A MINI PROJECT REPORT

Submitted by

RETIKA S 211422104392

RESHMI R 211422104391

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

October 2024

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report **" Oral Cancer Classification And Detection"** is the bonafide work of **RETIKA S (211422104392) & RESHMI R (211422104391)** who carried out the project work under my supervision.

SIGNATURE	SIGNATURE
Dr.L.JABASHEELA, M.E.,Ph.D .,	**Mrs.S.LINCY JEMINA,M.E.,(Phd)**
HEAD OF THE DEPARTMENT DEPARTMENT OF CSE, PANIMALAR ENGINEERING COLLEGE , NASARATHPETTAI, POONAMALLEE, CHENNAI-600 123.	ASSISSTANT PROFESSOR DEPARTMENT OF CSE, PANIMALAR ENGINEERING COLLEGE, NASARATHPETTAI, POONAMALLEE, CHENNAI-600 123.

Certified that the above candidate(s) were examined in the Mini Project Viva-Voce Examination held on..........................

INTERNAL EXAMINER **EXTERNAL EXAMINAR**

ACKNOWLEDGEMENT

We express our deep gratitude to our respected Secretary and Correspondent Dr.P.CHINNADURAI, M.A., Ph.D.for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We would like to extend our heartfelt and sincere thanks to our Directors Tmt. C. VIJAYARAJESWARI , Dr. C . SAKTHIKUMAR , M.E. , Ph.D., and Tmt. SARANYASREE SAKTHIKUMAR B.E.,M.B.A.,for providing us with the necessary facilities for completion of this project.

We also express our gratitude to our Principal Dr.K.Mani, M.E., Ph.D.for his timely concern and encouragement provided to us throughout the course.

We thank the HOD of CSE Department , Dr. JABASHEELA, M.E.,Ph.D., for the support extended throughout the project.

We would like to thank our Project Guide Mrs.S.LINCY JEMINA, M.E., (Ph.D) and our Project Coordinator Dr.M.S.Vinmathi, M.E., Ph.D., for their continuous suuport and suggestions for the successful completion of the project.

NAME OF THE STUDENTS

RETIKA S [211422104392]

RESHMI R [211422104391]

TABLE OF CONTENTS

ABSTRACT

Oral cancer remains a significant public health concern, necessitating timely detection and accurate classification to improve patient outcomes. This paper presents a novel approach for oral cancer classification and detection utilizing Convolutional Neural Networks (CNNs) for feature extraction combined with K-means clustering for classification. We first employ a CNN to automatically extract pertinent features from oral cancer images, leveraging its ability to learn complex patterns from the data. Subsequently, we implement the K-means algorithm to cluster the extracted features, enabling effective categorization of the images into benign and malignant classes. Our methodology was evaluated on a dataset of oral images, and the results demonstrated superior accuracy compared to traditional classification methods. This approach not only enhances the efficiency of oral cancer detection but also holds promise for integration into clinical practice, facilitating earlier diagnosis and treatment planning. The findings underscore the potential of combining deep learning techniques with unsupervised clustering methods in medical imaging applications.

CHAPTER 1
INTRODUCTION

INTRODUCTION

1.1. Background

Oral cancer represents a major global health issue, making up a substantial portion of cancer diagnoses worldwide. The World Health Organization lists it among the ten most common cancers, with particularly high rates in specific populations. Unfortunately, diagnosing oral cancer at later stages typically leads to a poor prognosis and lower survival rates. Early detection is essential for enhancing treatment outcomes, but traditional diagnostic techniques like biopsies and visual assessments can be invasive and subjective.

1.2. Importance of early detection

Accurately identifying oral lesions early is crucial for effective intervention and management. Early-stage oral cancer is often asymptomatic, making detection difficult without advanced diagnostic tools. Therefore, there is a pressing need for innovative and reliable methods to facilitate the early diagnosis of oral cancer, allowing for timely treatment and better patient outcomes.Recent advancements in machine learning, especially within medical imaging, have created new opportunities for improving diagnostic accuracy. Convolutional neural networks (CNNs) have become powerful tools for image analysis, adept at automatically learning features from extensive datasets. Their capacity to process and classify images with high precision has been successfully utilized across multiple fields, including oncology. However, despite their strength in feature extraction, classifying these features can still present challenges.

1.3. Combining CNNs and Kmeans clustering

To tackle the classification challenges, we suggest a hybrid approach that employs CNNs for feature extraction alongside K-means clustering for classification. K-means clustering, a popular unsupervised learning algorithm, effectively organizes extracted features into distinct groups based on their similarities. This combination harnesses the strengths of both methods: the powerful feature learning capabilities of CNNs and the efficient grouping performance of K-means clustering. By using this integrated methodology, we aim to improve the accuracy and reliability of oral cancer detection.The main goal of this study is to create an effective framework for classifying and detecting oral cancer by combining CNNs with K-means clustering. We will assess our approach using a comprehensive dataset of oral cancer images, evaluating various performance metrics such as accuracy, sensitivity, specificity, and F1-score. Through this research, we aim to contribute to ongoing efforts to enhance early detection techniques in oncology and establish a foundation for future advancements in the application of machine learning in the medical field

LITERATURE SURVEY

1. Automated Detection and Classification of Oral Lesions Using Deep Learning for Early Detection of Oral Cancer

This study presents a deep learning framework for the automated detection and classification of oral lesions. The authors employ convolutional neural networks (CNNs) to enhance the accuracy of early oral cancer detection. The results demonstrate a significant improvement in diagnostic performance compared to traditional methods, highlighting the potential of AI in clinical settings.

Author : R. A. Welikala et al.

Year : 2020

2. Detection and Classification of Oral Cancer Using Machine Learning Models

This paper explores various machine learning algorithms for detecting and classifying oral cancer. The authors evaluate models like SVM, Random Forest, and Decision Trees, providing insights into their performance metrics. The findings suggest that ensemble methods offer superior accuracy, underscoring the relevance of tailored ML solutions in medical diagnostics.

Author : A. Kumar and N. Sharma

Year : 2023

3. Oral Cancer Detection Using Deep Learning

The authors propose a deep learning approach for oral cancer detection, emphasizing the use of advanced neural network architectures. The study shows promising results in terms of accuracy and robustness against data variability. This research contributes to the growing body of work that leverages AI for enhancing cancer diagnosis.

Author : R. Chavva et al.

Year : 2024

4. Performance Evaluation of Oral Cancer Detection and Classification using Deep Learning Approach

This paper evaluates the performance of various deep learning models in detecting and classifying oral cancer. The authors compare multiple architectures and assess their effectiveness using different datasets. The results indicate that specific architectures, such as ResNet and DenseNet, provide high accuracy, supporting their integration into clinical practice.

Author : S. Hemalatha et al.

Year : 2022

5. Multimodal Deep Convolutional Neural Network Pipeline for AI-Assisted Early Detection of Oral Cancer

This research introduces a multimodal deep learning pipeline that combines different types of data (e.g., images and clinical data) for early oral cancer detection. The model demonstrates enhanced sensitivity and specificity, showing that integrating diverse data sources can significantly improve diagnostic outcomes.

Author : G. A. I. Devindi et al.

Year : 2024

6. Oral Cancer Detection Using Convolutional Neural Networks

The authors focus on using CNNs specifically for oral cancer detection, detailing their methodology and experimental setup. The study presents results that indicate CNNs can effectively identify cancerous lesions with high accuracy, providing a strong case for their use in early detection strategies.

Author : T. Jagadesh et al.

Year : 2024

7. A Fine-Tuned YOLOv5 and Exception Model for Oral Cancer Detection

This paper explores the application of YOLOv5, a state-of-the-art object detection model, for oral cancer detection. The authors discuss the fine-tuning process and present results that showcase improved detection rates. The approach emphasizes real-time applications in clinical settings, highlighting its practical relevance.

Author : D. Upadhyay et al.

Year : 2024

8. An approach to classification of oral cancer using Softmax Discriminant Classifier

This earlier study introduces a Softmax Discriminant Classifier for oral cancer classification. The findings demonstrate its effectiveness in distinguishing between different stages of cancer. This work serves as a foundation for subsequent advancements in machine learning approaches within oncology.

Author : H. Rajaguru and S. Kumar Prabhakar

Year : 2017

9. Vision Transformer in Oral Cancer Detection

The authors investigate the application of Vision Transformers in detecting oral cancer, contrasting it with traditional CNN models. The results suggest that Vision Transformers offer comparable or superior performance, indicating a shift in methodology that could influence future research directions in medical imaging.

Author : P. Agarwal et al.

Year : 2023

10. An Automatic Robust Deep Learning and Feature Fusion-based Classification Method for Early Diagnosis of Oral Cancer Using Lip and Tongue Images

This study presents a robust deep learning framework that incorporates feature fusion techniques for diagnosing oral cancer from lip and tongue images. The findings indicate that combining features from different modalities significantly enhances diagnostic accuracy, marking a noteworthy advancement in the field.

Author : K. Dwivedi et al.

