VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama Belagavi, Karnataka–590014

Project Report

On

"DETECTION AND CLASSIFICATION OF SKIN CANCER USING IMAGE PROCESSING AND SVM ALGORITHM"

Submitted in partial fulfillment of the requirements for the award of degree of

BACHELOR OF ENGINEERING

in

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

MANAN KARUN	**(1BI19EC069)**
NAMRATHA N A	**(1BI19EC086)**
RAMITHA R	**(1BI19EC112)**

Under the guidance of

DR D. VIJAYALAKSHMI

Associate Professor,

Dept. of ECE, BIT

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING
BANGALORE INSTITUTE OF TECHNOLOGY
K.R. Road, Bengaluru– 560004

2022-2023

BANGALORE INSTITUTE OF TECHNOLOGY

K.R. Road, V.V. Puram, Bengaluru- 560004

www.bit-bangalore.edu.in

DEPARTMENT OF ELECTRONICS AND COMMUNICATION

CERTIFICATE

This is to certify that the project work entitled **"DETECTION AND CLASSIFICATION OF SKIN CANCER USING IMAGE PROCESSING AND SVM ALGORITHM"** carried out by **Ms. MANAN KARUN (1BI19EC069), Ms. NAMRATHA N A (1BI19EC086), Ms. RAMITHA R (1BI19EC112)** are bonafide students of **Bangalore Institute of Technology** in partial fulfillment for the award of **Bachelor of Engineering** in **Electronics and Communication Engineering** of the **Visvesvaraya Technological University, Belgaum** during the year 2022-2023. It is certified that all corrections/ suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the above said Degree.

Signature of Guide	Signature of HOD	Signature of Principal
Dr. D. Vijayalakshmi	**Dr. Hemanth Kumar A. R**	**Dr. Aswath.M. U**
Associate Professor Dept. of ECE, BIT	Professor & HOD Dept. of ECE, BIT	Principal, BIT

External Viva

Name of the Examiners and Signature with date

1.

2.

ACKNOWLEDGEMENT

We would like to take this opportunity to thank all those who have being involved directly or indirectly in the completion of our Project.

We would like to express our sincere thanks to our internal guide **Dr D. VIJAYALAKSHMI,** Associate Professor, Department of Electronics and Communication Engineering, BIT Bengaluru.

We would also like to express our gratitude to **Dr. HEMANTH KUMAR A.R**, Professor and Head, Department of Electronics and Communication Engineering, BIT, Bengaluru for providing constant support and guidance.

We also thank **Dr. BYRAREDDY C.R.,** Professor**, Dr. MUKTHI S.L.,** Associate Professor, **Prof. GAHAN A.V,** Assistant Professor, and **Prof. HITHAISHI P,** Assistant Professor B.E Project Coordinators, Department of Electronics and Communication Engineering, BIT Bengaluru for coordinating and extending the support and guidance for accomplishment of seminar abide to the guidelines.

We immensely thank **Dr. ASWATH M.U**, Principal, BIT Bengaluru for providing excellent academic environment in the college.

We would express our thanks to all teaching and non-teaching staff of Department of Electronics and Communication for their support and cooperation throughout the completion of project. It would be our privilege to express our heartfelt gratitude and respect to the Bangalore Institute of Technology, which has given us an opportunity to present this project report

MANAN KARUN **(1BI19EC069)**
NAMRATHA N A **(1BI19EC086)**
RAMITHA R **(1BI19EC112)**

ABSTRACT

Skin cancers are the most common and deadly form of cancers in the present world. Skin cancer is a serious and potentially life-threatening disease that requires early detection and effective treatment. In recent years, computer-aided diagnosis (CAD) systems based on machine learning algorithms have been developed to assist dermatologists in detecting and classifying skin cancer using image processing techniques. SVM is a supervised learning algorithm that works by finding the best possible boundary between two classes that can separate them with maximum margin. In the context of skin cancer detection and classification, SVM can be trained on a dataset of skin cancer images and their corresponding labels. Image processing techniques such as image preprocessing, segmentation, feature extraction, and feature selection are used to extract relevant features from skin cancer images that can be used to train the SVM classifier. With continued research and development, CAD systems based on SVM classifier and image processing techniques have the potential to improve the accuracy and efficiency of skin cancer detection and classification, leading to better outcomes for patients. The reported cancer incidence in India in 2022 is estimated to be approximately 1.9 to 2 million and nearly 10 million across the world. Majority of patients suffering from skin cancer fail to diagnose themselves until their illness becomes terminal. The proposed project aims at overcoming this problem by developing a MATLAB application that assists in early skin cancer detection and classifies it into one of the five deadly types – Actinic Keratosis, Basal Cell Carcinoma, Cherry Nevus, Dermatofibroma, and Melanoma using image processing and machine learning classification algorithm SVM, which is a very efficient classification algorithm. It is a supervised learning algorithm that trains on a set of label data and groups it into different classes. The performance efficiency of the application is evaluated by computing parameters like accuracy, precision, sensitivity and specificity.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

BANGALORE INSTITUTE OF TECHNOLOGY

VISION

To establish and develop the Institute as a center of higher learning, ever abreast with expanding horizon of knowledge in the field of engineering and technology, with entrepreneurial thinking, leadership excellence for life-long success and solve societal problem.

MISSION

- Provide high quality education in the engineering disciplines from the undergraduate through doctoral levels with creative academic and professional programs.
- Develop the Institute as a leader in Science, Engineering, Technology and management, Research and apply knowledge for the benefit of society.
- Establish mutual beneficial partnerships with industry, alumni, local, state and central governments by public service assistance and collaborative research.
- Inculcate personality development through sports, cultural and extracurricular activities and engage in the social, economic and professional challenges.

LONG TERM GOALS

- To be among top 3 private engineering colleges in Karnataka and top 20 in India.
- To be the most preferred choice of students and faculty.
- To be the preferred partner of corporate.
- To provide knowledge through education and research in engineering.
- To develop in each student mastery of fundamentals, versatility of mind, motivation for learning, intellectual discipline and self-reliance which provide the best foundation for continuing professional achievement.
- To provide a liberal; as well as a professional education so that each student acquires a respect for moral values, a sense of their duties as a citizen, a feeling for taste and style, and a better human understanding.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION

VISION

Imparting **Quality Education** to achieve **Academic Excellence** in Electronics and Communication Engineering for **Global Competent Engineers**.

MISSION

- Create **state of art infrastructure** for quality education.
- Nurture **innovative concepts** and problem-solving **skills**.
- Delivering **Professional Engineers** to meet the **societal needs**.

PROGRAM EDUCATIONAL OBJECTIVES

- Prepare graduates to be **professionals**, Practicing engineers and entrepreneurs in the field of Electronics and communication.
- To acquire sufficient knowledge base for **innovative techniques** in design and development of systems.
- Capable of competing globally in **multidisciplinary** field.
- Achieve personal and professional success with awareness and commitment to **ethical and social responsibilities** as an individual as well as a team.
- Graduates will maintain and improve technical competence through **continuous learning process**.

PROGRAM SPECIFIC OUTCOMES

PSO1: Core Engineering: The graduates will be able to apply the principles of Electronics and Communication in core areas.

PSO2: Soft Skills: An ability to use latest hardware and software tools in Electronics and Communication engineering.

PSO3: Successful Career: Preparing Graduates to satisfy industrial needs and pursue higher studies with social-awareness and universal moral values.

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

Skin cancer is the unusual development of skin cells which frequently develops due to exposure to the sun. It is also possible that skin cancer develops on territories of the skin not exposed to the sunlight. Melanoma is generally the most serious and deadly form of skin cancer; it can be life threating most of the times.

The early discovery of melanoma and other skin cancer types can lead to be a very prominent solution in recovery and change the lives of many affected patients. On early detection of cancer, the patient can approach doctors and get professional medical help.

The types of Skin Cancers are -

1. **Actinic Keratoses**

   A rough, scaly patch on the skin caused by years of sun exposure. Actinic keratoses usually affects older adults. Reducing sun exposure can help reduce risk. It is most common on the face, lips, ears, back of hands, forearms, scalp and neck. The rough, scaly skin patch enlarges slowly and usually causes no other signs or symptoms. A lesion may take years to develop. Since it can become cancerous, it's usually removed as a precaution.

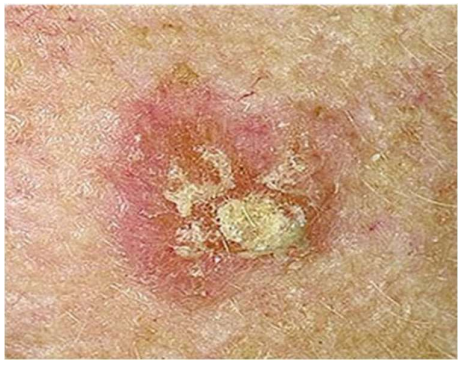


Fig 1.1 Actinic Keratoses

2. **Dermatofibroma**

   Dermatofibroma is a commonly occurring cutaneous entity usually centered within the skin's dermis. Dermatofibromas are referred to as benign fibrous histiocytomas of the

skin, superficial/cutaneous benign fibrous histiocytomas, or common fibrous histiocytoma. These mesenchymal cell lesions of the dermis clinically are firm subcutaneous nodules that occur on the extremities in the vast majority of cases and may or may not be associated with overlying skin changes. They are most commonly asymptomatic and usually relatively small, less than or equal to 1 centimetre in diameter. The term "fibrous histiocytoma" refers more to the morphologic appearance of the cell populations that comprise these lesions rather than strictly describing the cellular lineage.

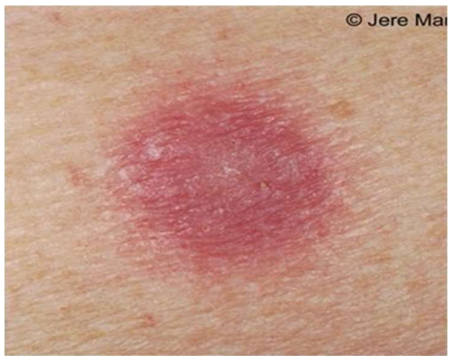


Fig. 1.2 Dermatofibroma

3. Basal Cell Carcinoma

Basal cell carcinoma (BCC) is the most common type of skin cancer. It arises from the basal cells, which are found in the lowest layer of the epidermis, the outermost layer of the skin.

BCC usually develops in areas of the body that are frequently exposed to the sun, such as the face, scalp, ears, neck, and shoulders, but it can occur in other areas as well.

The primary cause of basal cell carcinoma is prolonged exposure to ultraviolet (UV) radiation from the sun or artificial sources, such as tanning beds. The cumulative effects of sun exposure over time can lead to DNA damage in the skin cells, resulting in the development of cancerous growths.

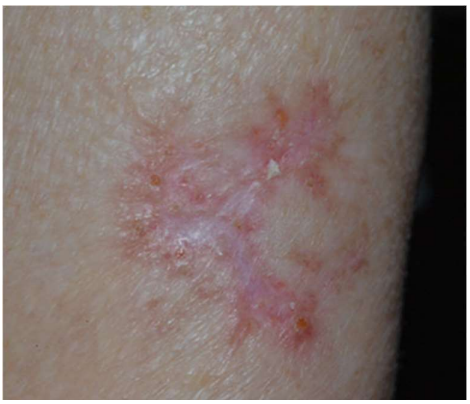


Fig.1.3 Basal Cell Carcinoma

4. Cherry Nevus

Cherry nevus, also known as cherry angioma or Campbell de Morgan spot, is a common benign skin growth characterized by small, bright red or cherry-red papules or spots on the skin's surface. These spots are usually round or oval in shape and can vary in size from a pinhead to about a quarter of an inch in diameter.

Cherry nevi are typically found on the trunk of the body, although they can appear on any part of the skin, including the face, scalp, arms, and legs. They tend to occur more frequently as people age and are more commonly seen in individuals over the age of 30.

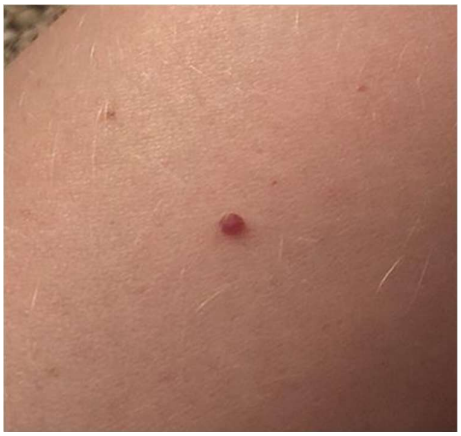


Fig.1.4 Cherry Nevus

5. Melanoma

It is the most serious type of skin cancer. Melanoma occurs when the pigment-producing cells that give colour to the skin become cancerous.

