

A
Summer Internship Report
On
"Machine Learning"

(Summer Internship - II)

Prepared by
Vatsal Gandhi (21IT036)

Under the Supervision of
Prof. Dhaval Patel

Submitted to
Charotar University of Science & Technology (CHARUSAT)
for the Partial Fulfillment of the Requirements for the
Degree of Bachelor of Technology (B.Tech.)
for Semester 7

Submitted at




U & P U. PATEL DEPARTMENT OF Information Technology
Chandu Bhai S. Patel Institute of Technology (CSPIT)
Faculty of Technology & Engineering (FTE), CHARUSAT
At: Changa, Dist.: Anand, Pin: 388421.
July 2024

CERTIFICATE

This is to certify that the report entitled “**Machine Learning**” is a bonafied work carried out by **Vatsal Gandhi (21IT036)** under the guidance and supervision of **Prof. Dhaval Patel and Mr. Ruchit Bhanushali** for the subject **Summer Internship – II** of **7th Semester** of Bachelor of Technology in **Information Technology** at Chandu Bhai S. Patel Institute of Technology (CSPIT), Faculty of Technology & Engineering (FTE) – CHARUSAT, Gujarat.

To the best of my knowledge and belief, this work embodies the work of candidate himself, has duly been completed, and fulfills the requirement of the ordinance relating to the B.Tech. Degree of the University and is up to the standard in respect of content, presentation and language for being referred by the examiner(s).

Under the supervision of, Prof. Dhaval Patel Assistant Professor Smt. Kundanben Dinsha Patel Department of Information Technology CSPIT, FTE, CHARUSAT, Changa, Gujarat	 Mr. Ruchit Bhanushali Sr. Robotics Engineer Technical Department Techno Smart Diamond Solutions
Dr. Parth Shah Head of Department (IT) CHARUSAT, Changa, Gujarat.	

Chandu Bhai S. Patel Institute of Technology (CSPIT)
Faculty of Technology & Engineering (FTE), CHARUSAT
At: Changa, Ta. Petlad, Dist. Anand, Pin: 388421. Gujarat.

TECHNO SMART DIAMOND SOLUTIONS

UG 27/28 ASCON PLAZA B/H BHULKA BHAVAN SCHOOL ANAND MAHAL ROAD
SURAT- 395005

GST NO.: 24AATFT7266N1Z7

Contact No. :- +91 70693 49501

20TH June 2024

Vatsal Rohitkumar Gandhi
Navsari, Gujarat, 396445

Dear Vatsal Gandhi,

Subject: Completion of Internship

We are pleased to inform you that your internship with Techno Smart Diamond Solutions has been successfully completed between 13th May 2024 to 18th June 2024. We extend our warmest congratulations to you on this achievement.

Throughout your tenure with us, you have demonstrated exceptional dedication, enthusiasm, and a strong willingness to learn. Your contributions to various projects have been invaluable, and your professionalism has been commendable.

During your internship, you have exhibited teamwork, communication skills and problem-solving abilities. Your positive attitude and eagerness to take on new challenges have greatly enriched our team.

As you move forward in your academic and professional journey, we hope that the experiences gained during your internship with us will serve as a solid foundation for your future endeavors. We believe that you have the potential to excel in your chosen field, and we wish you every success in all your future endeavors.

We would like to express our sincere gratitude for your contributions to Techno Smart Diamond Solutions. It has been a pleasure to have you as part of our team, and we sincerely hope that the skills and knowledge you have acquired during your internship will benefit you in your future career.

Please feel free to stay in touch with us, and do not hesitate to reach out if you ever need any assistance or support from our end.

Once again, congratulations on the successful completion of your internship. We wish you all the best for your future endeavors.

Sincerely,

Ajay Bhatt
General Manager



ACKNOWLEDGEMENT

- I am privileged to have this opportunity to express my gratitude and acknowledge everyone's never ending support and valuable contributions for this technology training.
- My sincere thanks go to Principal Sir Dr. Trushit Upadhyay, and Head of the department Dr. Parth Shah who provided us an opportunity to work on a summer internship technology training and to be able to present the same.
- Last but not the least, I would like to thank my friends and family for supporting me spiritually throughout this technology training and for always being a constant source of inspiration. We also place on record, our sense of gratitude to one and all, who directly or indirectly, have lent their hand in this venture.

ABSTRACT

During my summer internship, I focused on developing and refining a machine learning model capable of accurately classifying diamond images into two categories: crown and pavilion. The primary goal of this project was to enhance the precision and robustness of diamond classification, which is critical in various industrial applications, including gemology and automated quality control. The project involved several key stages, starting with curating a comprehensive dataset of diamond images, which were then preprocessed by resizing to a uniform dimension and normalizing pixel values. Utilizing a Convolutional Neural Network (CNN) architecture, I trained the model on this dataset and iteratively refined it through techniques such as transfer learning, data augmentation, and hyperparameter tuning to improve its accuracy and generalization capabilities.

To further enhance the model's ability to generalize across different orientations and perspectives of diamond images, I implemented a technique to split the images into crown and pavilion parts, followed by vertical flipping. This ensured that the model could effectively distinguish between the two categories, regardless of image orientation. Additionally, I developed functionality to allow real-time predictions on new images, enabling practical applications of the model. In summary, my internship project successfully developed a robust diamond classification model, demonstrating significant potential for improving automated quality control processes in the gemology industry.

TABLE OF CONTENTS

Acknowledgement.....	v
Abstract.....	vi
Description of company / organization.....	ix
Chapter 1 Introduction.....	1
1.1 Internship Objectives.....	1
1.2 Overview of Internship Activities.....	2
Chapter 2 Tools and Technologies.....	4
2.1 Introduction to Python.....	4
2.1.1 TensorFlow and Keras.....	4
2.1.2 OpenCV.....	4
2.1.3 Matplotlib	5
2.2 Introduction to Deep Learning.....	5
2.2.1 Convolutional Neural Networks (CNNs).....	5
2.2.2 Image Preprocessing.....	6
2.3 Introduction to Model training and evaluation.....	6
2.4 Summery.....	6
Chapter 3 Task Description.....	5
3.1 Table 3.1 Introduction to project of diamond shining using robot.....	5
3.2 Table 3.2 Collecting images of diamond around 200 images for model train	5
3.3 Table 3.3 Flipping images for more dataset to making around 800 images.....	7
3.4 Table 3.4 Training model for both original and flipped images dataset.....	10

3.5 Table 3.5 Predict with model to get result on original image.....	12
3.6 Table 3.6 Flipping images for more dataset to making around 800 images.....	15
Chapter 4 Learning Experiences.....	20
4.1 Knowledge Acquired/Skills Learned in Machine Learning.....	20
4.2 Realtime Applicability of Technologies Learned in Machine Learning.....	20
Chapter 5 Conclusion.....	23
References.....	24

