# SKIN CANCER DETECTION USING TENSORFLOW

**A Minor Project Report**

*Submitted by*

**LAKSHMAN SAI.P(RA2011027010202)**

**CHARAN RAM.P(RA2011027010186)**

*Under the Guidance of*

**Dr. SV. SHRI BHARATHI**

*In partial fulfilment of the requirements for the degree of*

**BACHELOR OF TECHNOLOGY**
**in**
**COMPUTER SCIENCE AND ENGINEERING**
**with a specialization in Big Data Analytics**


**DEPARTMENT OF DATA AND BUSINESS SYSTEM COLLEGE OF ENGINEERING AND TECHNOLOGY**
**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**
**(Under Section 3 of UGC Act, 1956)**
SRM NAGAR, KATTANKULATHUR – 603 203
CHENGALPATTU DISTRICT
**NOVEMBER 2023**

# SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

# KATTANKULATHUR – 603 203

## BONAFIDE CERTIFICATE

Certified that this B.Tech. Minor project report titled "**SKIN CANCER DETECTION USING TENSORFLOW**" is the bonafide work of **LAKSHMAN SAI.P(RA2011027010202) and CHARAN RAM.P(RA2011027010186)** who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation based on which a degree or award was conferred on an earlier occasion for this or any other candidate.


**Dr. SV. SHRI BHARATHI**
Assistant Professor
Department of Data Science and Business Systems,
SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

**Dr.M. Lakshmi**
Head of the Department
Department of of Data Science and Business Systems,
SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

## Department of Networking and Communications

## SRM Institute of Science and Technology

## Own Work Declaration Form

Degree/ Course : B. Tech in COMPUTER SCIENCE AND ENGINEERING with a specialization in Big Data Analytics

**Student Name : LAKSHMAN SAI.P and CHARAN RAM.P**

Registration Number : RA2011027010202 and RA2011027010186

**Title of Work : SKIN CANCER DETECTION USING TENSORFLOW**

We hereby certify that this assessment compiles with the University's Rules and Regulations relating to Academic misconduct and plagiarism, as listed in the University Website, Regulations, and the Education Committee guidelines.

We confirm that all the work contained in this assessment is our own except where indicated, and that We have met the following conditions:

- Clearly referenced / listed all sources as appropriate
- Referenced and put in inverted commas all quoted text (from books, web, etc.)
- Given the sources of all pictures, data etc. that are not my own
- Not made any use of the report(s) or essay(s) of any other student(s) either past or present
- Acknowledged in appropriate places any help that I / We have received from others (e.g.,fellow students, technicians, statisticians, external sources)
- Compiled with any other plagiarism criteria specified in the Course handbook / University website

We understand that any false claim for this work will be penalized in accordance with the University policies and regulations.

| **DECLARATION:** | |
|---|---|
| We are aware of and understand the University's policy on Academic misconduct and plagiarism and we certify that this assessment is our own work, except where indicated by referring, and that we have followed the good academic practices noted above. | |
| Lakshman Sai Paidipati<br>RA2011027010202 | Charan Ram Paidipati<br>RA2011027010186 |

# ACKNOWLEDGEMENT

We express our humble gratitude to **Dr. C. Muthamizhchelvan,** Vice-Chancellor, SRM Institute of Science and Technology, for the facilities extended for the project work and his continued support.

We extend our sincere thanks to Dean-CET, SRM Institute of Science and Technology, **Dr.T.V. Gopal,** for his invaluable support.

We wish to thank **Dr. Revathi Venkataraman, Professor & Chairperson**, School of Computing, SRM Institute of Science and Technology, for her support throughout the project work.

We are incredibly grateful to our Head of the Department, **Dr.M. Lakshmi**, Professor and Head, Department of Data Science and Business Systems, School of Computing, SRM Institute of Science and Technology, for her suggestions and encouragement at all the stages of the project work.

We want to convey our thanks to our Project Coordinator, **Dr.Hemavathi.D**, Associate Professor, Department of Data Science and Business Systems, School of Computing, SRM Institute of Science and Technology, for their inputs during the project reviews and support.

We register our immeasurable thanks to our Faculty Advisor, **Dr.SV.Shri Bharathi,** Department of Data Science and Business Systems, School of Computing, SRM Institute of Science and Technology, for leading and helping us to complete our course.

Our inexpressible respect and thanks to my / our guide, **Dr.SV.Shri Bharathi**, Associate Professor, Department Data Science and Business Systems, SRM Institute of Science and Technology, for providing us with an opportunity to pursue our project under his mentorship. He provided us with the freedom and support to explore the research topics of my/our interest. His passion for solving problems and making a difference in the world has always been inspiring.

We sincerely thank the Data Science and Business Systems, Department staff and students, SRM Institute of Science and Technology, for their help during our project. Finally, we would like to thank parents, family members, and friends for their unconditional love, constant support, and encouragement.

LAKSHMAN SAI.P(RA2011027010202)

CHARAN RAM.P (RA2011027010186)

# ABSTRACT

Skin cancers are more common than other diseases. Skin cancers may be caused by fungal infection, bacteria, allergy, or viruses, etc. The advancement of lasers and Photonics based medical technology has made it possible to diagnose the skin cancers much more quickly and accurately. But the cost of such diagnosis is still limited and very expensive. Using deep learning and neural networks, we'll be able to classify benign and malignant skin diseases, which may help the doctor diagnose cancer at an earlier stage. In this Project, we will make a skin disease classifier that tries to distinguish between benign (nevus and seborrheic keratosis) and malignant (melanoma) skin diseases from only photographic images using TensorFlow framework in Python.

In cancer, there are over 200 different forms. Out of 200, melanoma is the deadliest form of skin cancer. The diagnostic procedure for melanoma starts with clinical screening, followed by dermoscopic analysis and histopathological examination. Melanoma skin cancer is highly curable if it gets identified at the early stages. The first step of Melanoma skin cancer diagnosis is to conduct a visual examination of the skin's affected area. Dermatologists take the dermatoscopic images of the skin lesions by the high-speed camera, which have an accuracy of 65-80% in the melanoma diagnosis without any additional technical support. With further visual examination by cancer treatment specialists and dermatoscopic images, the overall prediction rate of melanoma diagnosis raised to 75-84% accuracy

**Keywords**: Skin cancer, TensorFlow, Malignant, Benign, CNN, Deep learning.

# TABLE OF CONTENTS

## LIST OF FIGURES



## LIST OF TABLES

# Chapter 1

## INTRODUCTION

Skin cancer is a deadly disease. Skin has three (3) basic layers. Skin cancer begins in outermost layer, which is made up of first layer squamous cells, second layer basal cells, and innermost or third layer melanocytes cell. Squamous cell and basal cell are sometimes called non-melanoma cancers. Non-melanoma skin cancer always responds to treatment and rarely spreads to other skin tissues. Melanoma is more dangerous than most other types of skin cancer. If it is not detected at beginning stage, it is quickly invading nearby tissues and spread to other parts ofthe body. Formal diagnosis method to skin cancer detection is Biopsy method. Abiopsy is a method to remove a piece of tissue or a sample of cells from patient body so that it can be analyzed in a laboratory. It is uncomfortable method. Biopsy Method is time consuming for patient as well as doctor because it takes lot of time for testing. Biopsy is done by removing skin tissues (skin cells) and that sample undergoes series of laboratory testing. There is possibility of spreading of disease into other part of body. It is more risk. Considering all the cases mentioned above, So Skin cancer detection using SVM is proposed. This methodology uses digital image processing technique and SVM for classification. This technique has inspired the early detection of skin cancers, and requires no oil to be applied to your skin to achieve clear sharp images of your moles. In this way, it's quicker and cleaner approach. But, most importantly, due to its higher magnification, Skin Cancer Detection Using SVM can prevent the unnecessary excision of perfectly harmless moles and skin lesions.

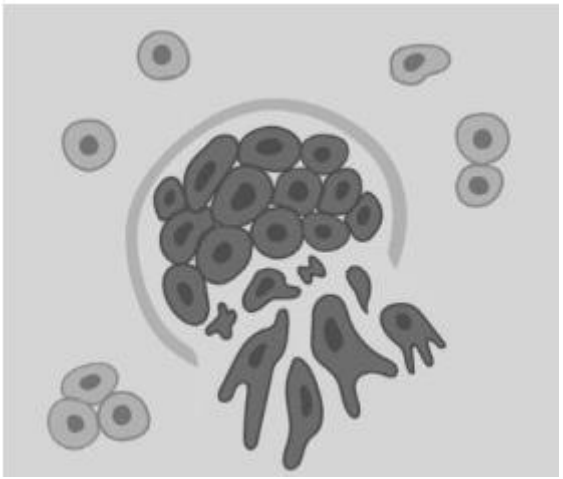


Fig.1.1: Malign Cells

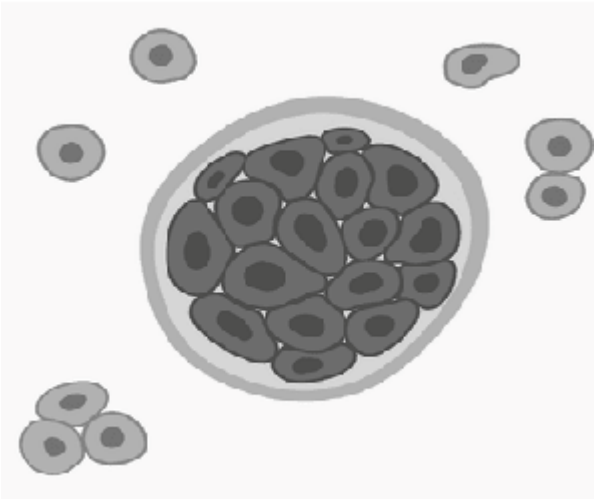


Fig.1.2: Benign Cells

Melanoma is a particularly deadly form of skin cancer and although it accounts for only 4% of all skin cancers it is responsible for 75% of all skin cancer deaths.If melanoma is diagnosed and treated in its early stages, it can be cured but if the other parts of the body. It's spread in other parts beyond the skin can be hazardousas it is difficult to treat. The presence of Melanocytes in any body part causes the Melanoma. Intensive Exposure of skin to ultraviolet radiation is the main cause of the melanoma. Dermoscopy is a non-invasive examination technique based on the useof incident light and oil immersion to make possible the visual examination of sub surface structures of the skin. Though the detection of melanoma using dermoscopy is higher than unaided observation based detection3, it's diagnostic accuracy depends on the training of the dermatologist. The diagnosis of melanoma from melanocytic nevi is not straight forward especially in the early stage. Thus, automatic diagnosis tool is essential for physicians. Even when the expert dermatologists use the dermoscopy for diagnosis, the accuracy of melanoma diagnosis is estimated to be about 75-84%. The computer aided diagnostics is helpful to increase the diagnosis accuracy as well as the speed. Computer is not more intelligent than human but it may be able to extract some information, like color variation, asymmetry, texture features, that may not be readily perceived by human eyes. There have been many proposed systems and algorithms such as the seven-point checklist, ABCD rule, and the Menzies method2,3 to improve the

diagnostics of the melanoma skin cancer. Thekey steps in a computer-vision based diagnosis of melanoma are: image acquisition of skin lesion image, segmentation ofthe skin lesion from skin region, extraction of features of the lesion blob and feature classification. Segmentation or border detection is the process of separating the lesion from the surrounding skin in order to form the region of interest. Feature extraction is used to extract the features; similar to those visually detected by dermatologists, that accurately characterizes a melanoma lesion. The feature extraction methodology of many computerized melanoma detection systems has been largely based on the conventional clinical algorithm of ABCD-rule of dermoscopy due to its effectiveness and simplicity of implementation. Its effectiveness stems from the fact that it incorporates the main features of a melanoma lesion such as asymmetry, border irregularity, color and diameter (or differential structures), where quantitative measures can be computed.

Skin cancer ap- pears to be of two kinds Benign and Melanoma form. Benignit's just the moles on the skin which does not penetrate inside, where- asMelanoma causes sores on the skin which leads to bleeding and it is named aftercells Melanocytes which is more hazardous. In United States, more than 700,000skin lesions are diagnosed annually under the estimation of American Cancer Society. According to statistics given by the Apollo and other hospitals it suggests that Melanoma affects the ages ranging from 41-60+. There are technologies thatare used to detect skin cancer at the early stages. Skin Cancer detected in advance can save people's lives and it eliminates the multiplication of cancer cells across the parts of the body. Although it affects the people within agelimits but high probably is for the bright skin people.

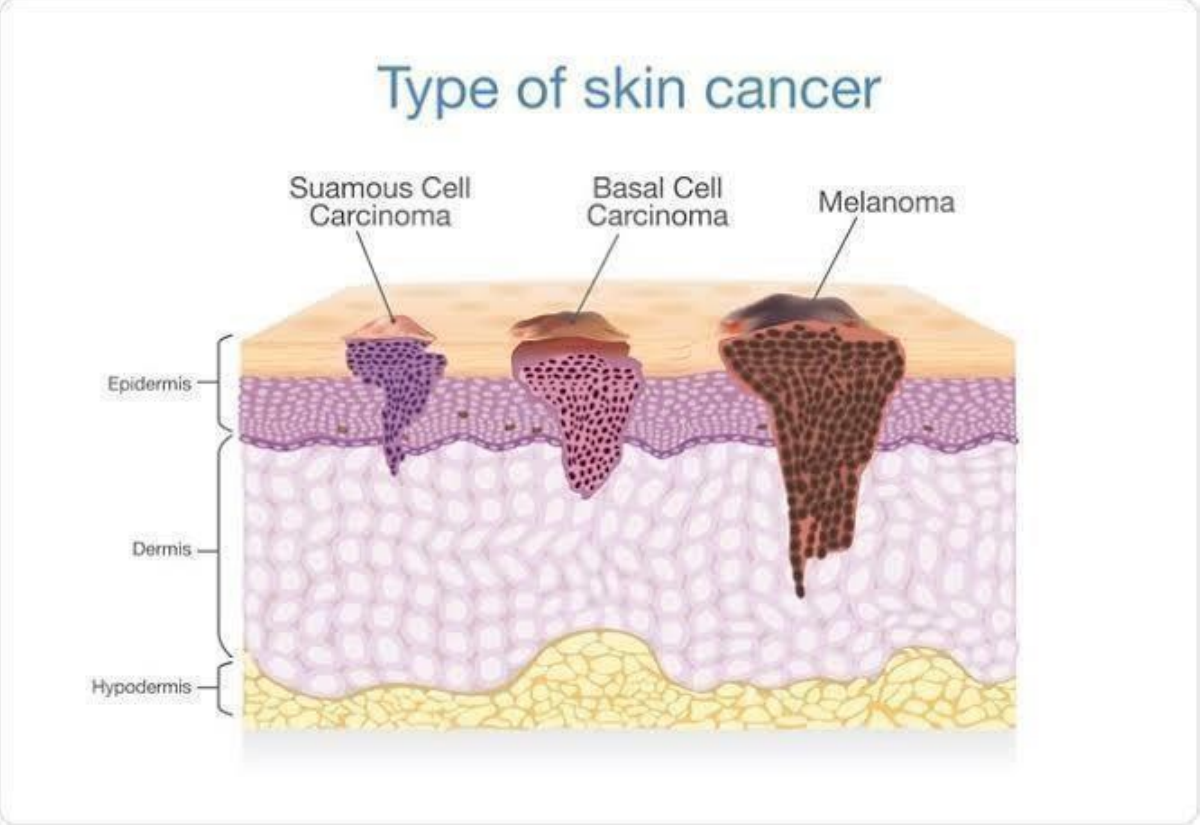



Fig.1.3: Types of skin cancer

Figure.1 shows the different types of skin cancer, the diagram clearly shows the Melanoma penetration inside the skin. It will be hard for even the experienceddermatologist to detect the skin cancer or to predict the stages. Therefore, many hardware & software devices and applications evolved. In order to detect Non- Melanoma Skin Cancer (NMSC) there are many hardware devices which includes, biopsy, molecular markers, ultra-sonography, Doppler, optical coherence tomography, Dermoscopy and spectroscopy. Dermoscopy is one suchhardware device that helps in examining the surface of the skin using skin surface microscopy for the detection of skin cancer; it is a non-invasive examination technique that lends the support to distinguish

between Melanoma and Non- Melanoma. It is the most expensive device that cannot be affordable for the poorcountries and developing countries like Africa. Moreover, later researchers cameup with many software applications, which eliminated the hardware cost and influence the ease of usage. The software does not require any expertise to handle them but developed with the intention of prior information about the cancer.

Image Processing is one of the traditional approaches which deals with analyzing and processing of an image. Image processing has proposed many methodologies which helps in the early detection of skin cancer. Proposed methodologies have influenced in the early detection which prevents the cancer from spreading across the skin. The methodology in an image processing approach involves the use of Noise removal, Edge detection, Image enhancement, Segmentation algorithms, Feature selection, Feature extraction, Neural network approach using Back propagation algorithms. In this paper, a model is proposed with an aim of curing the skin cancer along with providing cancer information to the people. Model detects the skin cancer on the color skinimage and uses pre-processing methods such as Image acquisition, Noise removal and plotting Histogram. Preprocessing methods helps to solve the illumination, contrast and noise problems. After the removal of noise various techniques such as Edge detection, Image enhancement, Segmentation. Feature extraction is used to extract the affected portion of skin iscalculated. Calculated values are fed into the Neural networks, using Back propagation algorithm, type of the skin cancer is predicted.

The main aim of the proposed system is to eliminate the risk caused by manycountries with respect to the skin cancer. Many hardware devices were developedbut those devices were not affordable. Patients are provided with the informationof type of cancer which helps to cure the cancer.

# Chapter 2

# SYSTEM ANALYSIS

## 2.1. MOTIVATION

The motivation behind this project is deeply rooted in the need to improve early detection of skin cancer. Skin cancer is a leading cause of cancer-related deaths worldwide, and the key to improving outcomes is catching it in its early stages.
Unfortunately, traditional diagnostic methods can be subjective and sometimes error-prone.
This motivates us to explore **a more accurate** and **efficient approach**.

## 2.2. PROBLEM STATEMENT

1. **Ineffectiveness of Traditional Methods:**
    - Traditional skin cancer diagnosis methods, such as biopsy, are often uncomfortable for patients and time-consuming for both patients and medical practitioners.
2. **Subjectivity and Accuracy Concerns:**
    - The subjective nature of some diagnostic processes may lead to misdiagnoses, posing risks to patients. Dermoscopy, while effective, relies heavily on the expertise of the dermatologist, and its diagnostic accuracy may vary.
3. **Risk of Disease Spread:**
    - In cases of melanoma, the most dangerous form of skin cancer, delayed detection can result in the disease spreading beyond the skin, making it more challenging to treat and potentially hazardous.
4. **Need for Non-Invasive and Accurate Solutions:**
    - There is a demand for non-invasive and more accurate diagnostic tools that can enhance the early detection of skin cancer, thereby improving patient outcomes and reducing the risks associated with delayed diagnoses.
5. **Existing Technologies and Challenges:**
    - While technologies exist for early skin cancer detection, their accuracy and efficiency are often contingent on factors such as the expertise of medical professionals. This highlights the need for automated and reliable diagnostic solutions.
6. **Prevalence and Impact:**
    - Given the prevalence of skin cancer and its significant impact on patient health, developing advanced, accessible, and accurate diagnostic tools becomes paramount for effective healthcare practices.

## 2.3. CHALLENGES AND LIMITATIONS

1. **Limited Dataset**:

A key challenge is the availability of a comprehensive and diverse dataset for training the Support Vector Machines (SVM) model. Limited data may impact the model's ability to generalize well to various skin types and conditions.

2. **Data Quality and Variability:**

The quality and variability of the input data, especially in photographic images, can be a limitation. Factors like lighting conditions, image resolution, and diverse skin tones may introduce variability that the model needs to handle robustly.

3. **Dependency on Image Quality:**

The effectiveness of the skin cancer detection model may heavily depend on the quality of input images. Blurred, low-resolution, or poorly lit images could hinder accurate classification.

4. **Complexity of Skin Cancer Types:**

The project focuses on distinguishing between benign and malignant skin diseases, specifically nevus, seborrheic keratosis, and melanoma. However, there are various other skin diseases that may present similar visual characteristics, leading to potential misclassifications.

5. **Need for Continuous Training:**

The model's performance may degrade over time, requiring continuous training to adapt to new data and emerging patterns in skin cancer images. This necessitates a mechanism for regular model updates.

6. **Interpretability of SVM:**

SVM models can be less interpretable compared to simpler models, making it challenging to understand the features contributing to classification decisions. This lack of interpretability could be a limitation in a medical context where transparency is crucial.

7. **Deployment Challenges:**

Implementing the skin cancer detection system in real-world healthcare settings may encounter practical challenges, including integration with existing systems, user training, and ensuring compliance with medical standards and regulations.

8. **Generalization to Diverse Populations:**

The model's generalization may be limited to the demographics represented in the training data. It's essential to assess its performance across diverse populations to avoid biases and ensure inclusivity.

9. **False Positives and Negatives:**

Like any diagnostic tool, the model may produce false positives or false negatives, leading to either unnecessary concern for patients or missed diagnoses.

10. **Ethical Considerations:**

The use of machine learning in healthcare raises ethical concerns related to patient privacy, informed consent, and the responsible deployment of technology. Ethical considerations should be thoroughly addressed.

