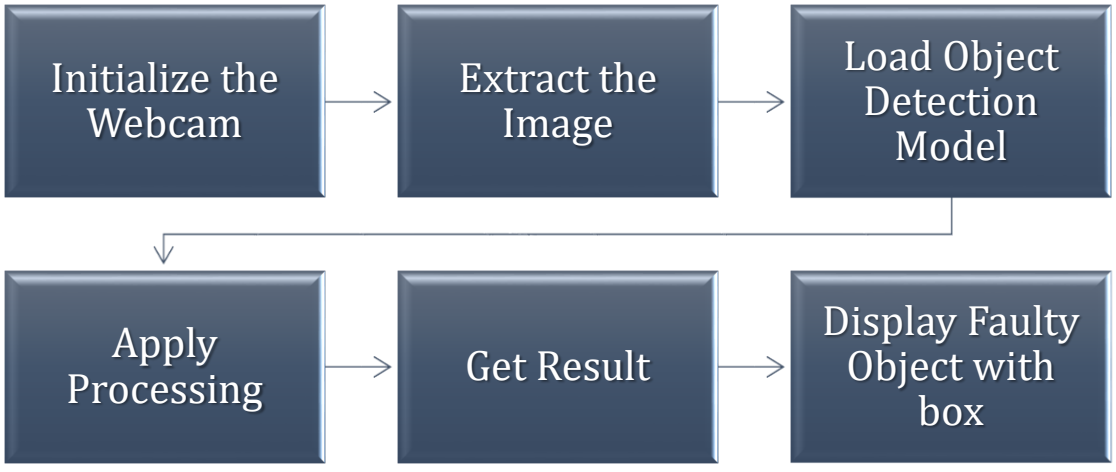
Identifying Perfect and faulty product.

Figure 3.7: Perfect Item

Figure 3.8: Faulty Item

3.3.1 UML DIAGRAM

Figure 3.9: UML Diagram



3.4 Algorithm

Step 1: Start

Step 2: Gather data for training

Step 3: Split data into two parts A and B. One for training and other for testing