Symptoms might include a new, unusual growth or a change in an existing mole. Melanomas can occur anywhere on the body. Treatment may involve surgery, radiation, medication or in some cases, chemotherapy.

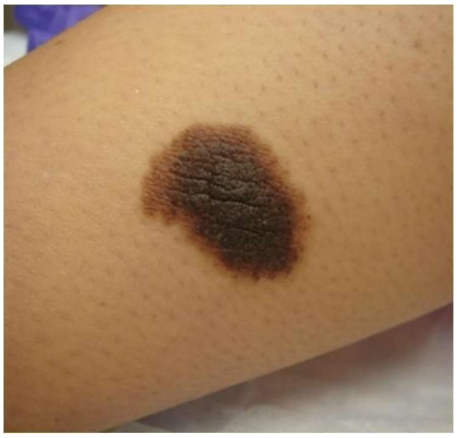


Fig.1.5 Melanoma

1.2 MOTIVATION

The images attached below shows the statistics of skin cancer affected in India as well asacross the world.

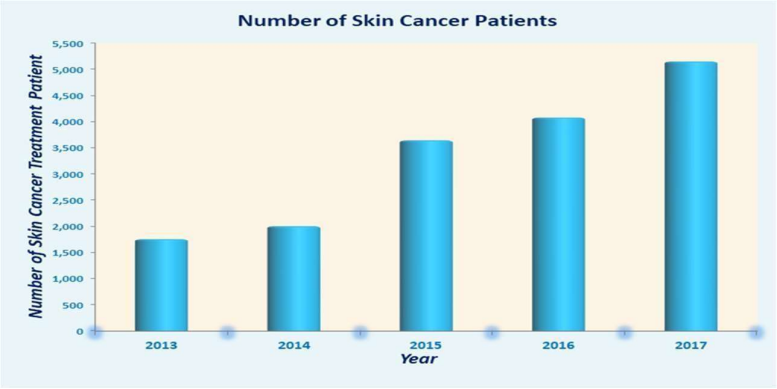
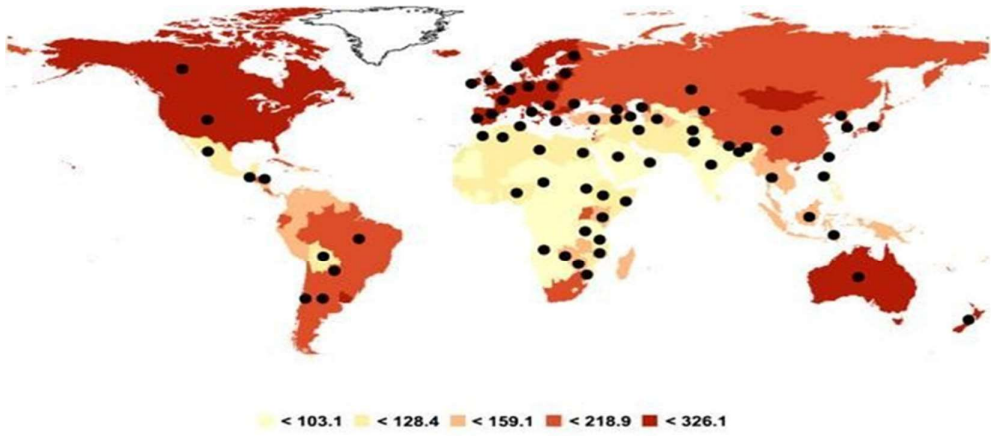


Fig.1.6 Statistical data of number of skin cancer patients receiving treatment



Source: GLOBOCAN cancer statistics, 2008. Estimated age-standardized incidence rate per 100,000 residents for all cancers, excluding non-melanoma skin cancer, both sexes and all ages.

Fig. 1.7 Worldwide estimated data of people affected with skin cancer

From the statistics given above, we can infer that the need for early detection is an extreme necessity.

The traditional methods followed since years is described below.

1.2.1 Diagnosis of skin cancer

The conventional methods to detect skin cancer are listed below:

- **Examine the skin:** The doctor may look at the skin to determine whether the skinchanges are likely to be skin cancer. Further testing may be needed to confirm that diagnosis.
- **Remove a sample of suspicious skin for testing (skin biopsy):** The doctor may remove the suspicious-looking skin for lab testing. A biopsy can determine whether you have skin cancer and, if so, what type of skin cancer you have.

1.2.2 Determining the extent of the skin cancer

If the doctor diagnosis a patient with skin cancer, then additional tests are performed to determinethe extent (stage) of the skin cancer.

Superficial skin cancers such as basal cell carcinoma rarely spread, a biopsy that removes the entire growth often is the only test needed to determine the cancer stage. But in case of large squamous cell carcinoma, Merkel cell carcinoma or melanoma, the doctor may recommend further tests to determine the extent of the cancer.

Additional tests might include imaging tests to examine the nearby lymph nodes for signs

of cancer or a procedure to remove a nearby lymph node and test it for signs of cancer (sentinel lymph node biopsy).

Doctors use the Roman numerals I through IV to indicate a cancer's stage. Stage I cancers are small and limited to the area where they began. Stage IV indicates advanced cancer that has spread to other areas of the body. The skin cancer's stage helps determine which treatment options will be most effective.

1.2.3 Treatment for skin cancer

The treatment options for skin cancer will vary, depending on the size, type, depth and location of the affected area. Small skin cancers limited to the surface of the skin may not require treatment beyond an initial skin biopsy that removes the entire growth.

Some of the advanced skin cancer techniques are listed below:

- **Freezing:** The doctor may destroy actinic keratoses and some small, early skin cancers by freezing them with liquid nitrogen (cryosurgery). The dead tissue sloughs off when it thaws.
- **Excisional surgery:** This type of treatment may be appropriate for any type of skin cancer. The doctor cuts out the cancerous tissue and a surrounding margin of healthy skin,
- **Mohs surgery:** This procedure is for larger, recurring or difficult-to-treat skin cancers, which may include both basal and squamous cell carcinomas. It's often used in areas where it's necessary to conserve as much skin as possible, such as on the nose.

  During Mohs surgery, the doctor removes the skin growth layer by layer, examining each layer under the microscope, until no abnormal cells remain. This procedure allows cancerous cells to be removed without taking an excessive amount of surrounding healthy skin.
- **Curettage and electrodesiccation or cryotherapy:** After removing most of a growth, the doctor scrapes away layers of cancer cells using a device with a circular blade (curet).An electric needle destroys any remaining cancer cells. In a variation of this procedure, liquid nitrogen can be used to freeze the base and edges of the treated area.

  These simple, quick procedures may be used to treat basal cell cancers or thin squamous cell cancers.
- **Radiation therapy:** Radiation therapy uses high-powered energy beams, such as X-rays, to kill cancer cells. Radiation therapy may be an option when cancer can't be completely

removed during surgery.

- **Chemotherapy:** In chemotherapy, drugs are used to kill cancer cells. For cancers limited to the top layer of skin, creams or lotions containing anti-cancer agents may be applied directly to the skin. Systemic chemotherapy can be used to treat skin cancers that have spread to other parts of the body.
- **Photodynamic therapy:** This treatment destroys skin cancer cells with a combination of laser light and drugs that makes cancer cells sensitive to light.
- **Biological therapy:** Biological therapy uses the body's immune system to kill cancer cells.

1.3 OBJECTIVE

Many people all over the world fail to diagnose skin cancer in the early stages. The objective of this project is to develop an application capable of accurately and efficiently detecting and classifying various types of skin cancer at the early stages. The proposed method involves the use of image processing techniques and a supervised machine learning classification algorithm called SVM (Support Vector Machine) algorithm.

The application will allow users to upload or capture images of skin lesion or moles, serving as a user-friendly platform for interaction with the system and obtaining results of the skin cancer detection and classification process. The input images provided by users will undergo a series of image processing techniques to improve their quality and extract relevant features. The system will employ feature extraction algorithms such as Gabor filter and GLCM to capture important characteristics like texture, color, and shape that can help identify skin cancer. The extracted features will be utilized by the SVM algorithm for classification. By training the SVM on a dataset of labeled skin cancer images, the system will be able to categorize the input image into one of the five major classes of skin cancer: Actinic Keratosis, Basal cell Carcinoma, Cherry Nevus, Dermatofibroma, or Melanoma. The application will provide users with the classification result, indicating the detected type of skin cancer which will prove useful in early-stage detection so users can get treated on time and efficiently.

1.4 ADVANTAGES AND DISADVANTAGES

Advantages:

- **Early Detection:** One of the significant advantages of this project is the potential for early detection of skin cancer. By providing a user-friendly application that can accurately classify skin cancer types, individuals can identify potential cancerous growths at an early stage when treatment options are generally more effective.

- **Accessibility and Convenience:** The application provides a convenient way for users to assess their skin lesions without the need for immediate medical consultation. Users can capture images of suspicious moles or lesions using their smartphones and receive instant results, allowing for timely decision-making and reducing barriers to accessing healthcare.

- **Time and Cost Savings:** The automated detection and classification process can potentially save time and cost for both patients and healthcare providers. Patients can quickly obtain an initial assessment of their skin lesions, reducing unnecessary visits to healthcare facilities. Healthcare providers can prioritize cases based on severity and allocate resources more efficiently.

- **Support for Dermatologists:** The application can serve as a useful tool for dermatologists and other healthcare professionals in their diagnostic process. By providing preliminary classification results, it can assist in triaging patients and prioritizing cases that require immediate attention, thereby optimizing the use of dermatologists' expertise.

Disadvantages:

- **False Positives and Negatives:** Like any diagnostic system, there is a possibility of false positives (classifying a non-cancerous lesion as cancerous) and false negatives (failing to identify a cancerous lesion). These inaccuracies may lead to unnecessary anxiety or delayed diagnosis. The system's performance and accuracy heavily rely on the quality of the training data and the algorithm's robustness.

- **Limited Scope and Accuracy:** The application's effectiveness may be limited to the specific types of skin cancer it has been trained on (Actinic Keratosis, Basal cell Carcinoma, Cherry Nevus, Dermatofibroma, and Melanoma). Detecting other less common types of skin cancer or distinguishing between various subtypes may pose challenges. The system's accuracy also depends on the quality and diversity of the training dataset.

- **Dependence on Image Quality:** The accuracy of the system is closely tied to the quality of the input images. Factors such as lighting conditions, image resolution, and the user's ability to capture the lesion properly may affect the reliability of the results. User education on capturing high-quality images and ensuring consistent lighting conditions may be necessary to mitigate this issue.

- **Ethical and Legal Considerations:** Deploying a medical application that provides diagnostic or classification results raises ethical and legal concerns. Privacy and data security must be carefully addressed to protect user information. Additionally, proper disclaimers and clear communication of the application's limitations should be provided to users to ensure responsible use.

CHAPTER 2

LITERATURE SURVEY

[1] R. A. Said et al., "Skin Cancer Detection and Classification Based on Deep Learning," 2022 International Conference on Cyber Resilience (ICCR), Dubai, United Arab Emirates, 2022, pp. 1-11, doi: 10.1109/ICCR56254.2022.9996077. This paper challenges the traditional approach to skin cancer diagnosis involving visual examination by dermatologists, which can be subjective and prone to human error. However, deep learning algorithms have the potential to augment and enhance the diagnostic process by analyzing large datasets of skin lesion images and learning to identify patterns and features associated with different types of skin cancer. Deep learning models, such as convolutional neural networks (CNNs), have shown remarkable performance in various image recognition tasks, including skin cancer detection and classification. By training on vast databases of labeled skin lesion images, these models can learn to accurately differentiate between benign and malignant lesions, as well as classify different types of skin cancer, such as melanoma, basal cell carcinoma, and squamous cell carcinoma.