## 2.4. OBJECTIVES

1. **Develop a Skin Disease Classifier:**
    - Design and implement a robust skin disease classifier using Support Vector Machines (SVM) and digital image processing techniques.
2. **Distinguish Between Benign and Malignant Skin Diseases:**
    - Train the classifier to accurately distinguish between benign skin diseases (nevus and seborrheic keratosis) and malignant skin diseases (melanoma).
3. **Improve Accuracy and Efficiency of Diagnosis:**
    - Enhance the accuracy and efficiency of skin cancer diagnosis by leveraging machine learning techniques, reducing misdiagnoses, and streamlining the diagnostic process**.**
4. **Explore the Use of SVM in Skin Cancer Detection:**
    - Investigate and demonstrate the effectiveness of Support Vector Machines as a classification tool for skin cancer detection, providing insights into its application in medical image analysis.
5. **Ensure Model Generalization:**
    - Develop a model that generalizes well across diverse skin types and conditions, minimizing biases and ensuring inclusivity in its application.

6. **Reduce Dependency on Invasive Diagnostic Methods:**
    - Provide an alternative to invasive diagnostic methods, such as biopsy, by creating a non-invasive and efficient solution for skin cancer detection.
7. **Contribute to Early Detection Practices:**
    - Contribute to the field of early skin cancer detection by proposing and implementing a solution that aids in the identification of skin diseases at an early stage.
8. **Minimize Unnecessary Excisions:**
    - Prevent unnecessary excision of harmless moles and skin lesions by offering a diagnostic tool with higher magnification and accuracy.
9. **Enhance Transparency and Interpretability:**
    - Strive to enhance the transparency and interpretability of the model, ensuring that medical professionals can understand and trust the features contributing to classification decisions.
10. **Address Ethical Considerations:**
    - Address ethical considerations related to patient privacy, informed consent, and responsible technology deployment, ensuring the ethical and responsible use of the skin cancer detection system.
11. **Contribute to Medical Image Analysis Research:**
    - Provide insights and findings that contribute to the broader field of medical image analysis, particularly in the context of skin cancer detection**.**

## 2.5. INNOVATION IDEA OF THE PROJECT:

In the realm of dermatology, where early detection is paramount, our project introduces a groundbreaking approach harnessing the power of artificial intelligence. The innovation lies in the seamless integration of advanced deep learning techniques and neural networks, specifically implemented through the TensorFlow framework in Python.

1. **Enhancing Diagnostic Accuracy:** Traditional diagnostic methods for skin cancer often rely on subjective assessments. Our project aims to revolutionize this process by introducing a skin disease classifier capable of discerning between benign (nevus and seborrheic keratosis) and malignant (melanoma) conditions with unprecedented accuracy. This shift towards a more objective and data-driven methodology is at the core of our innovation.
2. **Automated Image Analysis:** Utilizing TensorFlow, our project streamlines the diagnostic journey by training the model on photographic images. This automated image analysis not only expedites the diagnostic process but also ensures a consistent and reliable evaluation, minimizing the risk of human error and subjectivity.
3. **Empowering Dermatologists:** The innovation extends beyond technology to empower dermatologists with a powerful diagnostic tool. By leveraging the capabilities of deep learning, our system provides dermatologists with enhanced insights, aiding them in making well-informed decisions for early intervention and treatment.
4. **User-Friendly Framework:** Recognizing the need for accessibility, our project is designed with a user-friendly framework. The TensorFlow implementation in Python ensures that the tool is not only advanced but also practical for healthcare professionals, bridging the gap between cutting-edge technology and real-world clinical applications.
5. **Potential Impact on Healthcare:** This innovative project holds the potential to significantly impact the landscape of skin cancer diagnosis. Early and accurate identification of skin diseases is crucial for successful treatment, and our AI-driven precision aims to be a transformative force in improving patient outcomes.

## 2.6. SCOPE OF THE PROJECT

The scope of our project encompasses a comprehensive approach to advancing skin cancer detection methods, primarily focusing on the integration of artificial intelligence for precise diagnostics. The key aspects of the project scope include:

**1. Development of a Skin Disease Classifier:**

- Create a robust and versatile skin disease classifier using advanced deep learning techniques and neural networks, implemented through the TensorFlow framework in Python.

**2. Distinguishing Between Benign and Malignant Conditions:**

- Train the classifier to distinguish between benign skin diseases (nevus and seborrheic keratosis) and malignant skin diseases (melanoma) with a high degree of accuracy.

**3. Utilizing Photographic Images for Analysis:**

- Implement the classifier to analyze and interpret photographic images, enabling a non-invasive and efficient diagnostic process for dermatologists and healthcare professionals.

**4. Streamlining Diagnostic Workflows:**

- Streamline the diagnostic workflows for dermatologists by providing an automated tool that offers objective insights, reducing the reliance on traditional, time-consuming methods.

**5. Enhancing Early Detection Practices:**

- Contribute to the field of early skin cancer detection by leveraging AI-driven technologies. The project aims to assist medical professionals in identifying potential cases at the earliest and most treatable stages.

**6. Implementation of TensorFlow Framework:**

- Employ the TensorFlow framework in Python to implement the neural network architecture, ensuring a scalable and versatile solution that aligns with current industry standards.

**7. User-Friendly Interface:**

- Design a user-friendly interface to facilitate seamless integration into clinical settings. The project should empower dermatologists with an intuitive tool that enhances their diagnostic capabilities.

**8. Consideration of Ethical and Privacy Standards:**

- Adhere to ethical standards and privacy regulations to ensure responsible deployment of AI technologies in healthcare. Strive to maintain patient confidentiality and data security throughout the diagnostic process.

**9. Potential for Further Research and Innovation:**

- Lay the groundwork for potential future research and innovation in the domain of medical image analysis and skin cancer detection. Explore opportunities for refining the model, expanding the dataset, and addressing emerging challenges in the field.

**10.Educational Component:** -

Incorporate an educational component within the project, providing a tutorial that guides users through the process of building and implementing a skin disease classifier. This fosters knowledge transfer and promotes the adoption of advanced technologies in dermatology.

# Chapter 3

## LITERATURE SURVEY

1. **"Deep Learning Approaches for Skin Cancer Classification"**
    - *Authors: Esteva, A., Kuprel, B., Novoa, R. A., et al.*
    - This foundational study explores the application of deep learning, particularly convolutional neural networks (CNNs), in the classification of skin cancer. The research evaluates the potential of these advanced neural networks in achieving accuracy comparable to dermatologists. By training CNNs on a large dataset of skin images, the study demonstrates the model's ability to distinguish between benign and malignant lesions, setting the stage for subsequent research in AI-driven dermatology.
2. **"Dermatologist-level Classification of Skin Cancer with Deep Neural Networks"**
    - *Authors: Codella, N., Gutman, D., Celebi, M. E., et al.*
    - This collaborative effort investigates the partnership between dermatologists and deep neural networks for skin cancer classification. The study evaluates various neural network architectures, including Inception and ResNet, to achieve dermatologist-level accuracy in identifying melanoma. By comparing the performance of AI models with human experts, the research emphasizes the potential for combining the strengths of both to enhance diagnostic capabilities.
3. **"A Review on the Applications of Deep Learning in Skin Cancer Classification"**
    - *Authors: Haenssle, H. A., Fink, C., Schneiderbauer, R., et al.*
    - This comprehensive review offers a thorough examination of the applications of deep learning in skin cancer classification. It traces the evolution of techniques, from traditional machine learning approaches to the emergence of deep neural networks. The review discusses the challenges in using AI for dermatological diagnosis, such as the need for large annotated datasets and the interpretability of deep learning models.
4. **"A Survey on Automated Melanoma Detection"**
    - *Authors: Celebi, M. E., Kingravi, H. A., Vela, P. A., Iyatomi, H.*
    - Focused on melanoma detection, this survey provides a panoramic view of automated methods in dermatology. It covers various approaches, including classical image processing, machine learning, and deep learning techniques. The survey highlights the importance of feature extraction, image segmentation, and the role of computational intelligence in advancing melanoma detection technologies.
5. **"Skin Cancer Classification Using Convolutional Neural Networks: Systematic Review"**
    - *Authors: Fadzil, A. F. M., Lee, K. Y., Ramli, R., et al.*
    - This systematic review synthesizes findings from multiple studies related to the application of convolutional neural networks (CNNs) in skin cancer classification. It analyzes trends, challenges, and areas for improvement in automated skin cancer diagnosis. The review emphasizes the significance of CNNs in achieving high accuracy and explores the potential for transfer learning to address challenges related to limited datasets.

6. **"Skin Lesion Analysis Toward Melanoma Detection: A Challenge at the 2017 International Symposium on Biomedical Imaging (ISBI)"**

    - *Authors: Tschandl, P., Rosendahl, C., Kittler, H.*
    - Focusing on the challenges in skin lesion analysis, this study addresses the complexities involved in melanoma detection. The research specifically reviews contributions made during the 2017 ISBI challenge, highlighting the importance of benchmark datasets and the collaborative efforts of the research community in advancing the field. The study underscores the significance of standardization and evaluation metrics in fostering progress in skin cancer detection.

7. **"Skin Cancer Recognition Using Ensemble of Neural Network Models"**

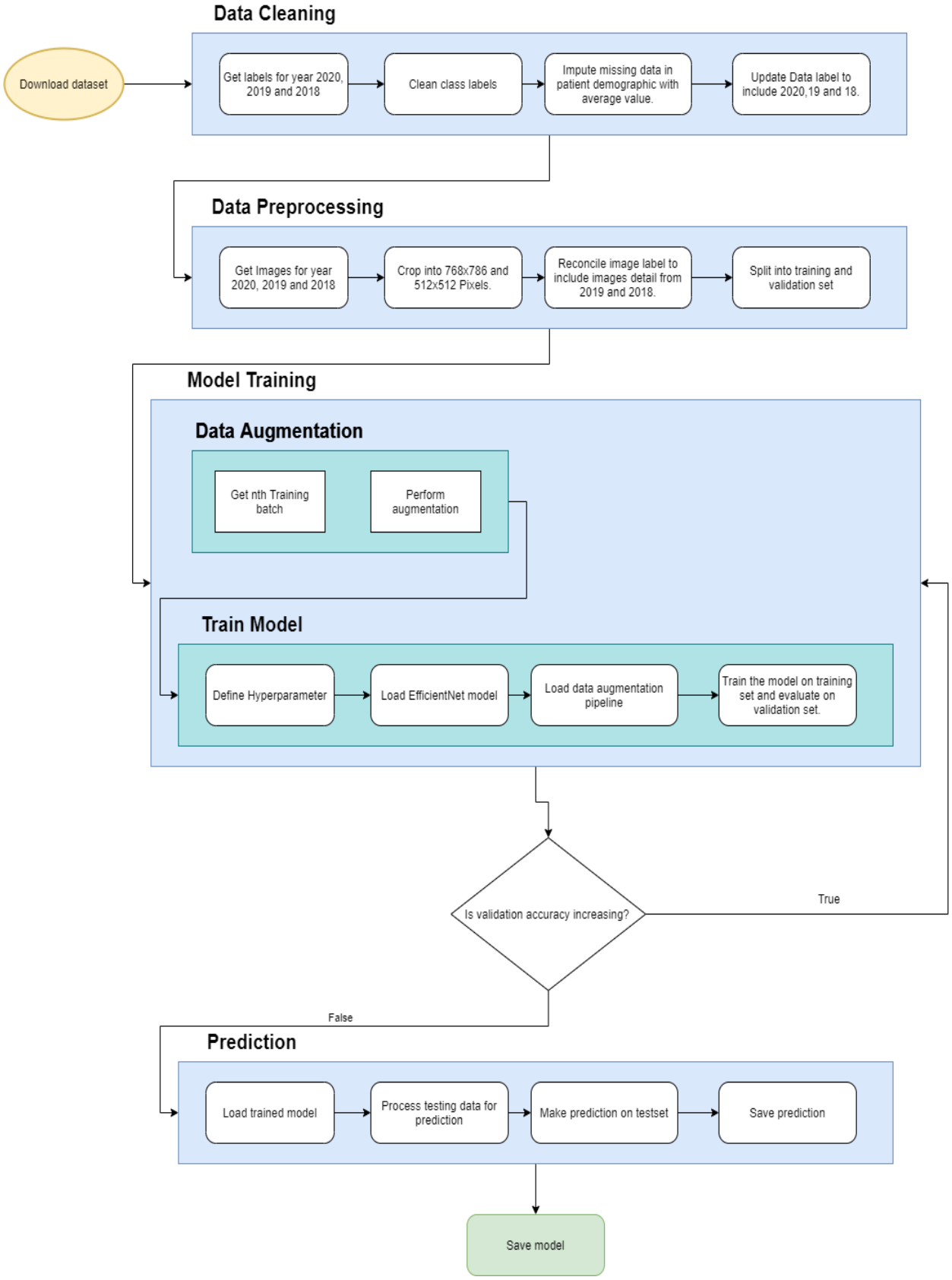
    - *Authors: Soni, R., Hasmukh, V.*
    - This research explores the application of ensemble learning, combining multiple neural network models for skin cancer recognition. The study investigates the effectiveness of ensemble methods in improving classification accuracy and robustness. By aggregating predictions from diverse models, the research aims to mitigate the limitations of individual models and enhance overall performance, contributing to the broader discussion on optimizing skin disease classifiers.

8. **"Skin Cancer Detection using Convolutional Neural Networks: A Review"**

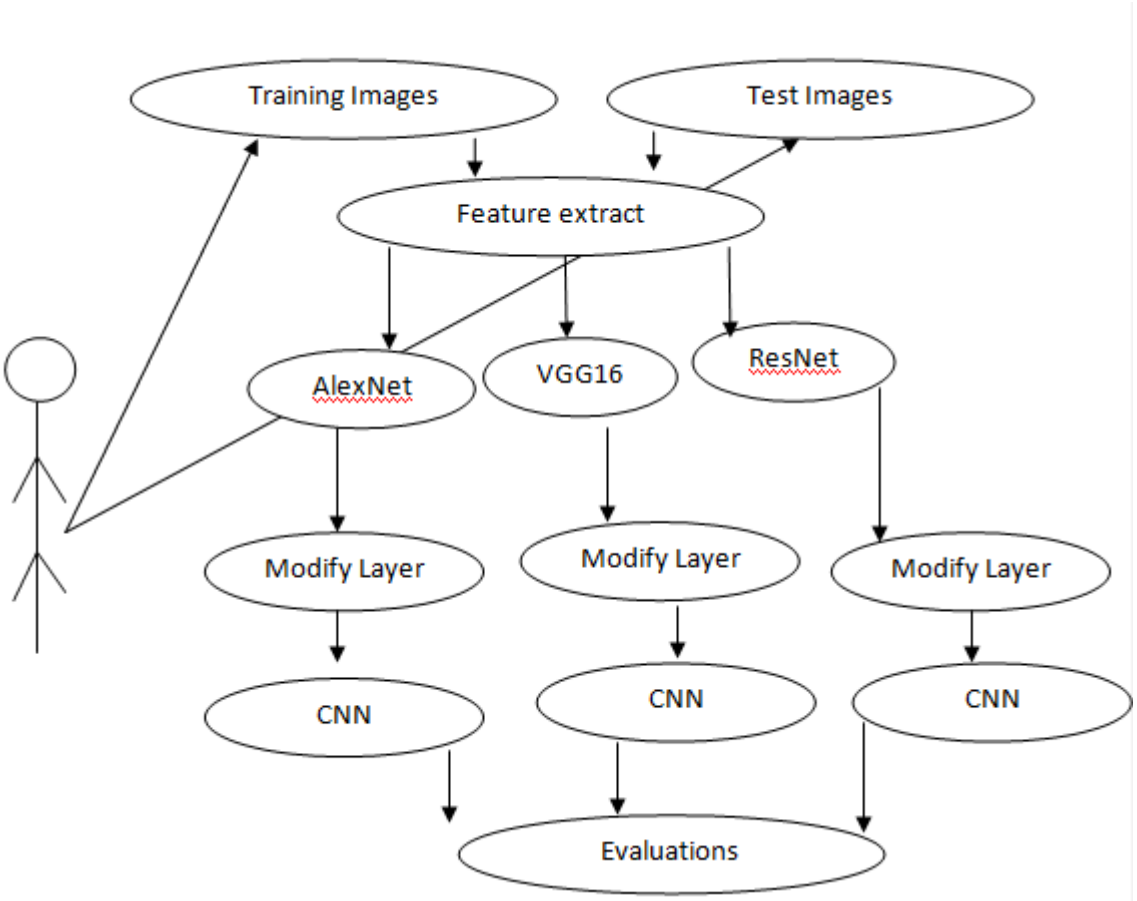
    - *Authors: Abbas, Q., Celebi, M. E., Garcia, I. F., et al.*
    - This review offers a comprehensive analysis of the application of convolutional neural networks (CNNs) in skin cancer detection. The study delves into the evolution of CNN-based models, discussing their effectiveness in classifying various skin conditions. The review emphasizes the role of CNNs in capturing intricate features from dermatological images, showcasing the potential of these architectures in advancing automated skin cancer diagnosis.

# Chapter 4
# UML DIAGRAM

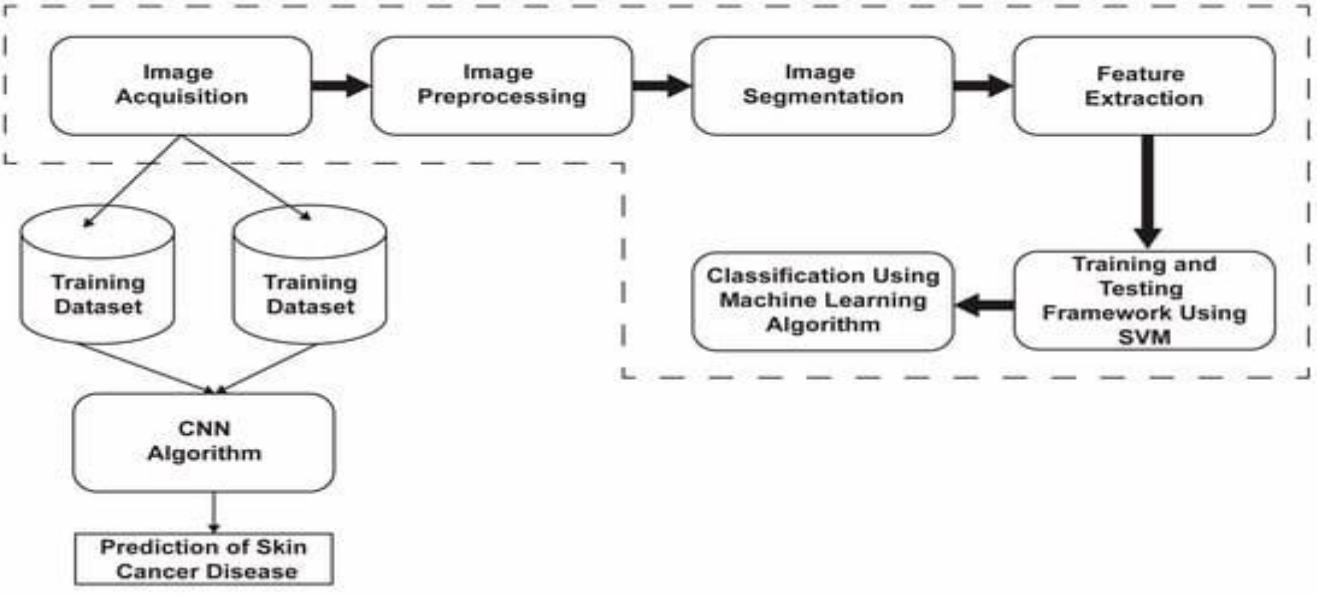
## 4.1. Model Training Flow Diagram



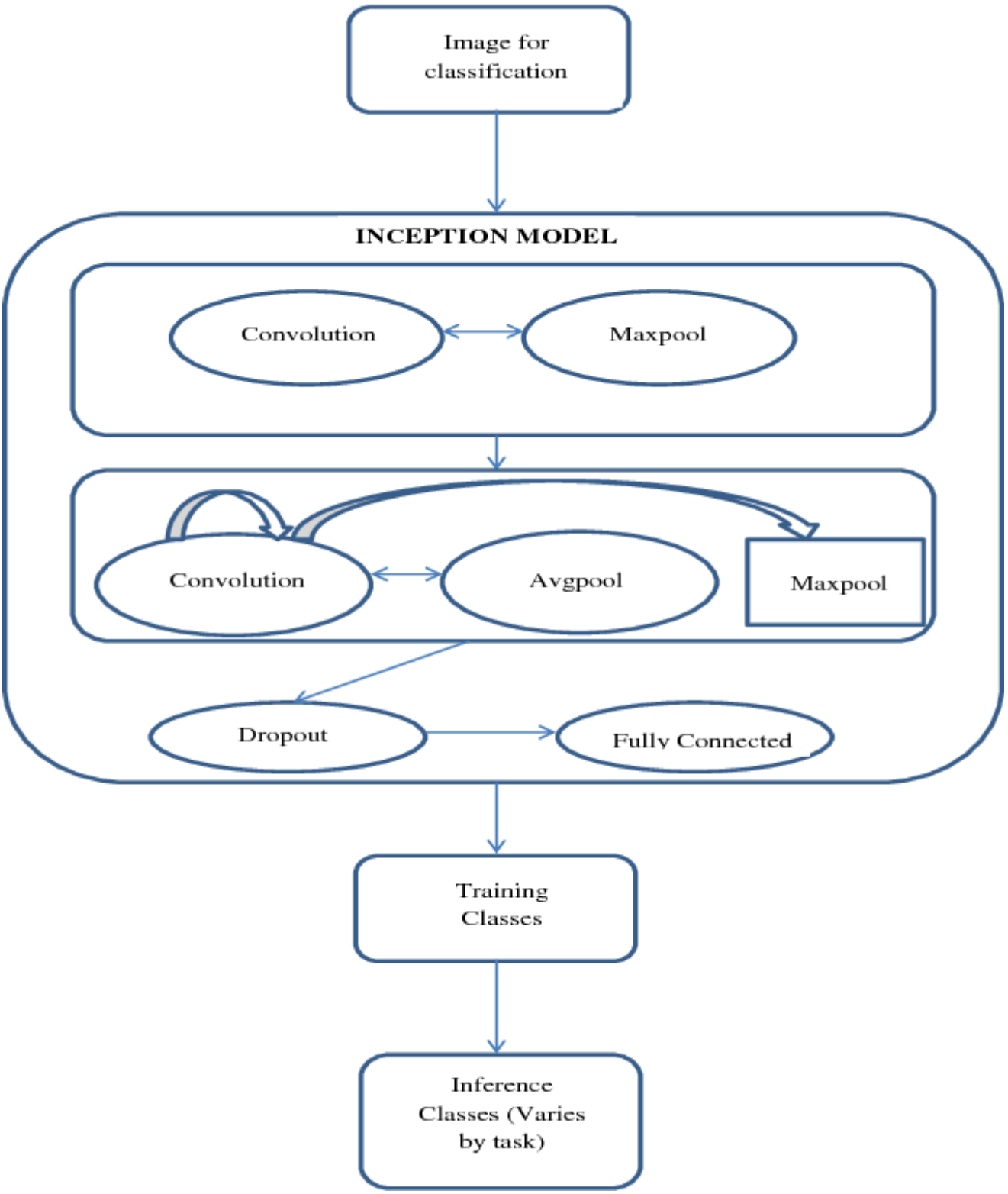
## 4.2. USE CASE DIAGRAM




## 4.3. BLOCK DIAGRAM



## 4.4. ARCHITECTURE DIAGRAM



# Chapter 5

## MODULES AND FUNCTIONALITIES

### Dataset for Image Recognition:

The ISIC (International Skin Imaging Collaboration Laboratory) database comprises images depicting benign, malignant, and normal skin lesions. These data are accessible in the ISIC repository, with additional images of various sizes hosted on Kaggle. A total of 673 images were amassed for training and testing purposes.