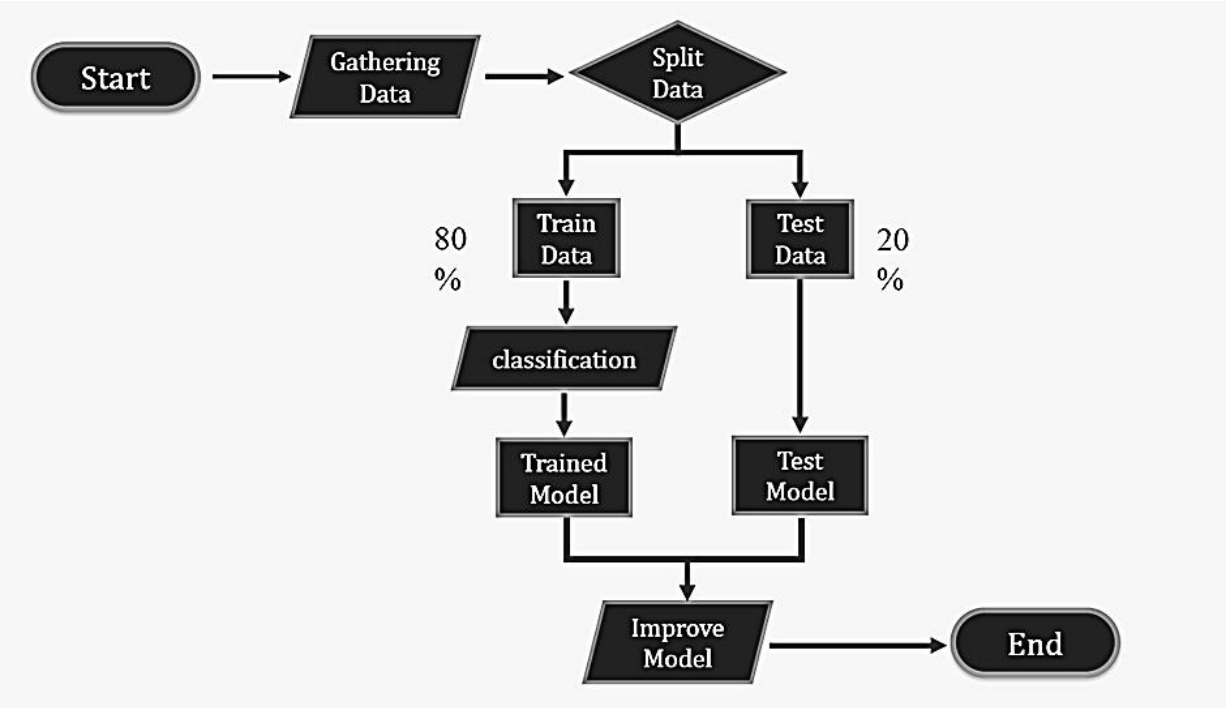
Step 4: Train model from the data

Step 5: Test the model from the data

Step 6: Re-iterate training for required result

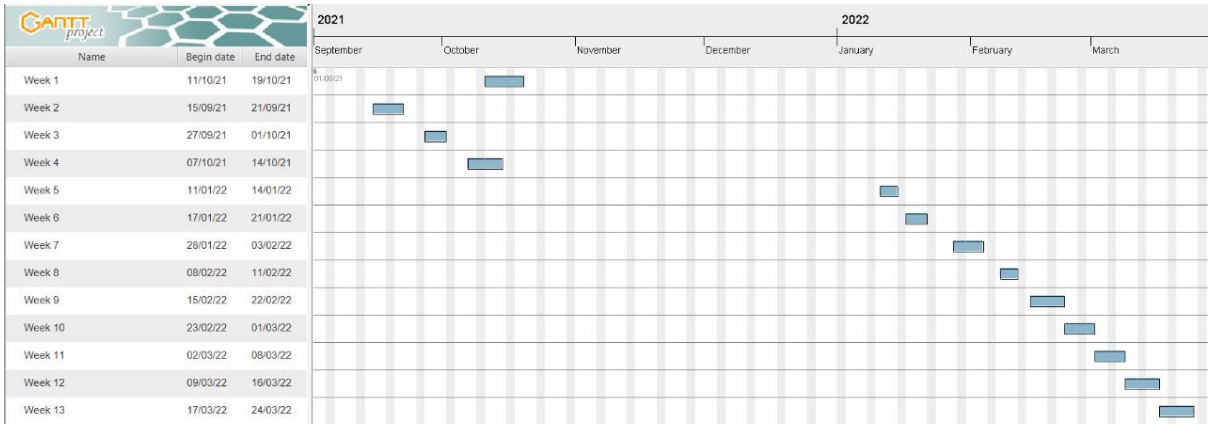
Step 7: End

Figure 3.10: Flow Chart



3.5 Timeline Chart

Figure 3.10: TimeLine Chart



3.6 Cost Estimation

Cost for deploying the system:

Table 3.1: Cost Estimation of the Project

Camera(8bit):	2000 INR
Installation kit (like connecting wires etc.):	1500 INR
Interaction System:	1500 INR

3.7 Feasibility Study

The already existing solutions to the Visual Inspection which exist are very expensive, and difficult to maintain. Hence small scale industries are very unlikely to deploy such solution for QC, but every industry wants to be competitive in market in order to not lose the customers. That's where our solution comes in, it promises high quality QA works at quarter the prices of any other average solution present.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Program

Training Program:

```
# import the necessary packages
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.layers import AveragePooling2D
from tensorflow.keras.layers import Dropout
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Input
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.preprocessing.image import load_img
from tensorflow.keras.utils import to_categorical
from sklearn.preprocessing import LabelBinarizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from imutils import paths
import matplotlib.pyplot as plt
import numpy as np
import os
import PIL
from PIL import Image
```

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from tensorflow.keras import Model, Input, regularizers
from tensorflow.keras.layers import Dense, Conv2D, MaxPool2D, UpSampling2D
from tensorflow.keras.callbacks import EarlyStopping
from keras.preprocessing import image
import glob
from PIL import Image
import warnings;
warnings.filterwarnings('ignore')
# initialize the initial learning rate, number of epochs to train for,
# and batch size
INIT_LR = 1e-4
EPOCHS = 20
BS = 32

DIRECTORY = r"D:\Project\Visual-Inspection\dataset"
CATEGORIES = ["faulty_item", "perfect_item"]
# grab the list of images in our dataset directory, then initialize
# the list of data (i.e., images) and class images
print("[INFO] loading images...")