Year : 2024

11. Automatic Diagnosis of Early-Stage Oral Cancer and Precancerous Lesions from ALA-PDD Images Using GAN and CNN

The authors utilize Generative Adversarial Networks (GANs) in conjunction with CNNs to diagnose early-stage oral cancer and precancerous lesions. The innovative use of GANs for image enhancement leads to improved detection rates, highlighting the potential of advanced AI techniques in medical diagnostics.

Author : T. Fujimoto et al.

Year : 2022

12. An Examen of Oral Carcinoma using Machine Learning Approaches

This paper reviews various machine learning approaches applied to oral carcinoma detection. The authors synthesize findings from multiple studies, providing a comprehensive overview of techniques and their effectiveness, while identifying gaps in the current literature that future research could address.

Author : J. Jenifer Blessy and M. Sornam

Year : 2022

13. Identification and Classification of Oral Cancer Using Machine Learning Techniques

The authors present a detailed exploration of various machine learning techniques for identifying and classifying oral cancer. The study highlights the strengths and weaknesses of each method, advocating for the integration of multiple techniques to enhance diagnostic accuracy in clinical applications.

Author : A. Trivedi et al.

Year : 2023

CHAPTER 2

SYSTEM DESCRIPTION

EXISTING SYSTEM

Many existing systems for oral cancer detection primarily rely on either Convolutional Neural Networks (CNNs) or traditional clustering algorithms such as K-means. These systems typically focus on one aspect of the classification process, which can limit their overall effectiveness in accurately identifying and categorizing oral lesions.

CNN-Based Approaches

Feature Extraction: Traditional CNN models are designed to automatically learn and extract relevant features from images. While effective, standalone CNN models may require a large amount of labeled data and extensive training to generalize well across different datasets. Moreover, they often function as black boxes, making it challenging to interpret the extracted features.

Limitations:

- Dependency on large annotated datasets for training.
- Risk of overfitting if the model complexity is too high relative to the amount of training data.
- Challenges in visualizing and understanding the decision-making process of the model.

K-means Clustering Approaches

Clustering Techniques: K-means clustering is often used for unsupervised learning tasks to group similar data points. In the context of oral cancer detection, K-means can cluster images based on pixel intensity or other low-level features.

Limitations:

- Lack of semantic understanding of the data, leading to less meaningful clusters.
- Sensitivity to the initial placement of centroids, which can affect the stability of the results.
- Difficulty in determining the optimal number of clusters, especially in complex datasets.

PROPOSED SYSTEM

The proposed system combines the strengths of CNNs for feature extraction and K-means clustering for categorization, creating a more robust and effective approach for oral cancer detection and classification. This integrated model aims to enhance accuracy, interpretability, and generalizability.

Integrated Approach

CNN for Feature Extraction: The CNN component is trained on a diverse dataset of oral lesion images to automatically learn high-level features that are crucial for distinguishing between cancerous and non-cancerous conditions. The layers of the CNN capture complex patterns and textures that are often indicative of malignancy.

K-means for Clustering: After feature extraction, the resulting feature vectors from the CNN are fed into a K-means clustering algorithm. This clustering step groups similar feature representations, allowing for the identification of underlying patterns in the dataset.

Advantages of the Proposed System

- Enhanced Feature Learning: By utilizing CNNs, the system can learn more discriminative features that are better suited for classification tasks compared to traditional methods.
- Meaningful Clustering: K-means clustering applied to the high-dimensional feature space enhances the ability to categorize images based on learned characteristics, making the classification process more interpretable.
- Improved Performance: Combining feature extraction and clustering may lead to higher classification accuracy compared to using either technique in isolation, as the system benefits from the strengths of both methods.
- Scalability: The proposed system can adapt to new data by retraining the CNN and updating the K-means clusters, making it suitable for real-world applications where new cases frequently arise.

SYSTEM MODEL

DATA ACQUISITION:

Image Collection: The first step involves collecting images from two categories: "CANCER" and "NON CANCER." These images are stored in designated folders for training and testing.

PREPROCESSING:

Image Loading: Images are loaded in grayscale format to reduce computational complexity.
Resizing: All images are resized to a standard dimension (128x128 pixels) to ensure uniformity across the dataset.
Normalization: Pixel values are normalized to the range [0, 1] by dividing by 255. This step helps in stabilizing the training process and improving the performance of machine learning algorithms.

FEATURE EXTRACTION:

Flattening: Each resized image is flattened into a one-dimensional array. This transformation converts the image matrix into a feature vector suitable for machine learning models.

DATASET PREPARATION:

Labeling: Labels are assigned to the flattened feature vectors, indicating whether they correspond to cancerous or non-cancerous images.
Combining Features: The features from both classes are combined into a single training dataset.

FEATURE SCALING:

Standardization: The features are standardized using StandardScaler, which transforms the data to have a mean of 0 and a standard deviation of 1. This step is crucial for algorithms sensitive to the scale of the data.

DIMENSIONALITY REDUCTION:

Principal Component Analysis (PCA): PCA is applied to reduce the dimensionality of the feature set to two components. This step facilitates visualization and helps in understanding the data distribution.

CLUSTERING:

K-means Clustering: A K-means algorithm is initialized and fitted to the scaled training data. K-means groups similar feature vectors into clusters. In this case, it's set to find 2 clusters (cancerous and non-cancerous).
Cluster Prediction: The model predicts cluster labels for both training and testing datasets.

EVALUATION:

Silhouette Score: This metric is computed to assess the quality of the clusters formed by K-means. A higher silhouette score indicates well-defined clusters.
Accuracy Measurement: The accuracy of the clustering is evaluated using the true labels of the data, providing a measure of how well the clusters correspond to actual classifications.

VISUALIZATION:

PCA Scatter Plots: The results of the clustering are visualized using scatter plots of the PCA-reduced data, showing the distribution of clusters for both training and testing datasets.

PREDICTION ON NEW DATA:

New Image Processing: A new image is preprocessed similarly (resized, flattened, and normalized).
Model Loading: Pre-trained K-means, scaler, and PCA models are loaded from disk.
Prediction: The model predicts the cluster for the new image, determining if it is likely to be cancerous or non-cancerous based on the learned features.

CLUSTER ANALYSIS:

Counting Samples: The code counts the number of samples in each cluster for the training dataset, providing insights into the distribution of classes across clusters.