**Preprocessing of Data:** To derive crucial information from noisy images, eliminate unwanted elements, and generate high-quality images [26], preprocessing steps are applied to skin disease images. Ensuring optimal image quality involves a multi-step process:

1. Resize the image to fill the frame.

2. Convert RGB images to grayscale. Grayscale facilitates automatic processing commonly employed in digital systems.

3. Subject the grayscale image to five filtering procedures, addressing noise reduction, identification and removal of artifacts, hair, and skin pigmentation, as well as healing skin damage.

These filtering methods encompass non-local averaging and block-matching 3D filters. The evaluation of results is based on the image size parameter. Key performance indicators for preprocessing include Mean Squared Error (MSE), Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), and the universal quality index (UQI). UQI assesses both the original and reconstructed images. Various prefiltering techniques are compared using these metrics to identify the most effective approach for noise removal from the database.

Mean Squared Error (MSE): Equation (1) quantifies MSE by summing the squared error between the compressed image and the original image. A lower MSE value indicates better alignment with the original, signifying superior image quality.

**MODEL EVALUATION:**

The confusion matrix, illustrated in Below Figure, serves as a table commonly employed to depict how well a classification model performs on recognized test data. This matrix enables the computation of diverse metrics, including accuracy, recall, precision, and F1 score, offering a comprehensive assessment of the classification model's effectiveness.

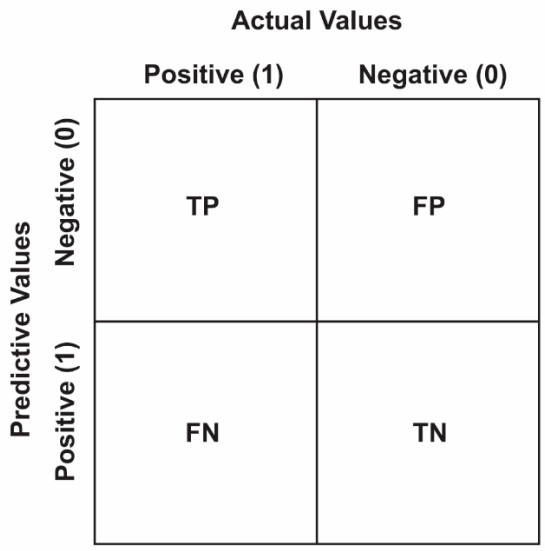



Fig.5.1: Confusion Matrix Model

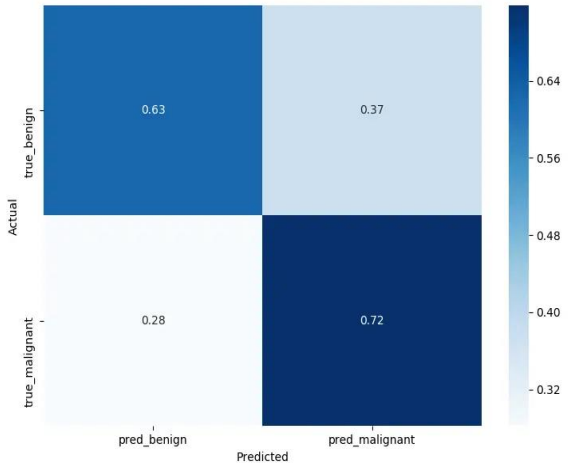



Fig.5.2: Confusion Matrix

True Positive (TP): An accurate prediction of a positive sample.

True Negative (TN): Correctly predicting negative class instances as negative instances.

False Positive (FP): Incorrectly predicting a negative class as positive (also referred to as a "Type I error").

False Negative (FN): Incorrectly predicting a positive sample as negative (also referred to as a "Type II error").

## Sensitivity

So our model gets about 0.72 probability of a positive test given that the patient has the disease (bottom right of the confusion matrix), that's often called sensitivity.

Sensitivity is a statistical measure that is widely used in medicine that is given by the following formula

$$\begin{aligned}\text{sensitivity} &= \frac{\text{number of true positives}}{\text{number of true positives} + \text{number of false negatives}} \\ &= \frac{\text{number of true positives}}{\text{total number of sick individuals in population}} \\ &= \text{probability of a positive test given that the patient has the disease}\end{aligned}$$

Fig.5.3: Sensitivity Formula

So in our example, out of all patients that have a malignant skin disease, we successfully predicted 72% of them as malignant, not bad but needs improvements.

## Specificity

The other metric is specificity, you can read it in the top left of the confusion matrix, we got about 63%. It is basically the probability of a negative test given that the patient is well:

$$\begin{aligned}\text{specificity} &= \frac{\text{number of true negatives}}{\text{number of true negatives} + \text{number of false positives}} \\ &= \frac{\text{number of true negatives}}{\text{total number of well individuals in population}} \\ &= \text{probability of a negative test given that the patient is well}\end{aligned}$$

Fig.5.4: Specificity Formula

In our example, out of all patients that has a benign, we predicted 63% of them as benign

## Image processing:

Image processing is used across many areas, one among them is the medicalfield. Detection of skin cancer using the image processing techniques provides precautionary measures and gives the accurate information to the patients. proposed system is a skin cancer detection and prediction system which is developed using Image

Processing Techniques. Affected skin image is taken asan input into the proposed system and the image is converted to grayscale imagewith which many functions can be applied. Once the Image acquisition is achieved pre-processing techniques are implemented using the Noise removal algorithms. After the noise removal, the image is enhanced using Histogram equalization to enhance the contrast & quality of the image. Histogram is also plotted to know the peak ratio of the affected pixels. Edge detection highlights the edges of the affected portionhelps to get an idea of where the cancer is majorly affected. Segmentation algorithms are implemented in order to threshold the image, so that the effected portion and unaffected portion is differentiated bydarkening the majorly infected portion. The Techniques used in the model are given below:

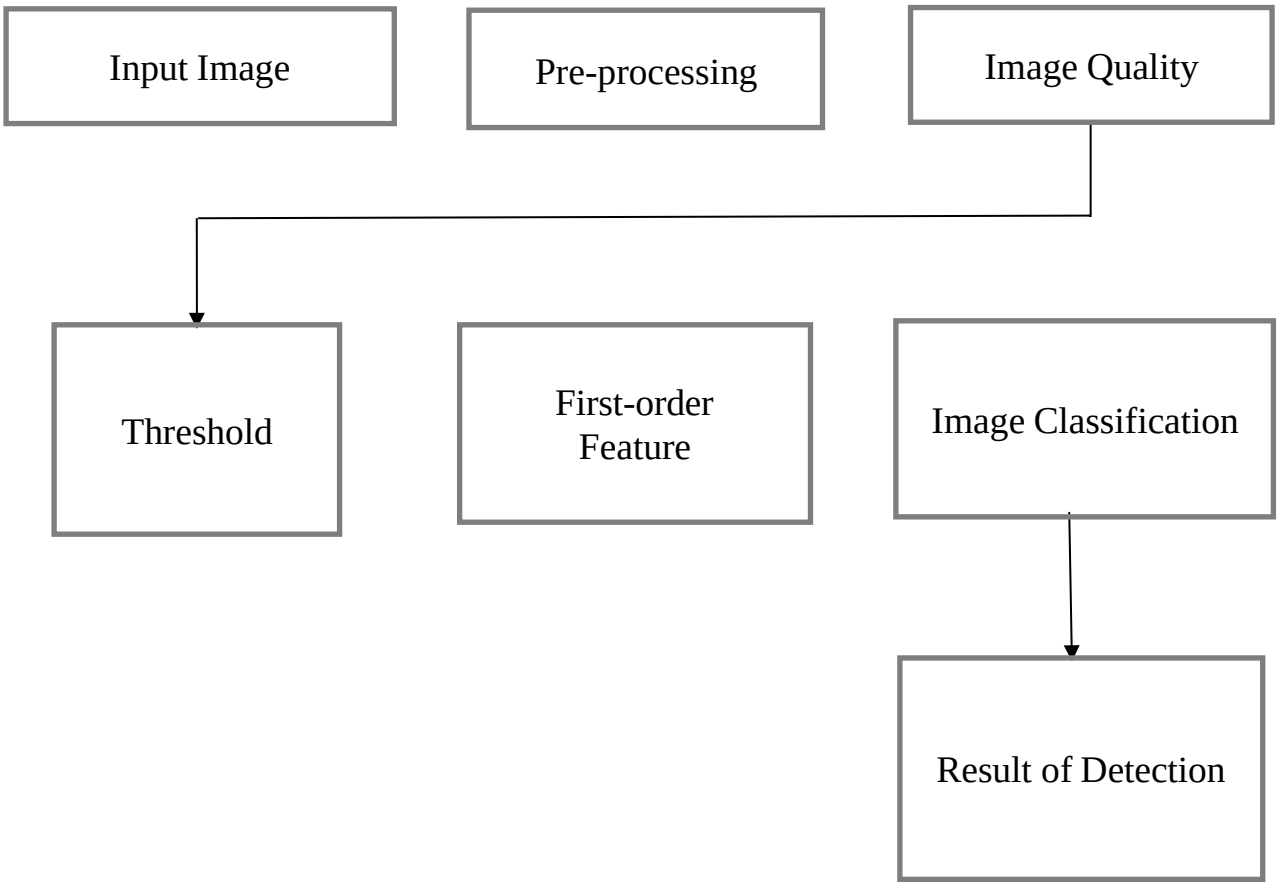



Fig.5.5: Proposed model

## Image acquisition:

Skin image is taken as an input from the patient through the proposed system. System is developed in such a way that it can accept only color images as its input. The color image is then converted to grayscale, which isa basic operation.

Fig.5.6: Original image

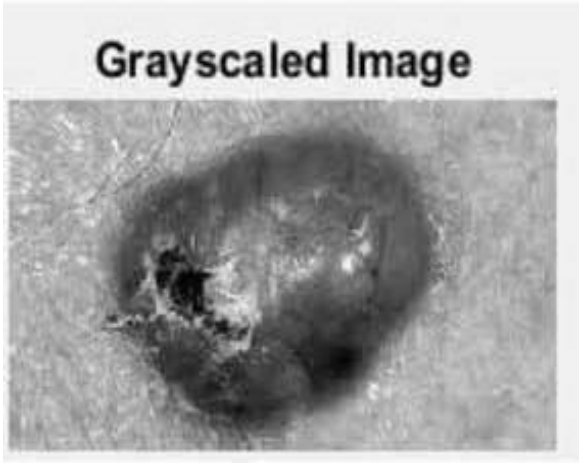



Fig.5.7: Gray scaled image

Conversion from RGB to Grayscale

## Noise removal:

Medical images are prone to much noises. Noise can be called as irrelevantinformation that surrounds the image in the current do- main. Obtained skin image is prone to much noise; these noises are removed using the Filter techniques. One of them is Median filter which eliminates the additional noisefrom the skin image and smoothens the quality of the image.

Fig.5.8: Noise removed image

Input image is added with a little noise to make it better for smoothening.Applies median filter on the pixels of the input im- age, which results in the process given in Fig5. The outcome of the noise removal process is a smoothened image.

## Image enhancement:

The outcome of the Noise removal is the filtered image which is given as aninput to stretch its quality and enhance its contrast that can be used based on thecontext of the domain. By enhancing the skin image clarity of the image increases more than the processed image. Below Figure is an example for Image Enhancement.

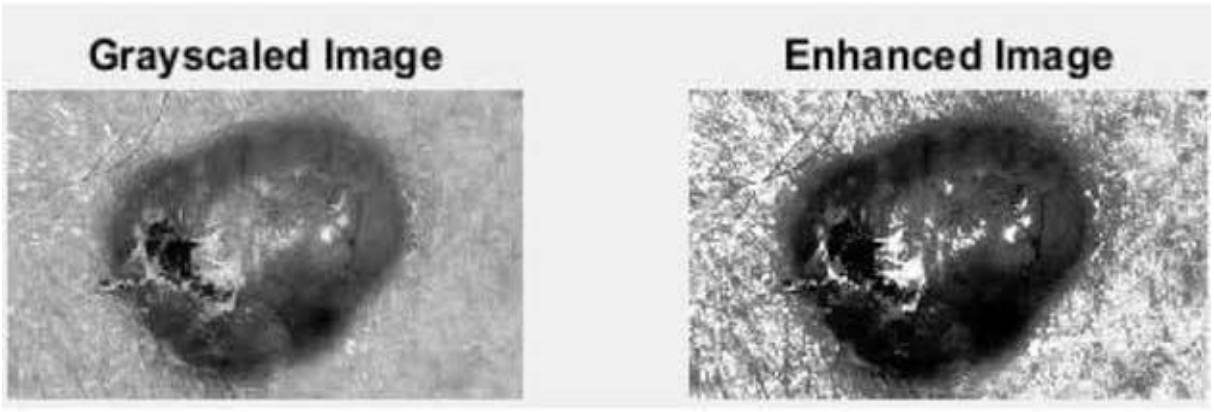



Fig.5.9: Image Is Enhanced from Grayscale to High Contrast Image.

## Edge detection:

As shown in Fig 7, Edges of the affected skin are detected using the canny edgedetector and applies binarization over the image to highlight into a black &white image to detect the affected portion.

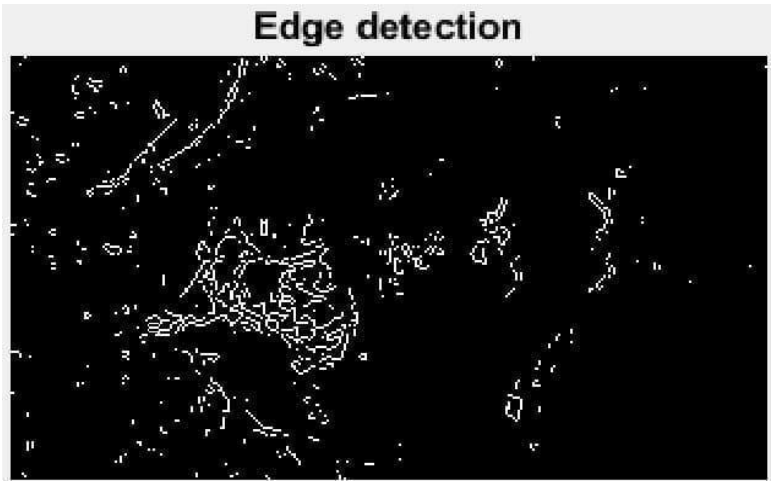



Fig.5.10: Edge Detected Image

## Plotting histogram:

To estimate the range of affected pixels' histogram is plotted, so that majorlyinfected portion can be easily determined.

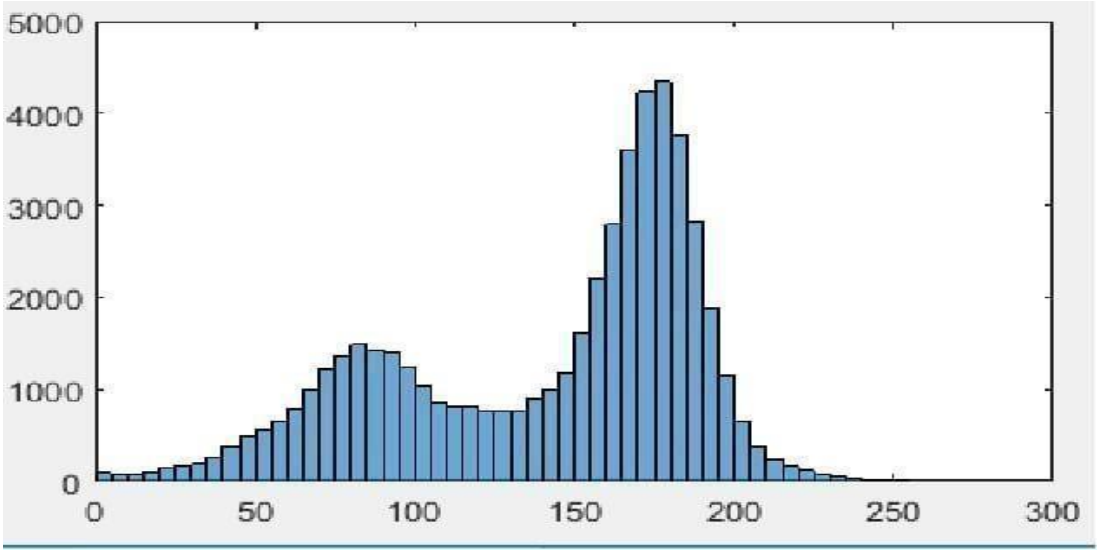



Fig.5.11: Histogram Representation

## Feature selection:

Enhanced image based on the domain is fed into the feature selection algorithms, where the proposed system follows the colour based feature selection model. Processed image's colour such as red, brown or pink is selectedand that part of the image is high- lighted & used for the further process. Figure9 exhibits the feature selection.

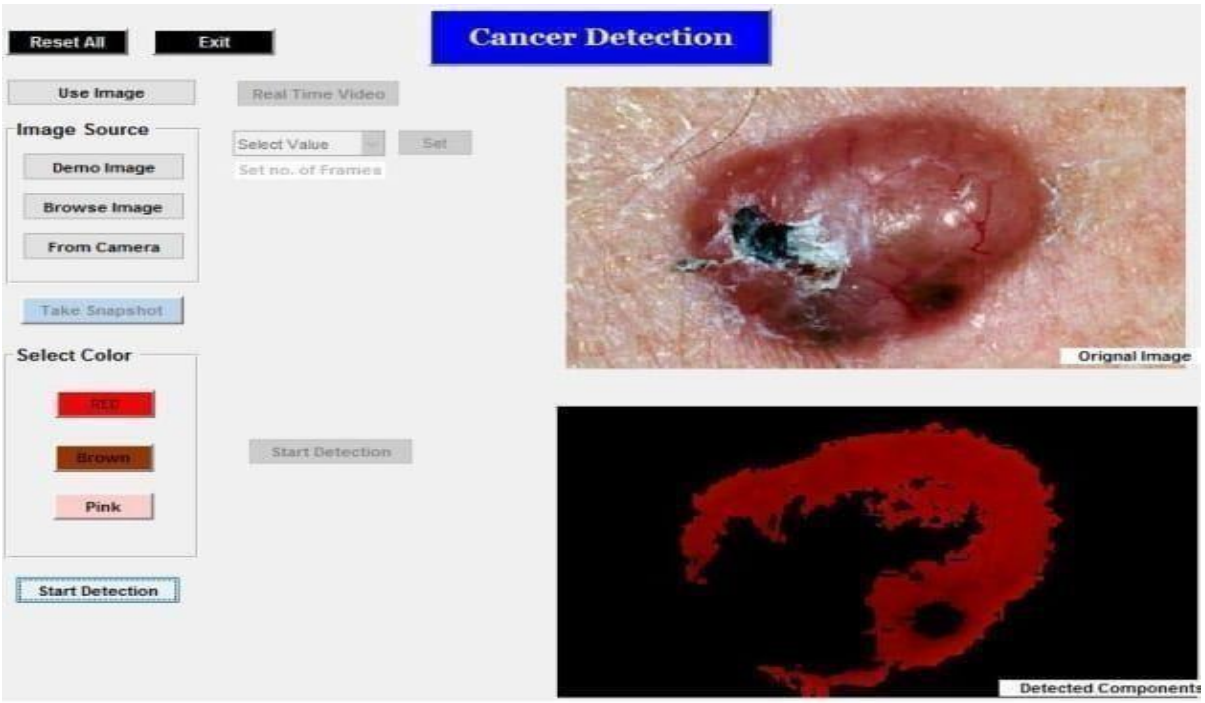



Fig.5.12: Feature Selection

## Segmentation:

The first level of segmentation is Thresholding. Thresholding value greaterthan170 is evaluated for all the channels of the input image and converts the output into a binary image for better system evaluation. Uses the individual channels tomap the affected portion on the top of the binary image

in order to highlight only the affected portion.