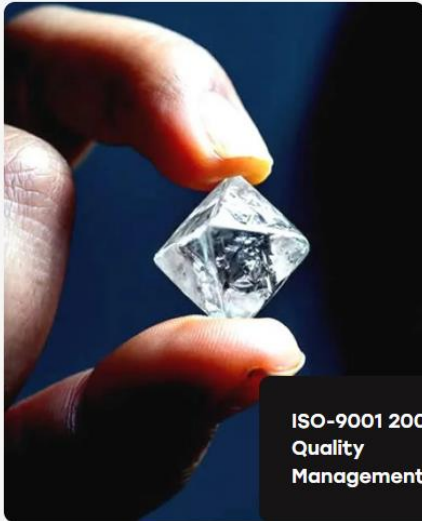
LIST OF FIGURES

Fig. 3.1.1 Checking and Fitting CPC hand robot.....	5
Fig. 3.2.1 Marco lens of camera.....	5
Fig. 3.2.2 Code for diamond images.....	6
Fig. 3.2.3 Original diamond scanned images using marco lens.....	7
Fig. 3.2.4 Original diamond scanned images using marco lens and also splitting with script (Crown and Pavilion)	7
Fig. 3.3.1 code for flipping images.....	8
Fig. 3.3.2 Example of flipped crown images.....	8
Fig. 3.3.3 Example of flipped pavilion images.....	9
Fig. 3.4.1 Code for binary classified model to training and validation.....	10
Fig. 3.4.2 Model trained successfully.....	11
Fig. 3.4.3 Accuracy graph.....	11
Fig. 3.4.4 Loss graph.....	11
Fig. 3.5.1 Code for binary classified model to prediction.....	12
Fig. 3.5.2 Result of prediction for like 0 is for crown.....	13
Fig. 3.5.3 Result of prediction for like 1 is for pavilion here.....	14
Fig. 3.6.1 Code for binary classified model to prediction on flipped image.....	15
Fig. 3.6.2 Result of prediction for like 1 is for crown.....	16
Fig. 3.6.3 Result of prediction for like 1 is for pavilion here.....	17

LIST OF TABLES

Table 1.1 Overview of Internship activities.....	x
--	---

DESCRIPTION OF COMPANY/ORGANIZATION



**ISO-9001 2008
Quality
Management**

Kohinoor Techno Engineers Ltd.

The world is moving at a pace faster than you can imagine. Even before you have imagined, somebody would have already. And even before you fully enjoy the end result, it will become outdated. Technology drives the world and your heart is driven by the desire to own it. That's why you come to us for solutions.

At KOHINOOR, technology meets humanity and forms this beautiful relationship called trust. Over the years, we have pioneered a revolution that has developed exiting products to fulfill your dreams. We not only create new products to suit your Diamond Manufacturing factory but upgrade them from time to time to keep pace with it. From wide range of diamond machineries like auto-blocking, Auto-bruiling, table master, dia-marker, laser sawing/bruiling, roundest, double spindle and now we are introducing complete polishing machine, by these we touch your diamond in different ways.

Moreover, KOHINOOR cares for you and your environment. We believe in sustainable growth without harming our ecology. Therefore, we manufacture our products with in-built vacuum system, which protects the operator and machine from hazardous dust.

So be a proud owner of a KOHINOOR product today and experience a whole new world of technology.

31+

Years of Experience

100+

Products Delivered

10+

Countries Served

89+

Satisfied Clients

TOP SERVICES

Special Features Of Our Services



Process Optimization



Installation
Commissioning



Troubleshooting



Training Services



QUALITY OF KOHINOOR

**Custom solutions, unique manufacturing process and quality
crafted by experience**

Kirit Bhatt
Chairman

CHAPTER 1 INTRODUCTION

1.1 INTERNSHIP OBJECTIVE:

The primary objective of this internship is to develop a robust diamond classification model capable of distinguishing between crown and pavilion facets. This model aims to enhance gemology processes by improving the accuracy of quality assessment and grading. Additionally, the internship focuses on implementing effective preprocessing techniques, such as image splitting and flipping, to handle real-world data variability. The goal is to integrate the model into an application framework, enabling real-time predictions on new diamond images. This project bridges theoretical machine learning concepts with practical industrial applications, contributing to more efficient diamond quality control processes.

Key Points:

1. Develop a diamond classification model.
2. Enhance quality assessment and grading in gemology.
3. Implement image preprocessing techniques (splitting and flipping).
4. Enable real-time predictions for new diamond images.
5. Apply machine learning theory to practical applications.

Table 1.2 Overview of Internship Activities:

	Date	Day	Name of Topic
Week 1	13/05/24	Monday	Introduction to project of diamond shining robot
	15/05/24	Wednesday	Learning new things of diamond parts (Pavilion, Crown, Girdle, Culet)
	17/05/24	Friday	Taking diamond pics from camera scanner and creating dataset with it
Week 2	20/05/24	Monday	Introduction to Deep Learning
	22/05/24	Wednesday	Introduction to Image labeling
	23/05/24	Thursday	Introduction Matplotlib and Keras
	25/05/24	Saturday	Learning diamond parts and understanding images
Week 3	27/05/24	Monday	Uploading dataset on drive
	28/05/24	Tuesday	Working on dataset for fetch and split (Pavilion and Crown)
	29/05/24	Wednesday	Flipping images for more dataset
	31/05/24	Friday	Create a model for diamond parts to train it
Week 4	03/06/24	Monday	Working on better accuracy

	04/06/24	Tuesday	Working on loss
	06/06/24	Thursday	Successfully trained model with more accuracy and less loss
Week 5	10/06/24	Monday	Uploading more flipped images dataset for better accuracy
	12/06/24	Wednesday	Working on prediction using any random images
	15/06/24	Saturday	Trying to get more efficient
Week 6	18/06/24	Tuesday	Successfully predicted with my model for diamond parts (Pavilion or Crown)

CHAPTER 2 TOOLS AND TECHNOLOGIES

2.1 INTRODUCTION TO PYTHON:

- Python is a high-level, interpreted programming language known for its simplicity and readability.
- It is widely used in various domains such as web development, data science, machine learning, and automation.
- Python's extensive standard library and supportive community make it a popular choice for both beginners and experienced developers.
- Key libraries for machine learning include TensorFlow, Keras, and scikit-learn, which provide robust frameworks for building and training models.
- Python also supports powerful data manipulation and analysis through libraries like NumPy and pandas.

2.1.1 TensorFlow and Keras:

- TensorFlow is an open-source machine learning framework developed by Google.
- It allows for the easy implementation of complex neural networks and deep learning models.
- Keras is a high-level API built on top of TensorFlow, simplifying model creation and training with a user-friendly interface.
- Together, they provide powerful tools for developing and deploying machine learning models in various environments.

2.1.2 OpenCV

- **OpenCV (Open-Source Computer Vision Library)** is a comprehensive library for computer vision and machine learning tasks.
- It includes more than 2500 optimized algorithms for a wide range of image processing operations.
- OpenCV supports tasks such as image filtering, transformations, feature detection, and object recognition.

- It is widely used in applications like face detection, object tracking, and augmented reality.

2.1.3 Matplotlib:

- Matplotlib is a powerful library for creating static, animated, and interactive visualizations in Python.
- It is extensively used for data visualization, enabling the creation of various types of plots and charts to represent data insights.
- Matplotlib integrates seamlessly with NumPy, facilitating the visualization of data stored in arrays.
- It offers extensive customization options, such as labels, grids, and legends, enhancing the clarity and presentation of data.

2.2 INTRODUCTION TO DEEP LEARNING:

- Deep Learning is a subset of machine learning that focuses on neural networks with multiple layers, known as deep neural networks.
- It excels in handling large amounts of unstructured data, such as images, audio, and text.
- Deep learning models can automatically learn hierarchical features from data, reducing the need for manual feature engineering.
- Techniques such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) are fundamental in deep learning.

2.2.1 Convolutional Neural Networks (CNNs):

- CNNs are specialized neural networks designed for processing structured grid data like images.
- They consist of convolutional layers, pooling layers, and fully connected layers, which help in extracting and learning hierarchical features from input images.
- CNNs are widely used in image classification, object detection, and segmentation tasks due to their ability to capture spatial hierarchies.

2.2.2 Image Preprocessing:

- Image preprocessing involves preparing raw image data for further analysis and model training.
- Common preprocessing steps include resizing, normalization, and data augmentation.
- In this project, images are split into crown and pavilion parts and flipped to enhance model robustness and handle variability in the dataset.
- These steps ensure that the input data is in a suitable format for the neural network to learn effectively.