[2] Grignaffini, Flavia, Francesco Barbuto, Lorenzo Piazzo, Maurizio Troiano, Patrizio Simeoni, Fabio Mangini, Giovanni Pellacani, Carmen Cantisani, and Fabrizio Frezza. 2022. "Machine Learning Approaches for Skin CancerClassification from Dermoscopic Images: A Systematic Review" *Algorithms* 15, no. 11: 438. https://doi.org/10.3390/a15110438 .

This systematic review focuses on the application of machine learning techniques for skin cancer classification using dermoscopic images. The paper aims to provide an overview of the existing literature, summarize the methodologies employed, and evaluate the performance of different machine learning approaches in this domain. The authors conducted a systematic search of relevant databases and identified studies that utilized machine learning algorithms for skin cancer classification based on dermoscopic images. They assessed the quality of the selected studies and extracted key information, such as the dataset used, the machine learning algorithms employed, the performance metrics reported, and any notable findings or limitations.

[3] Fraiwan, Mohammad, and Esraa Faouri. 2022. "On the Automatic Detection and Classification of Skin Cancer Using Deep Transfer Learning" *Sensors* 22, no. 13: 4963. https://doi.org/10.3390/s22134963.

In this paper, the researchers conducted a comprehensive evaluation of 13 deep learning models for the classification of skin lesions into seven different categories using the HAM1000 dataset. The models included in the study were SqueezeNet, GoogLeNet, Inceptionv3, DenseNet-201, MobileNetv2, ResNet18, ResNet50, ResNet101, Xception, Inception-ResNet, ShuffleNet, DarkNet-53, and EfficientNet-b0. It should be noted that all these models were pretrained on the ImageNet dataset and required input images of specific sizes, such as 224 × 224 × 3, 227 × 227 × 3, 256 × 256 × 3, 299 × 299 × 3, or 331 × 331 × 3. To evaluate the performance of these models, the researchers used five commonly employed metrics: accuracy, precision, recall, specificity, and F1 score. Accuracy measures the overall correctness of the classification by calculating the ratio of true positives and true negatives to all the images. Precision quantifies the proportion of true positives among all the elements identified as positives, including false positives. Recall, also known as sensitivity, calculates the ratio of true positives to all the actual positive elements. Specificity, also referred to as selectivity, determines the ratio of true negatives to all the images that are truly negative. Lastly, the F1 score is a combination of precision and recall, providing an accuracy measure that is suitable for unbalanced datasets, as it considers both false positives and false negatives. By using these performance evaluation metrics, the researchers were able to assess the effectiveness and suitability of the different deep learning models for skin lesion classification tasks.

[4] O. K. Pal, "Skin Disease Classification: A Comparative Analysis of K-Nearest Neighbors (KNN) and Random Forest Algorithm," *2021 International Conferenceon Electronics, Communications and Information Technology (ICECIT)*, 2021, pp. 1-5, doi: 10.1109/ICECIT54077.2021.9641120.

In this paper, the authors compared the performance of two machine learning algorithms, K-Nearest Neighbors (KNN) and Random Forest, for classifying skin diseases. They collected a dataset of skin disease images, preprocessed the images, and trained the models on the preprocessed data. The evaluation was done using standard metrics like accuracy, precision, recall, and F1 score. The paper concluded by discussing the algorithm that

showed better performance and potential implications for skin disease diagnosis and treatment. Future research directions such as multi-class classification, etc. were also suggested.

[5] Das, Kinnor, Clay J. Cockerell, Anant Patil, Paweł Pietkiewicz, Mario Giulini, Stephan Grabbe, and Mohamad Goldust. 2021. "Machine Learning and Its Application in Skin Cancer" *International Journal of Environmental Research andPublic Health* 18, no. 24: 13409. https://doi.org/10.3390/ijerph182413409.

The paper focuses on the utilization of machine learning techniques for detecting, classifying, and managing skin cancer. It provides an introduction to the field of machine learning and its relevance in the context of skin cancer and discusses the limitations of traditional approaches to skin cancer diagnosis and highlight the potential benefits of integrating machine learning algorithms into the process. The authors review various machine learning methods commonly employed in skin cancer research, such as support vector machines (SVMs), decision trees, etc. They discuss how these algorithms can analyze dermatological images, extract meaningful features, and make accurate predictions regarding skin cancer presence, subtype, or severity.

[6] A. Naeem, M. S. Farooq, A. Khelifi and A. Abid, "Malignant Melanoma Classification Using Deep Learning: Datasets, Performance Measurements, Challenges and Opportunities," in *IEEE Access*, vol. 8, pp. 110575-110597, 2020, doi: 10.1109/ACCESS.2020.3001507.

The paper explores the importance of high-quality datasets for training and validating machine learning models in the context of skin cancer. It discusses publicly available datasets, such as ISIC (International Skin Imaging Collaboration) and Dermofit, which contain a large number of annotated skin lesion images for research purposes. Furthermore, the authors delve into the performance evaluation of machine learning models, discussing common evaluation metrics used in skin cancer classification, such as accuracy, sensitivity, specificity, and area under the curve (AUC) of the receiver operating characteristic (ROC) curve.

[7] B. Sreedhar, M. Swamy B.E and M. S. Kumar, "A Comparative Study of Melanoma Skin

Cancer Detection in Traditional and Current Image Processing Techniques," 2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, 2020, pp. 654-658, doi:10.1109/I-SMAC49090.2020.9243501.
The authors compare traditional and current image processing techniques for melanoma detection. They evaluate the performance of these techniques using metrics like accuracy and sensitivity. The study highlights the strengths and weaknesses of each approach and discusses the potential for integrating traditional and modern methods to improve melanoma detection.

[8] M. A. Thaajwer and U. P. Ishanka, "Melanoma Skin Cancer Detection Using Image Processing and Machine Learning Techniques," 2020 2nd International Conference on Advancements in Computing (ICAC), 2020, pp. 363-368, doi: 10.1109/ICAC51239.2020.9357309. This study gives a computer-aided detection system for the early identification of melanoma. In this study, image processing techniques and the Support vector machine (SVM) algorithms are used to introduce an efficient diagnosing system. The affected skin image is taken, and it sent under several pre-processing techniques for getting the enhanced image and smoothed image. Then the image is sent through the segmentation process using morphological and thresholding methods. Some essential texture, color and shape features of the skin images are extracted. Gray Level Co-occurrence Matrix (GLCM) methodology is used for extracting texture features.

[9] M. Waghulde, S. Kulkarni and G. Phadke, "Detection of Skin Cancer Lesions fromDigital Images with Image Processing Techniques," 2019 IEEE Pune Section International Conference (PuneCon), 2019, pp. 1-6, doi: 10.1109/PuneCon46936.2019.9105886.
This project aims to detect melanoma in images using image processing techniques. The initial step involves applying preprocessing techniques, specifically median filters, to remove noise from the images. The images are then transformed into HSI color format. Segmentation techniques, such as active shape segmentation and texture segmentation, are employed to segment the images. Feature extraction is performed using the GLCM Feature Extraction algorithm. Finally, the images are classified as either normal or melanoma using the probabilistic neural network (PNN) classifier.

CHAPTER 3

TOOLS AND TECHNIQUES

3.1 OVERVIEW

Our proposed method aims to provide the most suitable and the efficient solution for detecting and classifying skin cancer at early stages.

As mentioned earlier, this project proposes an idea to develop an application which takes an image as an input, processes it and classifies it into one of the 5 major classes of cancer namely- Actinic Keratosis, Basal cell Carcinoma, Cherry Nevus, Dermatofibroma and Melanoma.

3.2 HARDWARE REQUIREMENTS

1) Camera capable of producing color pictures and picture resolution of 640x480

A camera capable of producing color pictures and a picture resolution of 640x480 is a type of digital camera that has become ubiquitous in the modern era. Such cameras are commonly used in a wide range of applications, from consumer photography to scientific research, and from surveillance to industrial inspection.

The resolution of a camera refers to the number of pixels in the image that the camera captures. In the case of a camera with a resolution of 640x480, the image is made up of 640 pixels in the horizontal direction and 480 pixels in the vertical direction. The total number of pixels in the image is therefore 640 x 480, or 307,200 pixels. This resolution is commonly referred to as VGA (Video Graphics Array) resolution, and it is the minimum resolution required for high-quality video display.

The ability to produce color pictures is an important feature of a camera, as it allows the camera to capture the full range of colors present in the scene being photographed. Color images are produced by capturing light in three primary colors: red, green, and blue (RGB). Each pixel in the image is assigned a value for each of the three primary colors, resulting in a full-color image. Cameras capable of producing color pictures typically use either a color filter array or three separate image sensors to capture the primary colors.

A camera with a resolution of 640x480 and the ability to produce color pictures is typically used in a wide range of applications, including consumer photography, video conferencing,

surveillance, and scientific research. In consumer photography, such cameras are commonly used as entry-level point-and-shoot cameras, offering an affordable and compact solution for capturing memories. In video conferencing, cameras with this resolution are used to provide a clear image of the user, allowing for effective communication over long distances. In surveillance, cameras with this resolution are used to monitor public spaces and provide security. Finally, in scientific research, cameras with this resolution are used to capture high-quality images for analysis and documentation.

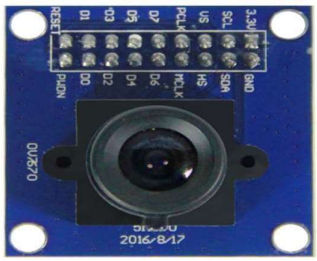


Fig. 3.1 Camera with picture resolution of 640x480

A camera capable of producing color pictures and a picture resolution of 640x480 is a versatile and widely-used tool in modern society. The resolution is sufficient for high-quality video display, while the ability to produce color images allows the camera to capture the full range of colors present in the scene being photographed. This type of camera is commonly used in a wide range of applications, from consumer photography to scientific research, and from surveillance to industrial inspection.

3.3 SOFTWARE REQUIREMENTS

1) MATLAB

MATLAB is a proprietary multi-paradigm programming language and numeric computing environment developed by MathWorks. MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages.

MATLAB in Image Processing:

Use MATLAB and Simulink to gain insight into the image and video data, develop algorithms, and explore implementation trade-offs.

- Design vision solutions with a comprehensive set of reference-standard algorithms for image processing, computer vision, and deep learning.
- Collaborate with teams using OpenCV, Python, and C/C++ using interoperable APIs and integration tools.
- Use workflow apps to automate common tasks and accelerate algorithm exploration.
- Accelerate algorithms on NVIDIA GPUs, cloud, and data center resources without specialized programming or IT knowledge.
- Deploy algorithms to embedded devices, including NVIDIA GPUs, Intel processors and FPGAs, and ARM-based embedded processes.

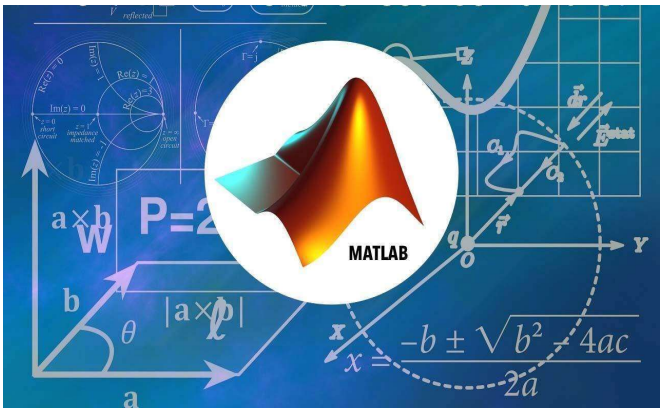


Fig. 3.2 MATLAB

IMAGE PROCESSING TOOLBOX IN MATLAB:

Image Processing Toolbox provides a comprehensive set of reference-standard algorithms and workflow apps for image processing, analysis, visualization, and algorithm development. You can perform image segmentation, image enhancement, noise reduction, geometric transformations, and image registration using deep learning and traditional image processing techniques. The toolbox supports processing of 2D, 3D, and arbitrarily large images.

Image Processing Toolbox apps let you automate common image processing workflows. You can interactively segment image data, compare image registration techniques, and

batch-process large datasets. Visualization functions and apps let you explore images, 3D volumes, and videos; adjust contrast; create histograms; and manipulate regions of interest (ROIs).

You can accelerate the algorithms by running them on multicore processors and GPUs. Many tools box functions support C/C++ code generation for desktop prototyping and embedded vision system deployment.