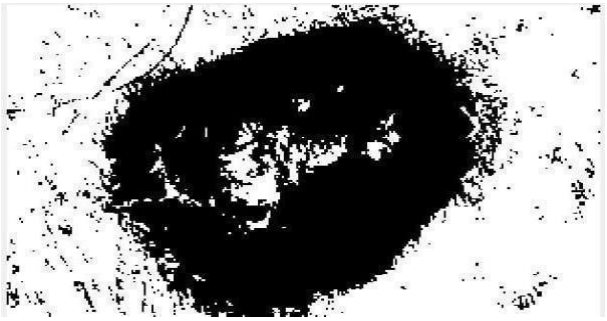


Fig.5.13: Affected Skin Background Removal

Fig.5.14: Affected Region Removal

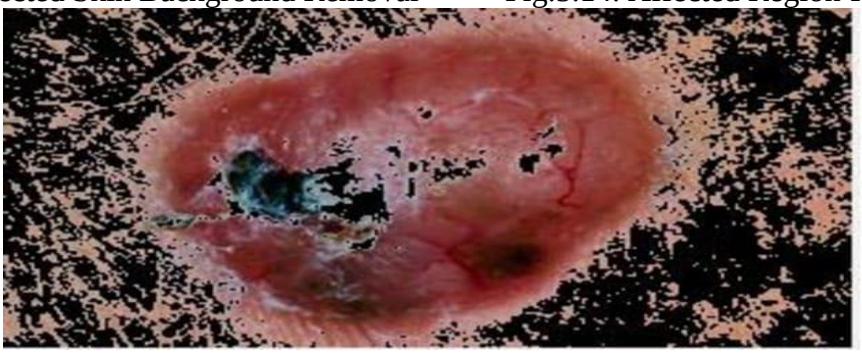


Fig.5.15: Mapping the affected region on the original background

The outcome of segmentation based on Thresholding approach

Uses binarization method to convert the image into binary, that can help inhighlighting the affected portion much easily.

## Feature extraction:

At the beginning, Convolutional Neural Network (CNN) is a set of stacked layers involving both nonlinear and linear processes. These layers are learned ina joint manner. The main building blocks of any CNN model are: convolutionallayer, pooling layer, nonlinear Rectified Linear Units (ReLU) layer connected toa regular multilayer neural network called fully connected layer, and a loss layerat the backend. CNN has known for its significant performance in applications as the visual tasks and natural language processing.

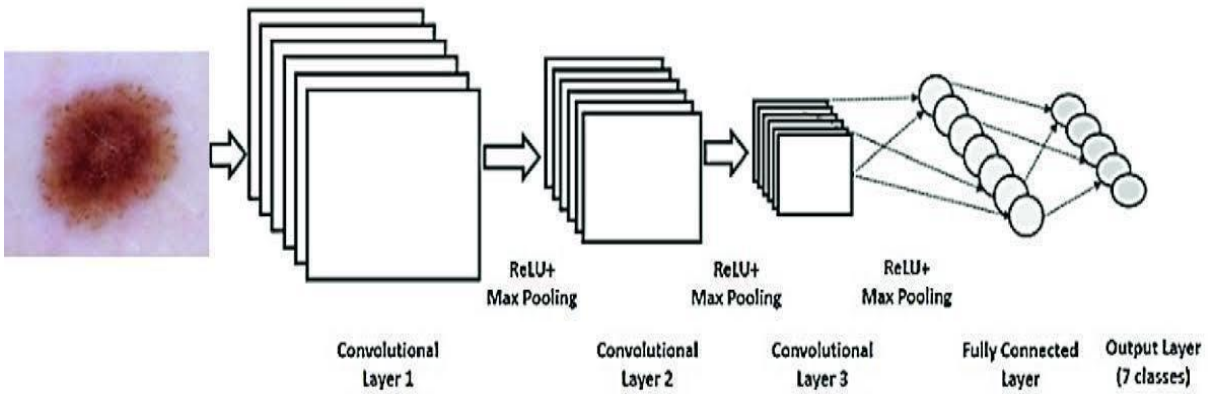



Fig.5.16: CNN block diagram

# Chapter 6
# APPENDIX

```
import tensorflow as tf
import tensorflow_hub as hub
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
from tensorflow.keras.utils import get_file
from sklearn.metrics import roc_curve, auc, confusion_matrix
from imblearn.metrics import sensitivity_score, specificity_score

import os
import glob
import zipfile
import random

# to get consistent results after multiple runs
tf.random.set_seed(7)
np.random.seed(7)
random.seed(7)

# 0 for benign, 1 for malignant
class_names = ["benign", "malignant"]


def download_and_extract_dataset():
  # dataset from https://github.com/udacity/dermatologist-ai
  # 5.3GB
  train_url = https://s3-us-west-1.amazonaws.com/udacity-dlnfd/datasets/skin-cancer/train.zip

  # 824.5MB
  valid_url = https://s3-us-west-1.amazonaws.com/udacity-dlnfd/datasets/skin-cancer/valid.zip

  # 5.1GB
  test_url  = https://s3-us-west-1.amazonaws.com/udacity-dlnfd/datasets/skin-cancer/test.zip

  for i, download_link in enumerate([valid_url, train_url, test_url]):
    temp_file = f"temp{i}.zip"
    data_dir = get_file(origin=download_link, fname=os.path.join(os.getcwd(), temp_file))
    print("Extracting", download_link)
    with zipfile.ZipFile(data_dir, "r") as z:
      z.extractall("data")
    # remove the temp file
```

```
    os.remove(temp_file)

# comment the below line if you already downloaded the dataset
download_and_extract_dataset()
# preparing data
# generate CSV metadata file to read img paths and labels from it
def generate_csv(folder, label2int):
    folder_name = os.path.basename(folder)
    labels = list(label2int)
    # generate CSV file
    df = pd.DataFrame(columns=["filepath", "label"])
    i = 0
    for label in labels:
        print("Reading", os.path.join(folder, label, "*"))
        for filepath in glob.glob(os.path.join(folder, label, "*")):
            df.loc[i] = [filepath, label2int[label]]
            i += 1
    output_file = f"{folder_name}.csv"
    print("Saving", output_file)
    df.to_csv(output_file)

# generate CSV files for all data portions, labeling nevus and seborrheic
keratosis
# as 0 (benign), and melanoma as 1 (malignant)
# you should replace "data" path to your extracted dataset path
# don't replace if you used download_and_extract_dataset() function
generate_csv("data/train", {"nevus": 0, "seborrheic_keratosis": 0, "melanoma":
1})
generate_csv("data/valid", {"nevus": 0, "seborrheic_keratosis": 0, "melanoma":
1})
generate_csv("data/test", {"nevus": 0, "seborrheic_keratosis": 0, "melanoma": 1})
# loading data
train_metadata_filename = "train.csv"
valid_metadata_filename = "valid.csv"
# load CSV files as DataFrames
df_train = pd.read_csv(train_metadata_filename)
df_valid = pd.read_csv(valid_metadata_filename)
n_training_samples = len(df_train)
n_validation_samples = len(df_valid)
print("Number of training samples:", n_training_samples)
print("Number of validation samples:", n_validation_samples)
train_ds = tf.data.Dataset.from_tensor_slices((df_train["filepath"],
df_train["label"]))
valid_ds = tf.data.Dataset.from_tensor_slices((df_valid["filepath"],
df_valid["label"]))
# preprocess data
def decode_img(img):
  # convert the compressed string to a 3D uint8 tensor
  img = tf.image.decode_jpeg(img, channels=3)
```

```
  # Use `convert_image_dtype` to convert to floats in the [0,1] range.
  img = tf.image.convert_image_dtype(img, tf.float32)
  # resize the image to the desired size.
  return tf.image.resize(img, [299, 299])


def process_path(filepath, label):
  # load the raw data from the file as a string
  img = tf.io.read_file(filepath)
  img = decode_img(img)
  return img, label


valid_ds = valid_ds.map(process_path)
train_ds = train_ds.map(process_path)
# test_ds = test_ds
for image, label in train_ds.take(1):
    print("Image shape:", image.shape)
    print("Label:", label.numpy())
# training parameters
batch_size = 64
optimizer = "rmsprop"
def prepare_for_training(ds, cache=True, batch_size=64,
shuffle_buffer_size=1000):
  if cache:
    if isinstance(cache, str):
      ds = ds.cache(cache)
    else:
      ds = ds.cache()
  # shuffle the dataset
  ds = ds.shuffle(buffer_size=shuffle_buffer_size)

  # Repeat forever
  ds = ds.repeat()
  # split to batches
  ds = ds.batch(batch_size)

  # `prefetch` lets the dataset fetch batches in the background while the model
  # is training.
  ds = ds.prefetch(buffer_size=tf.data.experimental.AUTOTUNE)

  return ds


valid_ds = prepare_for_training(valid_ds, batch_size=batch_size, cache="valid-
cached-data")
train_ds = prepare_for_training(train_ds, batch_size=batch_size, cache="train-
cached-data")
batch = next(iter(valid_ds))
```

```
def show_batch(batch):
  plt.figure(figsize=(12,12))
  for n in range(25):
      ax = plt.subplot(5,5,n+1)
      plt.imshow(batch[0][n])
      plt.title(class_names[batch[1][n].numpy()].title())
      plt.axis('off')

show_batch(batch)
# building the model
# InceptionV3 model & pre-trained weights
module_url = "https://tfhub.dev/google/tf2-preview/inception_v3/feature_vector/4"
m = tf.keras.Sequential([
    hub.KerasLayer(module_url, output_shape=[2048], trainable=False),
    tf.keras.layers.Dense(1, activation="sigmoid")
])

m.build([None, 299, 299, 3])
m.compile(loss="binary_crossentropy", optimizer=optimizer, metrics=["accuracy"])
m.summary()
model_name = f"benign-vs-malignant_{batch_size}_{optimizer}"
tensorboard = tf.keras.callbacks.TensorBoard(log_dir=os.path.join("logs",
model_name))
# saves model checkpoint whenever we reach better weights
modelcheckpoint = tf.keras.callbacks.ModelCheckpoint(model_name +
"_{val_loss:.3f}.h5", save_best_only=True, verbose=1)

history = m.fit(train_ds, validation_data=valid_ds,
                steps_per_epoch=n_training_samples // batch_size,
                validation_steps=n_validation_samples // batch_size, verbose=1,
epochs=100,
                callbacks=[tensorboard, modelcheckpoint])
# evaluation

# load testing set
test_metadata_filename = "test.csv"
df_test = pd.read_csv(test_metadata_filename)
n_testing_samples = len(df_test)
print("Number of testing samples:", n_testing_samples)
test_ds = tf.data.Dataset.from_tensor_slices((df_test["filepath"],
df_test["label"]))

def prepare_for_testing(ds, cache=True, shuffle_buffer_size=1000):
  # This is a small dataset, only load it once, and keep it in memory.
  # use `.cache(filename)` to cache preprocessing work for datasets that don't
  # fit in memory.
  if cache:
    if isinstance(cache, str):
      ds = ds.cache(cache)
```

```
    else:
      ds = ds.cache()

  ds = ds.shuffle(buffer_size=shuffle_buffer_size)

  return ds


test_ds = test_ds.map(process_path)
test_ds = prepare_for_testing(test_ds, cache="test-cached-data")
# convert testing set to numpy array to fit in memory (don't do that when testing
# set is too large)
y_test = np.zeros((n_testing_samples,))
X_test = np.zeros((n_testing_samples, 299, 299, 3))
for i, (img, label) in enumerate(test_ds.take(n_testing_samples)):
  # print(img.shape, label.shape)
  X_test[i] = img
  y_test[i] = label.numpy()

print("y_test.shape:", y_test.shape)
# load the weights with the least loss
m.load_weights("benign-vs-malignant_64_rmsprop_0.399.h5")
print("Evaluating the model...")
loss, accuracy = m.evaluate(X_test, y_test, verbose=0)
print("Loss:", loss, "  Accuracy:", accuracy)
from sklearn.metrics import accuracy_score

def get_predictions(threshold=None):
  """
  Returns predictions for binary classification given `threshold`
  For instance, if threshold is 0.3, then it'll output 1 (malignant) for that
sample if
  the probability of 1 is 30% or more (instead of 50%)
  """
  y_pred = m.predict(X_test)
  if not threshold:
    threshold = 0.5
  result = np.zeros((n_testing_samples,))
  for i in range(n_testing_samples):
    # test melanoma probability
    if y_pred[i][0] >= threshold:
      result[i] = 1
    # else, it's 0 (benign)
  return result

threshold = 0.23
# get predictions with 23% threshold
# which means if the model is 23% sure or more that is malignant,
# it's assigned as malignant, otherwise it's benign
```

```
y_pred = get_predictions(threshold)
accuracy_after = accuracy_score(y_test, y_pred)
print("Accuracy after setting the threshold:", accuracy_after)
import seaborn as sns
from sklearn.metrics import roc_curve, auc, confusion_matrix

def plot_confusion_matrix(y_test, y_pred):
  cmn = confusion_matrix(y_test, y_pred)
  # Normalise
  cmn = cmn.astype('float') / cmn.sum(axis=1)[:, np.newaxis]
  # print it
  print(cmn)
  fig, ax = plt.subplots(figsize=(10,10))
  sns.heatmap(cmn, annot=True, fmt='.2f',
              xticklabels=[f"pred_{c}" for c in class_names],
              yticklabels=[f"true_{c}" for c in class_names],
              cmap="Blues"
              )
  plt.ylabel('Actual')
  plt.xlabel('Predicted')
  # plot the resulting confusion matrix
  plt.show()


def plot_roc_auc(y_true, y_pred):
    """
    This function plots the ROC curves and provides the scores.
    """
    # prepare for figure
    plt.figure()
    fpr, tpr, _ = roc_curve(y_true, y_pred)
    # obtain ROC AUC
    roc_auc = auc(fpr, tpr)
    # print score
    print(f"ROC AUC: {roc_auc:.3f}")
    # plot ROC curve
    plt.plot(fpr, tpr, color="blue", lw=2,
                label='ROC curve (area = {f:.2f})'.format(d=1, f=roc_auc))
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('ROC curves')
    plt.legend(loc="lower right")
    plt.show()

plot_confusion_matrix(y_test, y_pred)
plot_roc_auc(y_test, y_pred)
sensitivity = sensitivity_score(y_test, y_pred)
```

```
specificity = specificity_score(y_test, y_pred)

print("Melanoma Sensitivity:", sensitivity)
print("Melanoma Specificity:", specificity)
def plot_images(X_test, y_pred, y_test):
  predicted_class_names = np.array([class_names[int(round(id))] for id in
y_pred])
  # some nice plotting
  plt.figure(figsize=(10,9))
  for n in range(30, 60):
      plt.subplot(6,5,n-30+1)
      plt.subplots_adjust(hspace = 0.3)
      plt.imshow(X_test[n])
      # get the predicted label
      predicted_label = predicted_class_names[n]
      # get the actual true label
      true_label = class_names[int(round(y_test[n]))]
      if predicted_label == true_label:
          color = "blue"
          title = predicted_label.title()
      else:
          color = "red"
          title = f"{predicted_label.title()}, true:{true_label.title()}"
      plt.title(title, color=color)
      plt.axis('off')
  _ = plt.suptitle("Model predictions (blue: correct, red: incorrect)")
  plt.show()

plot_images(X_test, y_pred, y_test)
# a function given a function, it predicts the class of the image
def predict_image_class(img_path, model, threshold=0.5):
  img = tf.keras.preprocessing.image.load_img(img_path, target_size=(299, 299))
  img = tf.keras.preprocessing.image.img_to_array(img)
  img = tf.expand_dims(img, 0) # Create a batch
  img = tf.keras.applications.inception_v3.preprocess_input(img)
  img = tf.image.convert_image_dtype(img, tf.float32)
  predictions = model.predict(img)
  score = predictions.squeeze()
  if score >= threshold:
    print(f"This image is {100 * score:.2f}% malignant.")
  else:
    print(f"This image is {100 * (1 - score):.2f}% benign.")

  plt.imshow(img[0])
  plt.axis('off')
  plt.show()
predict_image_class("data/test/melanoma/ISIC_0013767.jpg", m)
```

```
predict_image_class("data/test/melanoma/ISIC_0013767.jpg", m)

1/1 [==============================] - 0s 27ms/step
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or |
[[0.6803772]]
0.6803772
This image is 68.04% malignant.
```

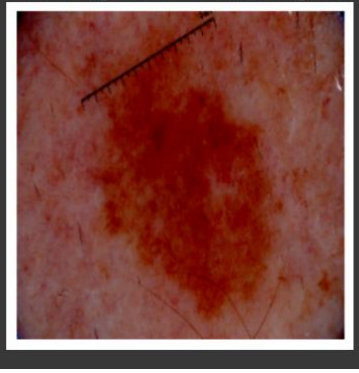


Fig.6.1:Output Of SkinCancer Detection

## IMPLEMENTATION

SkinCancerDetectionTensorFlow

colab.research.google.com/drive/1YXL8r1v88Nw2aiuc35_CZCZ_zrOuyR8u?usp=sharing#scrollTo=uUEJ9zVKoloS

+ Code + Text Cannot save changes

```
import tensorflow as tf
import tensorflow_hub as hub
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
from tensorflow.keras.utils import get_file
from sklearn.metrics import roc_curve, auc, confusion_matrix
from imblearn.metrics import sensitivity_score, specificity_score

import os
import glob
import zipfile
import random

# to get consistent results after multiple runs
tf.random.set_seed(7)
np.random.seed(7)
random.seed(7)

# 0 for benign, 1 for malignant
class_names = ["benign", "malignant"]


def download_and_extract_dataset():
  # dataset from https://github.com/udacity/dermatologist-ai
  # 5.3GB
  train_url = "https://s3-us-west-1.amazonaws.com/udacity-dlnfd/datasets/skin-cancer/train.zip"
```

Executing (1m 3s) <cell line: 6> › error_handler() › fit() › error_handler() › __call__() › _call() › call_function() › _call_flat() › flat_call() › __call__() › call_function() › quick_execute()

SkinCancerDetectionTensorFlow

colab.research.google.com/drive/1YXL8r1v88Nw2aiuc35_CZCZ_zrOuyR8u?usp=sharing#scrollTo=uUEJ9zVKoloS

+ Code + Text Cannot save changes

```
# building the model
# InceptionV3 model & pre-trained weights
module_url = "https://tfhub.dev/google/tf2-preview/inception_v3/feature_vector/4"
m = tf.keras.Sequential([
    hub.KerasLayer(module_url, output_shape=[2048], trainable=False),
    tf.keras.layers.Dense(1, activation="sigmoid")
])

m.build([None, 299, 299, 3])
m.compile(loss="binary_crossentropy", optimizer=optimizer, metrics=["accuracy"])
m.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 keras_layer (KerasLayer)    (None, 2048)              21802784

 dense (Dense)               (None, 1)                 2049

=================================================================
Total params: 21804833 (83.18 MB)
Trainable params: 2049 (8.00 KB)
Non-trainable params: 21802784 (83.17 MB)
_________________________________________________________________
```

```
model_name = f"benign-vs-malignant_{batch_size}_{optimizer}"
```

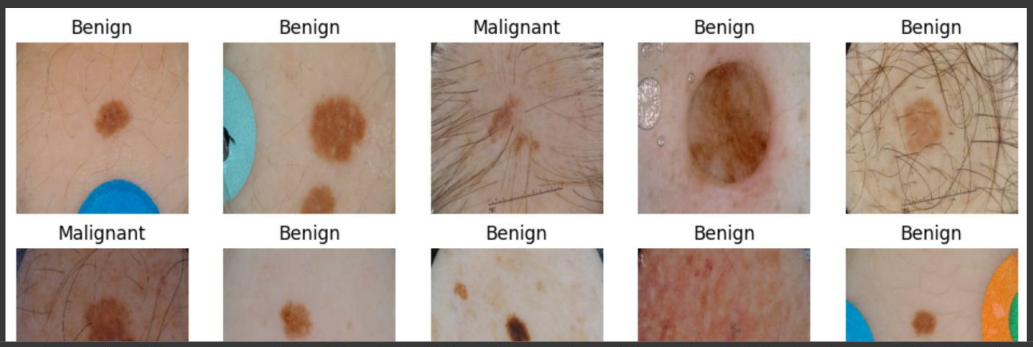
Executing (8m 0s) <cell line: 6> > error_handler() > fit() > error_handler() > __call__() > _call() > call_function() > _call_flat() > flat_call() > __call__() > call_function() > quick_execute()

