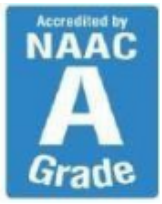
SAVEETHA

INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES

(Declared as Deemed to be University under Section 3 of UGC Act 1956)



SKIN CANCER DETECTION WITH CNN USING OPENCV

Submitted

By

T.Thanusree[192224193]

I. Sanjana Reddy[192224168]

Guided by

Dr.Manjula

Department of Logic Net

ABSTRACT

Skin Cancer Detection using Convolutional Neural Networks (CNN) and OpenCV represents a transformative approach in dermatological diagnostics for the early identification of skin cancer. This research explores the historical evolution, current landscape, and future prospects of leveraging cutting-edge technologies, particularly the integration of deep learning techniques, to enhance the accuracy of skin cancer detection. Employing a comprehensive dataset sourced from Kaggle, the project focuses on the implementation of CNN and OpenCV for sophisticated image processing, showcasing their potential to revolutionize dermatological diagnostics.

The abstracted framework serves as a testament to the project's commitment to advancing healthcare through the synergy of artificial intelligence and medical diagnostics. By meticulously analyzing skin lesion images, the research aims to contribute to the development of an efficient model for early detection, ultimately leading to improved patient outcomes. This study underscores the transformative impact of emerging technologies on healthcare practices, paving the way for a new era in skin cancer detection and patient Care.

Keywords: Skin Cancer Detection, Convolutional Neural Networks, OpenCV, Kaggle Dataset, Deep Learning, Image Processing,Dermatological Diagnostics.

INTRODUCTION

1.1. Introduction

In the realm of healthcare, the advent of advanced technologies has brought forth new opportunities for early detection and diagnosis. Skin cancer, being a prevalent and potentially fatal condition, necessitates innovative approaches for timely identification. This study focuses on the application of Convolutional Neural Networks (CNN) and OpenCV in the domain of skin cancer detection, leveraging data from a Kaggle dataset.[1] Historically, medical diagnostics relied on traditional methods, but recent years have seen a paradigm shift towards incorporating artificial intelligence and computer vision. The use of CNN, a deep learning technique, along with OpenCV, a computer vision library, offers a promising avenue for improving the accuracy and efficiency of skin cancer detection.

1.2 Statement of the Problem

Skin cancer poses a significant public health challenge, and early detection is critical for successful treatment. Traditional diagnostic methods may have limitations, prompting the exploration of advanced technologies. However, the integration of CNN and OpenCV in skin cancer detection presents challenges and requires thorough investigation.

1.3 Need for the Study

This study addresses the pressing need for more accurate and efficient methods of skin cancer detection. By harnessing the power of CNN and OpenCV, the research aims to contribute valuable insights into the effectiveness of these technologies in diagnosing skin cancer. The study seeks to bridge the gap between traditional diagnostic approaches and modern, technology-driven solutions for enhanced patient outcomes.

1.4 Scope of the Study

The scope of this study includes:

Evaluating the performance of Convolutional Neural Networks in analyzing dermatological images for skin cancer detection. Assessing the effectiveness of OpenCV in image processing and its contribution to improve diagnostic accuracy. Analyzing the potential of the Kaggle dataset for training and testing the skin cancer detection model.

1.5 Future Scope

Investigating the impact of integrating CNN and OpenCV on early detection rates and the overall diagnostic process. This research aims to provide a comprehensive understanding of the application of CNN and OpenCV in skin cancer detection, offering insights that can contribute to the advancement of diagnostic practices in the field of Dermatology.

LITERATURE REVIEW

Title: A comprehensive study on skin cancer detection using artificial neural network (ANN) and convolutional neural network (CNN)

Author: T. Thanusree, I. Sanjana Reddy

Year: 2024

Overview: The literature review in this study primarily focuses on advancements in skin cancer detection using Convolutional Neural Networks (CNN) and the OpenCV library. It explores key variables, performance dimensions, and the relevance of CNN in the context of dermatological image analysis.Additionally, the section discusses the significance of OpenCV in enhancing image processing capabilities, providing a foundation for efficient skin cancer diagnosis.

EXISTING SYSTEM

Existing System in Skin Cancer Detection:

Skin cancer detection has traditionally relied on manual examination by dermatologists, which is subject to human error and can be time-consuming. As a response to these challenges, computer-aided diagnosis systems, particularly those leveraging Artificial Neural Networks (ANN) and OpenCV, have emerged as promising tools for efficient and accurate skin cancer detection.

PROPOSED SYSTEM

Our proposed system seeks to revolutionize the field of dermatological diagnosis by introducing an advanced computer-aided detection system that combines the power of Convolutional Neural Networks (CNN) and OpenCV. This system aims to provide a more efficient, accurate, and timely method for the early detection of skin cancer lesions.

Automated Dermatological Image Analysis:

The heart of our proposed system lies in the integration of a state-of-the art CNN for automated dermatological image analysis. Unlike traditional methods, which rely on manual inspection, our system utilizes deep learning to automatically identify and classify potential skin cancer lesions in medical images. This automated analysis significantly reduces the dependency on human interpretation and accelerates the diagnostic process.

Convolutional Neural Networks (CNN):

The CNN architecture employed in our system is designed to understand intricate patterns and features within dermatological images. Trained on diverse datasets containing a wide range of skin lesions, the CNN can learn to distinguish between benign and malignant conditions. The hierarchical feature extraction capabilities of CNNs contribute to their ability to discern subtle details crucial for accurate skin cancer diagnosis.

RESULTS AND DISCUSSION

From this study we got the above result.