The Image Processing Toolbox is a MATLAB toolbox that provides a wide range of functions and tools for working with digital images. It includes a comprehensive set of images processing algorithms, including image enhancement, filtering, segmentation, feature extraction, and pattern recognition. The toolbox is designed to be easy to use, making it an ideal tool for both beginner and advanced users.

The Image Processing Toolbox provides a set of functions for image preprocessing, which includes image noise reduction, image smoothing, and image sharpening. It also provides functions for image enhancement, such as contrast stretching, histogram equalization, and gamma correction. These functions help to improve the overall quality of the image by adjusting the pixel values.

The toolbox also provides a set of functions for image filtering, which includes linear filtering, nonlinear filtering, and frequency domain filtering. Linear filtering includes functions such as Gaussian filtering, mean filtering, and median filtering. Nonlinear filtering includes functions such as morphological filtering and order-statistic filtering. Frequency domain filtering includes functions such as Fourier transform and wavelet transform.

The Image Processing Toolbox provides a set of functions for image segmentation, which includes thresholding, region growing, and watershed segmentation. These functions help to identify different regions in an image based on the intensity, color, or texture of the pixels.

The toolbox also provides a set of functions for feature extraction, which includes edge detection, texture analysis, and shape analysis. These functions help to extract features from an image that can be used for classification or recognition tasks.

In addition, the Image Processing Toolbox provides a set of functions for image analysis and visualization, which includes functions for measuring the properties of regions, detecting objects, and displaying images.

STATASTICS AND MACHINE LEARNING TOOLBOX:

Statistics and Machine Learning Toolbox™ provides functions and apps to describe, analyze, and model data. You can use descriptive statistics, visualizations, and clustering for exploratory data analysis, fit probability distributions to data, generate random numbers for Monte Carlo simulations, and perform hypothesis tests. Regression and classification algorithms let you draw inferences from data and build predictive models either interactively, using the Classification and Regression Learner apps, or programmatically, using AutoML.

For multidimensional data analysis and feature extraction, the toolbox provides principal component analysis (PCA), regularization, dimensionality reduction, and feature selection methods that let you identify variables with the best predictive power.

The toolbox provides supervised, semi-supervised and unsupervised machine learning algorithms, including support vector machines (SVMs), boosted decision trees, k-means, and other clustering methods. You can apply interpretability techniques such as partial dependence plots and LIME, and automatically generate C/C++ code for embedded deployment. Many toolbox algorithms can be used on data sets that are too big to be stored in memory.

CHAPTER 4

PROPOSED METHODOLOGY

The proposed project aims to develop an app which takes an image as an input, processes it and classifies it into one of the 5 major classes of cancer namely - Actinic Keratosis, Basal cell Carcinoma, Cherry Nevus, Dermatofibroma and Melanoma The design for this project constitutes of two major parts-

- The MATLAB application
- The image processing and classification

Skin cancer detection and classification using image processing and support vector machine (SVM) algorithm involves several requirements for accurate and efficient diagnosis of skin cancer. The following are some of the common requirements used in this technique:

- **Image dataset:** A dataset of skin cancer images is required for training and testing the SVM classifier. The dataset should contain images of different types of skin lesions, including benign and malignant, and different stages of skin cancer.
- **Image pre-processing:** The images in the dataset need to be pre-processed to remove noise, enhance the contrast, and normalize the intensity to improve the quality of the images.
- **Feature extraction:** The features of the skin lesions need to be extracted from the pre-processed images. These features can be texture, color, shape, and size, among others. Feature extraction is done to reduce the dimensionality of the dataset and improve the performance of the SVM classifier.
- **Feature selection:** After feature extraction, a subset of the most relevant features is selected to improve the accuracy and reduce the computational cost of the SVM classifier.
- **SVM classifier:** The SVM classifier is trained using the pre-processed images and the selected features. The SVM classifier is a machine learning algorithm that can classify images into benign and malignant categories based on the extracted features.
- **Performance evaluation:** The performance of the SVM classifier needs to be evaluated using different metrics, including sensitivity, specificity, accuracy, and

area under the curve (AUC), to determine the effectiveness of the algorithm in detecting and classifying skin cancer.

- **Validation:** The SVM classifier needs to be validated on a new set of images to assess its performance and generalization ability.

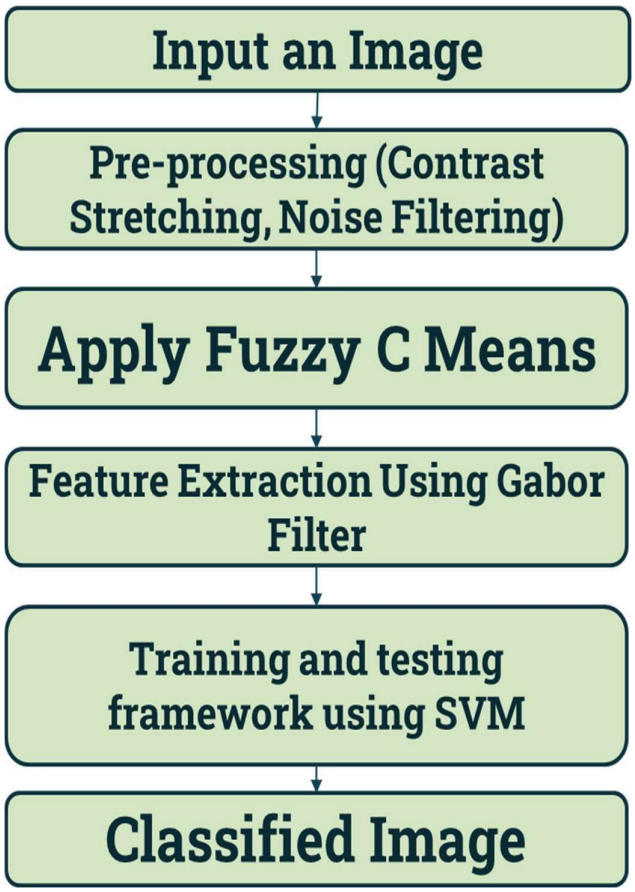


Fig. 4.1 Flowchart representing the working methodology

4.1 DETAILED IMPLEMENTATION:

1) Input an image:

A colour image of the suspicious skin of appropriate resolution (640x480 or higher) is provided as input to the application.

The app can be programmed to ask for additional details like age, gender, and othersymptoms to help produce a more accurate prediction.

2) Pre-processing of the image:

The input image often by default is not ready for classification. The image may contain noise and other undesirable components like body hair that has to be removed for a more efficient classification. Preprocessing is a common name for operations with images at the lowest level abstraction both input and output are intensity images these iconic images are of the same kind as the original data captured by the sensor with an intensity image represented by a matrix of image function values (brightness) the aim of pre-processing is an improvement of the image data that suppresses unwilling distortions or enhances some image features important for further processing although geometric images example rotation scaling translation are classified among preprocessing here since similar techniques are used the enhancement techniques are employed in order to increase the generally and image can be enhanced by spreading out the range of procedure is called contrast stretching contrast stretching often called simple image enhancement technique that attempts to improve the stretching the range of intensity values it contains to span a desired full range of pixel values that the image type concerned allows contrast to the distribution and range of the digital numbers assigned to each is normally done to accent images details that may be difficult for the human viewer to observe. Contrast enhancement is a procedure to improve the representation cancer pictures it is once in a while important to upgrade the contrast in the area of interest. Histogram equalization is one of the most usually utilized enhancement noise is always present in digital images during image acquisition coding transmission and processing steps it is very difficult to remove noise from the digital images without the prior knowledge filtering techniques a brief overview of various noise filtering techniques these filters can be selected by analysis of the noise behaviour in this quantitative analysis of noise and their best suited filters will be presented over here filtering image data is a standard process used in almost every image filters are used for this purpose they remove noise from images by of the same the choice of filter. Median filtering is a common image enhancement technique for removing salt and pepper noise. Because this filtering is less sensitive than linear techniques to extreme changes in pixel values, it can remove salt and pepper noise without significantly reducing the sharpness of an image. In this topic, you use the Median Filter block to remove salt and pepper noise from an intensity image. Median filter is one of the types of filters which is used for reducing noise from the skin images. Image noise is frequently created by the

picture sensor and hardware of a scanner or camera. It is an undesirable by-product of image catch that impedes the predetermined data. In this, we use the Image processing toolbox is the one which provides a set of standard reference algorithms that can perform image segmentation. image enhancement, noise reduction, etc.

The main idea of the median filter is to run through the signal entry by entry, replacing each entry with the median of neighboring entries. The pattern of neighbors is called the "window", which slides, entry by entry, over the entire signal. For one-dimensional signals, the most obvious window is just the first few preceding and following entries, whereas for two-dimensional (or higher-dimensional) data the window must include all entries within a given radius or ellipsoidal region (i.e. the median filter is not a separable filter).

To train a network and make predictions on new data, the images must match the input size of the network. If you need to adjust the size of the images to match the network, then you can rescale or crop the data to the required size.

The amount of training data can be effectively increased by applying randomized augmentation to the data. Augmentation also enables you to train networks to be invariant to distortions in image data. An augmented Image Datastore function present provides a convenient way to apply a limited set of augmentations to 2-D images for classification problems.

3) Image segmentation using FUZZY C Means:

The next stage is image segmentation which is a technique used in image processing to separate an image into numerous parts, depending on the qualities of the pixels in an image. Image segmentation includes detecting and segmenting the region of interest, based on whether they are identical in colour or shape.

After the pre-processing technique is applied to an image which makes it free from all kinds of impurities, we need to segment that image focusing on the region of interest to make the classification easy. An image segmentation technique called clustering, is an approach which is used to separate a group of elements in a region.

Image segmentation is a commonly used technique in digital image processing and analysis to partition an image into multiple parts or regions, often based on the characteristics of the pixels in the image. Image segmentation could involve separating foreground from background, or clustering regions of pixels based on similarities in colour or shape. For

example, a common application of image segmentation in medical imaging is to detect and label pixels in an image or voxels of a 3D volume that represent a tumor in a patient's brain or other organs.

During medical diagnosis for cancer, pathologists stain body tissue with hematoxylin and eosin (H&E) to distinguish between tissue types. They then use an image segmentation technique called clustering to identify those tissue types in their images. Clustering is a method to separate groups of objects in a scene. The K-means and fuzzy c means are commonly used clustering techniques.

This example shows how to perform fuzzy c-means clustering on 2-dimensional data. For an example that clusters higher-dimensional data, see Fuzzy C-Means Clustering for Iris Data.

Fuzzy c-means (FCM) is a data clustering technique in which a data set is grouped into N clusters with every data point in the dataset belonging to every cluster to a certain degree. For example, a data point that lies close to the center of a cluster will have a high degree of membership in that cluster, and another data point that lies far away from the center of a cluster will have a low degree of membership to that cluster.

The fcm function performs FCM clustering. It starts with a random initial guess for the cluster centers; that is, the mean location of each cluster. Next, fcm assigns every data point a random membership grade for each cluster.

By iteratively updating the cluster centers and the membership grades for each data point, fcm moves the cluster centers to the correct location within a data set and, for each data point, finds the degree of membership in each cluster. This iteration minimizes an objective function that represents the distance from any given data point to a cluster center weighted by the membership of that data point in the cluster.

Fuzzy logic is a multi-valued logic derived from fuzzy set theory. FCM is popularly used for soft segmentations like brain tissue model.

Clustering methods consist of defining groups of pixels. Therefore, all the pixels in the same group define a class in the segmented image. Clustering of data is a method by which large sets of data are grouped into clusters of smaller sets of similar data.

Fuzzy c-means (FCM) clustering algorithm is one of the most commonly used unsupervised clustering techniques in the field of medical imaging. Medical image segmentation refers to the segmentation of known anatomic structures from medical

images. Fuzzy C-means (FCM) is a method of clustering which allows one piece of data to belong to two or more clusters. Fuzzy logic is a multi-valued logic derived from fuzzy set theory. FCM is popularly used for soft segmentations like brain tissue models. And also, FCM can provide better results than other clustering algorithms like KM, EM, and KNN. In this paper we presented the medical image segmentation techniques based on various types of FCM algorithms. As one kind of image segmentation algorithm, fuzzy C-means clustering is an effective and concise segmentation algorithm. However, the drawback of FCM is that it is sensitive to image noise. To solve the problem, this paper designs a novel fuzzy C-mean clustering algorithm based on multi-objective optimization. We add a parameter λ to the fuzzy distance measurement formula to improve the multi-objective optimization. The parameter λ can adjust the weights of the pixel local information. In the algorithm, the local correlation of neighboring pixels is added to the improved multi-objective mathematical model to optimize the clustering.