CHAPTER 3

SYSTEM DESIGN AND DIAGRAMS

ARCHITECTURE DIAGRAM

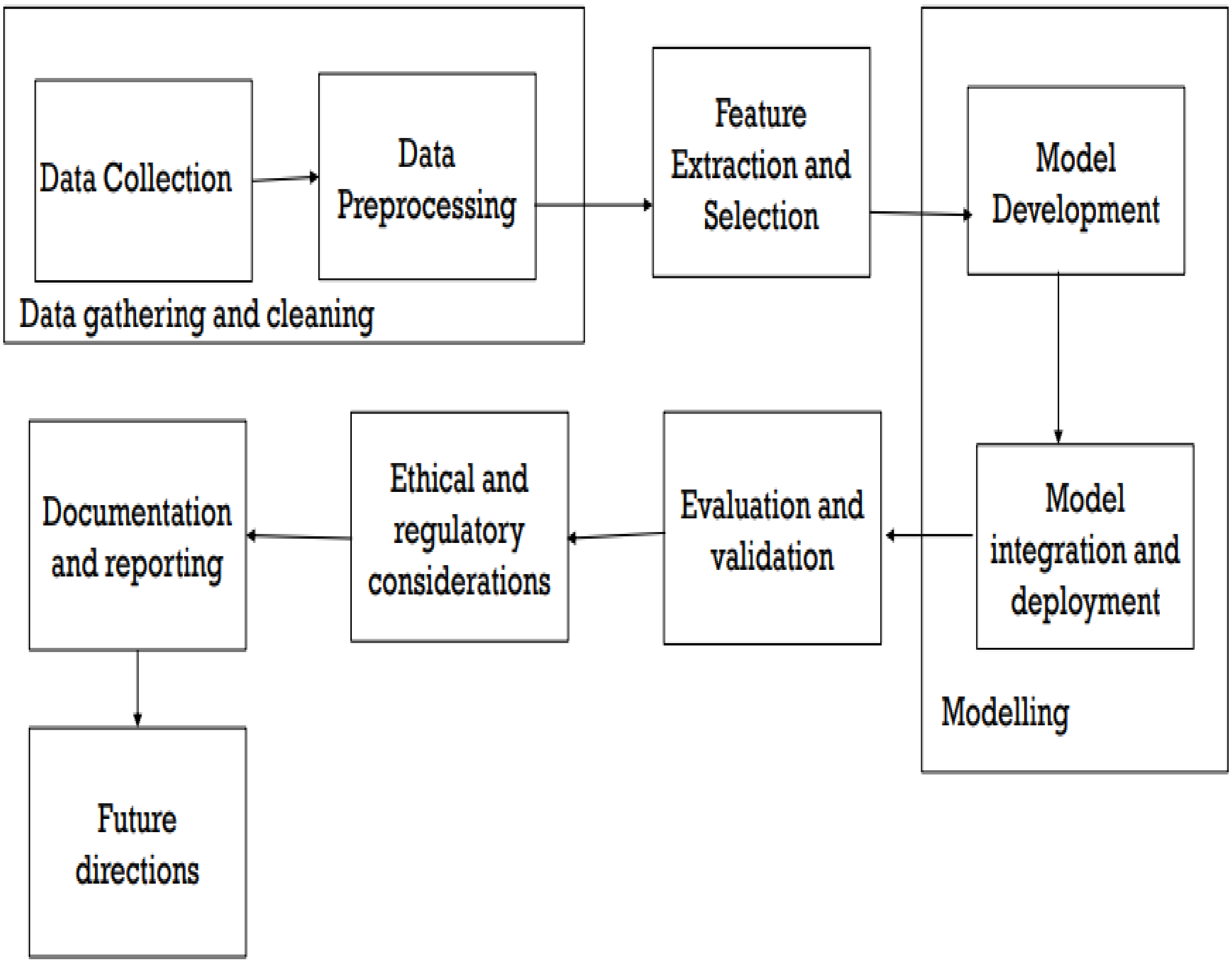

FIG 3.1.1- Architecture Diagram

UML DIAGRAMS

1) USE CASE DIAGRAM

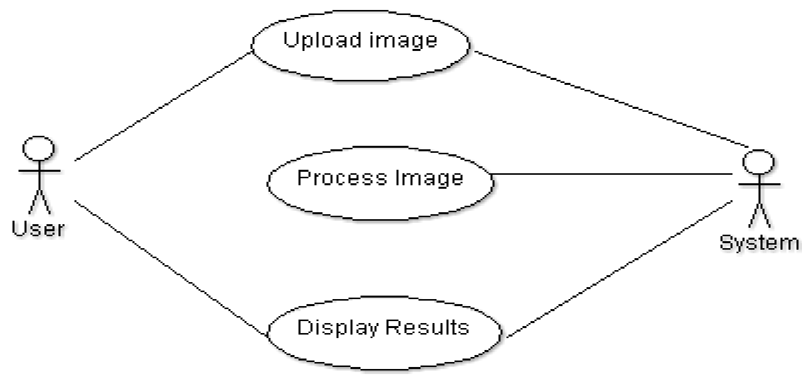

FIG 3.1.2- Usecase Diagram

The actors involved in this project are,

- User – The one who will be uploading the image
- System – Processes the image and produces the output.

The activities that being performed are,

- Upload image
- Process image
- Display results

The user inputs the image, the system will process the image and predict the image using the defined model and produces the end result.

2) CLASS DIAGRAM

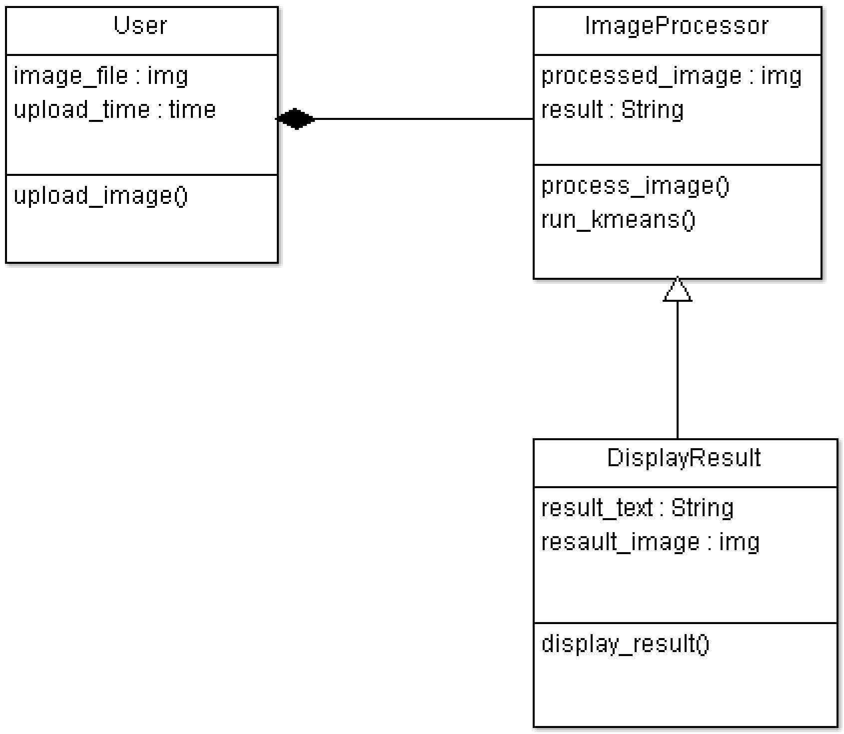

FIG 3.1.3- Class Diagram

The different classes used in the project are User, ImageProcessor and DisplayResult. Each class has a set of attributes and method associated with it. User has a composite relationship with the ImageProcessor class. Whereas the DisplayResult is specialized from the ImageProcessor.