#2 empty lists
data = []#append all the image arrays
labels = []#append corresponding images with/without masks

for category in CATEGORIES:
    #getting the path
    path = os.path.join(DIRECTORY, category)
    #looping througha specific category
    for img in os.listdir(path):
```

```
        #joining path with the corresponding image
        img_path = os.path.join(path, img)
        #convert all the images to the equal size for better training(keras.preprocessing)
        image = load_img(img_path, target_size=(224, 224))
        #convert image to array
        image = img_to_array(image)
        #preprocess input image
        image = preprocess_input(image)

        #append image to the data list
        data.append(image)
        #append corresponding label to the label list
        labels.append(category)
# perform one-hot encoding on the labels
lb = LabelBinarizer()
labels = lb.fit_transform(labels)
labels = to_categorical(labels)

data = np.array(data, dtype="float32")
labels = np.array(labels)

(trainX, testX, trainY, testY) = train_test_split(data, labels,
        test_size=0.20, stratify=labels, random_state=42)
# construct the training image generator for data augmentation
aug =
ImageDataGenerator(rotation_range=20,zoom_range=0.15,width_shift_range=0.2,height
_shift_range=0.2,shear_range=0.15,horizontal_flip=True,fill_mode="nearest")

# load the MobileNetV2 network, ensuring the head FC layer sets are
# left off
baseModel = MobileNetV2(weights="imagenet", include_top=False,
        input_tensor=Input(shape=(224, 224, 3)))
```

```
# construct the head of the model that will be placed on top of the
# the base model
headModel = baseModel.output
headModel = AveragePooling2D(pool_size=(7, 7))(headModel)
headModel = Flatten(name="flatten")(headModel)
headModel = Dense(128, activation="relu")(headModel)
headModel = Dropout(0.5)(headModel)
headModel = Dense(2, activation="softmax")(headModel)
# place the head FC model on top of the base model (this will become
# the actual model we will train)
model = Model(inputs=baseModel.input, outputs=headModel)

# loop over all layers in the base model and freeze them so they will
# *not* be updated during the first training process
for layer in baseModel.layers:
	layer.trainable = False
# compile our model
print("[INFO] compiling model...")
opt = Adam(lr=INIT_LR, decay=INIT_LR / EPOCHS)
model.compile(loss="binary_crossentropy", optimizer=opt,
	metrics=["accuracy"])
# train the head of the network
print("[INFO] training head...")
H = model.fit(aug.flow(trainX, trainY, batch_size=BS),steps_per_epoch=len(trainX) //
BS,validation_data=(testX, testY),validation_steps=len(testX) // BS,epochs=EPOCHS)
# make predictions on the testing set
print("[INFO] evaluating network...")
predIdxs = model.predict(testX, batch_size=BS)
# for each image in the testing set we need to find the index of the
# label with corresponding largest predicted probability
predIdxs = np.argmax(predIdxs, axis=1)
# show a nicely formatted classification report
print(classification_report(testY.argmax(axis=1), predIdxs,
```

```
        target_names=lb.classes_))
# serialize the model to disk
print("[INFO] saving object detector model...")
model.save("object_detector.model", save_format="h5")
# plot the training loss and accuracy
N = EPOCHS
plt.style.use("ggplot")
plt.figure()
plt.plot(np.arange(0, N), H.history["loss"], label="train_loss")
plt.plot(np.arange(0, N), H.history["val_loss"], label="val_loss")
plt.plot(np.arange(0, N), H.history["accuracy"], label="train_acc")
plt.plot(np.arange(0, N), H.history["val_accuracy"], label="val_acc")
plt.title("Training Loss and Accuracy")
plt.xlabel("Epoch #")
plt.ylabel("Loss/Accuracy")
plt.legend(loc="lower left")
plt.savefig("plot.png")
```

Detection Program:

```
# import the necessary packages
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.models import load_model
from imutils.video import VideoStream
import numpy as np
import imutils
import time
import cv2
```

```
import os

def detect_and_predict_object(frame, faceNet, maskNet):
	# grab the dimensions of the frame and then construct a blob
	# from it
	(h, w) = frame.shape[:2]
	blob = cv2.dnn.blobFromImage(frame, 1.0, (224, 224),
		(104.0, 177.0, 123.0))