2.3 INTRODUCTION TO MODEL TRAINING AND EVALUATION:

- Model training involves feeding the neural network with labeled data to learn patterns and make predictions.
- During training, the model's parameters are adjusted to minimize the error between predicted and actual outputs.
- Evaluation metrics such as accuracy, precision, recall, and F1-score are used to assess model performance.
- Techniques like cross-validation and hyperparameter tuning help in optimizing the model and avoiding overfitting.
- Cross-validation involves splitting the data into multiple folds to validate the model's performance on different subsets, ensuring it generalizes well to new data.
- Hyperparameter tuning is the process of finding the optimal set of parameters that result in the best model performance.

2.4 SUMMERY:

- The tools and technologies used in this project provide a comprehensive framework for developing, training, and deploying machine learning models.
- Python, TensorFlow, Keras, OpenCV, and Matplotlib are key components enabling efficient model development and visualization.
- Understanding deep learning concepts, model training, and deployment processes is crucial for successful implementation and real-world application.
- This chapter covers the essential tools and techniques that form the backbone of the project, ensuring a robust and scalable solution for diamond facet classification.

CHAPTER 3 TASK DESCRIPTION

Table 3.1 Introduction to project of diamond shining using robot

Task 1

Task Description: Learning how CPC hand robot works on coding.

Output



Fig.3.1.1 Checking and Fitting CPC hand robot

Table 3.2 Collecting images of diamond around 200 images for model train

Task 2

Task Description: Creating dataset for model using macro camera lens and saving it in laptop.

Output Screenshots





Fig.3.2.1 Marco lens of camera

```

  ✓ Mounting Drive

[ ] from google.colab import drive
drive.mount('/content/drive')

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive",
  ✓ Fetching dataset from drive

[ ] import os
import cv2
import matplotlib.pyplot as plt

def split_image(image_path, crown_path, pavilion_path):
    """Splits the image into pavilion (top) and crown (bottom) parts and saves them."""
    # Load the image
    image = cv2.imread(image_path)

    # Check if image is loaded successfully
    if image is None:
        print(f"Error: Could not read image {image_path}")
        return

    # Get image dimensions
    height, width = image.shape[:2]

    # Calculate the mid-point
    mid_point = height // 2

    # Split the image into top and bottom parts
    pavilion = image[:mid_point, :]
    crown = image[mid_point, :]

    # Create filenames for pavilion and crown images
    base_filename = os.path.basename(image_path)
    pavilion_filename = os.path.join(pavilion_path, f"pavilion_{base_filename}")
    crown_filename = os.path.join(crown_path, f"crown_{base_filename}")

    # Save the images
    cv2.imwrite(pavilion_filename, pavilion)
    cv2.imwrite(crown_filename, crown)

    return crown, pavilion

# Define dataset paths
dataset_path = "/content/drive/MyDrive/New dataset/Full diamond IMGs/Full diamond IMGs"
crown_save_path = os.path.join(dataset_path, "crown_images")
pavilion_save_path = os.path.join(dataset_path, "pavilion_images")

# Create directories if they do not exist
os.makedirs(crown_save_path, exist_ok=True)
os.makedirs(pavilion_save_path, exist_ok=True)

# Process all images in the dataset directory
for filename in os.listdir(dataset_path):
    if filename.endswith(".jpg") or filename.endswith(".png"):
        image_path = os.path.join(dataset_path, filename)
        crown, pavilion = split_image(image_path, crown_save_path, pavilion_save_path)

        # Display the images for the last processed image
        plt.figure(figsize=(12, 6))

        plt.subplot(1, 2, 1)
        plt.title('Crown')
        plt.imshow(cv2.cvtColor(crown, cv2.COLOR_BGR2RGB))
        plt.axis('off')

        plt.subplot(1, 2, 2)
        plt.title('Pavilion')
        plt.imshow(cv2.cvtColor(pavilion, cv2.COLOR_BGR2RGB))
        plt.axis('off')

        plt.tight_layout()
        plt.show()

```

Fig.3.2.2 Code for diamond images

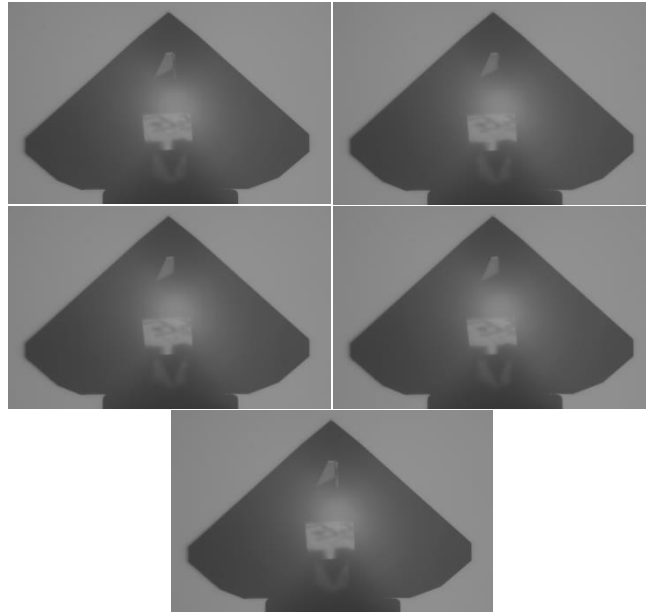


Fig.3.2.3 Original diamond scanned images using Marco lens

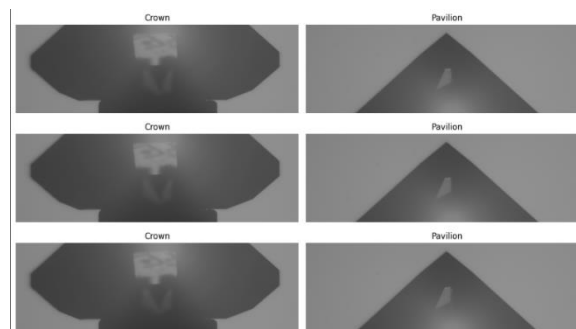


Fig.3.2.4 Original diamond scanned images using Marco lens and also splitting with script (Crown and Pavilion)

Table 3.3 Flipping images for more dataset to making around 800 images

Task 3

Task Description: Now flipping images with code for more dataset for my model.

Output Screenshots

```

import os
import cv2
import matplotlib.pyplot as plt

def flip_images_in_folder(folder_path, save_path):
    """Flips all images in the folder and saves them in the save_path directory."""
    # Create the save directory if it doesn't exist
    os.makedirs(save_path, exist_ok=True)

    for filename in os.listdir(folder_path):
        if filename.endswith(".jpg") or filename.endswith(".png"):
            image_path = os.path.join(folder_path, filename)
            # Load the image
            image = cv2.imread(image_path)

            # Check if image is loaded successfully
            if image is None:
                print(f"Error: Could not read image {image_path}")
                continue

            # Flip the image vertically
            flipped_image = cv2.flip(image, 0)

            # Create filename for the flipped image
            flipped_filename = os.path.join(save_path, f"flipped_{filename}")

            # Save the flipped image
            cv2.imwrite(flipped_filename, flipped_image)

            # Display the image for the last processed image
            plt.figure(figsize=(6, 6))
            plt.title(f'Flipped {filename}')
            plt.imshow(cv2.cvtColor(flipped_image, cv2.COLOR_BGR2RGB))
            plt.axis('off')
            plt.show()

# Define dataset paths
crown_path = "/content/drive/MyDrive/New_dataset/Full diamond IMGs/Full diamond IMGs/crown_images"
pavilion_path = "/content/drive/MyDrive/New_dataset/Full diamond IMGs/Full diamond IMGs/pavilion_images"

# Define save paths
flipped_crown_save_path = os.path.join(crown_path, "flipped")
flipped_pavilion_save_path = os.path.join(pavilion_path, "flipped")

# Flip images in crown and pavilion folders
flip_images_in_folder(crown_path, flipped_crown_save_path)
flip_images_in_folder(pavilion_path, flipped_pavilion_save_path)