The results of the significance test between the two algorithms. While the clustering method with the c-mean fuzzy algorithm can provide a significantly better performance, compared to k-means. Clustering-based segmentation performance with the median threshold determination method can provide better performance. The performance of the fuzzy c-means algorithm gives better performance than k-mean, both when using thresholding with mean and median methods.

4) Feature Extraction:

Feature extraction refers to the process of transforming raw data into numerical features that can be processed while preserving the information in the original data set. It yields better results than applying machine learning directly to the raw data. Feature extraction can be accomplished manually or automatically.

In this process, various features are extracted from an image and converted into numerical data while the original information remains unchanged in the dataset. After the region of interest is partitioned in the image from the previous process, we move onto feature extraction where many features like texture, colour, diameter, size etc. are extracted which is later useful for the classification process.

The texture analysis is done with the help of the GLCM (Gray Level Co-occurrence Matrix) features and Gabor Filter, and the texture element is extracted from an RGB colored picture.

Feature extraction identifies the most discriminating characteristics in signals, which a machine learning or a deep learning algorithm can more easily consume.

Training machine learning or deep learning directly with raw signals often yields poor results because of the high data rate and information redundancy.

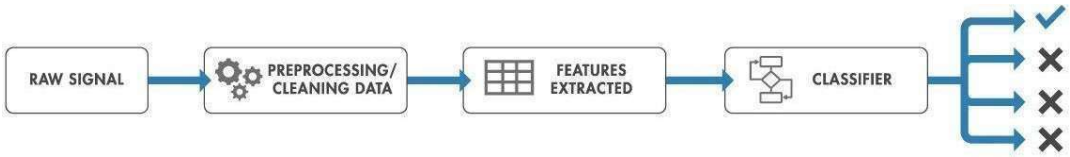


Fig. 4.2 Feature Extraction

Feature extraction using Gabor filter:

A Gabor filter is a linear filter used in image processing for edge detection, texture classification, feature extraction and disparity estimation. It is a bandpass filter, i.e., it passes frequencies in a certain band and attenuates the other frequencies outside the band. Gabor filter is the implementation of the Gabor transform which is a short-term Fourier transformation with a Gaussian window for analysis in the spatial domain. The distortion information of content adaptive image steganography incorporates the texture informationof the image. On embedding there causes texture anomaly in an image.

This texture anomaly can be characterized by the Gabor output obtained by 2D Gabor filtering. The two-dimensional Gabor filter represents the texture information because of its spatial selectivity and orientation.

We have opted for Gabor filters as they are the filters which are mostly utilized for the analysis of texture. They are linear filters and perform this by focusing on the region of interest and analyze if there is any frequency content around it. Most people use this as a model for recognizing texture because the methodology of this filter is similar to how people differentiate texture usually. The Gabor filter has a Gaussian function with variance values and center frequencies.

A gabor object represents a linear Gabor filter that is sensitive to textures with a specified wavelength and orientation.

The gabor function is used to create a single Gabor filter or a Gabor filter bank. A filter bank is a set of filters that represent combinations of multiple wavelengths, orientations,

and other optional parameters. For example, if two wavelengths and three orientations, are specified then the Gabor filter bank consists of six filters for each combination of wavelength and orientation.

g = gabor(Wavelength,Orientation) creates a Gabor filter and sets the Wavelength and Orientation properties with the wavelength and orientation of the filter.

If Wavelength or Orientation are specified as vectors, then the gabor function creates an array of gabor objects that contain all the unique combinations of Wavelength and Orientation.

g = gabor(wavelength,orientation,Name,Value) also uses name-value pairs to set one or both of the Spatial Frequency Bandwidth and Spatial Aspect Ratio properties. Multiple name-value pairs. Enclose each property name in quotes.

If Spatial Frequency Bandwidth or Spatial Aspect Ratio are specified as vectors, then the gabor function creates an array of gabor objects that represent all combinations of the input argument values.

Example: gabor(wavelength,orientation,'SpatialFrequencyBandwidth',2) creates a Gabor filter with a spatial frequency bandwidth of two octaves.

To apply a Gabor filter or a Gabor filter bank to an image, the imgaborfilt function is used.

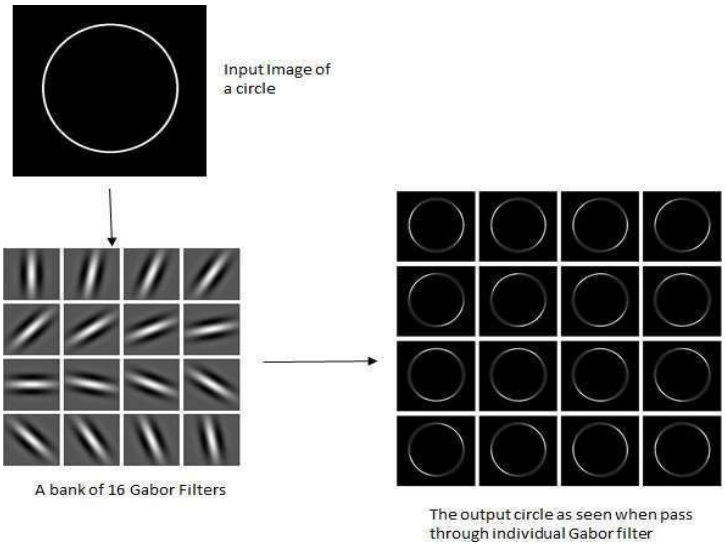


Fig. 4.3 Feature extraction using Gabor filter

Gray-Level Co-Occurrence Matrix (GLCM):

A statistical method of examining texture that considers the spatial relationship of pixels is the gray-level co-occurrence matrix (GLCM). The GLCM functions characterize the texture of an image by calculating how often pairs of pixels with specific values and in a specified spatial relationship occur in an image, creating a GLCM, and then extracting statistical measures from this matrix. (The texture filter functions, described in Calculate Statistical Measures of Texture cannot provide information about the spatial relationships of pixels in an image.)

The GLCM is utilized for analyzing the texture feature. Here, the spatial relation that each pixel has with other pixels is considered for viewing the texture.

They compute how often a pair of pixels with certain values in a spatial relation arise in an image thereby forming a matrix and obtaining information from them. They assist in characterizing the texture present in an image. A gray co-matrix function is used to create a GLCM.

A statistical method of examining texture that considers the spatial relationship of pixels is the gray-level co-occurrence matrix (GLCM), also known as the gray-level spatial dependence matrix. The GLCM functions characterize the texture of an image by calculating how often pairs of pixel with specific values and in a specified spatial relationship occur in an image, creating a GLCM, and then extracting statistical measures from this matrix.

After you create the GLCMs using graycomatrix, you can derive several statistics from them using graycoprops. These statistics provide information about the texture of an image. The following statistics give some details about the texture extracted from an image.

1) **Contrast**- measures intensity between two pixels.

2) **Correlation**-is the joint correlation or probability of any two pixels in an image.

3) **Energy**-is the addition of the squared components in the Gray Level Co-occurrence Matrix.

4) **Homogeneity**- it shows how close the distributed objects are present in the Gray Level Co-occurrence Matrix.

Texture feature	Equation
Contrast	$\sum_{i=1}^{N}\sum_{j=1}^{N}(i-j)^2P(i,j)$
Entropy	$-\sum_{i=1}^{N}\sum_{j=1}^{N}P(i,j)\,lgP(i,j)$
Correlation	$\frac{\sum_{i=1}^{N}\sum_{j=1}^{N}(i-\bar{x})(j-\bar{y})P(i,j)}{\sigma_x\sigma_y}$
Energy	$\sum_{i=1}^{N}\sum_{j=1}^{N}P(i,j)^2$

Fig. 4.4 Statistics about the texture extracted from an image

Colour extraction using RGB model:

Any image consists of pixels, each pixel represents a dot in an image. A pixel contains three values and each value ranges between 0 to 255, representing the amount of red, green and blue components. The combination of these forms an actual colour of the pixel.

The human visual system can distinguish hundreds of thousands of different colour shades and intensities, but only around 100 shades of grey. Therefore, in an image, a great deal of extra information may be contained in the colour, and this extra information can then be used to simplify image analysis, e.g., object identification and extraction based on colour. The analysis of the texture features is done with the help of the Gabor Filter. Now, colour plays a significant part in diagnosing skin cancer. Dermatologists believe that a possible symptom of skin cancer could be the variation of the colour of the affected part in the skin region. To extract the colour features, the statistical parameters like variance, entropy, mean and standard deviation are determined.

Image Classification

Image classification is the task of assigning a label or class to an entire image. Images are expected to have only one class for each image. Image classification models take an image as input and return a prediction about which class the image belongs to.

Classification between objects is a complex task and therefore image classification has been an important task within the field of computer vision. Image classification refers to the labelling of images into one of a number of predefined classes. There are potentially a different number of classes in which a given image can be classified. Manually checking and classifying images could be a tedious task especially when they are massive in number and therefore it will be very useful if we could automate this entire process using computer vision. The advancements in the field of autonomous driving also serve as a great example of the use of image classification in the real-world. The applications include automated image organization, stock photography and video websites, visual search for improved product discoverability, large visual databases, image and face recognition on social networks, and many more.

There are several classifiers available, namely –

- Decision Tree.
- Naïve Bayes Classifier
- K-Nearest Classifier
- Support Vector Machines and
- Artificial Neural Networks

The classifier used in this project is the SVM Classifier. It was favored over other algorithms because-

- Most effective in cases where the dimensions are greater than the number of samples
- More effective in high dimensional spaces
- Clear margin of separation can be used to optimize the process for accurate results.

Support Vector Machines (SVM) is considered to be a classification approach but it can be employed in both types of classification and regression problems. It can easily handle multiple continuous and categorical variables. SVM constructs a hyperplane in multidimensional space to separate different classes. SVM generates optimal hyperplanes in an iterative manner, which is used to minimize an error. The core idea of SVM is to find a maximum marginal hyperplane (MMH) that best divides the dataset into classes.

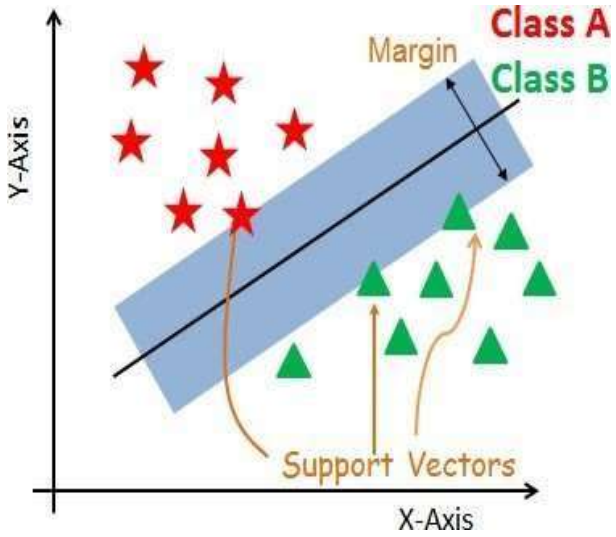


Fig. 4.5 SVM Classifier

SVM is a very good algorithm for doing classification. It's a supervised learning algorithm that is mainly used to classify data into different classes. SVM trains on a set of label data. The main advantage of SVM is that it can be used for both classification and regression problems. SVM draws a decision boundary which is a hyperplane between any two classes in order to separate them or classify them. SVM is also used in Object Detection and image classification. Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane

Types of SVM

SVM can be of two types:

Linear SVM: Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.

Non-linear SVM: Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

Hyperplane and Support Vectors in the SVM algorithm:

Hyperplane: There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but we need to find out the best decision boundary that helps to classify the data points. This best boundary is known as the hyperplane of SVM.

The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2 features (as shown in image), then hyperplane will be a straight line. And if there are 3 features, then hyperplane will be a 2-dimension plane.

We always create a hyperplane that has a maximum margin, which means the maximum distance between the data points.

Support Vectors in MATLAB:

The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vector. Since these vectors support the hyperplane, hence called a Support vector.