	# pass the blob through the network and obtain the face detections
	faceNet.setInput(blob)
	detections = faceNet.forward()
	print(detections.shape)

	# initialize our list of faces, their corresponding locations,
	# and the list of predictions from our face mask network
	faces = []
	locs = []
	preds = []

	# loop over the detections
	for i in range(0, detections.shape[2]):
		# extract the confidence (i.e., probability) associated with
		# the detection
		confidence = detections[0, 0, i, 2]

		# filter out weak detections by ensuring the confidence is
		# greater than the minimum confidence
		if confidence > 0.5:
			# compute the (x, y)-coordinates of the bounding box for
			# the object
			box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
			(startX, startY, endX, endY) = box.astype("int")
```

```
                # ensure the bounding boxes fall within the dimensions of
                # the frame
                (startX, startY) = (max(0, startX), max(0, startY))
                (endX, endY) = (min(w - 1, endX), min(h - 1, endY))

                # extract the face ROI, convert it from BGR to RGB channel
                # ordering, resize it to 224x224, and preprocess it
                face = frame[startY:endY, startX:endX]
                face = cv2.cvtColor(face, cv2.COLOR_BGR2RGB)
                face = cv2.resize(face, (224, 224))
                face = img_to_array(face)
                face = preprocess_input(face)

                # add the face and bounding boxes to their respective
                # lists
                faces.append(face)
                locs.append((startX, startY, endX, endY))

        # only make a predictions if at least one face was detected
        if len(faces) > 0:
                # for faster inference we'll make batch predictions on *all*
                # faces at the same time rather than one-by-one predictions
                # in the above `for` loop
                faces = np.array(faces, dtype="float32")
                preds = maskNet.predict(faces, batch_size=32)

        # return a 2-tuple of the face locations and their corresponding
        # locations
        return (locs, preds)
# load our serialized face detector model from disk
prototxtPath = r"object_detector\SSD_MobileNet.prototxt"
weightsPath = r"object_detector\SSD_MobileNet.caffemodel"
```

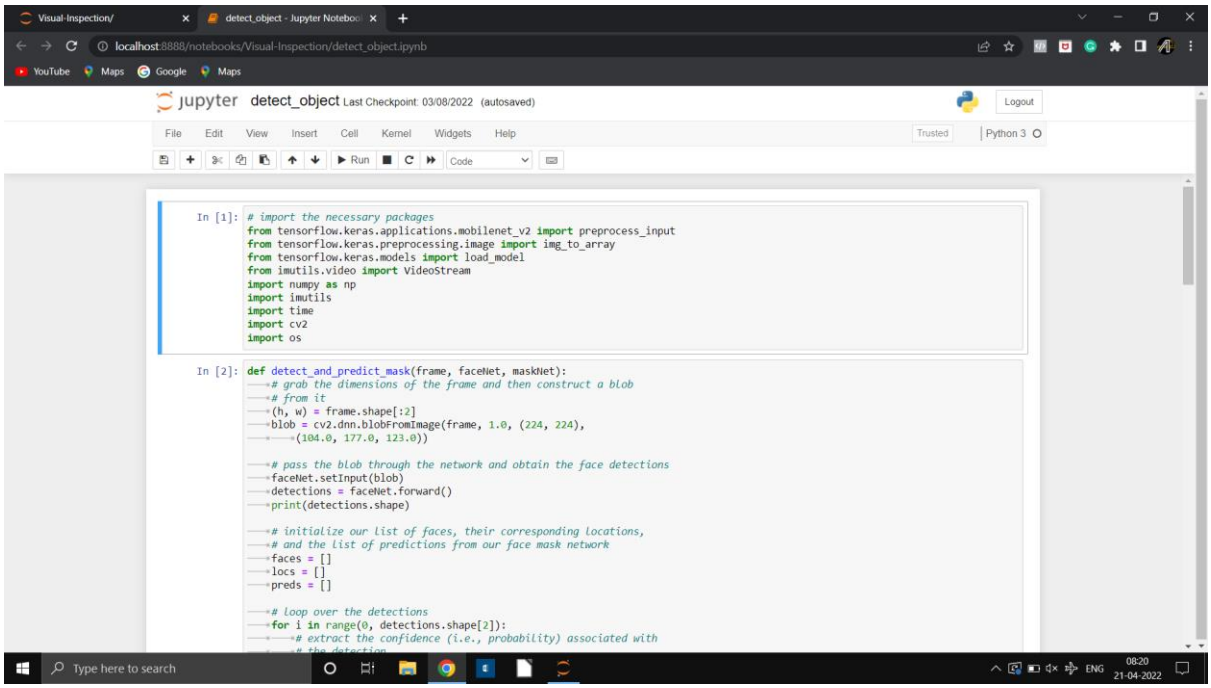
```
faceNet = cv2.dnn.readNet(prototxtPath, weightsPath)