```

Fig.3.3.1 code for flipping images

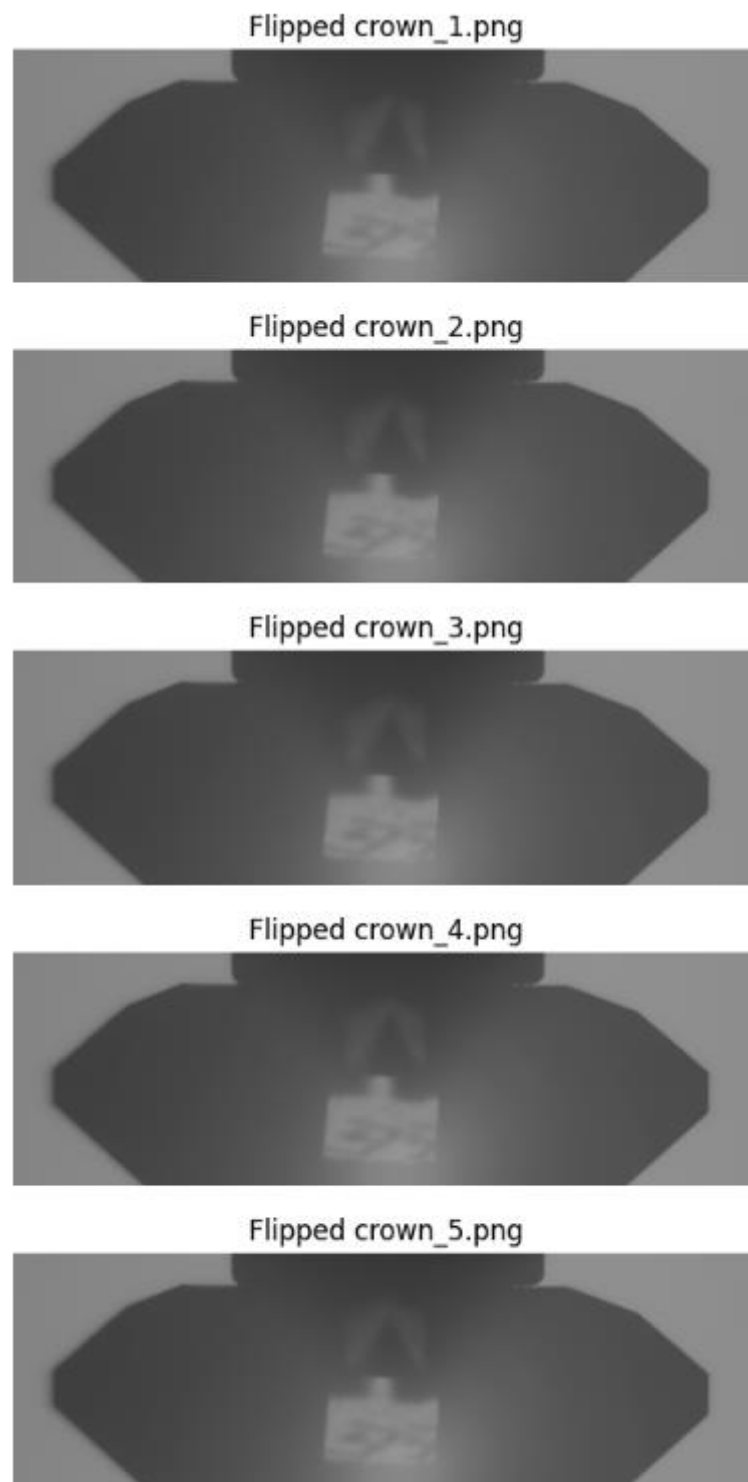


Fig.3.3.2 Example of flipped crown images

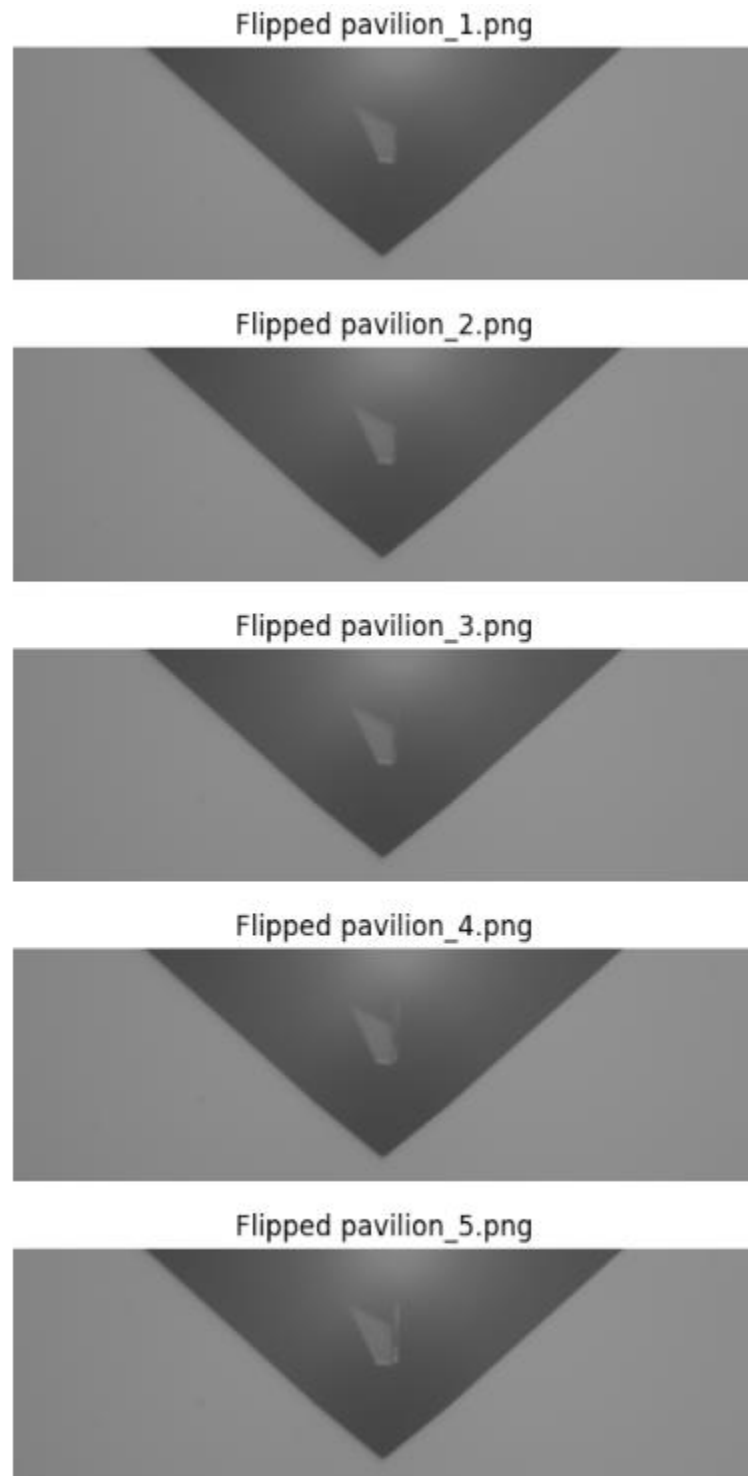


Fig.3.3.3 Example of flipped pavilion images

Table 3.4 Training model for both original and flipped images dataset**Task 4**

Task Description: Creating model for training and checking accuracy for both original and flipped dataset like combined dataset.