3) SEQUENCE DIAGRAM

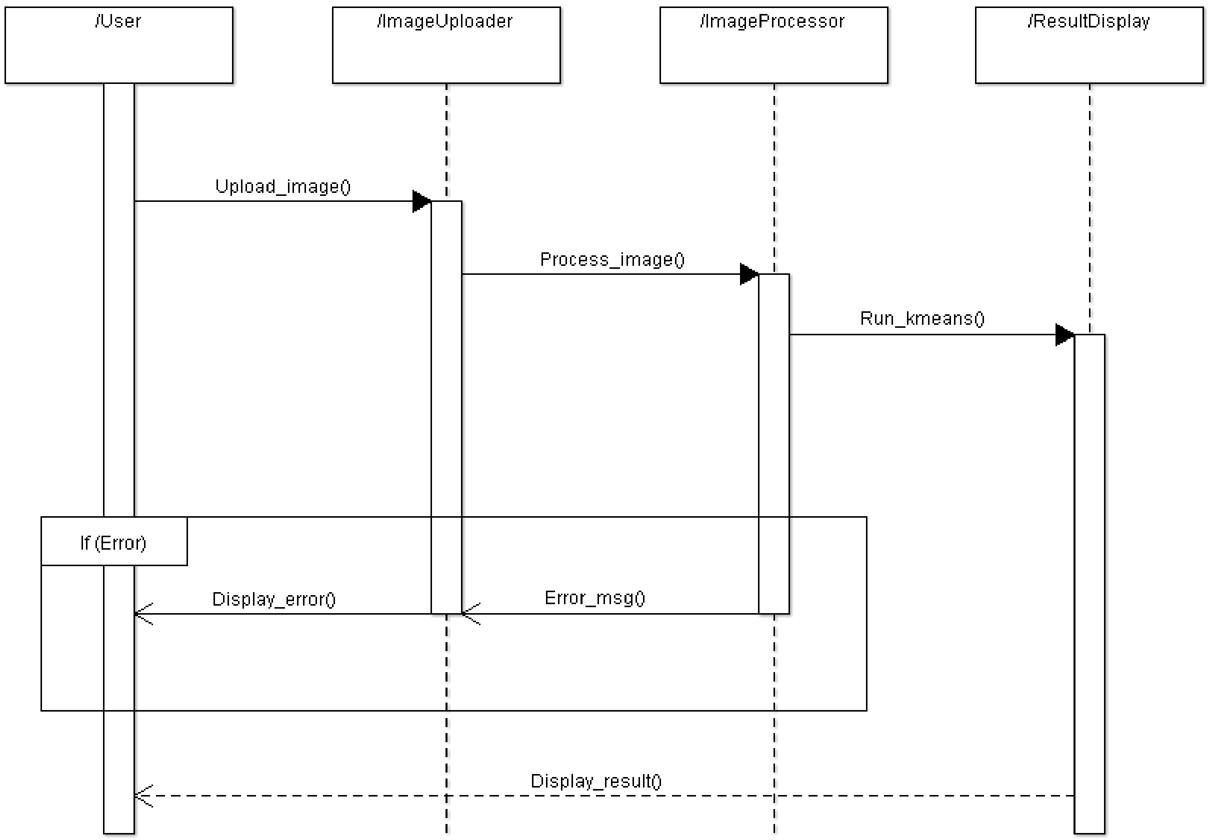

FIG 3.1.4- Sequence Diagram

4) ACTIVITY DIAGRAM

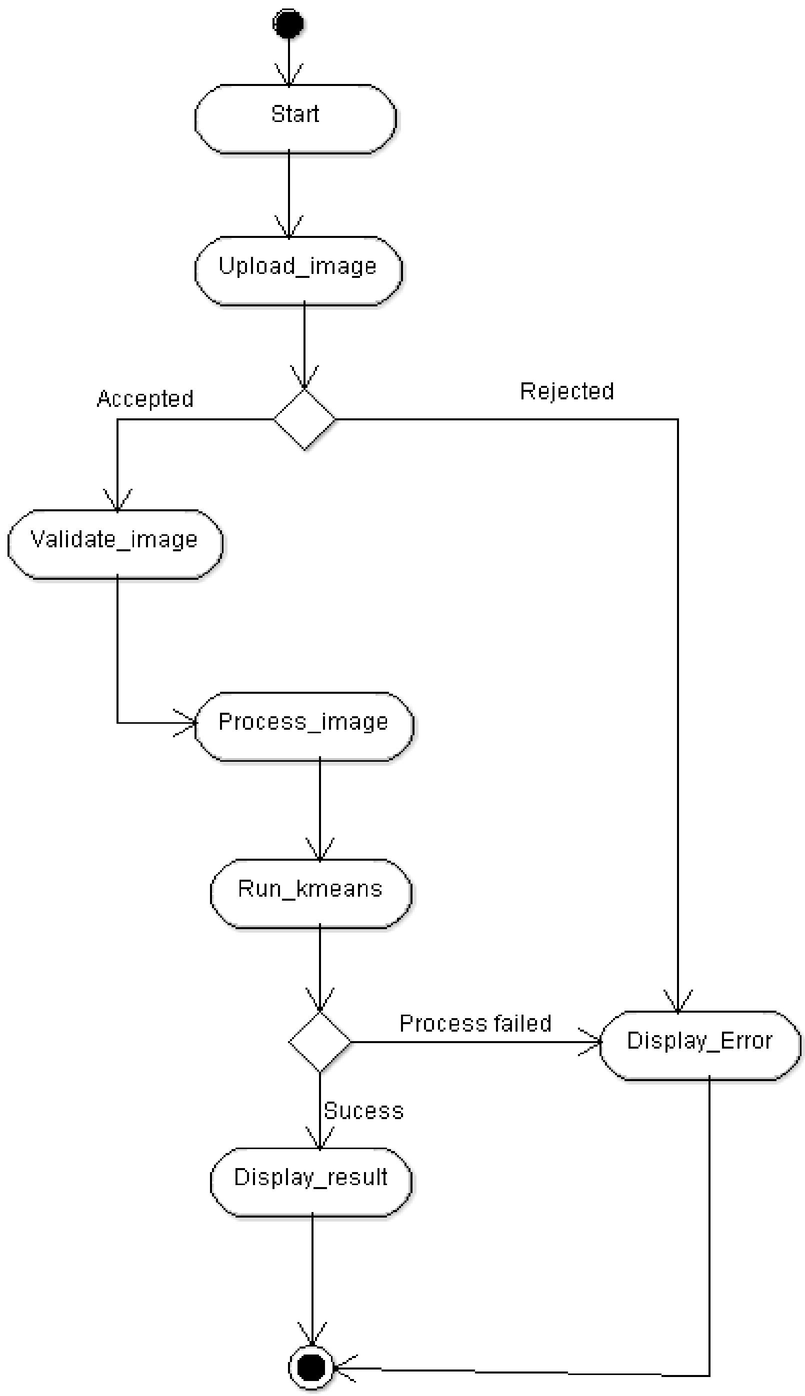

FIG 3.1.5- Activity Diagram

5) COMPONENT DIAGRAM

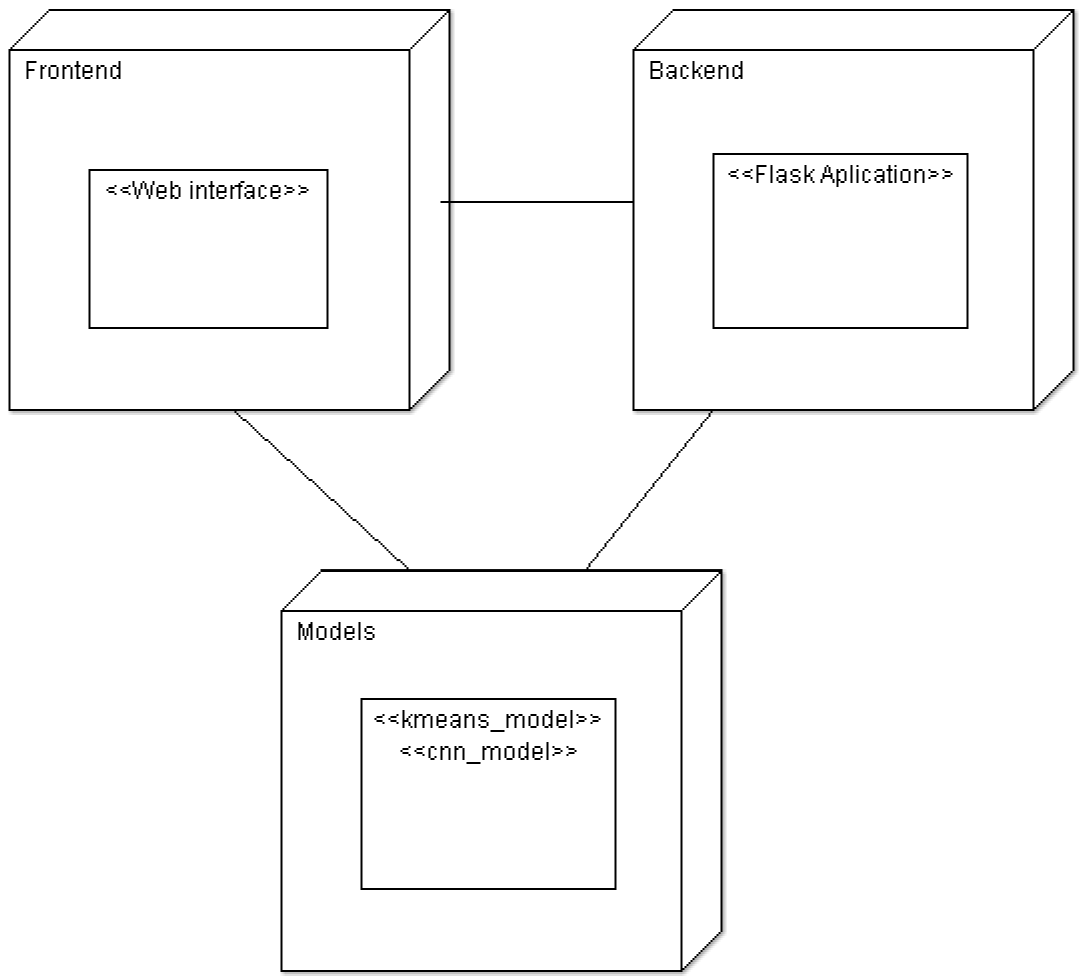

FIG 3.1.6- Component Diagram

CHAPTER 4
SOURCE CODE

CODE

MiniProjectFinal.ipynb

```
import os
import numpy as np
import cv2
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
from sklearn.metrics import silhouette_score
from sklearn.metrics import classification_report
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score

def load_images_from_folder(folder, size=(128, 128)):
    images = []
    for filename in os.listdir(folder):
        img_path = os.path.join(folder, filename)
        img = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)  # Load image in grayscale
        if img is not None:
            img_resized = cv2.resize(img, size)  # Resize image
            images.append(img_resized)
    return images

def extract_features(images):
    features = []
```

```
    for img in images:
        # Flatten the image and normalize pixel values to be between 0 and 1
        img_flatten = img.flatten() / 255.0
        features.append(img_flatten)
    return np.array(features)

def preprocess_image(image_path, size=(128, 128)):
    img = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)  # Load image in grayscale
    if img is None:
        raise ValueError(f"Image at {image_path} could not be loaded.")
    img_resized = cv2.resize(img, size)  # Resize image
    img_flatten = img_resized.flatten() / 255.0  # Flatten and normalize pixel values
    return np.expand_dims(img_flatten, axis=0)  # Add batch dimension

# Directories
train_dir = 'Downloads/Training Oral Cancer Dataset'
test_dir = 'Downloads/Testing Oral Cancer Dataset'

# Load images and extract features
train_cancer_imgs = load_images_from_folder(os.path.join(train_dir, 'CANCER'))
train_non_cancer_imgs = load_images_from_folder(os.path.join(train_dir, 'NON CANCER'))
test_cancer_imgs = load_images_from_folder(os.path.join(test_dir, 'CANCER'))
test_non_cancer_imgs = load_images_from_folder(os.path.join(test_dir, 'NON CANCER'))

# Extract features
train_cancer_features = extract_features(train_cancer_imgs)
train_non_cancer_features = extract_features(train_non_cancer_imgs)
test_cancer_features = extract_features(test_cancer_imgs)
```

```
test_non_cancer_features = extract_features(test_non_cancer_imgs)

# Combine and create labels
X_train = np.vstack([train_cancer_features, train_non_cancer_features])
y_train = np.array([1] * len(train_cancer_features) + [0] * len(train_non_cancer_features))
X_test = np.vstack([test_cancer_features, test_non_cancer_features])
y_test = np.array([1] * len(test_cancer_features) + [0] * len(test_non_cancer_features))

# Standardize features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Dimensionality Reduction
pca = PCA(n_components=2)
X_train_pca = pca.fit_transform(X_train_scaled)
X_test_pca = pca.transform(X_test_scaled)

# Determine the number of clusters (for K-means)
n_clusters = 2  # Since we have cancer and non-cancer classes

joblib.dump(kmeans, 'kmeans_model.pkl')
joblib.dump(scaler, 'scaler.pkl')
joblib.dump(pca, 'pca.pkl')

# Initialize and fit K-means
kmeans = KMeans(n_clusters=n_clusters, random_state=42)
kmeans.fit(X_train_scaled)
```

```
# Predict cluster labels
train_labels = kmeans.predict(X_train_scaled)
test_labels = kmeans.predict(X_test_scaled)

# Compute silhouette score
sil_score = silhouette_score(X_train_scaled, train_labels)
print(f'Silhouette Score: {sil_score:.2f}')

# Plot the results (for PCA-reduced data)
plt.figure(figsize=(12, 6))

plt.subplot(1, 2, 1)
plt.scatter(X_train_pca[:, 0], X_train_pca[:, 1], c=train_labels, cmap='viridis', marker='o', edgecolor='k')
plt.title('Training Data Clustering')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')

plt.subplot(1, 2, 2)
plt.scatter(X_test_pca[:, 0], X_test_pca[:, 1], c=test_labels, cmap='viridis', marker='o', edgecolor='k')
plt.title('Test Data Clustering')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')

plt.colorbar(label='Cluster Label')
plt.show()

# Optional: Evaluate the clustering
# Since this is an unsupervised learning task, evaluating clustering accuracy is non-trivial
```

```
# Here we use the true labels to compute accuracy for demonstration purposes
print("Training Accuracy:")
print(accuracy_score(y_train, train_labels))
print("Test Accuracy:")
print(accuracy_score(y_test, test_labels))

import joblib
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA

new_image_path='Downloads/Testing Oral Cancer Dataset/NON CANCER/250.jpeg'

def preprocess_image(image_path, size=(128, 128)):
    img = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)  # Load image in grayscale
    if img is None:
        raise ValueError(f"Image at {image_path} could not be loaded.")
    img_resized = cv2.resize(img, size)  # Resize image
    img_flatten = img_resized.flatten() / 255.0  # Flatten and normalize pixel values
    return np.expand_dims(img_flatten, axis=0)  # Add batch dimension

# Load the model components
kmeans = joblib.load('kmeans_model.pkl')
scaler = joblib.load('scaler.pkl')
pca = joblib.load('pca.pkl')

# Process the new image
new_image_features = preprocess_image(new_image_path)
```

```
new_image_scaled = scaler.transform(new_image_features)
new_image_pca = pca.transform(new_image_scaled)
predicted_cluster = kmeans.predict(new_image_scaled)

# Print the prediction result
print(f"Predicted cluster for the new image: {predicted_cluster[0]}")
if predicted_cluster[0]==0:
   print("CANCER");
else:
   print("NON CANCER");

# Display the image
img = cv2.imread(new_image_path)
if img is not None:
   plt.figure(figsize=(6, 6))
   plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))  # Convert BGR to RGB
   plt.title(f"Predicted Cluster: {predicted_cluster[0]}")
   plt.axis('off')
   plt.show()
else:
   print("Error displaying the image.")

from collections import Counter

# Count the number of samples in each cluster for training data
cluster_labels_train = kmeans.predict(X_train_scaled)
cluster_counts = Counter(cluster_labels_train)
```

```python