# load the object detector model from disk
maskNet = load_model("object_detector.model")

# initialize the video stream
print("[INFO] starting video stream...")
vs = VideoStream(src=0).start()

# loop over the frames from the video stream
while True:
	# grab the frame from the threaded video stream and resize it
	# to have a maximum width of 400 pixels
	frame = vs.read()
	frame = imutils.resize(frame, width=400)

	(locs, preds) = detect_and_predict_mask(frame, faceNet, maskNet)

	# loop over the detected face locations and their corresponding
	# locations
	for (box, pred) in zip(locs, preds):
		# unpack the bounding box and predictions
		(startX, startY, endX, endY) = box
		(defected, perfect) = pred

		# determine the class label and color we'll use to draw
		# the bounding box and text
		label = "defected" if defected > perfect else "perfect"
		color = (0, 0, 255) if label == "defected" else (0, 255, 0)

		# include the probability in the label
		label = "{}: {:.2f}%".format(label, max(defected, perfect) * 100)
```

```
                # display the label and bounding box rectangle on the output
                # frame
                cv2.putText(frame, label, (startX, startY - 10),
                        cv2.FONT_HERSHEY_SIMPLEX, 0.45, color, 2)
                cv2.rectangle(frame, (startX, startY), (endX, endY), color, 2)