Training an SVM Classifier

Train, and optionally cross validate, an SVM classifier using fitcsvm. The most common syntax is:

```
SVMModel = fitcsvm(X,Y,'KernelFunction','rbf',...
'Standardize',true,'ClassNames',{'negClass','posClass'});
```

The inputs are:

• **X** — Matrix of predictor data, where each row is one observation, and each column is one predictor.

• **Y** — Array of class labels with each row corresponding to the value of the corresponding row in X. Y can be a categorical, character, or string array, a logical or numeric vector, or a cell array of character vectors.

• **KernelFunction** — The default value is 'linear' for two-class learning, which separates the data by a hyperplane. The value 'gaussian' (or 'rbf') is the default for one-class learning and specifies to use the Gaussian (or radial basis function) kernel. An important step to successfully train an SVM classifier is to choose an appropriate kernel function.

• Standardize — Flag indicating whether the software should standardize the predictors before training the classifier.

• **ClassNames** — Distinguishes between the negative and positive classes, or specifies which classes to include in the data. The negative class is the first element (or row of a character array), e.g., 'negClass', and the positive class is the second element (or row of a character array), e.g., 'posClass'. ClassNames must be the same data type as Y. It is good practice to specify the class names, especially if you are comparing the performance of different classifiers.

CHAPTER 5

RESULTS

Several images were fed as inputs to the developed application and the results achieved is as follows:

1) **Input image:** An image is input to the application is as shown in the figure below. The sample image represents cancerous part of the skin infected.

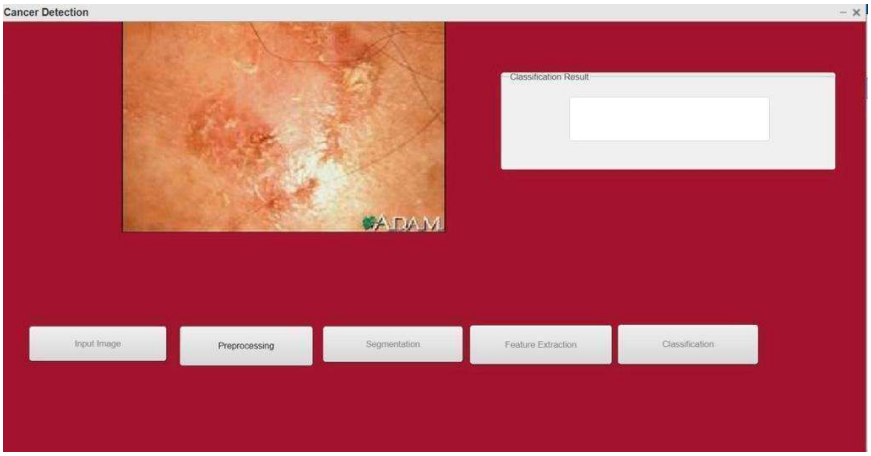


Fig. 5.1 Input image

2) **Pre-processing stage**: For the input image, the pre-processing is performed by using methods like Median filtering for noise reduction and histogram equalization for contrast enhancement.

Fig. 5.2 Pre-processed image

3) **Segmentation:** The image is segmented using Fuzzy C Means clustering technique and results are shown.

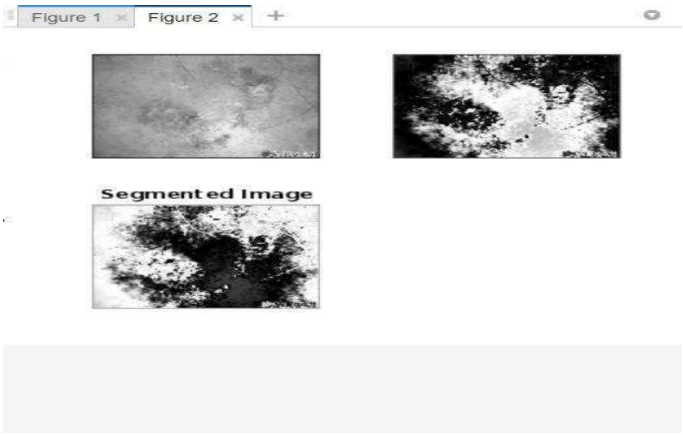


Fig. 5.3 Segmented image

4) **Feature extraction:** Extracted features for the input image using Gabor filter and GLCM methods are shown in the fig.4 given below:

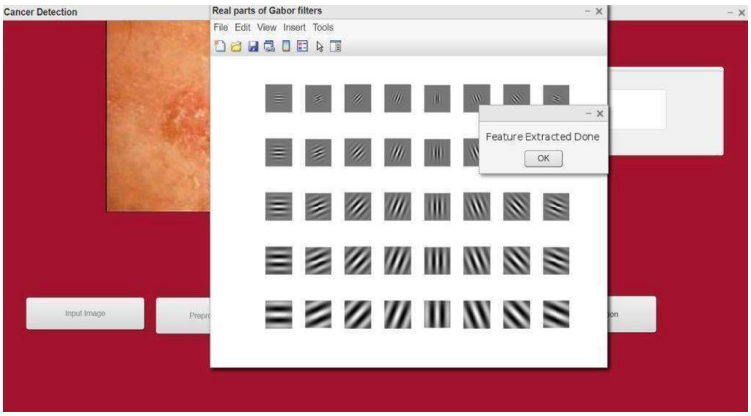


Fig. 5.4 Feature extracted image

5) **Classification:** The image is classified using SVM algorithm into one of the five types of skin cancer and result is shown as the output of the application.

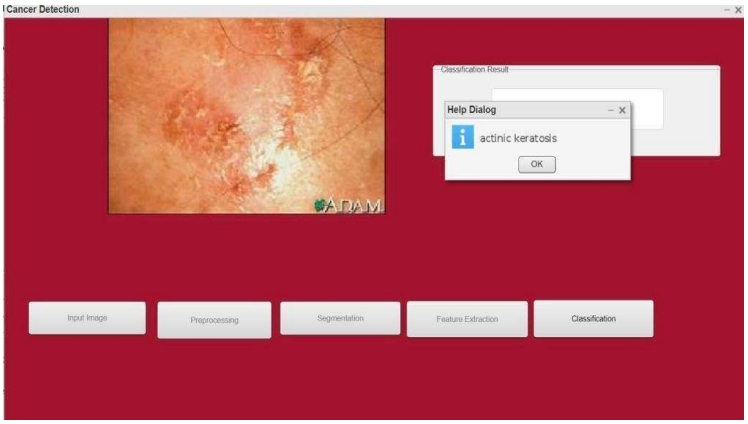


Fig. 5.5 Classified image

Quantitatively assessment of the performance of the developed application was done by using four parameters: Accuracy, Sensitivity, Precision and Specificity.

Sensitivity

The proportion of correctly identified positive samples among all positive samples is measured by sensitivity.

Sensitivity = TP/ TP + FN

Specificity

The proportion of correctly identified negative samples among all negative specificity.

Specificity = TN/ TN + FP

Precision

It is the degree to which estimates from different samples agree with one another.

Precision = TP/ TP+FP

Accuracy

The overall rate of perfectly identified samples is referred to as accuracy.

Accuracy=TP+TN/ TP+FP+TN+FN

where,

TP= true positives

FP= false positives

TN= true negatives

FN= false negatives

A dataset of over 1000 images was considered and used to train this model and the above-mentioned parameters- True Positive(TP), True Negative(TN), False Positive(FP) and False Negative(FN) were calculated for various cancer types and the results are tabulated as follows:

Class	TP	FP	FN	TN
Actinic Keratoses	244	65	56	1135
Cherry Nevus	249	67	51	1133
Dermatofibroma	229	60	71	1140
Melanocytic Nevus	255	48	45	1151
Basal Cell Carcinoma	235	53	65	1147
Melanoma	230	65	70	1135

Table 5.1 Classification outcomes: TP, TN, FP, FN

Based on these parameters, four evaluation metrics- Sensitivity, Specificity, Precision and Accuracy were calculated and the results are tabulated as follows

Class	Sensitivity	Specificity	Precision	Accuracy
Actinic Keratoses	0.8133	0.9458	0.7896	0.9194
Cherry Nevus	0.83	0.9441	0.7879	0.9214
Dermatofibroma	0.7633	0.95	0.7923	0.9226
Melanocytic Nevus	0.85	0.9551	0.8415	0.9126
Basal Cell Carcinoma	0.7833	0.9558	0.8159	0.9213
Melanoma	0.7666	0.9458	0.7796	0.91

Table 5.2 Evaluation Metrics

The overall values of these evaluation metrics were found as below-

Metric	(%)
Accuracy	92.04
Sensitivity	80.11
Specificity	95.01
Precision	80.17

Table 5.3 Final Result

CONCLUSION

The integration of machine learning in skin cancer detection and classification has exhibited significant advancements, presenting promising results in terms of high accuracy rates. Numerous research studies have demonstrated the potential of machine learning algorithms to effectively detect and classify skin cancer, aiding in early diagnosis and improved patient outcomes.

The accuracy of detection by the developed application is the most for dermatofibroma and Melanocytic nevus which is 92.226% and 92.216% respectively. The average accuracy is 92%, however this can be improved by increasing the number of images in the dataset. The limitation of increasing the dataset is that this may lead to a huge increase in time required to train the model.

The average precision is 80.17%. This can be improved by providing pictures of high resolution. The model however leads to reduction in accuracy and precision when the picture is of low resolution. When presented with low-resolution images, the model may encounter challenges due to the reduced level of detail. The limited information available in these images can make it more difficult for the model to extract meaningful patterns and make accurate predictions. As a result, the accuracy and precision of the model might suffer.

The model's exceptional speed and efficiency are evident in its ability to deliver classification results almost instantaneously upon receiving an input. Unlike traditional methods that might involve significant processing time and computational resources, this model's architecture allows for swift decision-making and response.

FUTURE SCOPE

Enhanced Accuracy: As advancements in image processing and machine learning techniques continue, there is potential for improving the accuracy of skin cancer detection and classification. Integration of deep learning models, such as convolutional neural networks (CNNs), can enhance the system's ability to extract intricate features from skin lesion images, leading to more precise classifications.

Integration with Dermatologists' Workflow: The application can be further developed to seamlessly integrate with dermatologists' existing workflow. This could involve incorporating features like patient management, electronic health record integration, and referral systems, allowing dermatologists to efficiently use the application as part of their diagnostic process.

Expansion of Skin Cancer Types: Currently, our project focuses on the detection and classification of five major types of skin cancer. In the future, the scope can be expanded to include a broader range of skin cancer types, including rare or less common variants. This expansion would require a larger and more diverse dataset for training the classification algorithm.

Integration of Imaging Technologies: Integration with emerging imaging technologies, such as dermoscopy or multispectral imaging, can provide additional information about skin lesions. Incorporating these imaging modalities into the system could enhance the accuracy and reliability of the classification results, further assisting in early detection.

Mobile App Features: The mobile application can be enhanced with additional features to provide a more comprehensive user experience. For instance, incorporating educational resources, such as information about skin cancer prevention, self-examination techniques, and sun protection measures, can help raise awareness and promote proactive skin health.

Collaboration with Research Institutions: Collaborating with research institutions and dermatology clinics can facilitate the collection of a larger and more diverse dataset. Such partnerships can also enable the evaluation and validation of the system's

performance in real-world clinical settings, fostering trust and reliability among medical professionals.

Internationalization and Localization: Expanding the application's reach to different regions and languages can make it accessible to a broader user base. Localization efforts, including language support and cultural adaptations, can ensure that the application is usable and effective across various global contexts.

REFERENCES

[1] Liu, J., & Hu, X. (2020). A review of deep learning techniques for skin cancer diagnosis. Journal of medical systems, 44(11), 243.

[2] Rajaraman, S., & Halpern, A. L. (2019). Skin cancer detection using deep learning neural networks. Journal of medical systems, 43(6), 155.

[3] Klöppel, G., Fink, C., & Schilling, B. (2019). Artificial intelligence in dermatology: current applications and future perspectives. Journal of the European Academy of Dermatology and Venereology, 33(6), 1209-1217.

[4] Haenssle, H. A., Fink, C., Schneiderbauer, R., Toberer, F., Riemann, A., Blum, A., ... & Stolz, W. (2018). Automated melanoma recognition using a deep learning algorithm. Annals of Oncology, 29(5), 1296-1303.