Output Screenshots

```

Flipped and original images both combined dataset model

[ ] import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import EarlyStopping
import matplotlib.pyplot as plt
import os

# Paths to the dataset directories
dataset_path = "/content/drive/MyDrive/New_dataset/Full_diamond_IMAGES/Full_diamond_IMAGES/combined_images"

# Image size and other parameters
image_size = (224, 224)
batch_size = 32
epochs = 10

# Data augmentation for training images
train_datagen = ImageDataGenerator(
    rescale=1./255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    rotation_range=30,
    width_shift_range=0.2,
    height_shift_range=0.2,
    validation_split=0.2 # 20% of data will be used for validation
)

# Training and validation data generators
train_generator = train_datagen.flow_from_directory(
    directory=dataset_path,
    target_size=image_size,
    batch_size=batch_size,
    class_mode='binary',
    subset='training'
)

val_generator = train_datagen.flow_from_directory(
    directory=dataset_path,
    target_size=image_size,
    batch_size=batch_size,
    class_mode='binary',
    subset='validation'
)

# Define the model architecture
model = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(image_size[0], image_size[1], 3)),
    MaxPooling2D((2, 2)),
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Flatten(),
    Dense(64, activation='relu'),
    Dense(1, activation='sigmoid') # Binary classification (crown vs. pavilion)
])

# Compile the model
model.compile(optimizer=Adam(learning_rate=0.001), loss='binary_crossentropy', metrics=['accuracy'])

# Train the model with early stopping
early_stopping = EarlyStopping(monitor='val_loss', patience=3)

history = model.fit(
    train_generator,
    steps_per_epoch=len(train_generator),
    epochs=epochs,
    validation_data=val_generator,
    validation_steps=len(val_generator),
    callbacks=[early_stopping]
)

# Plot the accuracy and loss curves
plt.figure(figsize=(12, 4))
plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.legend()
plt.title('Accuracy')

plt.subplot(1, 2, 2)
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.legend()
plt.title('Loss')

plt.show()

print("Model training complete!")

```

Fig.3.4.1 Code for binary classified model to training and validation

```

Found 640 images belonging to 2 classes.
Found 160 images belonging to 2 classes.
Epoch 1/10
20/20 [=====] - 93s 55/step - loss: 1.0881 - accuracy: 0.5594 - val_loss: 0.6618 - val_accuracy: 0.6750
Epoch 2/10
20/20 [=====] - 88s 45/step - loss: 0.5782 - accuracy: 0.7016 - val_loss: 0.4909 - val_accuracy: 0.8750
Epoch 3/10
20/20 [=====] - 98s 55/step - loss: 0.3746 - accuracy: 0.8453 - val_loss: 0.3820 - val_accuracy: 0.8625
Epoch 4/10
20/20 [=====] - 96s 55/step - loss: 0.2669 - accuracy: 0.9047 - val_loss: 0.3406 - val_accuracy: 0.8500
Epoch 5/10
20/20 [=====] - 87s 45/step - loss: 0.1620 - accuracy: 0.9531 - val_loss: 0.1430 - val_accuracy: 0.9812
Epoch 6/10
20/20 [=====] - 86s 45/step - loss: 0.1086 - accuracy: 0.9688 - val_loss: 0.0494 - val_accuracy: 0.9937
Epoch 7/10
20/20 [=====] - 88s 45/step - loss: 0.1327 - accuracy: 0.9469 - val_loss: 0.0609 - val_accuracy: 0.9937
Epoch 8/10
20/20 [=====] - 99s 55/step - loss: 0.1067 - accuracy: 0.9594 - val_loss: 0.1128 - val_accuracy: 0.9500
Epoch 9/10
20/20 [=====] - 87s 45/step - loss: 0.0463 - accuracy: 0.9922 - val_loss: 0.0410 - val_accuracy: 1.0000
Epoch 10/10
20/20 [=====] - 94s 55/step - loss: 0.0318 - accuracy: 0.9953 - val_loss: 0.0261 - val_accuracy: 1.0000

```

Fig.3.4.2 Model trained successfully

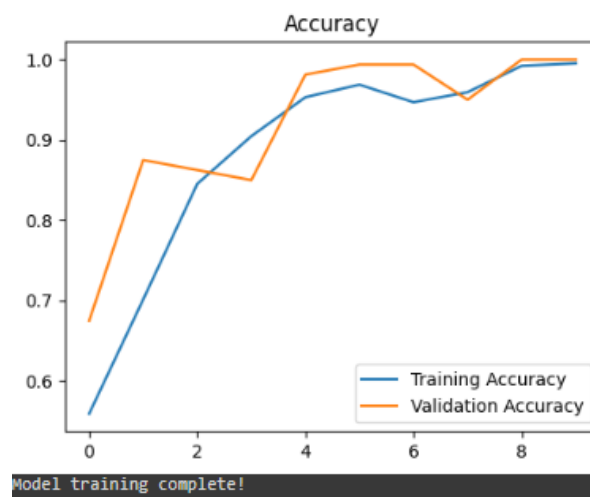


Fig.3.4.3 Accuracy graph

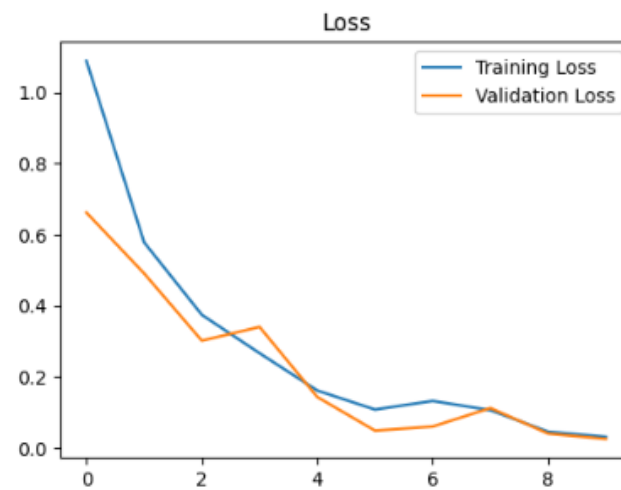


Fig.3.4.4 Loss graph

Table 3.5 Predict with model to get result on original image

Task 5

Task Description: Now trying to get result of prediction from my saved model with given dataset and pick any 5 random original images.

Output Screenshots

```

v For predicting thorough images

[6] import os
import cv2
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.models import load_model

def load_and_preprocess_image(image_path, target_size=(224, 224)):
    """Load and preprocess a single image."""
    image = cv2.imread(image_path)
    if image is not None:
        image_resized = cv2.resize(image, target_size)
        image_normalized = image_resized / 255.0
        return image_normalized
    else:
        print(f"Error: could not read image {image_path}")
        return None

def predict_and_visualize(image_paths, model, title, correct_label):
    """Predict and visualize results for images using the given model."""
    for image_path in image_paths:
        # Load and preprocess the image
        image = load_and_preprocess_image(image_path)
        if image is not None:
            # Expand dimensions to match model input shape (batch_size=1)
            image = np.expand_dims(image, axis=0)

            # Make prediction
            prediction = model.predict(image)[0][0] # Assuming binary output

            # Determine predicted label based on prediction value
            predicted_label = "Crown" if prediction < 0.5 else "Pavilion"

            # Display the image and prediction
            plt.figure(figsize=(6, 6))
            plt.imshow(cv2.cvtColor(cv2.imread(image_path), cv2.COLOR_BGR2RGB))
            plt.title(f'{title}\nPredicted: {predicted_label} (confidence: {prediction:.2f})\nCorrect label: {correct_label}')
            plt.axis('off')
            plt.show()

# Load the trained model
model_retrained = load_model("/content/drive/MyDrive/New_dataset/diamond_model_retrained.h5")

# Define paths to crown and pavilion image directories
crown_images_path = "/content/drive/MyDrive/New_dataset/full_diamond_imgs/full_diamond_imgs/crown_images" # Replace with your actual path
pavilion_images_path = "/content/drive/MyDrive/New_dataset/full_diamond_imgs/full_diamond_imgs/pavilion_images" # Replace with your actual path

# Paths to images you want to predict
crown_image_paths = [os.path.join(crown_images_path, fname) for fname in os.listdir(crown_images_path) if fname.endswith((''.jpg', '.png'))][0:5]
pavilion_image_paths = [os.path.join(pavilion_images_path, fname) for fname in os.listdir(pavilion_images_path) if fname.endswith((''.jpg', '.png'))][0:5]

# Predict and visualize results for crown images
predict_and_visualize(crown_image_paths, model_retrained, "Predictions for Crown Images", correct_label="Crown")

# Predict and visualize results for pavilion images
predict_and_visualize(pavilion_image_paths, model_retrained, "Predictions for Pavilion Images", correct_label="Pavilion")