        # show the output frame
        cv2.imshow("Frame", frame)
        key = cv2.waitKey(1) & 0xFF

        # if the `q` key was pressed, break from the loop
        if key == ord("q"):
                break

# do a bit of cleanup
cv2.destroyAllWindows()
vs.stop()
```

4.2 Software Results

This is a picture of the software used that enables us for proper implementation of the project.

Figure 4.1: Jupyter Notebook



JupyterLab is the latest web-based interactive development environment for notebooks, code, and data. Its flexible interface allows users to configure and arrange workflows in data science, scientific computing, computational journalism, and machine learning. A modular design invites extensions to expand and enrich functionality.

The notebook helped document every part of implementation and easy execution.
All the parts could be implemented in a single notebook.

4.3 Screenshots

Here are a pictures of the result window that appears when we run the object detection program.

This window requires camera support to work and it is able to recognise good biscuits from bad.

This window extracts the image from the real time and then runs classification model over it.

Figure 4.2: Detection of Faulty Object

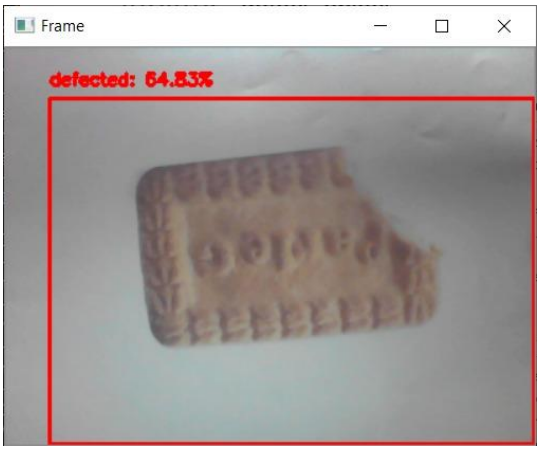
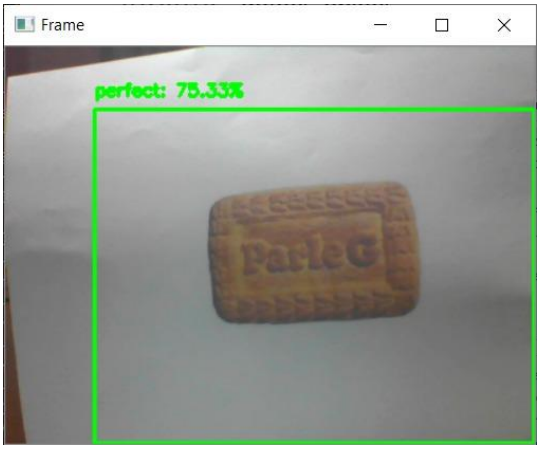


Figure 4.3: Detection of Perfect Object



4.4 Testing Results

Table 4.1: Testing Results

Id	Description	Test Steps	Test Type	Expected Output	Actual Output
T-101	Checking whether the model working correctly for perfect Product	Perfect Biscuit	Unit testing	Green colored box around the biscuit with labelled as Perfect	Green box visible and perfect label
T-102	Checking whether the model working correctly for Faulty Product	Broken Biscuits	Unit testing	Red colored box around the biscuit with labelled as Defected	Red box visible and Defected label

CHAPTER 5

CONCLUSION

5.1 Summary

The Data has been gathered, processed and trained using python as the programming language. The model is accurate to a good extent and is okay to go ahead with. The concepts like Open CV, Scikit learn, matplot lib and other supporting libraries are used for the purpose.

5.2 Future Scope

This project can be further enhanced to provide greater flexibility and performance with certain modification whenever necessary.
This system is expected to work with any standard product. This system should also be able to be delivered at a minimum cost. Also this system must be capable of handling large scale production.

CHAPTER 6

REFERENCES

6.1 Reference

[1] Serhii Maksymenko. The Value of AI-Based Visual Inspection in 2020 [J]. readwrite, July 2020.

[2] Jamshed Khan. Everything you need to know about Visual Inspection with AI [J]. NanoNets, May 2021.

[3] The use of Visual Inspection in Asset Management and Quality Control [B]. LimbleCMMS May 2021.

[4] Naveenkumar Mahamkali, Vadivel Ayyasamy. OpenCV for Computer Vision Applications [RJ]. ResearchGate, March 2015.

[5] Wes McKinney. Python For Data Analyst 2nd Edition [Book]. O'Rielly Publications, October 2017.

[6] Understanding Of Convolution Neuran Network by Saad Albawi, Saad Al-Zawi

[7] Image Based Surface Defect Detection By Prahar Bhatt, Rishi Malhan, Brual Shah

[8] Object Tracking And Failure Recovery by Saravanakumar Soman

[9] Machine Learning In Computer Vision by Ashraf I. Khan, Salim Al-Habsib

[10] Optimal Speed And Accuracy Of Object Detection by Alexey Bochkovsiky, Chien-Yao Wang, Hong-Yuan Liao

PUBLICATIONS

in IJARSCT, Volume 2, Issue 2, March 2022

International Journal of Advanced Research in Science, Communication and Technology

DOI: 10.48175/IJARSCT-2827


INTERNATIONAL JOURNAL OF ADVANCED RESEARCH IN SCIENCE, COMMUNICATION AND TECHNOLOGY
IJARSCT
CERTIFICATE OF PUBLICATION
INTERNATIONAL STANDARD SERIAL NUMBER ISSN NO: 2581-9429
THIS IS TO CERTIFY THAT
Chinmay Gaikawad
Vasantdada Patil Pratishthan's College of Engineering, Mumbai, Maharashtra, India
HAS PUBLISHED A RESEARCH PAPER ENTITLED
Visual Inspection for Industries
IN IJARSCT, VOLUME 2, ISSUE 2, MARCH 2022
Certificate No: 032022-A263
www.ijarsct.co.in
Crossref
DOI: 10.48175/IJARSCT-2827
www.doi.org
www.crossref.org
6.252
www.sjifactor.com