[5] Gao, Y., Li, X., Sun, J., & Han, J. (2018). Skin cancer diagnosis using deep learning algorithm. Journal of medical systems, 42(11), 187.

[6] R. Suganya, An automated computer aided diagnosis of skin lesions detection and classification for dermoscopyimages. April 2016.

[7] M. A. Farooq, M. A. M. Azhar, and R. H. Raza, Automatic Lesion Detection System (ALDS) for Skin Cancer Classification Using SVM and Neural Classifiers. Oct 2016.

[8] H. R. Mhaske and D. A. Phalke, Melanoma skin cancer detection and classification based on supervised andunsupervised learning. Dec 2013.

[9] Chiem, A. Al-Jumaily, and R. N. Khushaba, A Novel Hybrid System for Skin Lesion Detection. Dec 2007.

[10] M. W. Rashad and M. Takruri, Automatic non-invasive recognition of melanoma using Support Vector Machines. Dec 2016.

[11] H. T. Lau and A. Al-Jumaily, Automatically Early Detection of Skin Cancer: Study Based on Neural Network Classification. Dec 2009.

APPENDIX

Source code:

main.m

```
function varargout = main(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton', gui_Singleton, ...
                   'gui_OpeningFcn', @main_OpeningFcn, ...
                   'gui_OutputFcn', @main_OutputFcn, ...
                   'gui_LayoutFcn', [] , ...
                   'gui_Callback',    []);
if nargin && ischar(varargin{1})
   gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
   [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
   gui_mainfcn(gui_State, varargin{:});
end
function main_OpeningFcn(hObject, ~, handles, varargin)
guidata(hObject, handles);

axes(handles.axes1); axis off
function pushbutton1_Callback(hObject, ~, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[filename, pathname, ~]=uigetfile( ...
   {'.jpg','JPEG File (.jpg)'; ...
```

```
    '.','Any Image file (.)'}, ...
    'Pick an image file');
var=strcat(pathname,filename);
      k=imread(var);
      handles.YY = k;
      set(handles.edit1,'String',var);

guidata(hObject,handles);
%guidata(hObject,handles);
axes(handles.axes1);
  imshow(k);
% set(handles.axes1);
  title('Input Image');

set(handles.pushbutton1,'enable','off');
set(handles.pushbutton16,'enable','on');
function pushbutton2_Callback(~, ~, handles)
ei=25;
st=35;
%k=10
k=ei*st;
I=handles.YY;
%I = imread('1.jpg');
%h=filter matrx
h = ones(ei,st) / k;
I1 = imfilter(I,h,'symmetric');
IG=rgb2gray(I1);
%Converting to BW
I11 = imadjust(IG,stretchlim(IG),[]);
level = graythresh(I11);
BWJ = imbinarize(I11,level);
dim = size(BWJ);
```

```
IN=ones(dim(1),dim(2));
BW=xor(BWJ,IN);  %inverting
figure,subplot(2,2,2), imshow(BW), title('Black and White');
set(handles.pushbutton1,'enable','off');
set(handles.pushbutton2,'enable','off');
set(handles.pushbutton3,'enable','on');
set(handles.pushbutton4,'enable','off');



% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(~, ~, handles)
% hObject    handle to pushbutton5 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% --- Executes on button press in pushbutton6.
function pushbutton6_Callback(~, ~, handles)
% hObject    handle to pushbutton6 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%I=imread('cancer.bmp');
I=handles.YY;
[y,x,z]=size(I);
myI=double(I);
H=zeros(y,x);
S=H;
HS_I=H;
for i=1:x
   for j=1:y
      HS_I(j,i)=((myI(j,i,1)+myI(j,i,2)+myI(j,i,3))/3);
      S(j,i)=1-3*min(myI(j,i,:))/(myI(j,i,1)+myI(j,i,2)+myI(j,i,3));
```

```
        if ((myI(j,i,1)==myI(j,i,2))&(myI(j,i,2)==myI(j,i,3)))
            Hdegree=0;
        else
            Hdegree=acos(0.5*(2*myI(j,i,1)-myI(j,i,2)-myI(j,i,3))/((myI(j,i,1)-
myI(j,i,2))^2+(myI(j,i,1)-myI(j,i,3))*(myI(j,i,2)-myI(j,i,3)))^0.5);
        end
        if (myI(j,i,2)>=myI(j,i,3))
            H(j,i)=Hdegree;
        else
            H(j,i)=(2*pi-Hdegree);
        end
    end
end

Hth1=0.9*2*pi; Hth2=0.1*2*pi;
Nred=0;
for i=1:x
    for j=1:y
        if ((H(j,i)>=Hth1)||(H(j,i)<=Hth2))
            Nred=Nred+1;
        end
    end
end

Ratio=Nred/(x*y);

if (Ratio>=0.6)
    Red=1;
else
Red=0
end
```

```
HS_I=uint8(HS_I);


figure(1);
imshow(I);
figure(2);
imshow(HS_I);
s=get(handles.edit1,'String');
I =  imread(s);
I=rgb2gray(I);
glcms = graycomatrix(I);
getData()
% Derive Statistics from GLCM
stats = graycoprops(glcms,'Contrast Correlation Energy Homogeneity');
Contrast = stats.Contrast;
Correlation = stats.Correlation;
Energy = stats.Energy;
Homogeneity = stats.Homogeneity;
seg_img=imread('seg.jpg');
Mean = mean2(seg_img);
Standard_Deviation = std2(seg_img);
Entropy = entropy(seg_img);
RMS = mean2(rms(seg_img));
%Skewness = skewness(img)
Variance = mean2(var(double(seg_img)));
a = sum(double(seg_img(:)));
Smoothness = 1-(1/(1+a));
Kurtosis = kurtosis(double(seg_img(:)));
Skewness = skewness(double(seg_img(:)));
% Inverse Difference Movement
m = size(seg_img,1);
n = size(seg_img,2);
in_diff = 0;
```

```
for i = 1:m
    for j = 1:n
        temp = seg_img(i,j)./(1+(i-j).^2);
        in_diff = in_diff+temp;
    end
end
IDM = double(in_diff);

feat = [Contrast,Correlation,Energy,Homogeneity, Mean, Standard_Deviation,
Entropy, RMS, Variance, Smoothness, Kurtosis, Skewness, IDM];
img = imread(s);
 gaborArray = gaborFilterBank(5,8,39,39); % Generates the Gabor filter bank
 featureVector = gaborFeatures(img,gaborArray,4,4);
 disp('Gabor Feature');
 featureVector
 save('Gaborfeature.mat','featureVector');
% GLCM2 = graycomatrix(I,'Offset',[2 0;0 2]);
% features = GLCM_Features1(GLCM2,0)
set(handles.pushbutton14,'enable','on');
set(handles.pushbutton13,'enable','off');
msgbox('Feature Extracted Done');
clear all;
function [out] = GLCM_Features1(glcmin,pairs)
if ((nargin > 2) || (nargin == 0))
   error('Too many or too few input arguments. Enter GLCM and pairs.');
elseif ( (nargin == 2) )
    if ((size(glcmin,1) <= 1) || (size(glcmin,2) <= 1))
       error('The GLCM should be a 2-D or 3-D matrix.');
    elseif ( size(glcmin,1) ~= size(glcmin,2) )
        error('Each GLCM should be square with NumLevels rows and
NumLevels cols');
    end
```

```
elseif (nargin == 1)
   pairs = 0;
   if ((size(glcmin,1) <= 1) || (size(glcmin,2) <= 1))
      error('The GLCM should be a 2-D or 3-D matrix.');
   elseif ( size(glcmin,1) ~= size(glcmin,2) )
      error('Each GLCM should be square with NumLevels rows and NumLevels
cols');
   end
endg

format long e
if (pairs == 1)
  newn = 1;
   for nglcm = 1:2:size(glcmin,3)
      glcm(:,:,newn) = glcmin(:,:,nglcm) + glcmin(:,:,nglcm+1);
      newn = newn + 1;
   end
elseif (pairs == 0)
   glcm = glcmin;
end

size_glcm_1 = size(glcm,1);
size_glcm_2 = size(glcm,2);
size_glcm_3 = size(glcm,3);

out.autoc = zeros(1,size_glcm_3);
out.contr = zeros(1,size_glcm_3);
out.corrm = zeros(1,size_glcm_3);
out.corrp = zeros(1,size_glcm_3);
out.cprom = zeros(1,size_glcm_3);
out.cshad = zeros(1,size_glcm_3);
out.dissi = zeros(1,size_glcm_3);
```

```
out.energ = zeros(1,size_glcm_3);
out.entro = zeros(1,size_glcm_3);
out.homom = zeros(1,size_glcm_3);
out.homop = zeros(1,size_glcm_3);
out.maxpr = zeros(1,size_glcm_3);

out.sosvh = zeros(1,size_glcm_3);
out.savgh = zeros(1,size_glcm_3);
out.svarh = zeros(1,size_glcm_3);
out.senth = zeros(1,size_glcm_3);
out.dvarh = zeros(1,size_glcm_3);

out.denth = zeros(1,size_glcm_3);
out.inf1h = zeros(1,size_glcm_3);
out.inf2h = zeros(1,size_glcm_3);

out.indnc = zeros(1,size_glcm_3);
out.idmnc = zeros(1,size_glcm_3);

glcm_sum = zeros(size_glcm_3,1);
glcm_mean = zeros(size_glcm_3,1);
glcm_var = zeros(size_glcm_3,1);

u_x = zeros(size_glcm_3,1);
u_y = zeros(size_glcm_3,1);
s_x = zeros(size_glcm_3,1);
s_y = zeros(size_glcm_3,1);

p_x = zeros(size_glcm_1,size_glcm_3);
p_y = zeros(size_glcm_2,size_glcm_3);
p_xplusy = zeros((size_glcm_1*2 - 1),size_glcm_3);
p_xminusy = zeros((size_glcm_1),size_glcm_3);
```

```
hxy = zeros(size_glcm_3,1);
hxy1 = zeros(size_glcm_3,1);
hx  = zeros(size_glcm_3,1);
hy = zeros(size_glcm_3,1);
hxy2 = zeros(size_glcm_3,1);

for k = 1:size_glcm_3

   glcm_sum(k) = sum(sum(glcm(:,:,k)));
   glcm(:,:,k) = glcm(:,:,k)./glcm_sum(k);
   glcm_mean(k) = mean2(glcm(:,:,k));
   glcm_var(k)  = (std2(glcm(:,:,k)))^2;

   for i = 1:size_glcm_1

      for j = 1:size_glcm_2

         out.contr(k) = out.contr(k) + (abs(i - j))^2.*glcm(i,j,k);
         out.dissi(k) = out.dissi(k) + (abs(i - j)*glcm(i,j,k));
         out.energ(k) = out.energ(k) + (glcm(i,j,k).^2);
         out.entro(k) = out.entro(k) - (glcm(i,j,k)*log(glcm(i,j,k) + eps));
         out.homom(k) = out.homom(k) + (glcm(i,j,k)/( 1 + abs(i-j) ));
         out.homop(k) = out.homop(k) + (glcm(i,j,k)/( 1 + (i - j)^2));

         out.sosvh(k) = out.sosvh(k) + glcm(i,j,k)*((i - glcm_mean(k))^2);

         out.indnc(k) = out.indnc(k) + (glcm(i,j,k)/( 1 + (abs(i-j)/size_glcm_1) ));
         out.idmnc(k) = out.idmnc(k) + (glcm(i,j,k)/( 1 + ((i -
j)/size_glcm_1)^2));
         u_x(k)       = u_x(k) + (i)*glcm(i,j,k);
         u_y(k)       = u_y(k) + (j)*glcm(i,j,k);
```

```
        end

    end
    out.maxpr(k) = max(max(glcm(:,:,k)));
end

for k = 1:size_glcm_3

    for i = 1:size_glcm_1

        for j = 1:size_glcm_2
            p_x(i,k) = p_x(i,k) + glcm(i,j,k);
            p_y(i,k) = p_y(i,k) + glcm(j,i,k); % taking i for j and j for i
            if (ismember((i + j),2:2*size_glcm_1))
                p_xplusy((i+j)-1,k) = p_xplusy((i+j)-1,k) + glcm(i,j,k);
            end
            if (ismember(abs(i-j),0:(size_glcm_1-1)))
                p_xminusy((abs(i-j))+1,k) = p_xminusy((abs(i-j))+1,k) +...
                    glcm(i,j,k);
            end