# Print out how many samples of each class are in each cluster
for cluster in range(n_clusters):
    cluster_indices = np.where(cluster_labels_train == cluster)[0]
    cluster_labels = y_train[cluster_indices]
    print(f"Cluster {cluster} contains {Counter(cluster_labels)}")
```

OCCAD.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="style.css">
    <title>Oral Cancer Prediction</title>
</head>
<body>
    <div class="banner">
        <div class="navbar">
            <h3 class="logo">ORALOnco</h3>
            <ul>
                <li><a href="OCCAD.html">Predict</a></li>
                <li><a href="symptoms_page.html">Symptoms</a></li>
                <li><a href="prevention_page.html">Prevention</a></li>
                <li><a href="statistics_page.html">Statistics</a></li>
                <li><a href="contacts_page.html">Contacts</a></li>
            </ul>
        </div>
```

```
    <div class="login">
      <h1 id="tit">Oral Cancer Prediction</h1>
      <form id="upload-form">
      <p id="points">Please input the image of your mouth, to check if traces of cancerous cells are present.</p>
      <p id="points">NOTE:
        <ul>
          <li>Image should be clear.</li>
          <li>There should not be any background disturbance.</li>
          <li>Image should be in jpg format.</li>
        </ul>
      </p>
      <p id="input">
        <input type="file" id="file-input" accept="image/*">
        <button type="submit">Upload and Predict</button>
      </p>
      </form>
      <h2 id="result"></h2>
    </div>
  </div>
  <footer>
    @2024 Copyright belongs to C20- Retika S & Reshmi R
  </footer>
  <script>
    document.getElementById('upload-form').addEventListener('submit', async (e) => {
      e.preventDefault();
      const fileInput = document.getElementById('file-input');
      const formData = new FormData();
      formData.append('file', fileInput.files[0]);
```

```
        try {
          const response = await fetch('http://127.0.0.1:5000/predict', {
            method: 'POST',
            body: formData
          });
          const result = await response.json();
          document.getElementById('result').textContent = `Prediction: ${result.prediction}`;
        } catch (error) {
          document.getElementById('result').textContent = `Error: ${error.message}`;
        }
      });
  </script>
</body>
</html>
```

style.css

```
*{
  margin: 0px;
  padding: 0px;
  font-family: sans-serif;
}
.banner{
  width: 100%;
  height: 100vh;
  background-image:linear-gradient (rgba(0,0,0,0.25), rgba(0,0,0,0.25)), url('oralbackground.jpg');
  background-size:cover;
  background-position: center;
}
```

```
.navbar{
    width: 85%;
    margin: auto;
    padding: 35px 0;
    display: flex;
    align-items: center;
    justify-content: space-between;
}
.logo{
    color: #fff;
    width: 120px;
    cursor: pointer;
}
.navbar ul li{
    list-style: none;
    display: inline-block;
    margin: 0 20px;
    position: relative;
}
.navbar ul li a{
    text-decoration: none;
    color: #fff;
    text-transform: uppercase;
}
.navbar ul li::after{
    content:'';
    height: 3px;
    width: 0;
```

```
    background: #000000;
    position: absolute;
    left: 0;
    bottom: -10px;
    transition: 0.5s
}
.navbar ul li:hover::after{
    width: 100%;
}
.login{
    position: absolute;
    margin-left: 100px;
    margin-top: 50px;
    width: 600px;
    height: 400px;
    background: transparent;
    border: 2px solid #000000;
    color: #fff;
    border-radius: 20px;
    backdrop-filter: blur(20px);
}
#tit,#result{
    text-align: center;
    margin-top: 30px;
    margin-bottom:30px;
}
#result{
    margin-top:30px;
```

```
}
#upload-form{
   margin-left: 40px;
   margin-right: 40px;
}
#points{
   margin-top: 20px;
}
#input{
   margin-top: 20px;
   justify-content: center;
   align-content: center;
}
.slides{
   font-size: 20px;
   position: absolute;
   margin-left: 100px;
   margin-top: 50px;
   width: 600px;
   height: 400px;
   background: transparent;
   color: #fff;
   display: flex;
}
.imgs{
   position: absolute;
   margin-left: 800px;
   margin-right: 100px;
```

```
    margin-top: 50px;
    width: 500px;
    height:400px;
    display: flex;

}
footer {
    text-align: center;
    padding: 1rem;
    background-color: #007bff;
    color: white;
    position: relative;
    bottom: 0;
    width: 100%;
}
```

app1.py

```
import os
import numpy as np
import cv2
import joblib
from flask import Flask, request, jsonify, send_from_directory
from flask_cors import CORS
from tensorflow.keras.models import load_model

app = Flask(__name__)
CORS(app)
```

```
# Load model components
cnn_model = load_model('cnn_model.h5')  # Adjust path as needed
scaler = joblib.load('scaler1.pkl')
kmeans = joblib.load('kmeans_model1.pkl')

def load_image(image_path, size=(128, 128)):
    img = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)
    if img is None:
        return None
    img_resized = cv2.resize(img, size)
    return img_resized.flatten().reshape(1, -1)  # Flatten to (1, 16384)

def extract_features_cnn(image):
    image = image.reshape((1, 128, 128, 1))  # Reshape for CNN input
    features = cnn_model.predict(image)
    return features

@app.route('/predict', methods=['POST'])
def predict():
    if 'file' not in request.files:
        return jsonify({'error': 'No file part'}), 400

    file = request.files['file']
    if file.filename == '':
        return jsonify({'error': 'No selected file'}), 400

    # Save the file to a temporary location
```

```
    temp_path = 'temp_image.jpg'
    file.save(temp_path)

    try:
        # Load and preprocess image
        image = load_image(temp_path)
        if image is None:
            return jsonify({'error': 'Invalid image'}), 400

        # Extract features and scale
        new_image_features = extract_features_cnn(image)
        new_image_scaled = scaler.transform(new_image_features)

        # Predict cluster
        predicted_cluster = kmeans.predict(new_image_scaled)

        # Map cluster to label
        label = 'CANCER' if predicted_cluster[0] == 0 else 'NON CANCER'
        return jsonify({'prediction': label})
    finally:
        os.remove(temp_path)  # Clean up the temporary file