SkinCancerDetectionTensorFlow

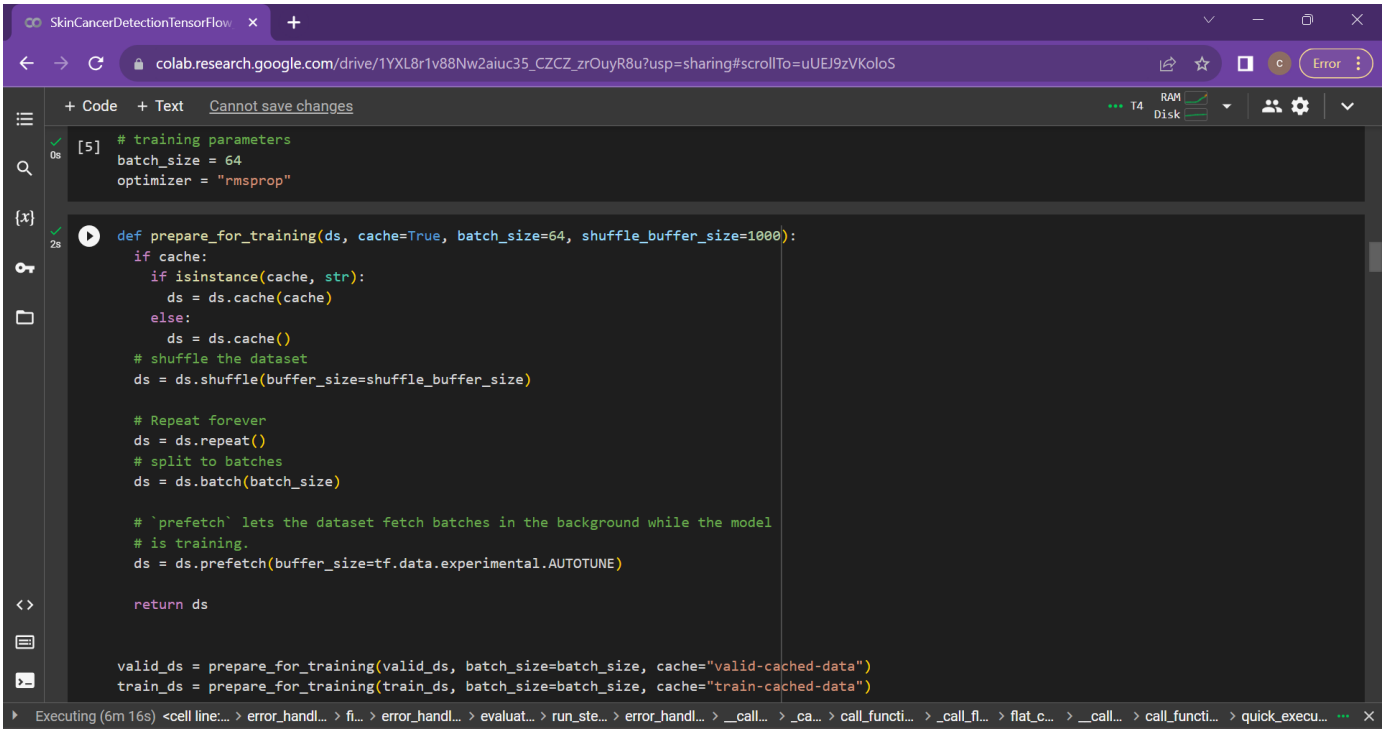
colab.research.google.com/drive/1YXL8r1v88Nw2aiuc35_CZCZ_zrOuyR8u?usp=sharing#scrollTo=uUEJ9zVKoloS

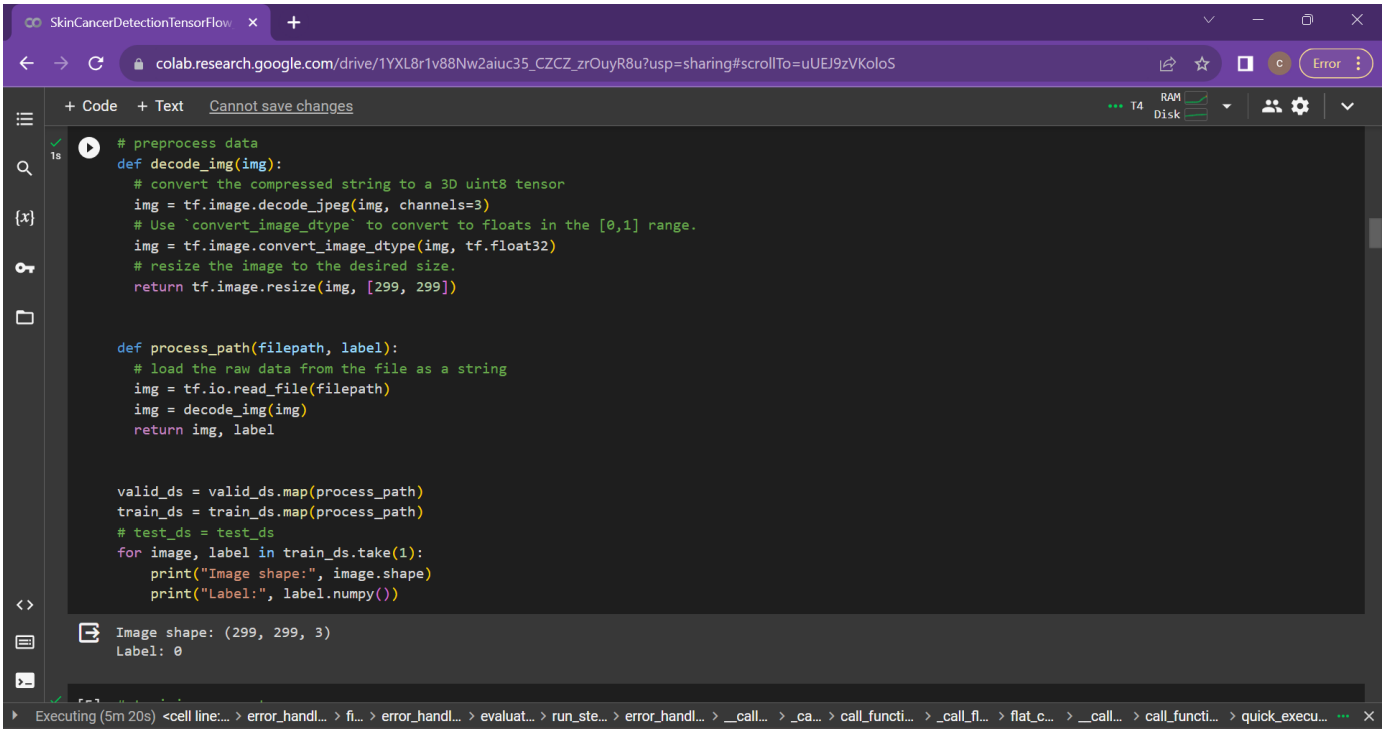
+ Code + Text Cannot save changes

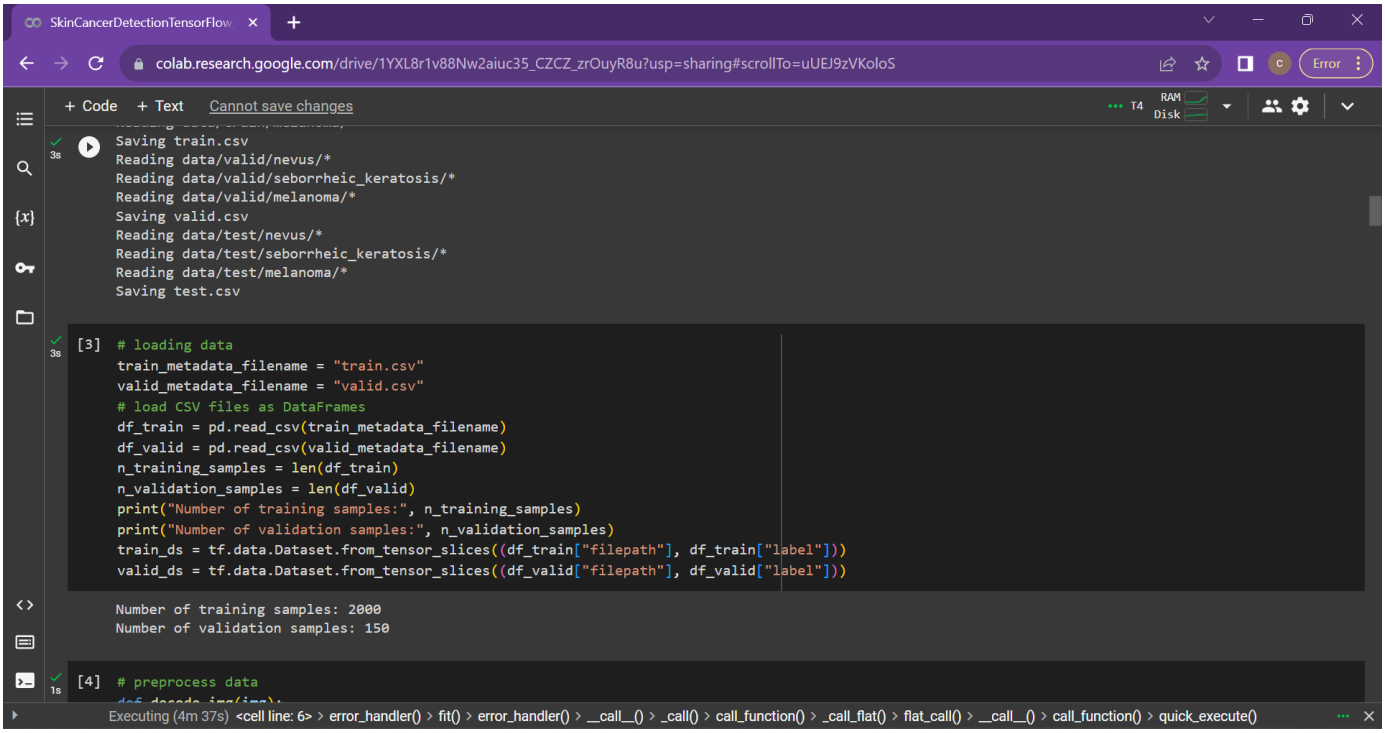
```
batch = next(iter(valid_ds))

def show_batch(batch):
  plt.figure(figsize=(12,12))
  for n in range(25):
      ax = plt.subplot(5,5,n+1)
      plt.imshow(batch[0][n])
      plt.title(class_names[batch[1][n].numpy()].title())
      plt.axis('off')

show_batch(batch)
```

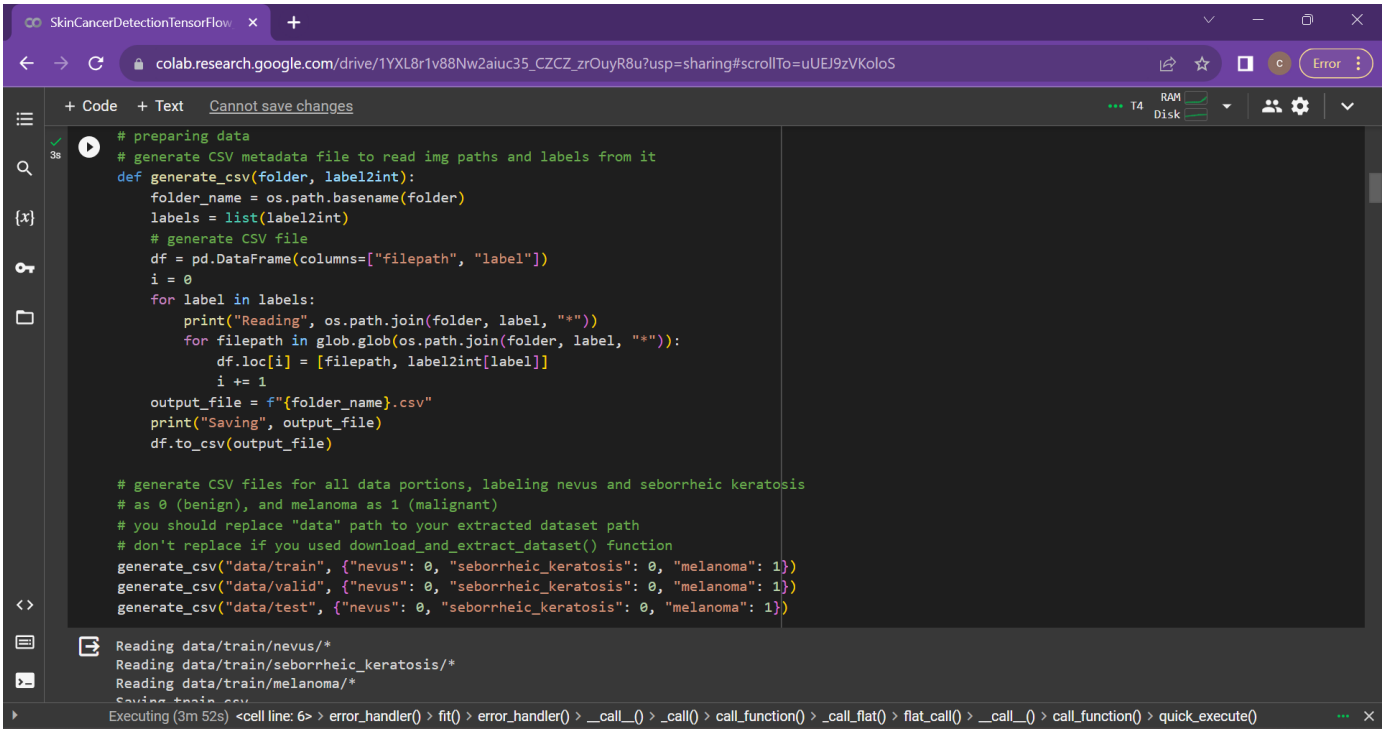


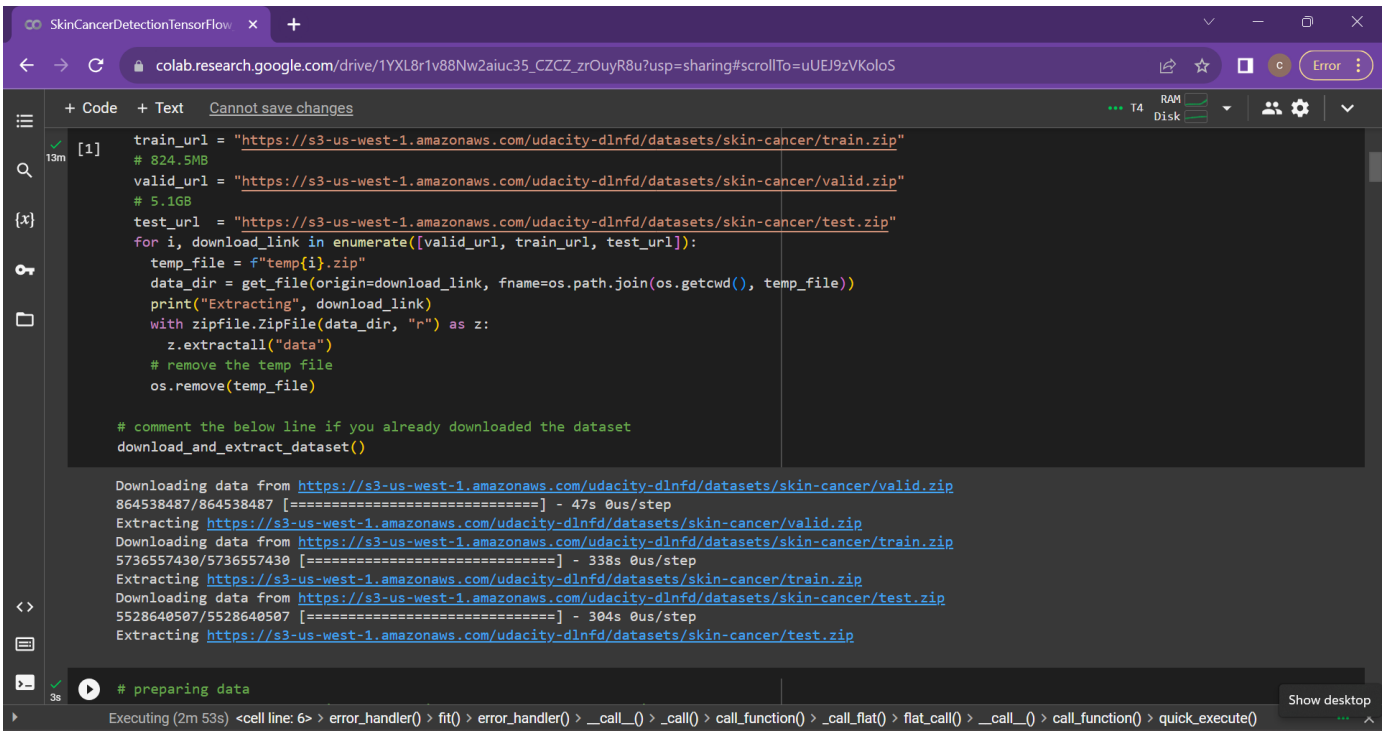

Executing (7m 9s) <cell line: 6> > error_handler() > fit() > error_handler() > __call__() > _call() > call_function() > _call_flat() > flat_call() > __call__() > call_function() > quick_execute()

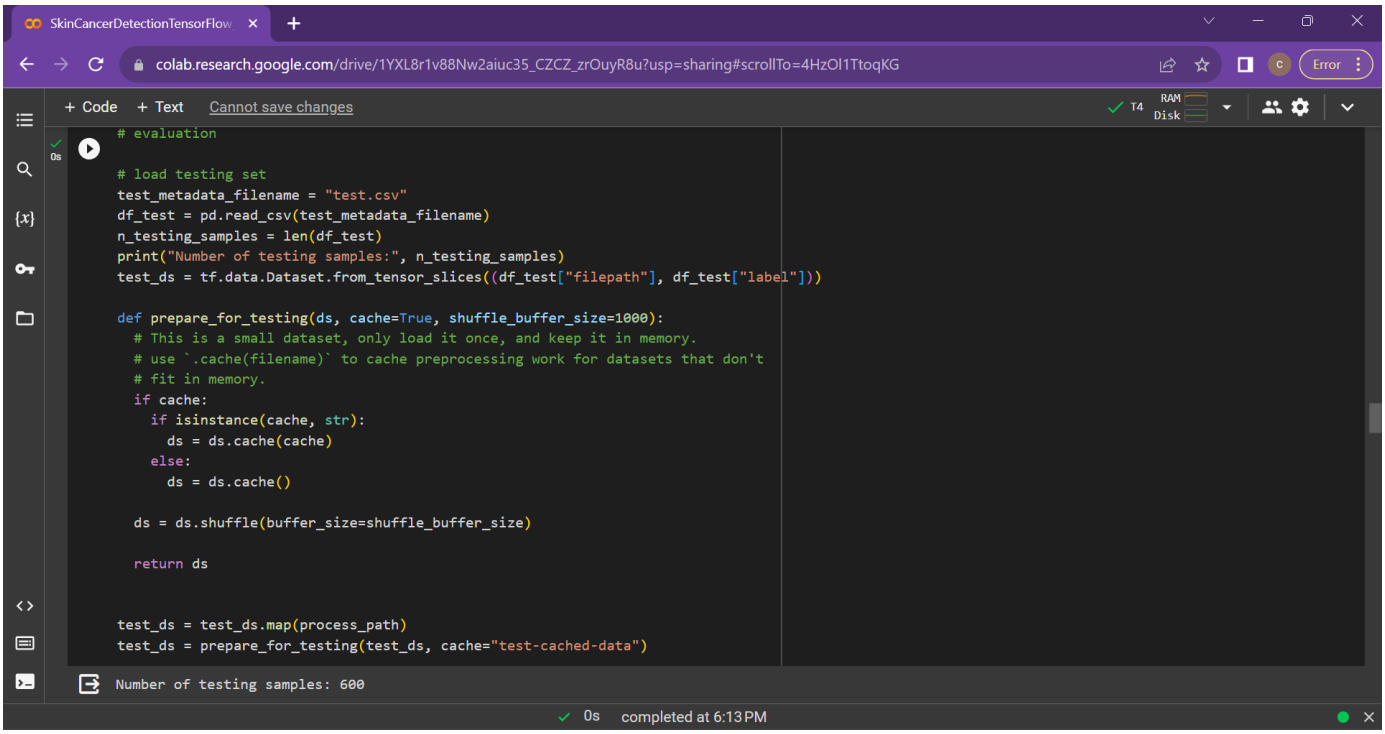

SkinCancerDetectionTensorFlow
colab.research.google.com/drive/1YXL8r1v88Nw2aiuc35_CZCZ_zrOuyR8u?usp=sharing#scrollTo=uUEJ9zVKoloS
Error
+ Code
+ Text
Cannot save changes
T4
RAM
Disk
[5]
0s
# training parameters
batch_size = 64
optimizer = "rmsprop"
2s
def prepare_for_training(ds, cache=True, batch_size=64, shuffle_buffer_size=1000):
if cache:
if isinstance(cache, str):
ds = ds.cache(cache)
else:
ds = ds.cache()
# shuffle the dataset
ds = ds.shuffle(buffer_size=shuffle_buffer_size)
# Repeat forever
ds = ds.repeat()
# split to batches
ds = ds.batch(batch_size)
# `prefetch` lets the dataset fetch batches in the background while the model
# is training.
ds = ds.prefetch(buffer_size=tf.data.experimental.AUTOTUNE)
return ds
valid_ds = prepare_for_training(valid_ds, batch_size=batch_size, cache="valid-cached-data")
train_ds = prepare_for_training(train_ds, batch_size=batch_size, cache="train-cached-data")
Executing (6m 16s)