        end
    end
end

for k = 1:(size_glcm_3)

    for i = 1:(2*(size_glcm_1)-1)
        out.savgh(k) = out.savgh(k) + (i+1)*p_xplusy(i,k);

        out.senth(k) = out.senth(k) - (p_xplusy(i,k)*log(p_xplusy(i,k) + eps));
    end
```

```
end

for k = 1:(size_glcm_3)

   for i = 1:(2*(size_glcm_1)-1)
      out.svarh(k) = out.svarh(k) + (((i+1) - out.senth(k))^2)*p_xplusy(i,k);

   end

end

for k = 1:size_glcm_3

   for i = 0:(size_glcm_1-1)
      out.denth(k) = out.denth(k) - (p_xminusy(i+1,k)*log(p_xminusy(i+1,k) +
eps));
      out.dvarh(k) = out.dvarh(k) + (i^2)*p_xminusy(i+1,k);
   end
end



for k = 1:size_glcm_3
   hxy(k) = out.entro(k);
   for i = 1:size_glcm_1

      for j = 1:size_glcm_2
         hxy1(k) = hxy1(k) - (glcm(i,j,k)*log(p_x(i,k)*p_y(j,k) + eps));
         hxy2(k) = hxy2(k) - (p_x(i,k)*p_y(j,k)*log(p_x(i,k)*p_y(j,k) + eps));

      end
      hx(k) = hx(k) - (p_x(i,k)*log(p_x(i,k) + eps));
      hy(k) = hy(k) - (p_y(i,k)*log(p_y(i,k) + eps));
```

```
    end
    out.inf1h(k) = ( hxy(k) - hxy1(k) ) / ( max([hx(k),hy(k)]) );
    out.inf2h(k) = ( 1 - exp( -2*( hxy2(k) - hxy(k) ) ) )^0.5;

end

corm = zeros(size_glcm_3,1);
corp = zeros(size_glcm_3,1);

for k = 1:size_glcm_3
    for i = 1:size_glcm_1
        for j = 1:size_glcm_2
            s_x(k)  = s_x(k)  + (((i) - u_x(k))^2)*glcm(i,j,k);
            s_y(k)  = s_y(k)  + (((j) - u_y(k))^2)*glcm(i,j,k);
            corp(k) = corp(k) + ((i)*(j)*glcm(i,j,k));
            corm(k) = corm(k) + (((i) - u_x(k))*((j) - u_y(k))*glcm(i,j,k));
            out.cprom(k) = out.cprom(k) + (((i + j - u_x(k) - u_y(k))^4)*...
                glcm(i,j,k));
            out.cshad(k) = out.cshad(k) + (((i + j - u_x(k) - u_y(k))^3)*...
                glcm(i,j,k));
        end
    end

    s_x(k) = s_x(k) ^ 0.5;
    s_y(k) = s_y(k) ^ 0.5;
    out.autoc(k) = corp(k);
    out.corrp(k) = (corp(k) - u_x(k)*u_y(k))/(s_x(k)*s_y(k));
    out.corrm(k) = corm(k) / (s_x(k)*s_y(k));

end
function gaborArray = gaborFilterBank(u,v,m,n)
```

```
if (nargin ~= 4)    % Check correct number of arguments
   error('There must be four input arguments (Number of scales and orientations
and the 2-D size of the filter)!')
end

%% Create Gabor filters
% Create u*v gabor filters each being an m by n matrix

gaborArray = cell(u,v);
fmax = 0.25;
gama = sqrt(2);
eta = sqrt(2);

for i = 1:u

   fu = fmax/((sqrt(2))^(i-1));
   alpha = fu/gama;
   beta = fu/eta;

   for j = 1:v
      tetav = ((j-1)/v)*pi;
      gFilter = zeros(m,n);

      for x = 1:m
         for y = 1:n
            xprime = (x-((m+1)/2))*cos(tetav)+(y-((n+1)/2))*sin(tetav);
            yprime = -(x-((m+1)/2))*sin(tetav)+(y-((n+1)/2))*cos(tetav);
            gFilter(x,y) = (fu^2/(pi*gama*eta))exp(-
((alpha^2)(xprime^2)+(beta^2)*(yprime^2)))*exp(1i*2*pi*fu*xprime);
         end
      end
      gaborArray{i,j} = gFilter
```

```
    end
end

%% Show Gabor filters (Please comment this section if not needed!)

% Show magnitudes of Gabor filters:
figure('NumberTitle','Off','Name','Magnitudes of Gabor filters');
for i = 1:u
    for j = 1:v
        subplot(u,v,(i-1)*v+j);
        imshow(abs(gaborArray{i,j}),[]);
    end
end

% Show real parts of Gabor filters:
figure('NumberTitle','Off','Name','Real parts of Gabor filters');
for i = 1:u
    for j = 1:v
        subplot(u,v,(i-1)*v+j);
        imshow(real(gaborArray{i,j}),[]);
    end
end

function featureVector = gaborFeatures(img,gaborArray,d1,d2)

if (nargin ~= 4)
    error('Please use the correct number of input arguments!')
end

if size(img,3) == 3
    warning('The input RGB image is converted to grayscale!')
    img = rgb2gray(img);
```

```
end
img = double(img);

[u,v] = size(gaborArray);
gaborResult = cell(u,v);
for i = 1:u
   for j = 1:v
      gaborResult{i,j} = imfilter(img, gaborArray{i,j});
   end
end

featureVector = [];
for i = 1:u
   for j = 1:v
      gaborAbs = abs(gaborResult{i,j});
      gaborAbs = downsample(gaborAbs,d1);
      gaborAbs = downsample(gaborAbs.',d2);
      gaborAbs = gaborAbs(:);

      gaborAbs = (gaborAbs-mean(gaborAbs))/std(gaborAbs,1);

      featureVector = [featureVector; gaborAbs];

   end
end

function getData()
%adds an extraction angle per pixel
offsets = [0 1; -1 1;-1 0;-1 -1;2 2];
jpgImagesDir = fullfile('Dataset/Train', '*.jpg');
total = numel( dir(jpgImagesDir) );
jpg_files = dir(jpgImagesDir);
```

```
jpg_counter = 0;
%total=length(filename);
gambar={total};
data_feat={total};
stats={total};
data_label={total};
label=1;
limit=5;

j=1;
for i=1:total

   %msgbox(jpg_files(j).name)
   s=strcat(num2str(i),'.jpg');
   file=fullfile('Dataset/Train',s);
   %file=fullfile(pathname,filename{i});
   gambar{i}=imread(file);
   gambar{i}=imresize(gambar{i},[600 600]);
   gambar{i}=rgb2gray(gambar{i});
%     gambar{i}=imadjust(gambar{i},stretchlim(gambar{i} ));
%     gambar{i}=imsharpen(gambar{i},'Radius',1,'Amount',0.5);
   glcm=graycomatrix(gambar{i}, 'Offset', offsets, 'Symmetric', true);
   stats{i}=graycoprops(glcm);

   iglcm=1;
   for x=1:5
    data_feat{i,x}=stats{i}.Contrast(iglcm);
    iglcm=iglcm+1;
   end
   iglcm=1;
   for x=6:10
      data_feat{i,x}=stats{i}.Correlation(iglcm);
```

```
        iglcm=iglcm+1;
    end
    iglcm=1;
    for x=12:16
        data_feat{i,x}=stats{i}.Energy(iglcm);
        iglcm=iglcm+1;
    end
        iglcm=1;
    for x=18:22
        data_feat{i,x}=stats{i}.Homogeneity(iglcm);
        iglcm=iglcm+1;
    end
        data_feat{i,24}=mean2(gambar{i});
        data_feat{i,25}=std2(gambar{i});
        data_feat{i,26}=entropy(gambar{i});
        data_feat{i,27}= mean2(var(double(gambar{i}))); %average image
variance
        data_feat{i,28}=kurtosis(double(gambar{i}(:)));
        data_feat{i,29}=skewness(double(gambar{i}(:)));

        %labeling
        if i>limit
            label=label+1;
            data_label{i}=label;
            limit=limit+5;
        else
            data_label{1,i}=label;
        end
end
% data is converted to the appropriate data type so that svm is not confused
data_feat=cell2mat(data_feat);
disp(data_feat);
```

```
data_label=cell2mat(data_label);
save('data_1.mat','data_feat','data_label');
% --- Executes on button press in pushbutton14.
function pushbutton14_Callback(hObject, ~, handles)
% hObject    handle to pushbutton14 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
s=get(handles.edit1,'String');
test=imread(s);

test=imresize(test,[600 600]);
test=rgb2gray(test);
offsets = [0 1; -1 1;-1 0;-1 -1;2 2];

 glcm=graycomatrix(test, 'Offset', offsets, 'Symmetric', true);

stats=graycoprops(glcm);
data_glcm=struct2array(stats);
iglcm=1;
glcm_contrast={5};
glcm_correlation={5};
glcm_energy={5};
glcm_homogeneity={5};
   for x=1:5
     glcm_contrast{x}=data_glcm(iglcm);
     iglcm=iglcm+1;
   end
   for x=1:5
      glcm_correlation{x}=data_glcm(iglcm);
      iglcm=iglcm+1;
   end
   for x=1:5
```

```
        glcm_energy{x}=data_glcm(iglcm);
        iglcm=iglcm+1;
    end
    for x=1:5
        glcm_homogeneity{x}=data_glcm(iglcm);
        iglcm=iglcm+1;
    end
rata2=mean2(test);
std_deviation=std2(test);
glcm_entropy=entropy(test);
rata2_variance= mean2(var(double(test)));
glcm_kurtosis=kurtosis(double(test(:)));
glcm_skewness=skewness(double(test(:)));
buat_train=[glcm_contrast(1:5),glcm_correlation(1:5),glcm_energy(1:5),glcm_
homogeneity(1:5),rata2,std_deviation,glcm_entropy,rata2_variance,glcm_kurto
sis,glcm_skewness];

test_data=cell2mat(buat_train);

disp('GLCM Feature')
disp('Contrast(1) Correlation(2) Energy(3) Homogeneity(4) Mean(5)
Standard_Deviation(6) Entropy(7) RMS(8) Variance(9) smoothness(10)
Kurtosis(11) Skewness(12) IDM(13)')

input_Feature=test_data

load('data_1.mat');
%load('Test_data.mat');
%disp('Gabor Feature');
load('Gaborfeature.mat');
%Input_Feat=test_data
```

```
result = multisvm(data_feat,data_label,test_data);

%result1 = multisvm(data_feat,data_label,data_feat1);
%[Cmat,Accuracy]=
confusion_matrix(result1,data_label,{'Desert','Forest','Mountain','Residential','R
iver'});
%[c_matrixp,Result]= confusion.getMatrix(data_label,result1);
%C = confusionmat(data_label,result1)

if result == 1
   A1 = 'actinic keratosis';
   set(handles.edit2,'string',A1);
   helpdlg('actinic keratosis');
   disp('actinic keratosis');
elseif result == 2
   A2 = 'Basel cell carcinoma';
   set(handles.edit2,'string',A2);
   helpdlg('Basel cell carcinoma');
   disp('Basel cell carcinoma');
elseif result == 3
   A3 = 'cherry nevus';
   set(handles.edit2,'string',A3);
   helpdlg('cherry nevus');
   disp('cherry nevus');
elseif result == 4
   A4 = 'dermatofibroma';
   set(handles.edit2,'string',A4);
   helpdlg('dermatofibroma');
   disp('dermatofibroma');
elseif result == 5
   A5 = 'Melanocytic nevus';
   set(handles.edit2,'string',A5);
```

```
            helpdlg('Melanocytic nevus');
            disp('Melanocytic nevus');
            elseif result == 6
            A5 = 'Melanoma';
            set(handles.edit2,'string',A5);
            helpdlg('Melanoma');
            disp('Melanoma');
end
        str=get(handles.edit1,'String');
        I1 = imread(str);
        I4 = imadjust(I1,stretchlim(I1));
        I5 = imresize(I4,[300,400]);
        figure
        imshow(I5);title(' Processed Image ');
        set(handles.pushbutton1,'enable','off');
        set(handles.pushbutton16,'enable','off');
        set(handles.pushbutton12,'enable','on');
% --- Executes on button press in pushbutton17.
function pushbutton17_Callback(~, ~, ~)
```