Editor-in-Chief


INTERNATIONAL JOURNAL OF ADVANCED RESEARCH IN SCIENCE, COMMUNICATION AND TECHNOLOGY
IJARSCT
CERTIFICATE OF PUBLICATION
INTERNATIONAL STANDARD SERIAL NUMBER ISSN NO: 2581-9429
THIS IS TO CERTIFY THAT
Pratik Maurya
Vasantdada Patil Pratishthan's College of Engineering, Mumbai, Maharashtra, India
HAS PUBLISHED A RESEARCH PAPER ENTITLED
Visual Inspection for Industries
IN IJARSCT, VOLUME 2, ISSUE 2, MARCH 2022
Certificate No: 032022-A262
www.ijarsct.co.in
Crossref
DOI: 10.48175/IJARSCT-2827
www.doi.org
www.crossref.org
6.252
www.sjifactor.com
Editor-in-Chief

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 2, Issue 2, March 2022
Impact Factor: 6.252

Visual Inspection for Industries

Pratik Maurya[1], Chinmay Gaikawad[2], Suyash Salvi[3]
Students, Department of Computer Engineering[1,2,3]
Vasantdada Patil Pratishthan's College of Engineering, Mumbai, Maharashtra, India

Abstract: *The system is an AI-ML image recognition-based application that gives Quality Control and Quality Assurance functionality at the same time to the small-scale industries. The system continuously scans the product for any defects, anomalies, imperfections, damages, foreign substances, or any other non-standard qualities on the product. The system basically is a governing/supervising authority that keeps that fact in check that the standard procedures and standard quality has been maintained throughout the production. This system works automatically and there is little to no human input required. It is fully automatic and requires no human input. There are no chances of error that are very plausible in a human-assisted or human-moderated system. The system only requires regular maintenance from a well-trained technician and no further human input. The system is smart and gives warnings, can halt production when needed for the QA.*

Keywords: Machine learning, Image recognition, Industries, Assembly line, Image Recognition

I. Introduction

Over the past 20 years, automation in manufacturing has been transforming factory floors, the character of producing employment, and therefore the economics of the many manufacturing sectors. Today, we are on the cusp of a replacement automation era: rapid advances in robotics, AI, and machine learning are enabling machines to match or outperform humans in a range of labor activities, including ones requiring cognitive capabilities.

Industry executives those whose companies have already embraced automation, those that are just getting started, and people who haven't yet begun fully reckoning with the implications of this new automation age got to consider the subsequent three fundamental perspectives: what automation is making possible with current technology and is probably going to form possible because the technology continues to evolve; what factors besides technical feasibility to think about when making decisions about automation; and the way to start brooding about where and how much to automate so as to best capture value from automation over the long term.

The Visual Inspection systems that are available in India are expensive and are ranged from 60K-700K INR. A small scale company is unable to acquire such devices without spending huge chunk of their funds in the systems. Also after scaling up their production. There's a need for a system that will be able to handle the conveyer speed. Here are few of the companies that manufacture such systems in India. (Source: IndiaMart)

Infinity Automation Systems PVT LTD	5 lakhs
Rabro Systems	2 lakhs
Analte Control	3 lakhs
Nexgen Robotic Automation Private Limited	4 lakhs
Radheykrishna Technology	3 lakhs
Mayura Automation & Robotic Systems Pvt. Ltd	60k

(Source: IndiaMart)

II. Research Approach

2.1 Study of Algorithms for Object Detection

Various methods for image detection were being studied for the project. All of this was done in the pursuit of creating the most efficient model that'd work on lower end devices. Here are some of the algorithms that can be used for object detection:

1. Fast R-CNN
2. Faster R-CNN

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 2, Issue 2, March 2022
Impact Factor: 6.252

3. Histogram of Oriented Gradients (HOG)