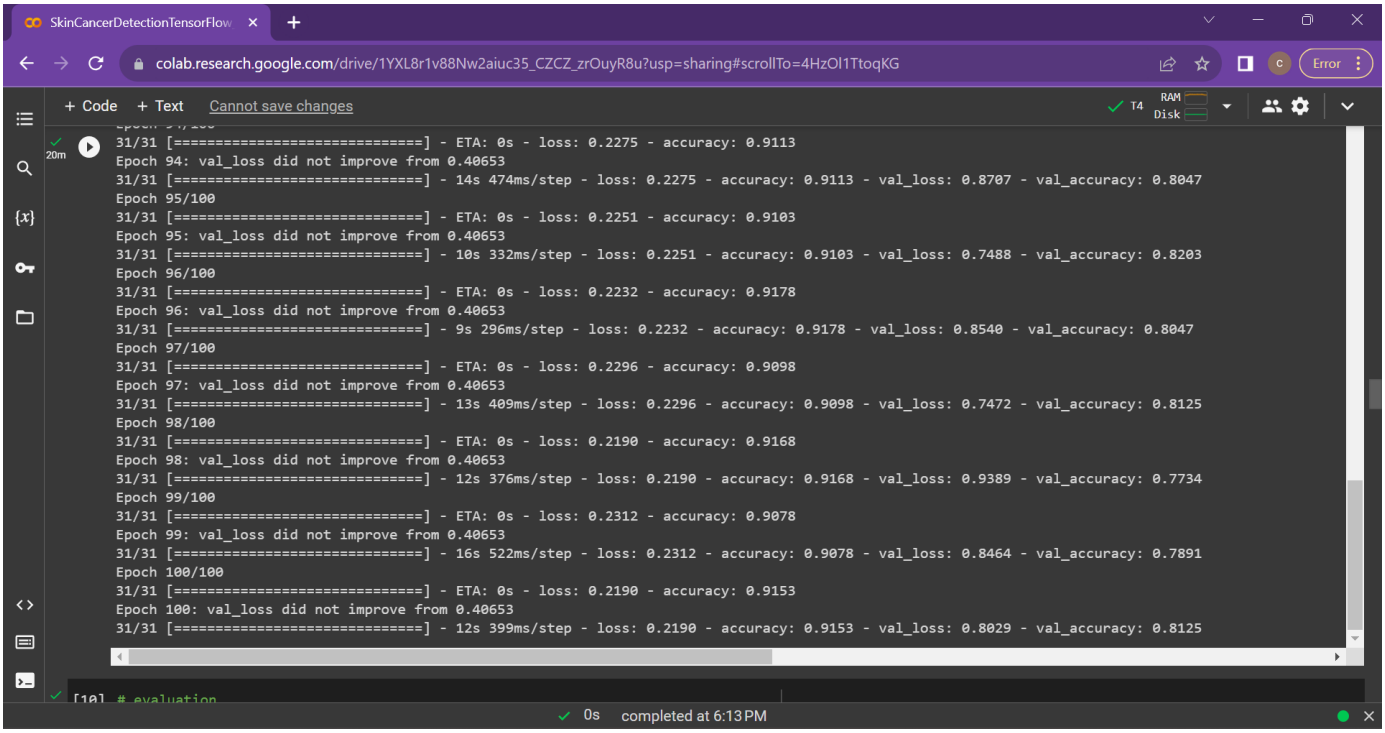

SkinCancerDetectionTensorFlow
colab.research.google.com/drive/1YXL8r1v88Nw2aiuc35_CZCZ_zrOuyR8u?usp=sharing#scrollTo=uUEJ9zVKoloS
Error
+ Code
+ Text
Cannot save changes
T4
RAM
Disk
1s
# preprocess data
def decode_img(img):
# convert the compressed string to a 3D uint8 tensor
img = tf.image.decode_jpeg(img, channels=3)
# Use `convert_image_dtype` to convert to floats in the [0,1] range.
img = tf.image.convert_image_dtype(img, tf.float32)
# resize the image to the desired size.
return tf.image.resize(img, [299, 299])
def process_path(filepath, label):
# load the raw data from the file as a string
img = tf.io.read_file(filepath)
img = decode_img(img)
return img, label
valid_ds = valid_ds.map(process_path)
train_ds = train_ds.map(process_path)
# test_ds = test_ds
for image, label in train_ds.take(1):
print("Image shape:", image.shape)
print("Label:", label.numpy())
Image shape: (299, 299, 3)
Label: 0
Executing (5m 20s)

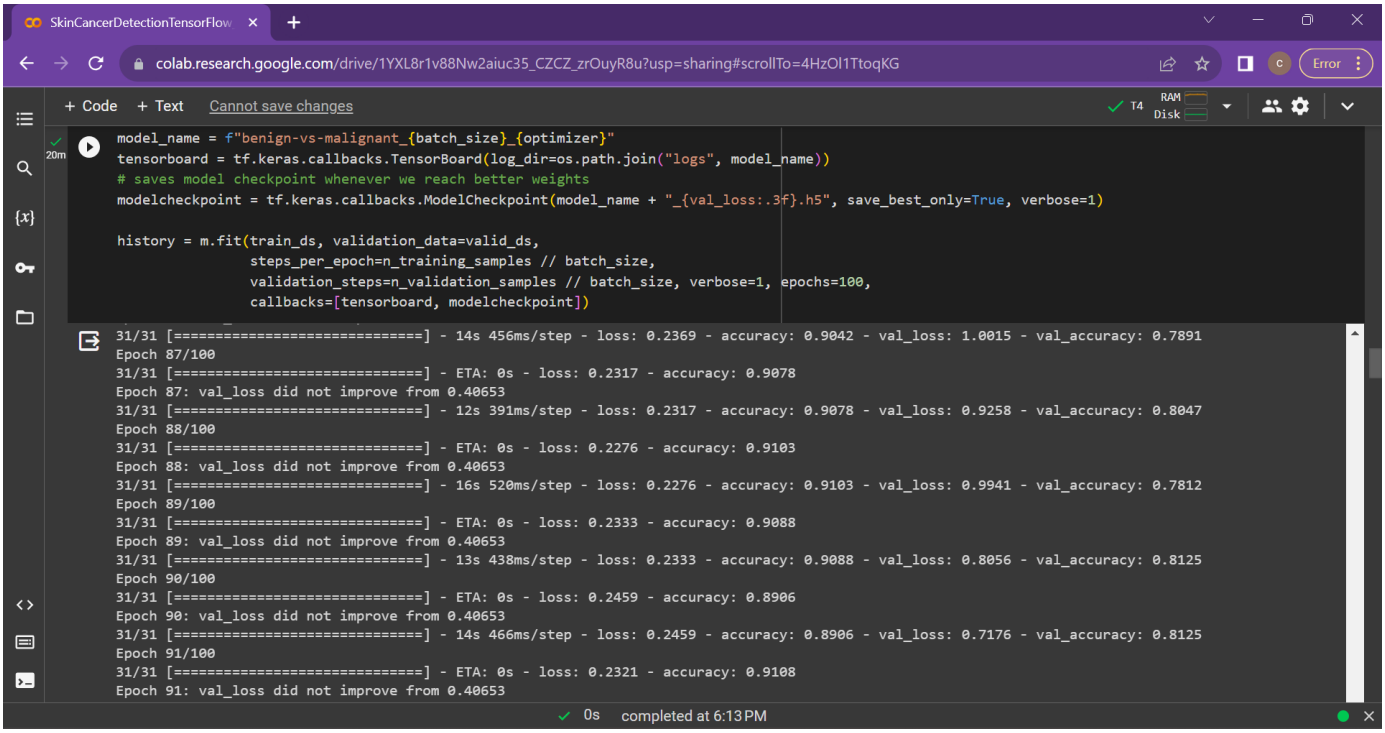

SkinCancerDetectionTensorFlow
colab.research.google.com/drive/1YXL8r1v88Nw2aiuc35_CZCZ_zrOuyR8u?usp=sharing#scrollTo=uUEJ9zVKoloS
Error
+ Code
+ Text
Cannot save changes
T4
RAM
Disk
Saving train.csv
Reading data/valid/nevus/*
Reading data/valid/seborrheic_keratosis/*
Reading data/valid/melanoma/*
Saving valid.csv
Reading data/test/nevus/*
Reading data/test/seborrheic_keratosis/*
Reading data/test/melanoma/*
Saving test.csv
[3] # loading data
train_metadata_filename = "train.csv"
valid_metadata_filename = "valid.csv"
# load CSV files as DataFrames
df_train = pd.read_csv(train_metadata_filename)
df_valid = pd.read_csv(valid_metadata_filename)
n_training_samples = len(df_train)
n_validation_samples = len(df_valid)
print("Number of training samples:", n_training_samples)
print("Number of validation samples:", n_validation_samples)
train_ds = tf.data.Dataset.from_tensor_slices((df_train["filepath"], df_train["label"]))
valid_ds = tf.data.Dataset.from_tensor_slices((df_valid["filepath"], df_valid["label"]))
Number of training samples: 2000
Number of validation samples: 150
[4] # preprocess data
Executing (4m 37s) <cell line: 6> › error_handler() › fit() › error_handler() › __call__() › _call() › call_function() › _call_flat() › flat_call() › __call__() › call_function() › quick_execute()


SkinCancerDetectionTensorFlow
colab.research.google.com/drive/1YXL8r1v88Nw2aiuc35_CZCZ_zrOuyR8u?usp=sharing#scrollTo=uUEJ9zVKoloS
Error
+ Code
+ Text
Cannot save changes
T4
RAM
Disk
# preparing data
# generate CSV metadata file to read img paths and labels from it
def generate_csv(folder, label2int):
folder_name = os.path.basename(folder)
labels = list(label2int)
# generate CSV file
df = pd.DataFrame(columns=["filepath", "label"])
i = 0
for label in labels:
print("Reading", os.path.join(folder, label, "*"))
for filepath in glob.glob(os.path.join(folder, label, "*")):
df.loc[i] = [filepath, label2int[label]]
i += 1
output_file = f"{folder_name}.csv"
print("Saving", output_file)
df.to_csv(output_file)
# generate CSV files for all data portions, labeling nevus and seborrheic keratosis
# as 0 (benign), and melanoma as 1 (malignant)
# you should replace "data" path to your extracted dataset path
# don't replace if you used download_and_extract_dataset() function
generate_csv("data/train", {"nevus": 0, "seborrheic_keratosis": 0, "melanoma": 1})
generate_csv("data/valid", {"nevus": 0, "seborrheic_keratosis": 0, "melanoma": 1})
generate_csv("data/test", {"nevus": 0, "seborrheic_keratosis": 0, "melanoma": 1})
Reading data/train/nevus/*
Reading data/train/seborrheic_keratosis/*
Reading data/train/melanoma/*
Executing (3m 52s) <cell line: 6> › error_handler() › fit() › error_handler() › __call__() › _call() › call_function() › _call_flat() › flat_call() › __call__() › call_function() › quick_execute()


SkinCancerDetectionTensorFlow
colab.research.google.com/drive/1YXL8r1v88Nw2aiuc35_CZCZ_zrOuyR8u?usp=sharing#scrollTo=uUEJ9zVKoloS
Error
+ Code
+ Text
Cannot save changes
T4
RAM
Disk
[1]
13m
train_url = "https://s3-us-west-1.amazonaws.com/udacity-dlnfd/datasets/skin-cancer/train.zip"
# 824.5MB
valid_url = "https://s3-us-west-1.amazonaws.com/udacity-dlnfd/datasets/skin-cancer/valid.zip"
# 5.1GB
test_url = "https://s3-us-west-1.amazonaws.com/udacity-dlnfd/datasets/skin-cancer/test.zip"
for i, download_link in enumerate([valid_url, train_url, test_url]):
temp_file = f"temp{i}.zip"
data_dir = get_file(origin=download_link, fname=os.path.join(os.getcwd(), temp_file))
print("Extracting", download_link)
with zipfile.ZipFile(data_dir, "r") as z:
z.extractall("data")
# remove the temp file
os.remove(temp_file)
# comment the below line if you already downloaded the dataset
download_and_extract_dataset()
Downloading data from https://s3-us-west-1.amazonaws.com/udacity-dlnfd/datasets/skin-cancer/valid.zip
864538487/864538487 [==========] - 47s 0us/step
Extracting https://s3-us-west-1.amazonaws.com/udacity-dlnfd/datasets/skin-cancer/valid.zip
Downloading data from https://s3-us-west-1.amazonaws.com/udacity-dlnfd/datasets/skin-cancer/train.zip
5736557430/5736557430 [==========] - 338s 0us/step
Extracting https://s3-us-west-1.amazonaws.com/udacity-dlnfd/datasets/skin-cancer/train.zip
Downloading data from https://s3-us-west-1.amazonaws.com/udacity-dlnfd/datasets/skin-cancer/test.zip
5528640507/5528640507 [==========] - 304s 0us/step
Extracting https://s3-us-west-1.amazonaws.com/udacity-dlnfd/datasets/skin-cancer/test.zip
# preparing data
3s
Executing (2m 53s) <cell line: 6> › error_handler() › fit() › error_handler() › __call__() › _call() › call_function() › _call_flat() › flat_call() › __call__() › call_function() › quick_execute()
Show desktop


SkinCancerDetectionTensorFlow
colab.research.google.com/drive/1YXL8r1v88Nw2aiuc35_CZCZ_zrOuyR8u?usp=sharing#scrollTo=4HzOl1TtoqKG
Error
+ Code
+ Text
Cannot save changes
T4
RAM
Disk
0s
# evaluation
# load testing set
test_metadata_filename = "test.csv"
df_test = pd.read_csv(test_metadata_filename)
n_testing_samples = len(df_test)
print("Number of testing samples:", n_testing_samples)
test_ds = tf.data.Dataset.from_tensor_slices((df_test["filepath"], df_test["label"]))
def prepare_for_testing(ds, cache=True, shuffle_buffer_size=1000):
# This is a small dataset, only load it once, and keep it in memory.
# use `.cache(filename)` to cache preprocessing work for datasets that don't
# fit in memory.
if cache:
if isinstance(cache, str):
ds = ds.cache(cache)
else:
ds = ds.cache()
ds = ds.shuffle(buffer_size=shuffle_buffer_size)
return ds
test_ds = test_ds.map(process_path)
test_ds = prepare_for_testing(test_ds, cache="test-cached-data")
Number of testing samples: 600
0s completed at 6:13 PM


SkinCancerDetectionTensorFlow
colab.research.google.com/drive/1YXL8r1v88Nw2aiuc35_CZCZ_zrOuyR8u?usp=sharing#scrollTo=4HzOl1TtoqKG
Error
+ Code
+ Text
Cannot save changes
T4
RAM
Disk
20m
31/31 [==============================] - ETA: 0s - loss: 0.2275 - accuracy: 0.9113
Epoch 94: val_loss did not improve from 0.40653
31/31 [==============================] - 14s 474ms/step - loss: 0.2275 - accuracy: 0.9113 - val_loss: 0.8707 - val_accuracy: 0.8047
Epoch 95/100
31/31 [==============================] - ETA: 0s - loss: 0.2251 - accuracy: 0.9103
Epoch 95: val_loss did not improve from 0.40653
31/31 [==============================] - 10s 332ms/step - loss: 0.2251 - accuracy: 0.9103 - val_loss: 0.7488 - val_accuracy: 0.8203
Epoch 96/100
31/31 [==============================] - ETA: 0s - loss: 0.2232 - accuracy: 0.9178
Epoch 96: val_loss did not improve from 0.40653
31/31 [==============================] - 9s 296ms/step - loss: 0.2232 - accuracy: 0.9178 - val_loss: 0.8540 - val_accuracy: 0.8047
Epoch 97/100
31/31 [==============================] - ETA: 0s - loss: 0.2296 - accuracy: 0.9098
Epoch 97: val_loss did not improve from 0.40653
31/31 [==============================] - 13s 409ms/step - loss: 0.2296 - accuracy: 0.9098 - val_loss: 0.7472 - val_accuracy: 0.8125
Epoch 98/100
31/31 [==============================] - ETA: 0s - loss: 0.2190 - accuracy: 0.9168
Epoch 98: val_loss did not improve from 0.40653
31/31 [==============================] - 12s 376ms/step - loss: 0.2190 - accuracy: 0.9168 - val_loss: 0.9389 - val_accuracy: 0.7734
Epoch 99/100
31/31 [==============================] - ETA: 0s - loss: 0.2312 - accuracy: 0.9078
Epoch 99: val_loss did not improve from 0.40653
31/31 [==============================] - 16s 522ms/step - loss: 0.2312 - accuracy: 0.9078 - val_loss: 0.8464 - val_accuracy: 0.7891
Epoch 100/100
31/31 [==============================] - ETA: 0s - loss: 0.2190 - accuracy: 0.9153
Epoch 100: val_loss did not improve from 0.40653
31/31 [==============================] - 12s 399ms/step - loss: 0.2190 - accuracy: 0.9153 - val_loss: 0.8029 - val_accuracy: 0.8125
[10] # evaluation
0s
completed at 6:13 PM


SkinCancerDetectionTensorFlow
colab.research.google.com/drive/1YXL8r1v88Nw2aiuc35_CZCZ_zrOuyR8u?usp=sharing#scrollTo=4HzOl1TtoqKG
Error
+ Code
+ Text
Cannot save changes
T4
RAM
Disk
20m
model_name = f"benign-vs-malignant_{batch_size}_{optimizer}"
tensorboard = tf.keras.callbacks.TensorBoard(log_dir=os.path.join("logs", model_name))
# saves model checkpoint whenever we reach better weights
modelcheckpoint = tf.keras.callbacks.ModelCheckpoint(model_name + "_{val_loss:.3f}.h5", save_best_only=True, verbose=1)
history = m.fit(train_ds, validation_data=valid_ds,
steps_per_epoch=n_training_samples // batch_size,
validation_steps=n_validation_samples // batch_size, verbose=1, epochs=100,
callbacks=[tensorboard, modelcheckpoint])
31/31 [==============================] - 14s 456ms/step - loss: 0.2369 - accuracy: 0.9042 - val_loss: 1.0015 - val_accuracy: 0.7891
Epoch 87/100
31/31 [==============================] - ETA: 0s - loss: 0.2317 - accuracy: 0.9078
Epoch 87: val_loss did not improve from 0.40653
31/31 [==============================] - 12s 391ms/step - loss: 0.2317 - accuracy: 0.9078 - val_loss: 0.9258 - val_accuracy: 0.8047
Epoch 88/100
31/31 [==============================] - ETA: 0s - loss: 0.2276 - accuracy: 0.9103
Epoch 88: val_loss did not improve from 0.40653
31/31 [==============================] - 16s 520ms/step - loss: 0.2276 - accuracy: 0.9103 - val_loss: 0.9941 - val_accuracy: 0.7812
Epoch 89/100
31/31 [==============================] - ETA: 0s - loss: 0.2333 - accuracy: 0.9088
Epoch 89: val_loss did not improve from 0.40653
31/31 [==============================] - 13s 438ms/step - loss: 0.2333 - accuracy: 0.9088 - val_loss: 0.8056 - val_accuracy: 0.8125
Epoch 90/100
31/31 [==============================] - ETA: 0s - loss: 0.2459 - accuracy: 0.8906
Epoch 90: val_loss did not improve from 0.40653
31/31 [==============================] - 14s 466ms/step - loss: 0.2459 - accuracy: 0.8906 - val_loss: 0.7176 - val_accuracy: 0.8125
Epoch 91/100
31/31 [==============================] - ETA: 0s - loss: 0.2321 - accuracy: 0.9108
Epoch 91: val_loss did not improve from 0.40653
0s
completed at 6:13 PM

SkinCancerDetectionTensorFlow

colab.research.google.com/drive/1YXL8r1v88Nw2aiuc35_CZCZ_zrOuyR8u?usp=sharing#scrollTo=VVEdeCwmo1q9

+ Code + Text Cannot save changes

T4 RAM Disk

```
Number of testing samples: 600
```

```
# convert testing set to numpy array to fit in memory (don't do that when testing
# set is too large)
y_test = np.zeros((n_testing_samples,))
X_test = np.zeros((n_testing_samples, 299, 299, 3))
for i, (img, label) in enumerate(test_ds.take(n_testing_samples)):
  # print(img.shape, label.shape)
  X_test[i] = img
  y_test[i] = label.numpy()

print("y_test.shape:", y_test.shape)
```

```
y_test.shape: (600,)
```

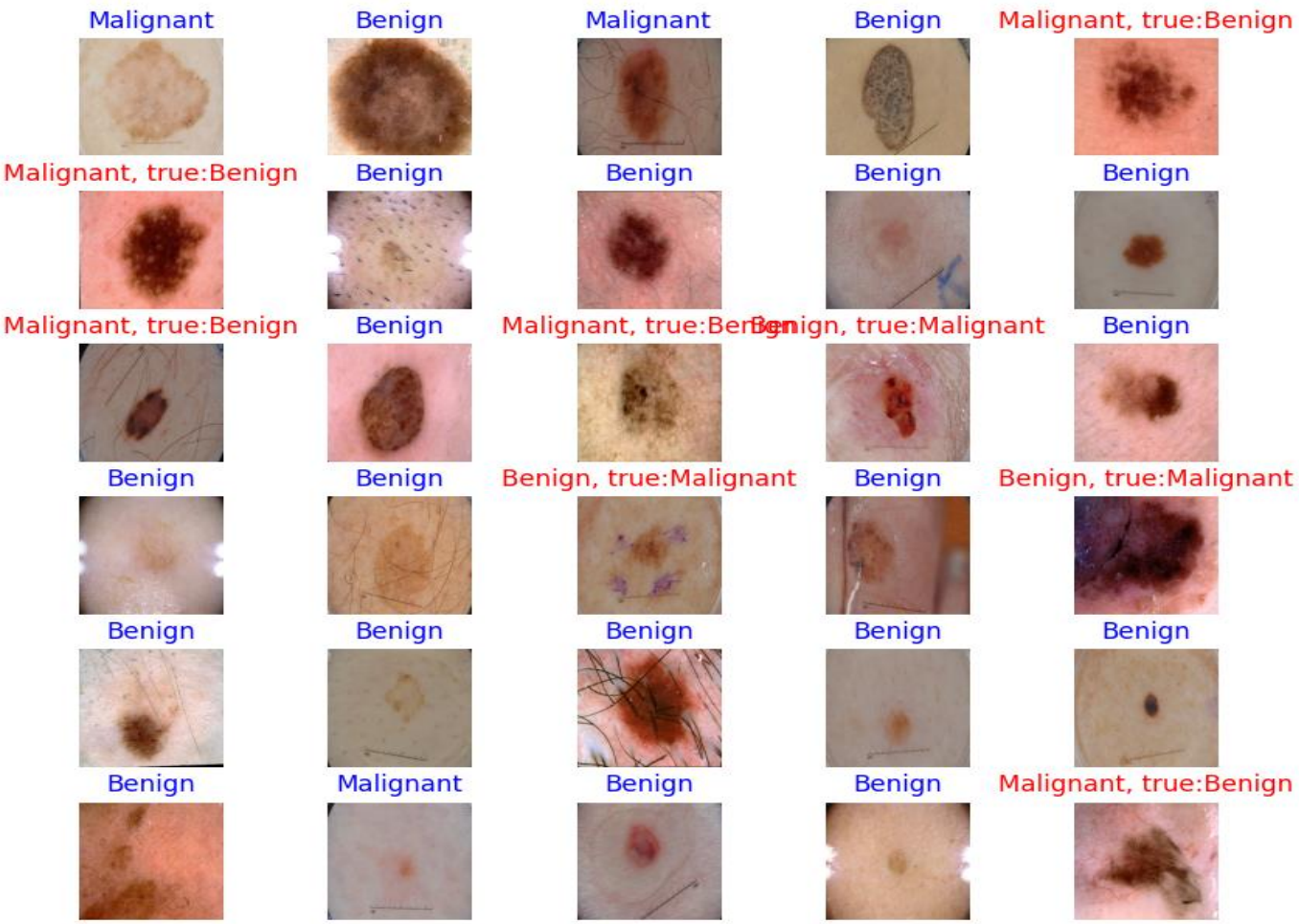
```
[12] # load the weights with the least loss
m.load_weights("benign-vs-malignant_64_rmsprop_0.407.h5")
```