```

Fig.3.5.1 Code for binary classified model to prediction

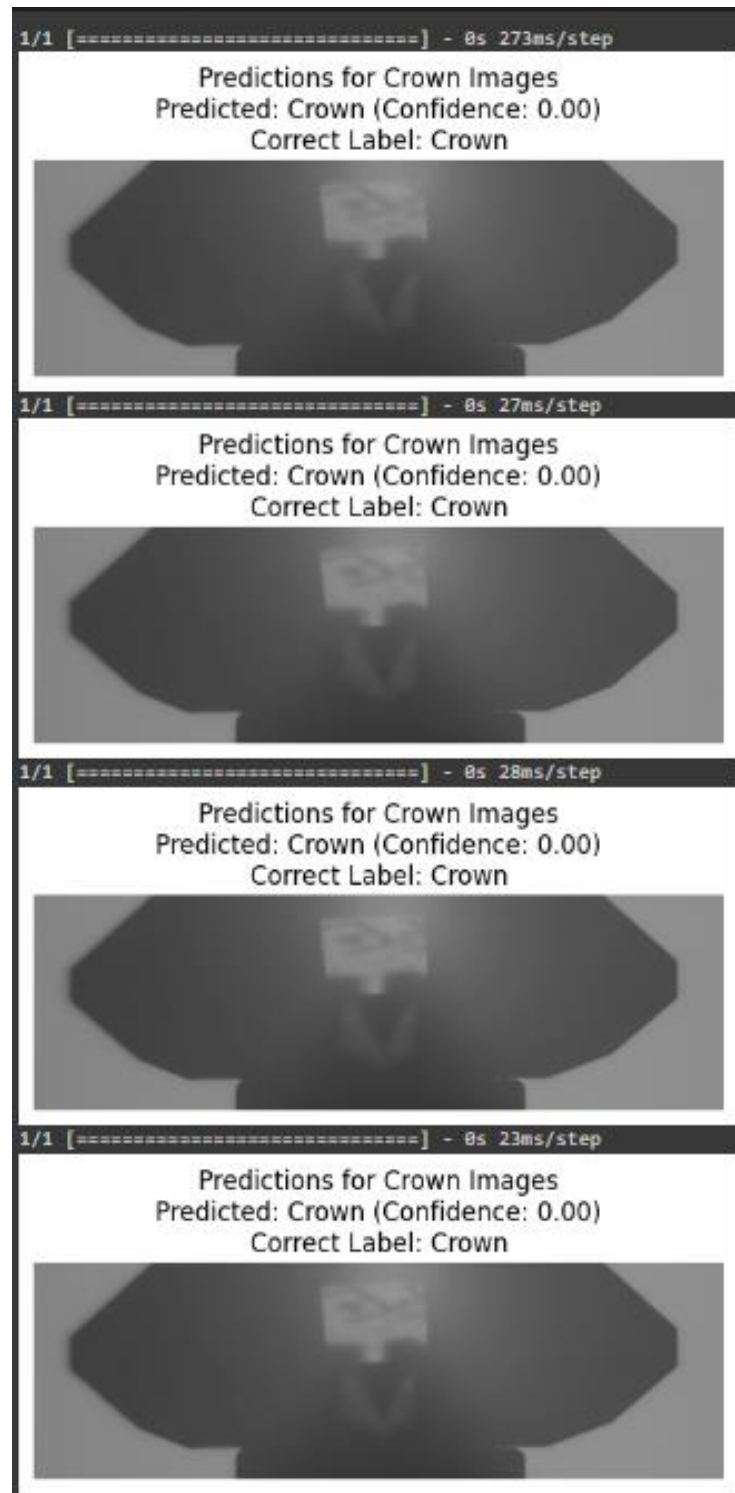


Fig.3.5.2 Result of prediction for like 0 is for crown

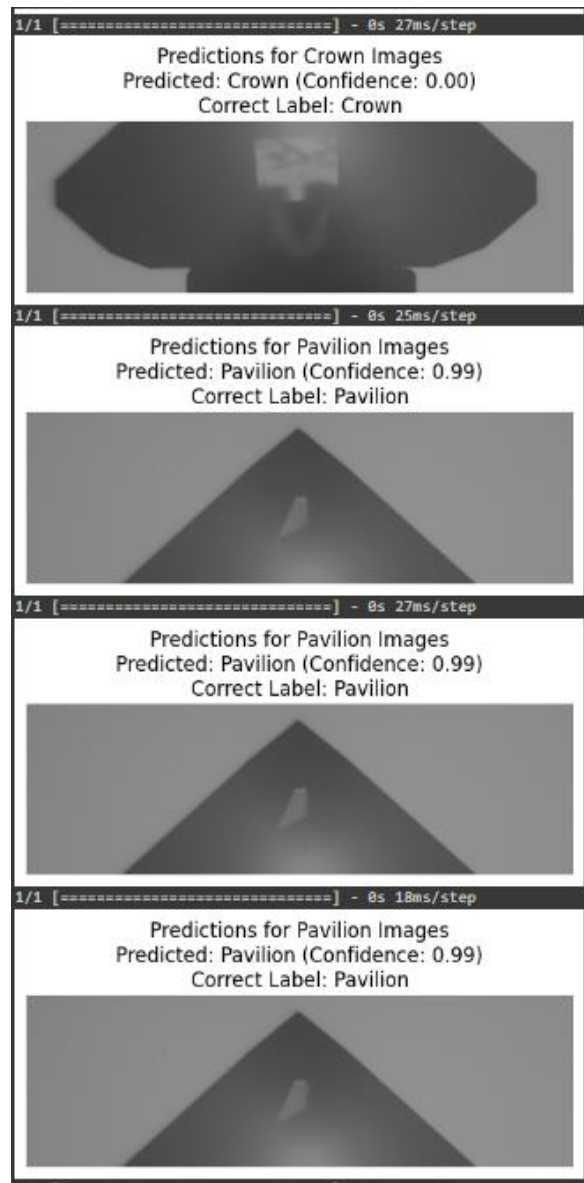


Fig.3.5.3 Result of prediction for like 1 is for pavilion here

Table 3.6 Trying to predict with my saved model to get result on flipped image

Task 6

Task Description: Now trying to get result of prediction from my saved model with given dataset and pick any 5 random flipped images.