CONCLUSION

The use of Convolutional Neural Networks (CNNs) for skin cancer detection with OpenCV demonstrates a promising advancement in medical image analysis and diagnostic accuracy. CNNs, due to their ability to learn and extract features from images, significantly improve the accuracy and precision of skin cancer detection compared to traditional methods. They can effectively distinguish between benign and malignant lesions.

Utilizing OpenCV for image preprocessing and CNNs for detection allows for automation in the diagnostic process. This reduces the workload on dermatologists and speeds up the detection process, enabling quicker diagnosis and treatment.

Early detection of skin cancer is crucial for effective treatment. CNNs can detect subtle changes in skin lesions that may not be noticeable to the human eye, thus facilitating early intervention.

REFERENCE

- Monica KM, Shreeharsha J, Falkowski-Gilski P, Falkowska-Gilska B, Awasthy M, Phadke R. Melanoma skin cancer detection using mask-RCNN with modified GRU model. Front Physiol. 2023;14: 1324042.
- Viknesh CK, Kumar PN, Seetharaman R, Anitha D. Detection and Classification of Melanoma Skin Cancer Using Image Processing Technique. Diagnostics (Basel). 2023;13. doi:10.3390/diagnostics13213313
- Moturi D, Surapaneni RK, Avanigadda VSG. Developing an efficient method for melanoma detection using CNN techniques. J Egypt Natl Canc Inst. 2024;36: 6.
- Kumar Lilhore U, Simaiya S, Sharma YK, Kaswan KS, Rao KBVB, Rao VVRM, et al. A precise model for skin cancer diagnosis using hybrid U-Net and improved MobileNet-V3 with hyperparameters optimization. Sci Rep. 2024;14: 4299.
- Rahman MM, Nasir MK, Nur-A-Alam M, Khan MSI. Proposing a hybrid technique of feature fusion and convolutional neural network for melanoma skin cancer detection. J Pathol Inform. 2023;14: 100341.
- Di Biasi L, De Marco F, Auriemma Citarella A, Castrillón-Santana M, Barra P, Tortora G. Refactoring and performance analysis of the main CNN architectures: using false

negative rate minimization to solve the clinical images melanoma detection problem. BMC Bioinformatics. 2023;24: 386.

- Rai HM, Yoo J. A comprehensive analysis of recent advancements in cancer detection using machine learning and deep learning models for improved diagnostics. J Cancer Res Clin Oncol. 2023;149: 14365–14408.
- Behara K, Bhero E, Agee JT. Grid-Based Structural and Dimensional Skin Cancer Classification with Self-Featured Optimized Explainable Deep Convolutional Neural Networks. Int J Mol Sci. 2024;25. doi:10.3390/ijms25031546
- Ali MU, Khalid M, Alshanbari H, Zafar A, Lee SW. Enhancing Skin Lesion Detection: A Multistage Multiclass Convolutional Neural Network-Based Framework. Bioengineering (Basel). 2023;10. doi:10.3390/bioengineering10121430
- Sharma P, Nayak DR, Balabantaray BK, Tanveer M, Nayak R. A survey on cancer detection via convolutional neural networks: Current challenges and future directions. Neural Netw. 2024;169: 637–659.

ANNEXURE

```
import cv2
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelBinarizer
import os
# Load and preprocess the dataset
def load_data(data_dir):
    images = []
    labels = []
    for label in os.listdir(data_dir):
        if label not in ['cancer', 'non_cancer']:
```

```
            continue
        label_dir = os.path.join(data_dir, label)
        for image_file in os.listdir(label_dir):
            image_path = os.path.join(label_dir, image_file)
            image = cv2.imread(image_path)
            image = cv2.resize(image, (128, 128))  # Resize images to 128x128
            images.append(image)
            labels.append(label)
    return np.array(images), np.array(labels)
data_dir = 'path_to_dataset'  # Replace with the path to your dataset
images, labels = load_data(data_dir)
# Normalize images
images = images.astype('float32') / 255.0
# Encode labels
lb = LabelBinarizer()
labels = lb.fit_transform(labels)
# Split the data
X_train, X_test, y_train, y_test = train_test_split(images, labels, test_size=0.2, random_state=42)
# Define the CNN model
def build_model():
    model = Sequential()
    model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(128, 128, 3)))
    model.add(MaxPooling2D((2, 2)))
    model.add(Dropout(0.25))
    model.add(Conv2D(64, (3, 3), activation='relu'))
    model.add(MaxPooling2D((2, 2)))
    model.add(Dropout(0.25))
    model.add(Conv2D(128, (3, 3), activation='relu'))
    model.add(MaxPooling2D((2, 2)))
    model.add(Dropout(0.25))
    model.add(Flatten())
```

```
    model.add(Dense(512, activation='relu'))
    model.add(Dropout(0.5))
    model.add(Dense(1, activation='sigmoid'))
    model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
    return model
model = build_model()
model.summary()
# Data augmentation
datagen = ImageDataGenerator(
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    horizontal_flip=True
)
datagen.fit(X_train)
# Train the model
history = model.fit(
    datagen.flow(X_train, y_train, batch_size=32),
    epochs=20,
    validation_data=(X_test, y_test)
)
# Evaluate the model
loss, accuracy = model.evaluate(X_test, y_test)
print(f"Test Loss: {loss}")
print(f"Test Accuracy: {accuracy}")
# Predict on new data
def predict(image_path, model, lb):
    image = cv2.imread(image_path)
    image = cv2.resize(image, (128, 128))
    image = image.astype('float32') / 255.0
    image = np.expand_dims(image, axis=0)
```

```
    prediction = model.predict(image)
    label = lb.inverse_transform(prediction > 0.5)[0]
    return label
test_image_path = 'path_to_test_image'  # Replace with the path to your test image
predicted_label = predict(test_image_path, model, lb)
print(f"Predicted Label: {predicted_label}")
```