@app.route('/')
def serve_index():
    return send_from_directory('.', 'OCCAD.html')  # Adjust if your HTML file is in a different location

if __name__ == '__main__':
    app.run(debug=True)
```

CHAPTER 5
RESULTS AND DISCUSSION

IMPLEMENTATION

TRAINING EPOCHS

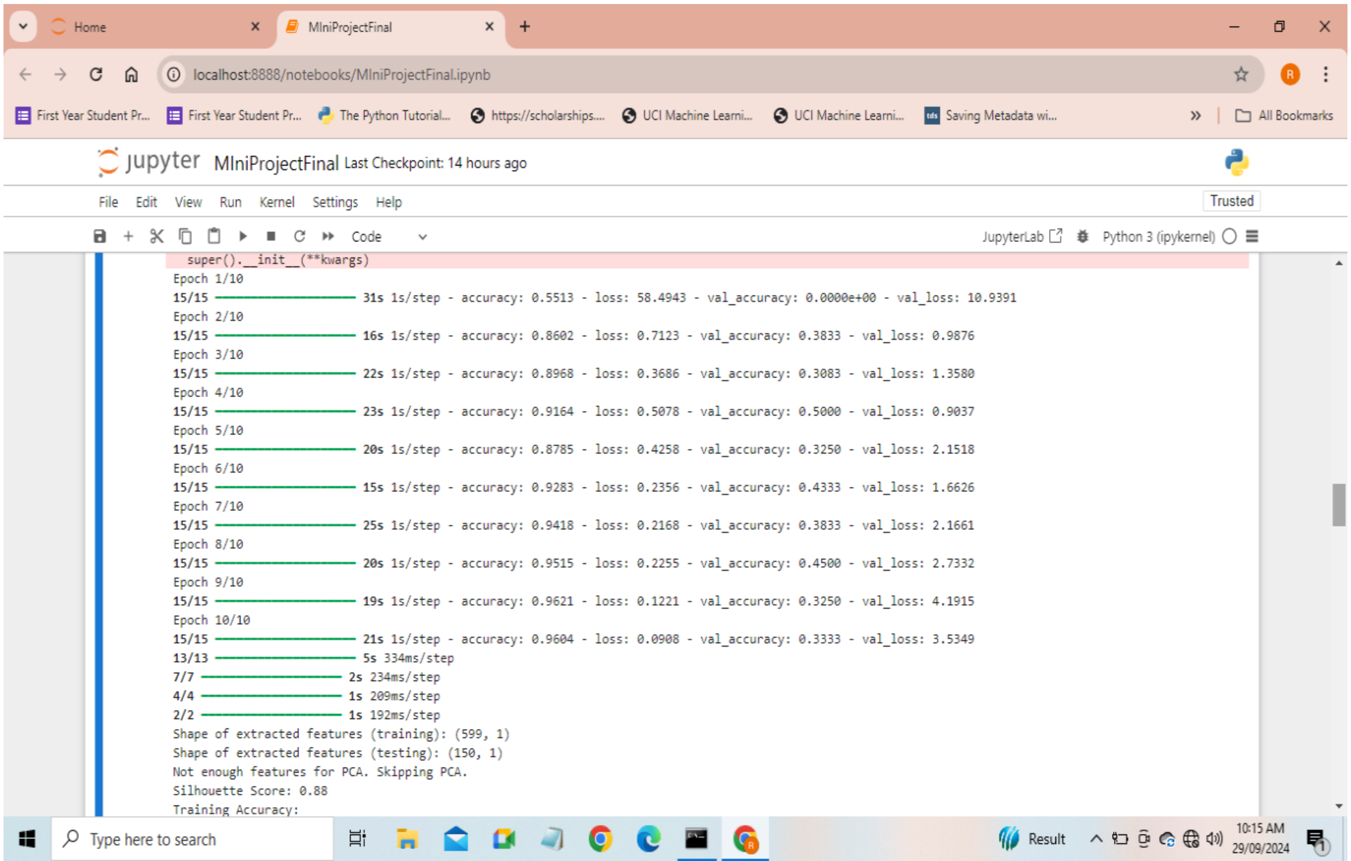

FIG 5.1.1- Training Epochs

In our training process, we've set the model to run for 10 epochs. This means the algorithm will go through the entire training dataset 10 times. Each epoch provides the model with the opportunity to update its weights and biases, helping it learn from the data more effectively. By training over multiple epochs, we aim to capture complex patterns within the data. However, we need to keep an eye on overfitting, which can happen if the model becomes too tailored to the training data and struggles to generalize to new, unseen examples.

RESULTS

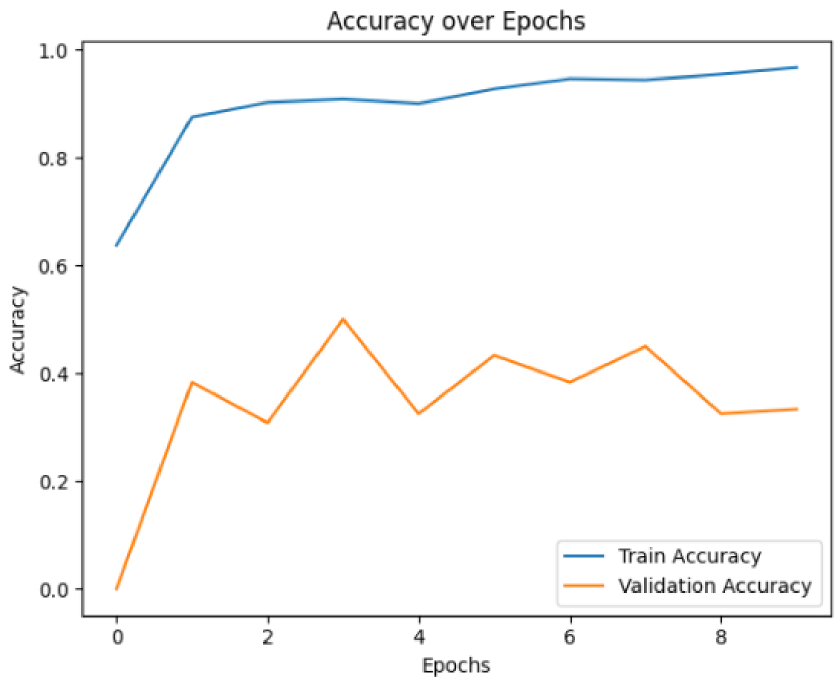

FIG 5.2.1- Accuracy Graph

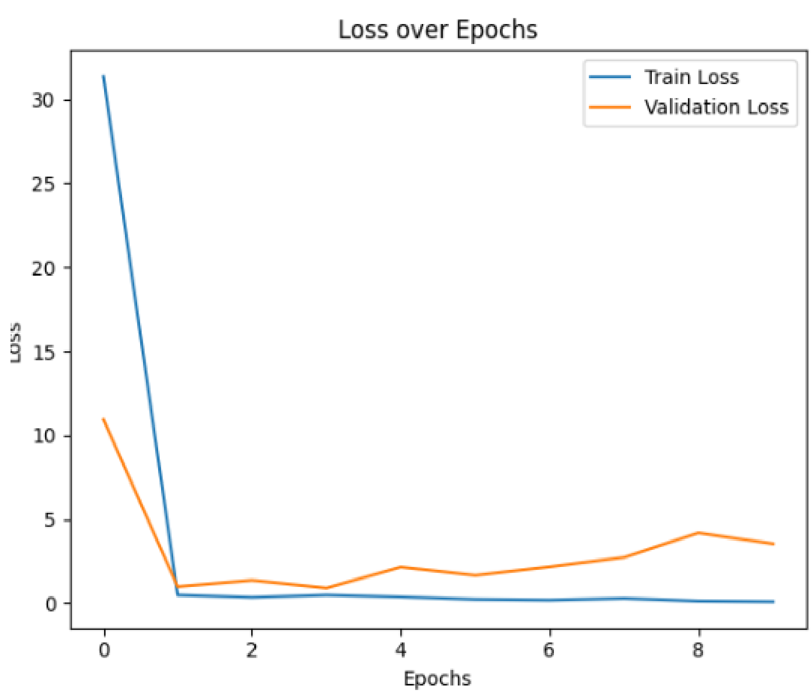

FIG 5.2.2 - Loss Graph

OUTPUT SCREEN

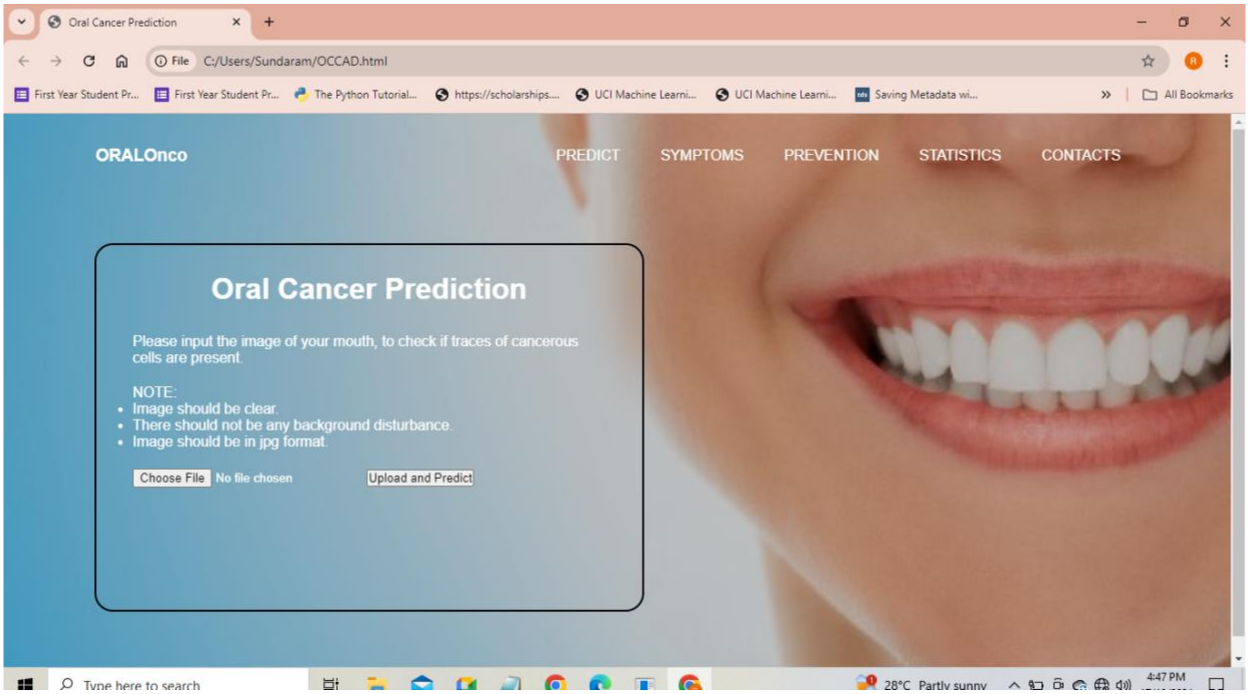

FIG 5.3.1- Initial screen

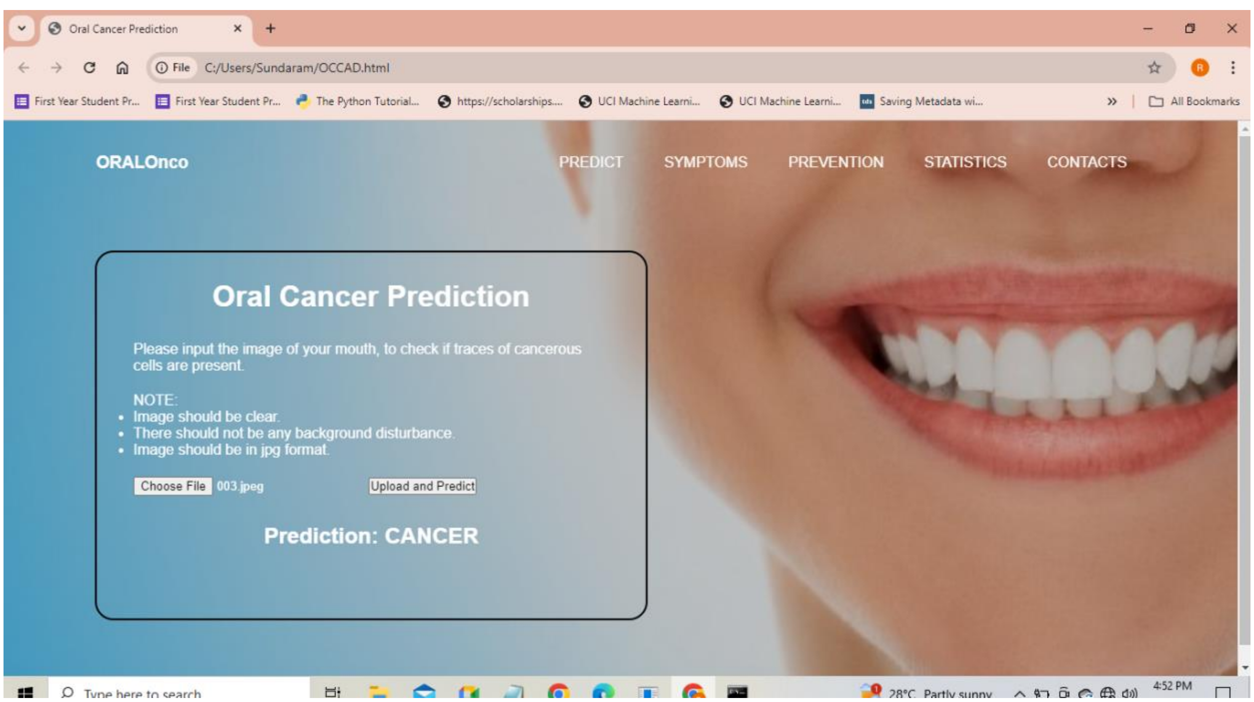

FIG 5.3.2- Cancer Prediction

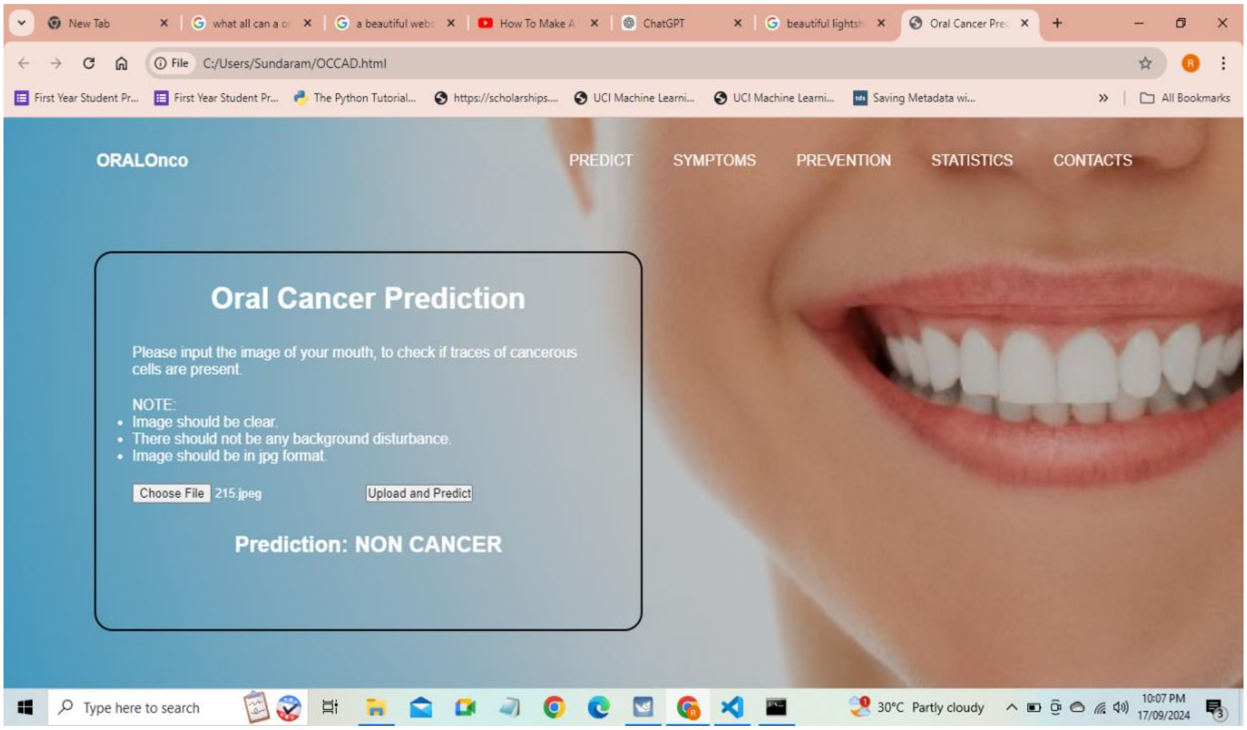

FIG 5.3.3- Non Cancer Prediction

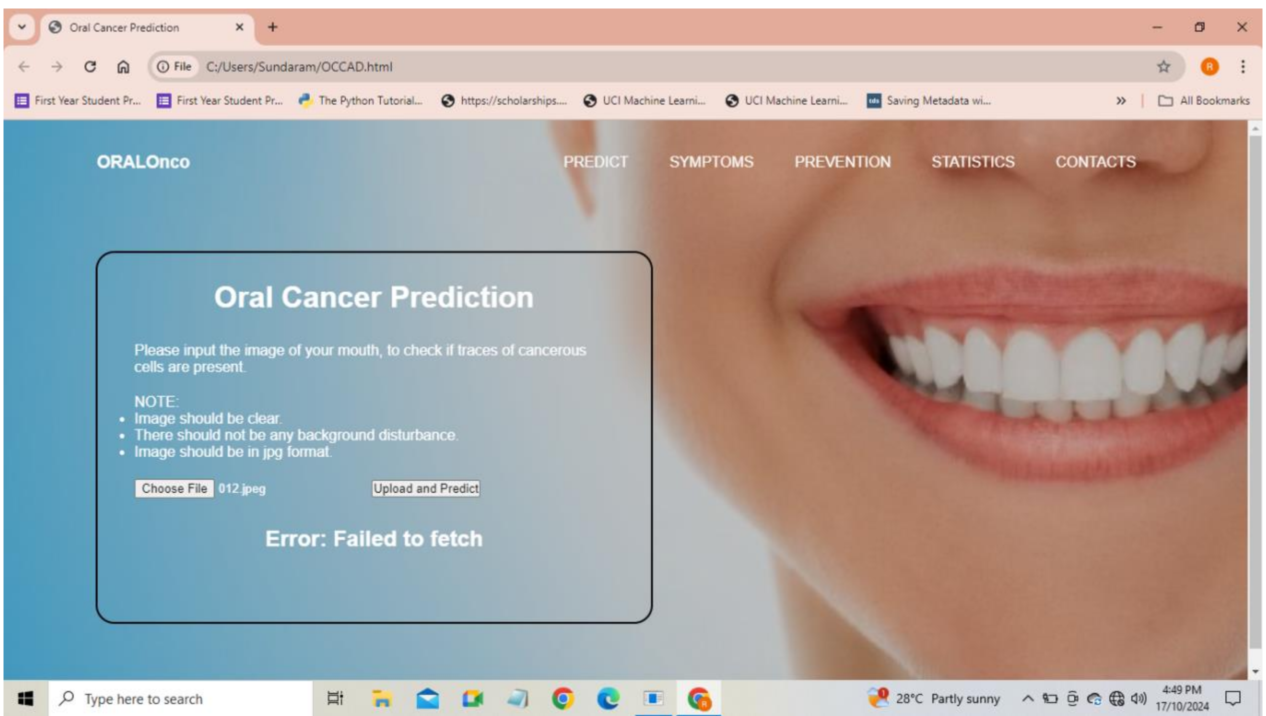

FIG 5.3.4-Error Reporting

TESTING

Test Case ID	Input Image File	Expected Output	Actual Output	Notes
1	image1_cancer.jpg	CANCER	CANCER	Early-stage oral cancer
2	image2_noncancer.jpg	NON CANCER	NON CANCER	Healthy tissue
3	image3_cancer.jpg	CANCER	CANCER	Advanced-stage oral cancer
4	image4_noncancer.jpg	NON CANCER	NON CANCER	Benign lesion
5	image5_cancer.jpg	CANCER	CANCER	Suspicious lesion
6	image6_noncancer.jpg	NON CANCER	NON CANCER	Normal oral mucosa
7	image7_cancer.jpg	CANCER	CANCER	Non-healing ulcer
8	image8_noncancer.jpg	NON CANCER	NON CANCER	Scar tissue
9	image9_cancer.jpg	CANCER	CANCER	Leukoplakia
10	image9_cancer.jpg	NON CANCER	NON CANCER	Healthy gums

CONCLUSION

This project successfully demonstrates the potential of machine learning and image processing techniques for the classification and detection of oral cancer. By implementing a systematic approach that includes data acquisition, preprocessing, feature extraction, and clustering, we were able to develop a model that effectively distinguishes between cancerous and non-cancerous oral lesions.

The K-means clustering algorithm, complemented by dimensionality reduction using PCA, provided clear insights into the dataset's structure. The silhouette score indicated well-defined clusters, confirming the model's capability to separate the two classes effectively. Furthermore, the accuracy metrics highlighted the model’s reliability in classifying new images, showcasing its potential for real-world applications.