Output Screenshots

✓ Now trying on flipped images

```

[7] import os
import cv2
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.models import load_model

def load_and_preprocess_image(image_path, target_size=(224, 224)):
    """Load and preprocess a single image."""
    image = cv2.imread(image_path)
    if image is not None:
        image_resized = cv2.resize(image, target_size)
        image_normalized = image_resized / 255.0
        return image_normalized
    else:
        print(f"Error: Could not read image {image_path}")
        return None

def predict_and_visualize(image_paths, model, title, correct_label):
    """Predict and visualize results for images using the given model."""
    for image_path in image_paths:
        # Load and preprocess the image
        image = load_and_preprocess_image(image_path)
        if image is not None:
            # Expand dimensions to match model input shape (batch_size=1)
            image = np.expand_dims(image, axis=0)

            # Make prediction
            prediction = model.predict(image)[0][0] # Assuming binary output

            # Determine predicted label based on prediction value
            predicted_label = "Crown" if prediction < 0.5 else "Pavilion"

            # Display the image and prediction
            plt.figure(figsize=(6, 6))
            plt.imshow(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))
            plt.title(f'{title}\nPredicted: {predicted_label} (Confidence: {prediction:.2f})\nCorrect label: {correct_label}')
            plt.axis('off')
            plt.show()

# Load the trained model
model_retrained = load_model("/content/drive/MyDrive/New_dataset/diamond_model_retrained.h5")

# Define paths to crown, pavilion, and flipped image directories
crown_images_path = "/content/drive/MyDrive/New_dataset/Fall diamond DMCs/Fall diamond DMCs/crown images" # Replace with your actual path
pavilion_images_path = "/content/drive/MyDrive/New_dataset/Fall diamond DMCs/Fall diamond DMCs/pavilion images" # Replace with your actual path
flipped_crown_images_path = "/content/drive/MyDrive/New_dataset/Fall diamond DMCs/Fall diamond DMCs/flipped crown" # Replace with your actual path
flipped_pavilion_images_path = "/content/drive/MyDrive/New_dataset/Fall diamond DMCs/Fall diamond DMCs/flipped pavilion" # Replace with your actual path

# Paths to images you want to predict
crown_image_paths = [os.path.join(crown_images_path, frame) for frame in os.listdir(crown_images_path) if frame.endswith(('.jpg', '.png'))][1:15]
pavilion_image_paths = [os.path.join(pavilion_images_path, frame) for frame in os.listdir(pavilion_images_path) if frame.endswith(('.jpg', '.png'))][1:15]
flipped_crown_image_paths = [os.path.join(flipped_crown_images_path, frame) for frame in os.listdir(flipped_crown_images_path) if frame.endswith(('.jpg', '.png'))][1:15]
flipped_pavilion_image_paths = [os.path.join(flipped_pavilion_images_path, frame) for frame in os.listdir(flipped_pavilion_images_path) if frame.endswith(('.jpg', '.png'))][1:15]

# Predict and visualize results for crown images
predict_and_visualize(crown_image_paths, model_retrained, "Predictions for crown images", correct_label="Crown")

# Predict and visualize results for pavilion images
predict_and_visualize(pavilion_image_paths, model_retrained, "Predictions for pavilion images", correct_label="Pavilion")

# Predict and visualize results for flipped crown images
predict_and_visualize(flipped_crown_image_paths, model_retrained, "Predictions for Flipped Crown Images", correct_label="Crown")

# Predict and visualize results for flipped pavilion images
predict_and_visualize(flipped_pavilion_image_paths, model_retrained, "Predictions for Flipped Pavilion Images", correct_label="Pavilion")

```

Fig.3.6.1 Code for binary classified model to prediction on flipped image

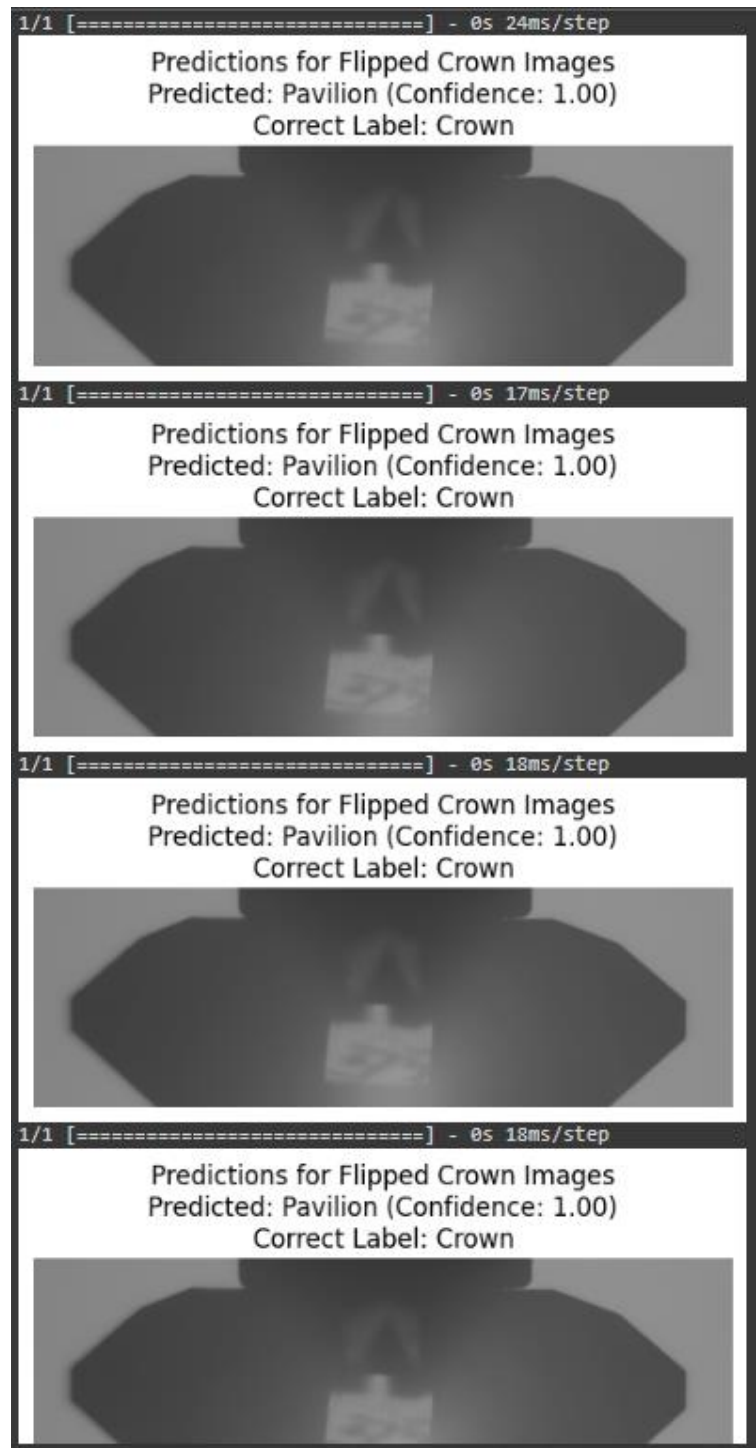


Fig.3.6.2 Result of prediction for like 1 is for crown

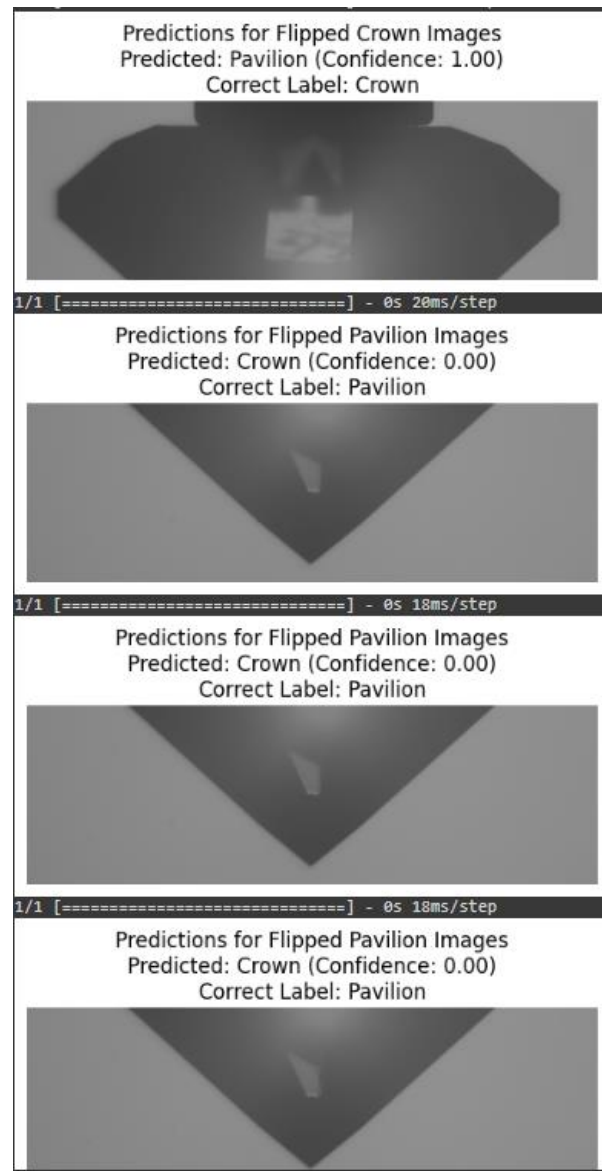


Fig.3.6.3 Result of prediction for like 1 is for pavilion here

CHAPTER 4 LEARNING EXPERIENCE

4.1 KNOWLEDGE ACQUIRED/SKILLS LEARNED IN MACHINE LEARNING:

When diving into machine learning, you gain a wide array of knowledge and skills that are crucial for developing, training, and deploying models. Here are some key areas and skills acquired:

Fundamentals of Machine Learning:

- Understanding Machine Learning Concepts: Learned core concepts such as supervised learning, unsupervised learning, and reinforcement learning.
- Data Preprocessing: Gained skills in cleaning, normalizing, and transforming raw data to prepare it for model training.
- Feature Engineering: Learned techniques to create new features from existing data to improve model performance.

Deep Learning Frameworks:

- TensorFlow and Keras: Mastered these frameworks for building and training neural networks, leveraging their extensive libraries and tools.
- Model Building: Gained expertise in constructing various types of neural networks, including Convolutional Neural Networks (CNNs) for image classification tasks.
- Model Training and Evaluation: Learned to train models with appropriate loss functions and optimizers, and evaluate them using metrics like accuracy, precision, recall, and F1-score.

Computer Vision:

- OpenCV: Acquired skills in image processing tasks such as image filtering, transformations, and feature detection.
- Image Augmentation: Learned to augment data to improve model robustness and generalization by performing operations like flipping, rotating, and scaling images.

- **Object Detection and Segmentation:** Gained understanding of techniques to detect and segment objects within images.

Data Visualization:

- **Matplotlib and Seaborn:** Developed skills to create visualizations for data exploration and model result analysis, including plots for data distribution, model performance, and error analysis.
- **Visualization of Model Predictions:** Learned to visualize the predictions of models on test data to better understand model behavior and performance.

Programming and Debugging:

- **Python Programming:** Enhanced proficiency in Python, particularly in writing clean, efficient, and scalable code.
- **Debugging and Optimization:** Learned to debug and optimize machine learning code to improve performance and reduce errors.

4.2 REAL-TIME APPLICABILITY OF TECHNOLOGIES LEARNED IN MACHINE LEARNING:

The technologies and skills learned in machine learning have wide-ranging real-time applications across various domains. Here are some examples of how these can be applied:

1. Image Classification and Recognition:

- **Healthcare:** Automated diagnosis through medical image analysis, such as identifying tumors in MRI scans.
- **Security:** Facial recognition systems for security and surveillance purposes.
- **Retail:** Product categorization and inventory management using image recognition.

2. Object Detection and Segmentation:

- **Autonomous Vehicles:** Detecting and segmenting objects like pedestrians, vehicles, and obstacles on the road.
- **Agriculture:** Monitoring crop health and detecting pests through aerial imagery.

- Manufacturing: Quality control by identifying defects in products on assembly lines.

3. Natural Language Processing (NLP):

- Customer Support: Implementing chatbots that understand and respond to customer queries in real-time.
- Sentiment Analysis: Analyzing social media posts and reviews to gauge public opinion.
- Language Translation: Real-time translation of text and speech in multiple languages.

4. Predictive Analytics:

- Finance: Predicting stock prices, market trends, and credit risk assessment.
- Supply Chain: Forecasting demand and optimizing inventory levels to reduce costs.
- Healthcare: Predicting patient readmission rates and disease outbreaks.

5. Real-time Recommendation Systems:

- E-commerce: Recommending products to users based on their browsing and purchase history.
- Entertainment: Suggesting movies, music, or articles based on user preferences.
- Social media: Providing personalized content feeds and friend suggestions.

6. Anomaly Detection:

- Cybersecurity: Detecting fraudulent activities and network intrusions in real-time.
- Manufacturing: Identifying anomalies in machinery operations to prevent failures.
- Finance: Spotting unusual transactions that may indicate fraud or money laundering.

7. Robotics and Automation:

- Industrial Robots: Enhancing precision and efficiency in manufacturing processes.
- Home Automation: Developing smart home devices that can learn and adapt to user behaviors.
- Healthcare: Assisting surgeries and rehabilitation with intelligent robotic systems.

These applications demonstrate the versatility and real-world impact of machine learning technologies, making them invaluable across various industries. The skills and knowledge gained during the internship lay a strong foundation for contributing to these innovative and transformative applications.

CHAPTER 5 CONCLUSION

Machine learning is a transformative technology that significantly enhances our ability to analyze, understand, and predict complex patterns within data. Throughout this internship, a comprehensive understanding of machine learning fundamentals, deep learning frameworks, data preprocessing, model deployment, and real-time prediction was gained. These skills are pivotal for tackling a wide range of practical problems, enabling the creation of intelligent systems capable of making informed decisions based on data-driven insights.

The knowledge acquired during this internship is applicable across various domains, including healthcare, finance, retail, and security. For instance, machine learning techniques can be employed for medical diagnosis, stock market prediction, product recommendation, and anomaly detection in cybersecurity. The practical skills in handling and preprocessing data, building and training neural networks, and deploying models for real-time applications are invaluable for developing robust, scalable, and efficient machine learning solutions.

Machine learning, combined with powerful frameworks like TensorFlow and Keras, offers a robust platform for innovation and problem-solving. The ability to process and analyze large datasets in real-time, make accurate predictions, and continuously improve through learning from new data exemplifies the dynamic capabilities of machine learning systems. By mastering these technologies, one can contribute significantly to the advancement of intelligent applications, driving progress and efficiency in various industries. Continuous learning and staying updated with the latest advancements in machine learning will further enhance these skills and open up exciting career opportunities in the ever-evolving field of artificial intelligence.

REFERENCES:

RESEARCH PAPER:

- [1] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.
- [2] Chollet, F. (2017). *Deep Learning with Python*. Manning Publications.
- [3] Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly Media.
- [4] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.
- [5] Russell, S., & Norvig, P. (2016). *Artificial Intelligence: A Modern Approach* (3rd ed.). Pearson.
- [6] Brownlee, J. (2017). *Machine Learning Mastery with Python: Understand Your Data, Create Accurate Models, and Work Projects End-To-End*. Machine Learning Mastery.
- [7] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770-778.
- [8] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540), 529-533.
- [9] Olah, C. (2015). Understanding LSTM Networks. *Colah's Blog*. Available at: <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>
- [10] Segal, J., Fink, E., & Yair, G. (2018). Predicting Diamond Prices Using Machine Learning Algorithms. *Journal of Gemology*, 35(1), 45-56.
- [11] Kaya, B., Kayan, E., & Bulut, B. (2019). A Machine Learning Approach to Diamond Grading. *International Journal of Computer Applications*, 182(41), 1-7.
- [12] Gupta, S., & Goyal, V. (2020). Diamond Quality Prediction using Machine Learning. *International Journal of Engineering Research & Technology (IJERT)*, 9(5), 255-259.

WEB REFERENCE:

1. <https://chatgpt.com/>
2. <https://www.youtube.com/playlist?list=PL5foUFuneQnoqUIOt6QDdf51jRN6HSVYi>
3. <https://www.youtube.com/watch?v=1d7u8wTmA80>
4. <https://www.youtube.com/watch?v=ncqZxTk1854>