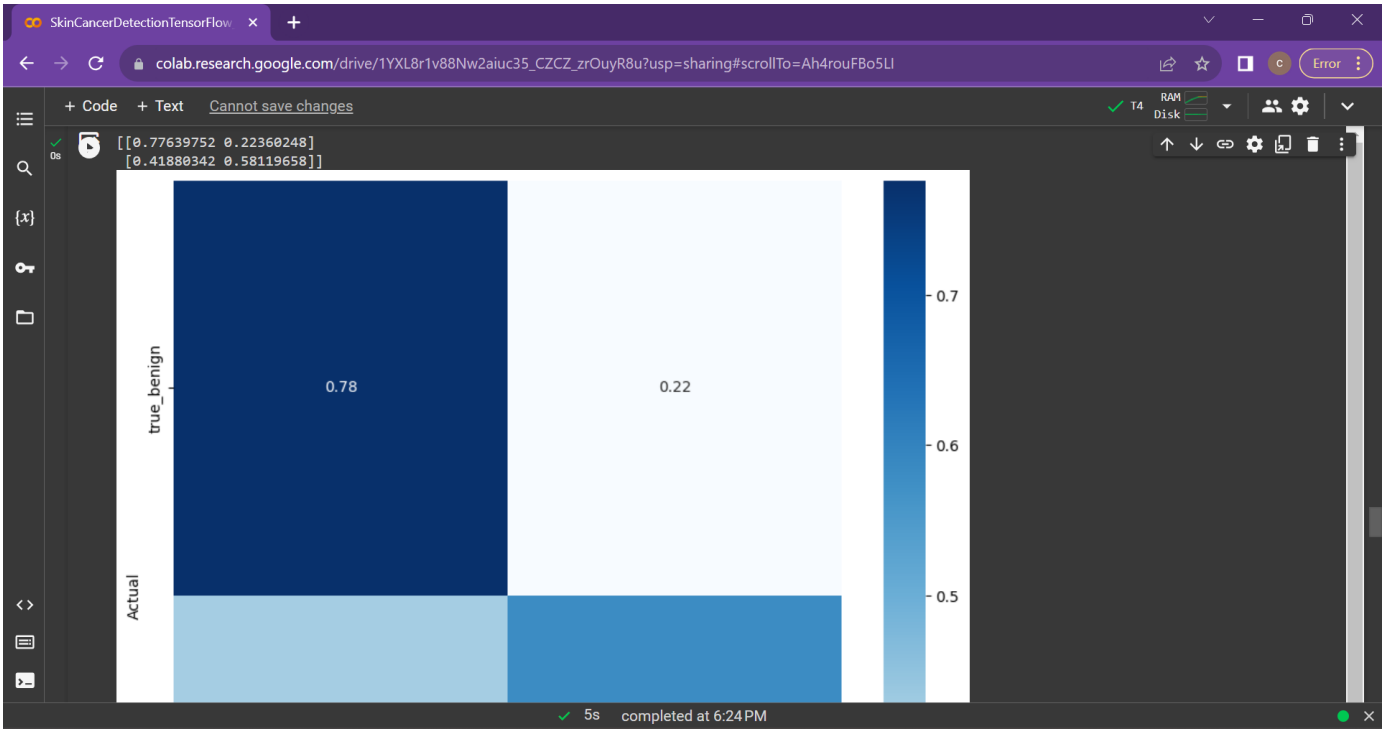
```
print("Evaluating the model...")
loss, accuracy = m.evaluate(X_test, y_test, verbose=0)
print("Loss:", loss, "  Accuracy:", accuracy)
```

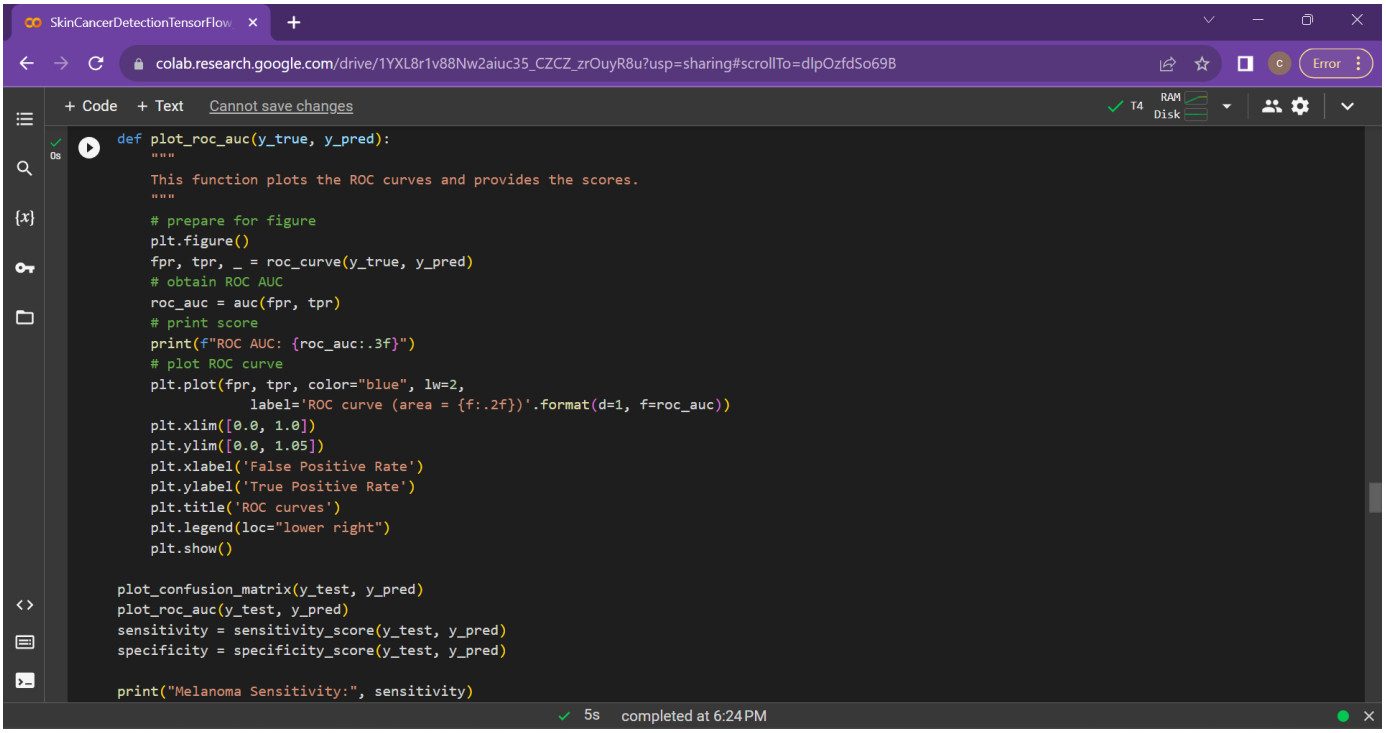
```
Evaluating the model...
Loss: 0.4478985667228699   Accuracy: 0.7983333468437195
```

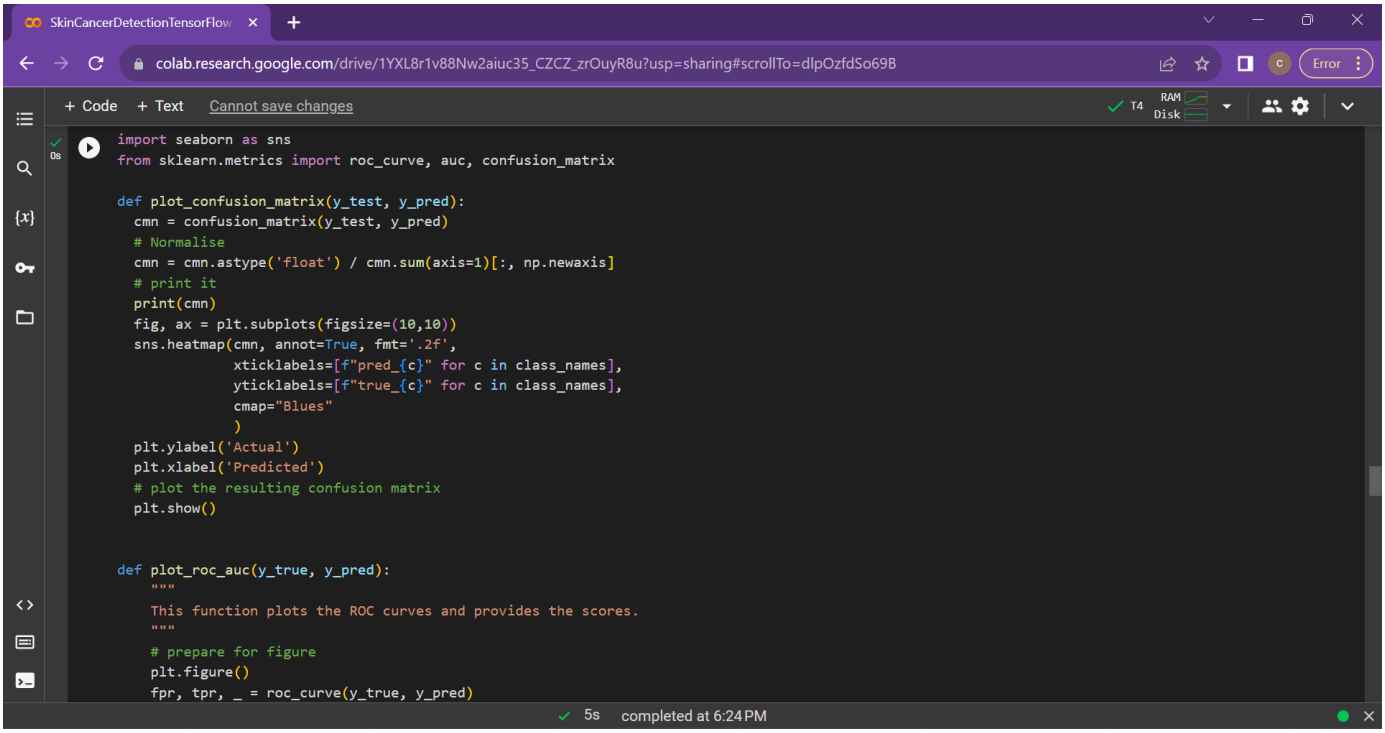
15s completed at 6:19 PM

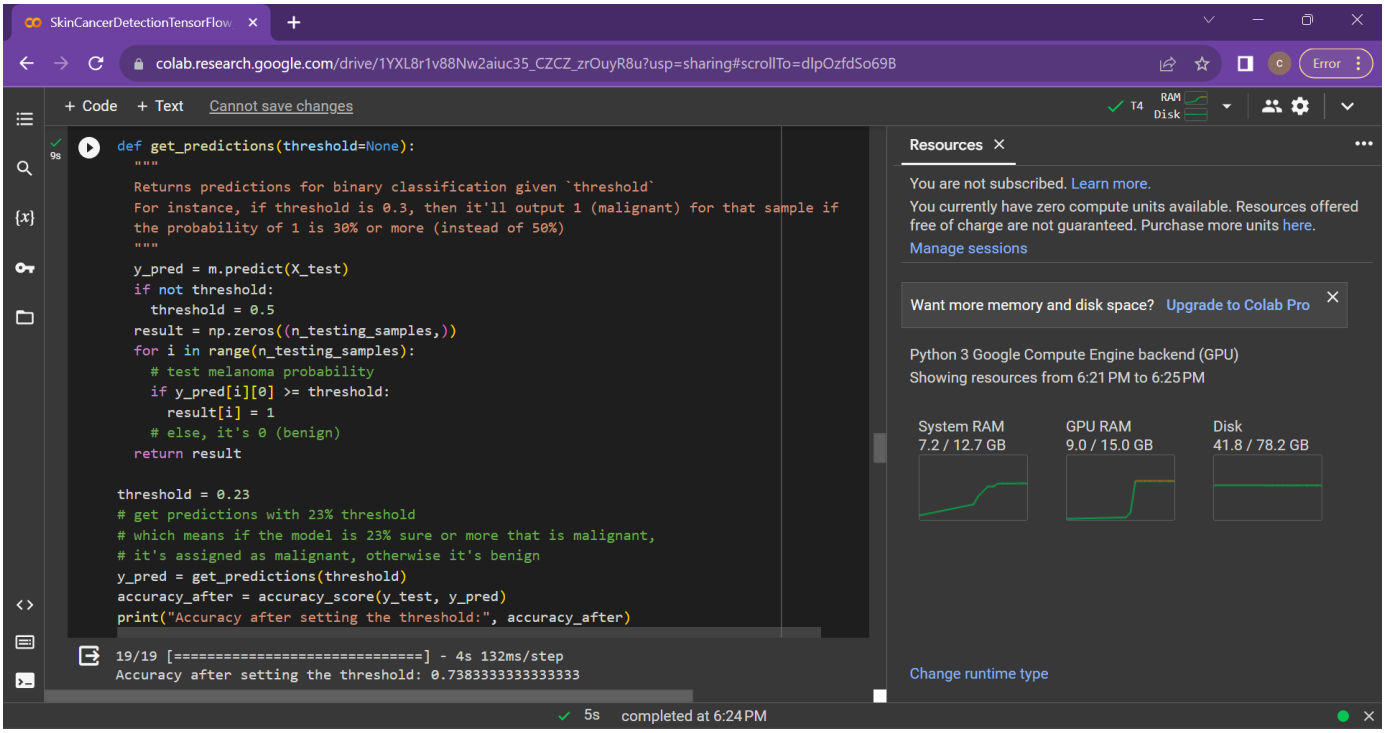
Model predictions (blue: correct, red: incorrect)

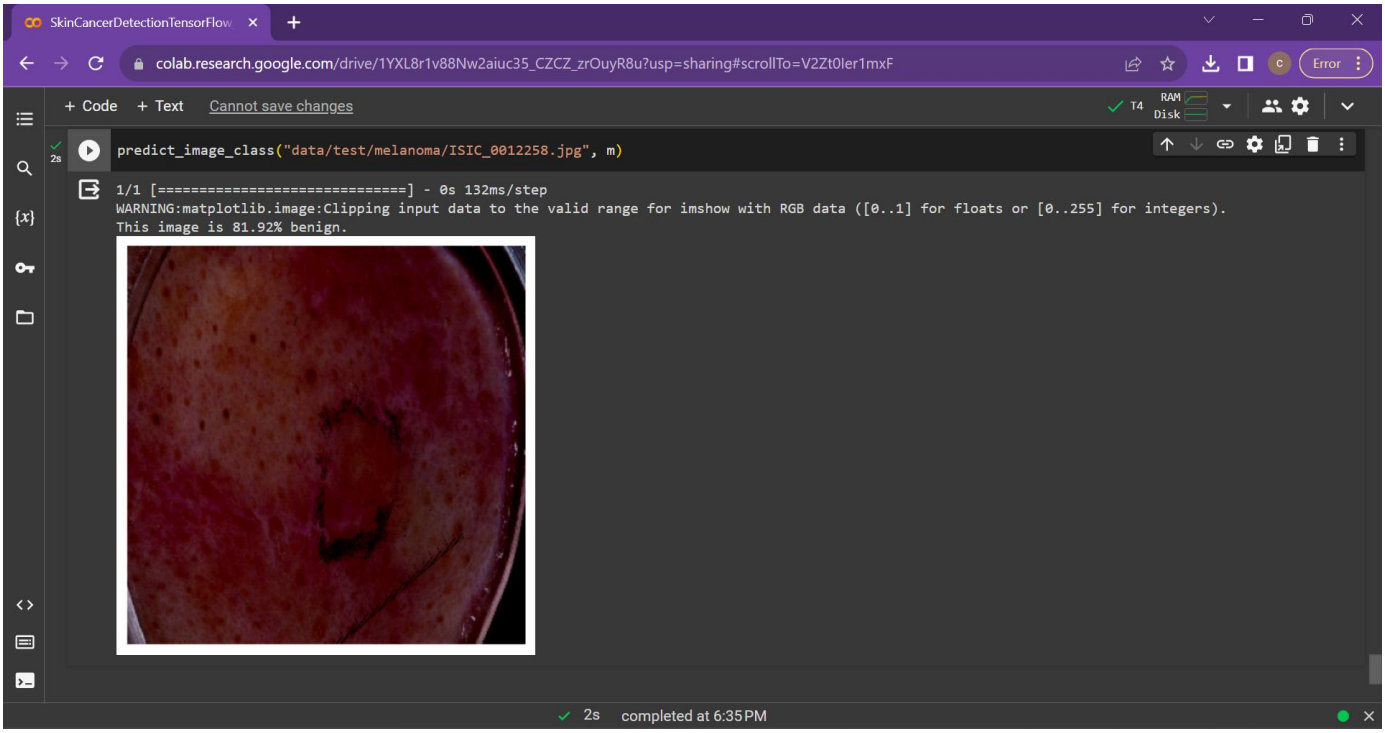



SkinCancerDetectionTensorFlow
colab.research.google.com/drive/1YXL8r1v88Nw2aiuc35_CZCZ_zrOuyR8u?usp=sharing#scrollTo=Ah4rouFBo5LI
Error
+ Code
+ Text
Cannot save changes
T4
RAM
Disk
[[0.77639752 0.22360248]
[0.41880342 0.58119658]]
true_benign
Actual
0.78
0.22
0.7
0.6
0.5
5s
completed at 6:24 PM


SkinCancerDetectionTensorFlow
colab.research.google.com/drive/1YXL8r1v88Nw2aiuc35_CZCZ_zrOuyR8u?usp=sharing#scrollTo=dlpOzfdSo69B
Error
+ Code
+ Text
Cannot save changes
T4
RAM
Disk
def plot_roc_auc(y_true, y_pred):
"""
This function plots the ROC curves and provides the scores.
"""
# prepare for figure
plt.figure()
fpr, tpr, _ = roc_curve(y_true, y_pred)
# obtain ROC AUC
roc_auc = auc(fpr, tpr)
# print score
print(f"ROC AUC: {roc_auc:.3f}")
# plot ROC curve
plt.plot(fpr, tpr, color="blue", lw=2,
label='ROC curve (area = {f:.2f})'.format(d=1, f=roc_auc))
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC curves')
plt.legend(loc="lower right")
plt.show()
plot_confusion_matrix(y_test, y_pred)
plot_roc_auc(y_test, y_pred)
sensitivity = sensitivity_score(y_test, y_pred)
specificity = specificity_score(y_test, y_pred)
print("Melanoma Sensitivity:", sensitivity)
5s
completed at 6:24 PM


SkinCancerDetectionTensorFlow
colab.research.google.com/drive/1YXL8r1v88Nw2aiuc35_CZCZ_zrOuyR8u?usp=sharing#scrollTo=dlpOzfdSo69B
Error
+ Code
+ Text
Cannot save changes
T4
RAM
Disk
import seaborn as sns
from sklearn.metrics import roc_curve, auc, confusion_matrix
def plot_confusion_matrix(y_test, y_pred):
cmn = confusion_matrix(y_test, y_pred)
# Normalise
cmn = cmn.astype('float') / cmn.sum(axis=1)[:, np.newaxis]
# print it
print(cmn)
fig, ax = plt.subplots(figsize=(10,10))
sns.heatmap(cmn, annot=True, fmt='.2f',
xticklabels=[f"pred_{c}" for c in class_names],
yticklabels=[f"true_{c}" for c in class_names],
cmap="Blues"
)
plt.ylabel('Actual')
plt.xlabel('Predicted')
# plot the resulting confusion matrix
plt.show()
def plot_roc_auc(y_true, y_pred):
"""
This function plots the ROC curves and provides the scores.
"""
# prepare for figure
plt.figure()
fpr, tpr, _ = roc_curve(y_true, y_pred)
5s
completed at 6:24 PM


SkinCancerDetectionTensorFlow
colab.research.google.com/drive/1YXL8r1v88Nw2aiuc35_CZCZ_zrOuyR8u?usp=sharing#scrollTo=dlpOzfdSo69B
Error
+ Code
+ Text
Cannot save changes
T4
RAM
Disk
def get_predictions(threshold=None):
"""
Returns predictions for binary classification given `threshold`
For instance, if threshold is 0.3, then it'll output 1 (malignant) for that sample if
the probability of 1 is 30% or more (instead of 50%)
"""
y_pred = m.predict(X_test)
if not threshold:
threshold = 0.5
result = np.zeros((n_testing_samples,))
for i in range(n_testing_samples):
# test melanoma probability
if y_pred[i][0] >= threshold:
result[i] = 1
# else, it's 0 (benign)
return result
threshold = 0.23
# get predictions with 23% threshold
# which means if the model is 23% sure or more that is malignant,
# it's assigned as malignant, otherwise it's benign
y_pred = get_predictions(threshold)
accuracy_after = accuracy_score(y_test, y_pred)
print("Accuracy after setting the threshold:", accuracy_after)
19/19 [==============================] - 4s 132ms/step
Accuracy after setting the threshold: 0.7383333333333333
Resources
You are not subscribed. Learn more.
You currently have zero compute units available. Resources offered free of charge are not guaranteed. Purchase more units here.
Manage sessions
Want more memory and disk space? Upgrade to Colab Pro
Python 3 Google Compute Engine backend (GPU)
Showing resources from 6:21 PM to 6:25 PM
System RAM
7.2 / 12.7 GB
GPU RAM
9.0 / 15.0 GB
Disk
41.8 / 78.2 GB
Change runtime type
5s
completed at 6:24 PM




## DATASET:

The ISIC (International Skin Imaging Collaboration) dataset is a widely used and comprehensive collection of dermatoscopic images for research and development in the field of dermatology, particularly in the context of skin cancer detection. Here are more details about the ISIC dataset:

**1. ISIC Archive:**

- **Link:** ISIC Archive

## Key Features:

- **Diversity of Lesions:** The ISIC dataset encompasses a diverse range of skin lesions, including both benign and malignant cases. This diversity is crucial for training machine learning models to recognize various skin conditions accurately.
- **Annotations and Ground Truth:** Many images in the ISIC dataset come with expert annotations and ground truth labels. These annotations provide information about the type of skin lesion (e.g., nevus, melanoma) and serve as valuable training data for supervised machine learning algorithms.
- **High-Quality Dermatoscopic Images:** The dataset includes high-quality dermatoscopic images, which are images taken using a dermatoscope—a specialized tool that allows for the examination of skin lesions with increased magnification and illumination.
- **Large Scale:** With a large number of images, the ISIC dataset provides a significant amount of data for both training and testing machine learning models. This large-scale nature helps in developing robust models that can generalize well to different types of skin lesions.
- **Contributions from the Research Community:** The dataset is continually updated and expanded with contributions from the research community. Researchers and institutions worldwide contribute images to the ISIC archive, making it a dynamic and evolving resource.