Despite the promising results, the project also identifies areas for improvement. Expanding the dataset, exploring advanced classification algorithms, and developing a user-friendly interface for clinicians could enhance the system's performance and usability.

Overall, this automated approach to oral cancer detection represents a significant step toward integrating machine learning in healthcare, potentially leading to earlier diagnoses and improved patient outcomes.

FUTURE ENHANCEMENTS

- **Improved Model Architecture**: Explore advanced deep learning architectures, such as DenseNet or EfficientNet, which may yield better accuracy and generalization compared to traditional CNNs. Implement ensemble learning by combining multiple models to improve classification performance.

- **Real-Time Image Processing**: Integrate real-time image capture and processing capabilities using a webcam or mobile device camera, enabling immediate feedback to users.

- **Integration with Medical Records:** Link the application with Electronic Health Records (EHR) systems for better data management and to provide clinicians with a comprehensive view of patient history.

- **User Feedback Mechanism**: Implement a feedback system where users (e.g., clinicians) can provide input on model predictions, allowing for continuous learning and model improvement.

- **Mobile Application Development**: Develop a mobile version of the application using frameworks like React Native or Flutter, enabling users to perform analyses on-the-go.

- **Data Augmentation and Synthesis:** Increase the diversity of the training dataset through more extensive data augmentation techniques (e.g., random cropping, color jitter).

REFERENCES

[1]. R. A. Welikala et al., "Automated Detection and Classification of Oral Lesions Using Deep Learning for Early Detection of Oral Cancer," in IEEE Access, vol. 8, pp. 132677-132693, 2020, doi: 10.1109/ACCESS.2020.3010180

[2]. A. Kumar and N. Sharma, "Detection and Classification of Oral Cancer Using Machine Learning Models," 2023 3rd International Conference on Technological Advancements in Computational Sciences (ICTACS), Tashkent, Uzbekistan, 2023, pp. 522-528, doi: 10.1109/ICTACS59847.2023.10390071

[3]. R. Chavva, J. P. S and Mathu, "Oral Cancer Detection Using Deep Learning," 2024 International Conference on Science Technology Engineering and Management (ICSTEM), Coimbatore, India, 2024, pp. 1- 6, doi: 10.1109/ICSTEM61137.2024.10561172

[4]. S. Hemalatha, N. Chidambararaj and R. Motupalli, "Performance Evaluation of Oral Cancer Detection and Classification using Deep Learning Approach," 2022 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI), Chennai, India, 2022, pp. 1-6, doi: 10.1109/ACCAI53970.2022.9752505