4. Region-based Convolutional Neural Networks (R-CNN)
5. Region-based Fully Convolutional Network (R-FCN)
6. Single Shot Detector (SSD)
7. Spatial Pyramid Pooling (SPP-net)
8. YOLO (You Only Look Once)

We found that the best suited algorithm for our purpose is the CNN algorithm. To make it usable for mobile devices, we also used mobileNetV2.

2.2 Learning about CNN in Detail

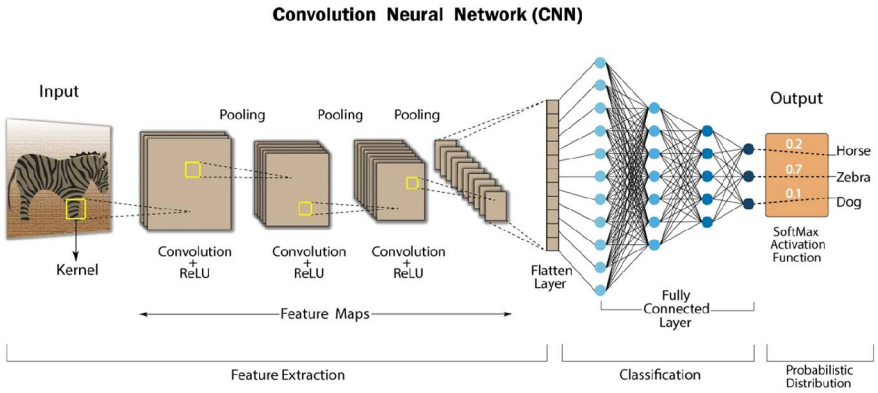


Figure: Convolutional Neural Networks (CNN)

CNN is different from other Neural Networks because of its hidden layers known as 'Convolutions'. The communications done by these convolutions are called as Convolution Operation. The convolutions try to imitate human neurons are responsible for recognising patterns, shapes and colors in an image recognition system.

III. METHODS

For the working visual inspection system, one has to choose a product. Any standard product can be chosen for the purpose. These are the steps taken further are as follows:

A. Collection of Data and taking Inference from the Trained Model

The data was collected using a camera. The photos were cleaned and made consistent for the algorithm to make a model out of it. The photos were classified into two distinct classes for the model to interpret on its own. A standard product was chosen as the subject. The subject was captured in its perfect form and the damaged form in order to make the data ready for model making.

B. Trial and Error Approach for the Accurate Model

The cleaned data is ran through the selected algorithms and the accuracy levels are noted down. The procedure is repeated till a suitable accuracy is obtained. This trial and error helped us get to the completion of the visual inspection system. This also concluded that the CNN works best for such projects

C. The Model is Tested with Real-Time Objects

A suitable object detection model is used for detection. One can use general purpose models available in Kaggle or make a specific model. The object detection model is ran to detect the object. The classification algorithm is then ran on the detected object. That is how the detected object can be classified into perfect or damaged.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 2, Issue 2, March 2022
Impact Factor: 6.252

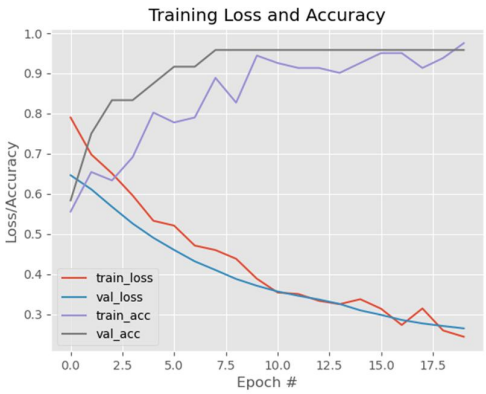


Figure: Training loss and Accuracy

IV. FINDINGS

The model was trained and the model wasn't accurate at first. Despite having 89% accuracy, the model was inaccurately producing results. We had to repeat the process until we made a model with 96% accuracy that was able to identify the objects accurately. Once the model was tested and found to be accurate, we could successfully identify the faulty objects on the assembly line.

V. CONCLUSION

The Project enables an assembly line to automate the inspection of manufactured products. It is scalable as well as can be programmed for multiple products. Automated technologies aren't only executing iterative tasks but also augmenting workforce capabilities significantly. In fact, automated machines are expected to exchange almost half the worldwide workforce. Multiple industries from banking to manufacturing, are adopting automation to drive productivity, safety, profitability, and quality. Automation will support connectivity and reliability in a hyper-competitive ecosystem. The future of automation looks hopeful where everything will be made easily available and accessible.

ACKNOWLEDGMENT

We would like to thank Professor Prajakta Khelkar, our project guide from Department of Computer Engineering for providing us the required guidance throughout the project. We would also like to acknowledge Dr. Mahavir Devmane, HOD, Department of Computer Engineering, VPPCOEVA and Professor Vinod Alone, Project head, Department of Computer Engineering, VPPCOEVA for hosting meetings that helped us choose this topic for our project.

REFERENCES

[1]. AI-based visual inspection for defect detection by Serhii Maksymenko
[2]. Everything you need to know about visual inspection with AI by Jamshed Khan
[3]. The use of visual inspection in asset management and quality control opencv for computer vision application by Naveenkumar Mahamkali
[4]. Book Python for data analyst