## Potential Use Cases:

**Skin Cancer Classification:** Researchers and practitioners use the ISIC dataset to train and evaluate machine learning models for the classification of skin lesions into benign and malignant categories.

**Algorithm Development:** The dataset serves as a benchmark for developing and refining algorithms

related to dermatoscopic image analysis, feature extraction, and pattern recognition.

**Deep Learning Research:** Given the increasing interest in deep learning for medical image analysis, the ISIC dataset is often used in research on convolutional neural networks (CNNs) and other deep learning architectures for skin disease detection.

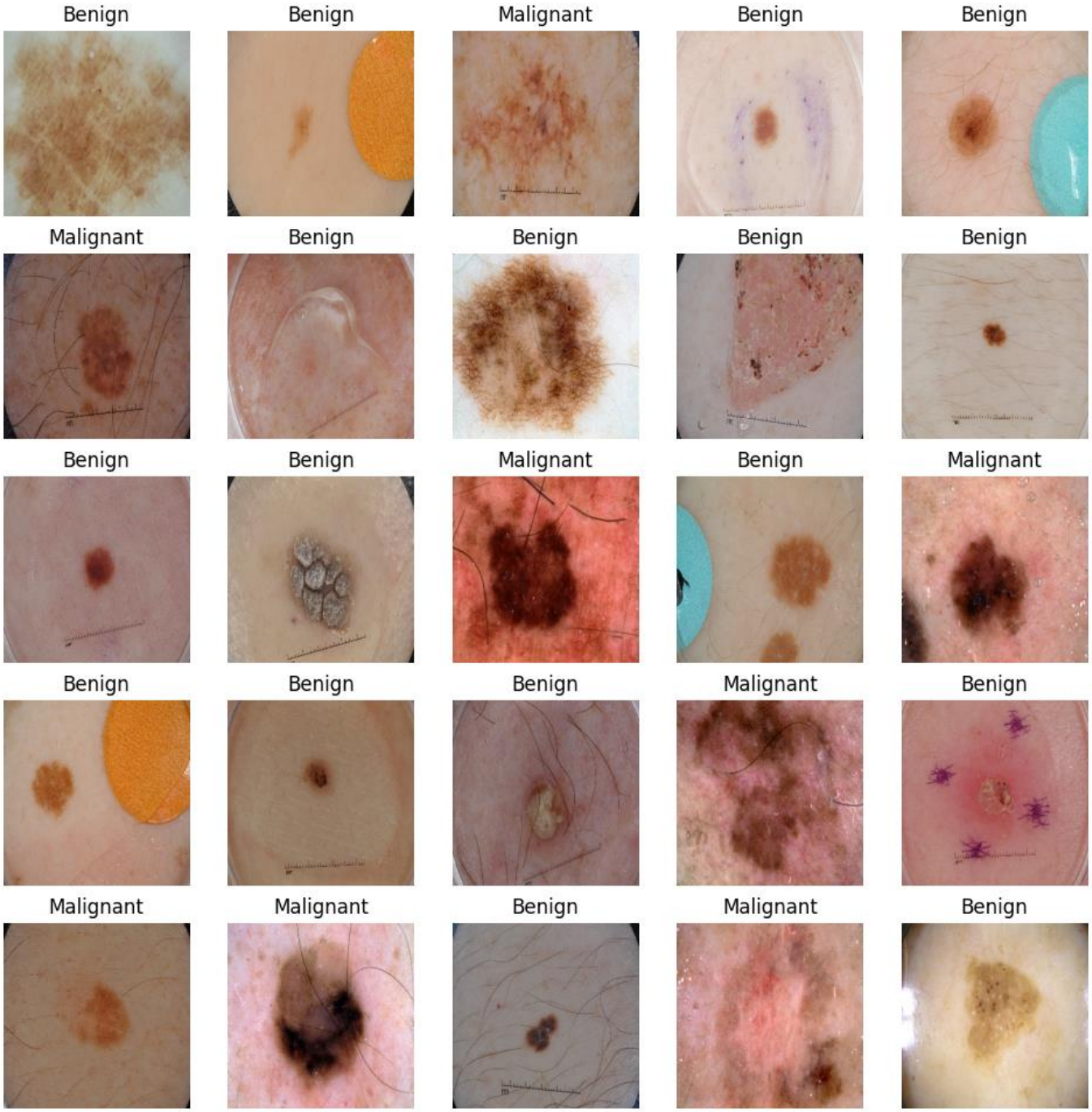


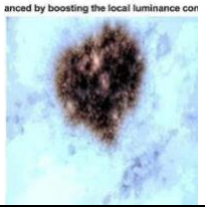
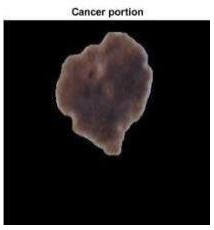

Fig 6.2 Dataset

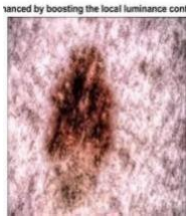
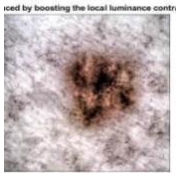
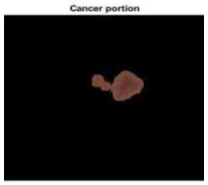
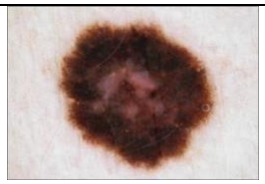
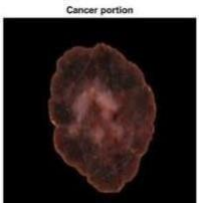
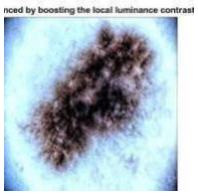
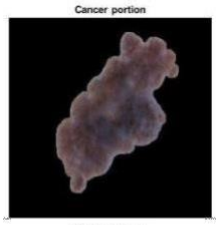
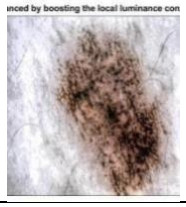
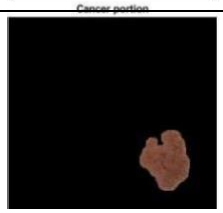
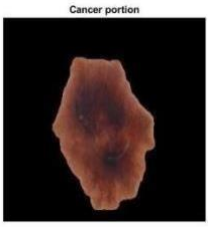
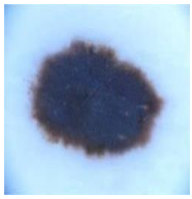
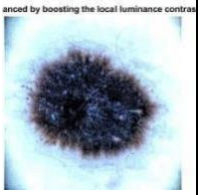
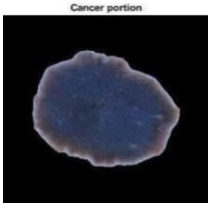
# Chapter 7

## RESULT ANALYSIS

The proposed system was tested for a ISIC (International Skin Imaging Collaboration) dataset of image dimensions 620*250 pixels. System mainlyconcentrated on the results of the skin characters. Proposed model deals with the Skin cancer detection and stage prediction that provides the relevant information for the patients, so that the appropriate treatment can be taken. Proposed system employs Convolutional Neural Network which provides a major advantage of reproducing the expected outcome based onthe weight factors and activation function. This eliminates the cost that is spent on expensive treatment. Result obtained for selected images is as follows:

Table 7.1: Output samples of skin cancer

| S. No | Input Image | Pre-Processed Image | Segmented Output | CancerType | Accuracy (%) |
|---|---|---|---|---|---|
| 1 |  |  |  | Benign – Melanocytic Nevi | 98.73 |
| 2 |  |  |  | Benign – Melanocytic Nevi | 99.86 |
| 3 |  |  |  | Malignant – Squamous Cell Carcinoma | 96.83 |

| | | | | | |
|---|---|---|---|---|---|
| **4** |  |  |  | Benign – Melanocytic Nevi | 98.99 |
| **5** |  |  |  | Benign - Acrochordon | 98.98 |
| **6** |  |  |  | Benign – Melanocytic Nevi | 99.47 |
| **7** |  |  |  | Malignant - Melanoma | 95.62 |
| **8** |  |  |  | Benign - Acrochordon | 99.54 |
| **9** |  |  |  | Benign – Melanocytic Nevi | 95.26 |
| **10** |  |  |  | Malignant - Melanoma | 95.03 |

# Chapter 8

## CONCLUSION

Skin cancer is one of the dangerous forms of cancer as the affected cells canspread easily across the body. It can be either Melanoma or Non-Melanoma.There are various solutions such as Dermoscopy and other devices to detect theskin cancer, these devices involve costs as well as requires adoctor to equip themon the patients. Proposed method aims at detecting and prediction of skin cancerusing Image Processing Techniques that can be easily used by Doctors for the Patient's skin cancer analysis. The system employs methods such as Preprocessing, Feature Selection, Feature Extraction and Classification. The outcome of the model is determined by the CNN (Convolutional Neural Network) that predicts the type of the cancer. This kind of models help the patients to take care of their skin as well as take precautionary measures if the Skin cancer is encountered. The model was applied on an ISIC (InternationalSkin Imaging Collaboration) dataset and resulted in the classification of the cancer types. The confidence level 99.86%.

It may be helpful for medical doctors who do not have enough experience in dermatology. Finally, this system provide a fast diagnosis for the medical fieldand it will a non-invasive tool for the patient, therefore it can help prevent skin cancer.

# REFERENCES


1. Subramanian, R.R.; Dintakurthi, A.; Kumar, S.; Reddy, K.; Amara, S.; Chowdary, A. Skin Cancer Classification Using Convolutional Neural Networks. In Proceedings of the 2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence), Noida, India, 28–29 January 2021. [Google Scholar]
2. Harsha, P.; Sahruthi, G.; Vaishnavi, A.; Varshini, P. Spotting Skin Cancer Using CNN. *Int. J. Eng. Tech.* **2022**, *8*, 47–51. [Google Scholar]
3. Dubai, P.; Bhatt, S.; Joglekar, C.; Patii, S. Skin Cancer Detection and Classification. In Proceedings of the 2017 6th International Conference on Electrical Engineering and Informatics (ICEEI), Langkawi, Malaysia, 25–27 November 2017; pp. 1–6. [Google Scholar] [CrossRef]
4. Jana, E.; Subban, R.; Saraswathi, S. Research on Skin Cancer Cell Detection Using Image Processing. In Proceedings of the 2017 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), Coimbatore, India, 14–16 December 2017. [Google Scholar]
5. Dildar, M.; Akram, S.; Irfan, M.; Khan, H.U.; Ramzan, M.; Mahmood, A.R.; Alsaiari, S.A.; Saeed, A.H.M.; Alraddadi, M.O.; Mahnashi, M.H. Skin Cancer Detection: A Review Using Deep Learning Techniques. *Int. J. Environ. Res. Public Health* **2021**, *18*, 5479. [Google Scholar] [CrossRef] [PubMed]
6. Hosny, K.; Kassem, M.; Fouad, M. Skin Cancer Classification Using Deep Learning and Transfer Learning. In Proceedings of the 2018 9th Cairo International Biomedical Engineering Conference (CIBEC), Cairo, Egypt, 20–22 December 2018. [Google Scholar]
7. Ashraf, R.; Kiran, I.; Mahmood, T.; Butt, A.; Razzaq, N.; Farooq, Z. An Efficient Technique for Skin Cancer Classification Using Deep Learning. In Proceedings of the 2020 IEEE 23rd International Multitopic Conference (INMIC), Bahawalpur, Pakistan, 5–7 November 2020. [Google Scholar]
8. Chaturvedi, S.S.; Tembhurne, J.V.; Diwan, T. A Multi-Class Skin Cancer Classification Using Deep Convolutional Neural Networks. *Multimed. Tools Appl.* **2020**, *79*, 28477–28498. [Google Scholar] [CrossRef]
9. Marka, A.; Carter, J.B.; Toto, E.; Hassanpour, S. Automated Detection of Nonmelanoma Skin Cancer Using Digital Images: A Systematic Review. *BMC Med. Imaging* **2019**, *19*, 21. [Google Scholar] [CrossRef] [PubMed]
10. Jojoa Acosta, M.F.; Caballero Tovar, L.Y.; Garcia-Zapirain, M.B.; Percybrooks, W.S. Melanoma Diagnosis Using Deep Learning Techniques on Dermatoscopic Images. *BMC Med. Imaging* **2021**, *21*, 6. [Google Scholar] [CrossRef]
11. Javaid, A.; Orakzai, M.; Akram, F. Skin Cancer Classification Using Image Processing and Machine Learning. In Proceedings of the 2021 International Bhurban Conference on Applied Sciences and Technologies (IBCAST), Islamabad, Pakistan, 12–16 January 2021. [Google Scholar]
12. Hong, J.; Yu, S.C.H.; Chen, W. Unsupervised Domain Adaptation for Cross-Modality Liver Segmentation via Joint Adversarial Learning and Self-Learning. *Appl. Soft Comput.* **2022**, *121*, 108729. [Google Scholar] [CrossRef]
13. Hong, J.; Zhang, Y.D.; Chen, W. Source-Free Unsupervised Domain Adaptation for Cross-Modality Abdominal Multi-Organ Segmentation. *Knowl.-Based Syst.* **2022**, *250*, 109155. [Google Scholar] [CrossRef]
14. Zhu, N.; Zhang, D.; Wang, W.; Li, X.; Yang, B.; Song, J.; Zhao, X.; Huang, B.; Shi, W.; Lu, R.; et al. A Novel Coronavirus from Patients with Pneumonia in China, 2019. *N. Engl. J. Med.* **2020**, *382*, 727–733. [Google Scholar] [CrossRef]
15. Ali, M.; Miah, M.S.; Haque, J.; Rahman, M.M.; Islam, M. An Enhanced Technique of Skin Cancer Classification Using Deep Convolutional Neural Network with Transfer Learning Models. *Mach. Learn. Appl.* **2021**, *5*, 100036. [Google Scholar] [CrossRef]
16. Younis, H.; Bhatti, M.H.; Azeem, M. Classification of Skin Cancer Dermoscopy Images Using Transfer Learning. In Proceedings of the 2019 15th International Conference on Emerging Technologies (ICET), Peshawar, Pakistan, 2–3 December 2019; pp. 1–4. [Google Scholar]
17. Dascalu, A.; Walker, B.N.; Oron, Y.; David, E.O. Non-Melanoma Skin Cancer Diagnosis: A Comparison between Dermoscopic and Smartphone Images by Unified Visual and Sonification Deep Learning Algorithms. *J. Cancer Res. Clin. Oncol.* **2022**, *148*, 2497–2505. [Google Scholar] [CrossRef]
18. Kawaguchi, Y.; Hanaoka, J.; Ohshio, Y.; Okamoto, K.; Kaku, R.; Hayashi, K.; Shiratori, T.; Yoden, M. Sarcopenia Predicts Poor Postoperative Outcome in Elderly Patients with Lung Cancer. *Gen. Thorac. Cardiovasc. Surg.* **2019**, *67*, 949–954. [Google Scholar] [CrossRef]
19. Esteva, A.; Kuprel, B.; Novoa, R.A.; Ko, J.; Swetter, S.M.; Blau, H.M.; Thrun, S. Dermatologist-Level Classification of Skin Cancer with Deep Neural Networks. *Nature* **2017**, *542*, 115–118. [Google Scholar] [CrossRef] [PubMed]
20. Meshram, A.A.; Abhimanyu Dutonde, A.G. A Review of Skin Melanoma Detection Based on Machine Learning. *Int. J. New Pract. Manag. Eng.* **2022**, *11*, 15–23. [Google Scholar] [CrossRef]
21. Babu, G.N.K.; Peter, V.J. Skin Cancer Detection Using Support Vector Machine with Histogram of Oriented Gradients Features. *ICTACT J. Soft Comput.* **2021**, *6956*, 2301–2305. [Google Scholar] [CrossRef]

22. Faiza, F.; Irfan Ullah, S.; Salam, A.; Ullah, F.; Imad, M.; Hassan, M. Diagnosing of Dermoscopic Images Using Machine Learning Approaches for Melanoma Detection. In Proceedings of the 2020 IEEE 23rd International Multitopic Conference (INMIC), Bahawalpur, Pakistan, 5–7 November 2020. [**Google Scholar**]
23. Majumder, S.; Ullah, M.A. Feature Extraction from Dermoscopy Images for Melanoma Diagnosis. *SN Appl. Sci.* **2019**, *1*, 753. [**Google Scholar**] [**CrossRef**]
24. Chaitanya, K.N.; Pavani, I. Melanoma Early Detection Using Dual Classifier. *Int. J. Sci. Eng. Technol. Res.* **2015**, *1*, 1–6. [**Google Scholar**]
25. Bernstein, M.N.; Gladstein, A.; Latt, K.Z.; Clough, E.; Busby, B.; Dillman, A. Jupyter Notebook-Based Tools for Building Structured Datasets from the Sequence Read Archive. *F1000Research* **2020**, *9*, 376. [**Google Scholar**] [**CrossRef**]
26. Fan, L.; Zhang, F.; Fan, H.; Zhang, C. Brief Review of Image Denoising Techniques. *Vis. Comput. Ind. Biomed. Art* **2019**, *2*, 7. [**Google Scholar**] [**CrossRef**]
27. Ansari, U.; Sarode, T.K. Skin Cancer Detection Using Image Processing. In Proceedings of the WSCG 2021: Full Papers Proceedings: 29. International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, Pilsen, Czech Republic, 17–21 May 2021. [**Google Scholar**]
28. Senan, E.; Jadhav, M. Analysis of Dermoscopy Images by Using ABCD Rule for Early Detection of Skin Cancer. *Glob. Transitions Proc.* **2021**, *2*, 1–7. [**Google Scholar**] [**CrossRef**]
29. Mabrouk, M.; Sheha, M.; Sharawi, A. Automatic Detection of Melanoma Skin Cancer Using Texture Analysis. *Int. J. Comput. Appl.* **2012**, *42*, 22–26. [**Google Scholar**] [**CrossRef**]
30. Seeja, R.D.; Suresh, A. Deep Learning Based Skin Lesion Segmentation and Classification of Melanoma Using Support Vector Machine (SVM). *Asian Pac. J. Cancer Prev.* **2019**, *20*, 1555–1561. [**Google Scholar**] [**CrossRef**]
31. Do, T.T.; Hoang, T.; Pomponiu, V.; Zhou, Y.; Chen, Z.; Cheung, N.M.; Koh, D.; Tan, A.; Tan, S.H. Accessible Melanoma Detection Using Smartphones and Mobile Image Analysis. *IEEE Trans. Multimed.* **2017**, *20*, 2849–2864. [**Google Scholar**] [**CrossRef**]
32. Ozkan, I.A.; Koklu, M. Skin Lesion Classification Using Machine Learning Algorithms. *Int. J. Intell. Syst. Appl. Eng.* **2017**, *5*, 285–289. [**Google Scholar**] [**CrossRef**]
33. Murugan, A.; Nair, A.; Kumar, K.P. Research on SVM and KNN Classifiers for Skin Cancer Detection. *Int. J. Eng. Adv. Technol.* **2020**, *9*, 4627–4632. [**Google Scholar**] [**CrossRef**]
34. Mijwil, M.M. Skin Cancer Disease Images Classification Using Deep Learning Solutions. *Multimed. Tools Appl.* **2021**, *80*, 26255–26271. [**Google Scholar**] [**CrossRef**]
35. Naeem, A.; Farooq, S.; Khelifi, A.; Abid, A. Malignant Melanoma Classification Using Deep Learning: Datasets, Performance Measurements, Challenges and Opportunities. *IEEE Access* **2020**, *8*, 110575–110597. [**Google Scholar**] [**CrossRef**]
36. Junayed, M.S.; Anjum, N.; Noman, A.; Islam, B. A Deep CNN Model for Skin Cancer Detection and Classification. In Proceedings of the WSCG 2021: Full Papers Proceedings: 29. International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, Pilsen, Czech Republic, 17–21 May 2021. [**Google Scholar**]