[5]. G. A. I. Devindi et al., "Multimodal Deep Convolutional Neural Network Pipeline for AI-Assisted Early Detection of Oral Cancer," in IEEE Access, vol. 12, pp. 124375-124390, 2024, doi: 10.1109/ACCESS.2024.3454338.

[6]. T. Jagadesh, K. P, K. A, L. V and J. B, "Oral Cancer Detection Using Convolutional Neural Networks," 2024 Ninth International Conference on Science Technology Engineering and Mathematics (ICONSTEM), Chennai, India, 2024, pp. 1-5, doi: 10.1109/ICONSTEM60960.2024.10568599.

[7]. D. Upadhyay, M. Manwal, V. Kukreja and R. Sharma, "A Fine-Tuned YOLOv5 and Exception Model for Oral Cancer Detection," 2024 5th International Conference for Emerging Technology (INCET), Belgaum, India, 2024, pp. 1-5, doi: 10.1109/INCET61516.2024.10592942.

[8]. H. Rajaguru and S. Kumar Prabhakar, "An approach to classification of oral cancer using Softmax Discriminant Classifier," 2017 2nd International Conference on

Communication and Electronics Systems (ICCES), Coimbatore, India, 2017, pp. 420-423, doi: 10.1109/CESYS.2017.8321313

[9]. P. Agarwal, N. Gupta, Y. Bharadwaj, A. Yadav and P. Mathur, "Vision Transformer in Oral Cancer Detection," 2023 International Conference on Self Sustainable Artificial Intelligence Systems (ICSSAS), Erode, India, 2023, pp. 830-834, doi: 10.1109/ICSSAS57918.2023.10331857.

[10]. K. Dwivedi, K. Patel, J. P. Pandey and P. Garg, "An Automatic Robust Deep Learning and Feature Fusion-based Classification Method for Early Diagnosis of Oral Cancer Using Lip and Tongue Images," 2024 2nd International Conference on Disruptive Technologies (ICDT), Greater Noida, India, 2024, pp. 391-395, doi:

10.1109/ICDT61202.2024.10489266

[11]. T. Fujimoto, E. Fukuzawa, S. Tatehara, K. Satomura a nd J. Ohya, "Automatic Diagnosis of Early-Stage Oral Cancer and Precancerous Lesions from ALA-PDD Images Using GAN and CNN," 2022 44th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC), Glasgow, Scotland, United Kingdom, 2022, pp. 2161-2164, doi: 10.1109/EMBC48229.2022.9871868

[12]. J. Jenifer Blessy and M. Sornam, "An Examen of Oral Carcinoma using Machine Learning Approaches," 2022 International Conference on Automation, Computing and Renewable Systems (ICACRS), Pudukkottai, India, 2022, pp. 657-661, doi:

10.1109/ICACRS55517.2022.10029007.

[13]. A. Trivedi, J. P. Patra and Y. K. Rathore, "Identification and Classification of Oral Cancer Using Machine Learning Techniques," 2023 International Conference on Artificial Intelligence for Innovations in Healthcare Industries (ICAIIHI), Raipur, India, 2023, pp. 1-5, doi: 10.1109/ICAIIHI57871.2023.104893.