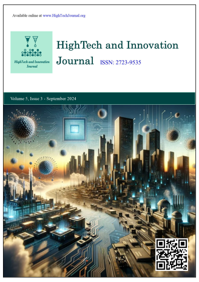
Available online at www.HighTechJournal.org

HighTech and Innovation Journal

ISSN: 2723-9535

Vol. 5, No. 3, September, 2024



Early Identification of Skin Cancer Using Region Growing Technique and a Deep Learning Algorithm

Suhendro Y. Irianto [1*], Rian Yunandar [1], M. S. Hasibuan [1], Deshinta Arrova Dewi [2], Nualyai Pitsachart [3]

[1] *Department of Informatics, Institute Informatics and Business Darmajaya, Lampung 35144, Indonesia.*

[2] *Faculty of Data Science and Information Technology, INTI International University, Nilai, Malaysia.*

[3] *School of Nursing, Shinawatra University, 99 Moo 10, Bangtoey, Samkhok, Pathum Thani 12160, Thailand.*

Received 10 February 2024; Revised 19 August 2024; Accepted 27 August 2024; Published 01 September 2024

Abstract

Skin cancer, comprising both melanoma and non-melanoma forms, is a significant public health concern, constituting approximately 5.9% to 7.8% of annual cancer diagnoses. In Indonesia, the predominant form of the disease is basal cell carcinoma (65.5%), followed by squamous cell carcinoma (23%), and malignant melanoma (7.9%). Several studies have shown that early detection of its melanoma form is essential due to the heightened mortality risk. Therefore, this study aimed to assess the efficacy of the Region Growing + LSTM algorithm in improving detection accuracy compared to LSTM. The novelty of the study lay in addressing the inefficiencies of manual dermoscopy image examination and introducing a novel combination of Region Growing segmentation and Deep Learning LSTM for enhanced detection precision. The results showed that the proposed model could identify segmented areas before classification and achieved 96.62% accuracy, outperforming LSTM's 84%. However, LSTM exhibited shorter training and prediction times (39.3 seconds and 3.2 seconds, respectively) compared to Region Growing + LSTM (17 minutes and 2 seconds for training, 3 minutes and 49 seconds for prediction). Although Region Growing + LSTM offered superior accuracy, it required more time than LSTM, showing potential trade-offs between accuracy and efficiency in skin cancer image detection.

Keywords: LSTM, Region Growing; Corn Leaf Disease; Public Health; Health Risk.

1. Introduction

Skin cancer is a dermatologic disease caused by injury to skin cells, leading to their transition from normal to malignant states. This transition is characterized by DNA damage, which initiates uncontrolled and abnormal cell division. Although the symptoms of skin cancer are often visible without specialized tools, achieving a prompt and accurate diagnosis relies on the expertise of dermatologists regarding its complexities and treatment. To prevent the occurrence of the disease, self-examinations every six months, coupled with annual visits to a dermatologist, are recommended [1, 2]. Over the past decade, rapid advancements in image processing and computer vision have occurred. In addition, Wang et al. [1] have applied these technologies to detect skin cancer lesions, achieving a segmentation accuracy of 95% using the region-growing technique. Another study by Dildar et al. [3] using Deep Learning RNN achieved a detection accuracy of 93%. Building on these findings and addressing skin cancer issues in Indonesia, this

[*] Corresponding author: suhendro@darmajaya.ac.id

http://dx.doi.org/10.28991/HIJ-2024-05-03-07

study introduces the integration of image processing and Deep Learning to enable early skin cancer detection. The chosen methods, namely Region Growing segmentation and RNN-LSTM are widely known for their quick and accurate detection.

According to previous studies, skin cancer is a disease that attacks the outermost organ of the body, namely the skin, which is composed of billions of cells. Several studies have shown that unhealthy lifestyles, frequent exposure to ultraviolet rays, toxins, and specific genetic factors can cause the abnormal growth of these cells, leading to the development of cancerous forms [4]. In addition, skin cancer can be classified into two types based on the level of danger, including benign and malignant. Benign, commonly referred to as tumors, is rarely life-threatening, can be removed, does not recur, and does not spread to other parts of the body (e.g., moles). Meanwhile, malignant refers to aggressive cancer cells, potentially fatal, can regrow after removal, and often spread to other parts of the body [5]. Common examples of malignant skin cancer include 1) melanoma, which attacks through pigment skin cells, 2) basal cell carcinoma, which targets basal skin cells, often due to UV radiation and affects the face, and 3) squamous cell carcinoma, which targets squamous skin cells, often affecting dark-skinned individuals and areas not exposed to sunlight, such as the legs [6].

In line with previous studies, an image serves as a representation of a two-dimensional object in the form of a collection of points or pixels with colors. Several reports have also shown that digital image processing is a technique for processing both still and moving images to enhance their quality for easy understanding by humans or computer systems [7]. This technique is a crucial component in various industrial and commercial applications and is a major component in the field of computer vision [8]. According to Arnal Barbedo [9], there are four types of digital images, including 1) Binary Image, which is the simplest type with only two values, namely black and white; 2) Grayscale Image, which is considered a monochrome image with 8 bits or pixels representing brightness levels from 0 to 255; 30 indexed image that comprises an array alongside a matrix for the color map; and 4) RGB Image, which depicts image through the utilization of three-color components, namely red, green, and blue.

To enhance the information obtained from the image, several studies have proposed the use of pre-processing, which comprises reducing unwanted distortions or strengthening features for further processing. Examples of pre-processing techniques include dynamic resizing and shaping of images, noise filtration, image conversion, and image enhancement [8]. Meanwhile, median filtering is an image processing technique that improves quality by reducing salt-and-pepper noise. The mechanism of action comprises replacing the gray level value of each pixel with the median gray level value of surrounding pixels. Before applying median filtering, zeros are added around the edges of the image to ensure proper filtering [9]. Another common image processing technique is Contrast Limited Adaptive Histogram Equalization (CLAHE), which is a method of Histogram Equalization [10]:

$$(k) = (L - 1) \times \sum n(j)/n \tag{1}$$

For values of "k" within the interval of 0 to $L - 1$, "L" denotes the total number of gray levels in the image, "j" ranges from 0 to "k". In addition, "$n(j)$" indicates the count of pixels with a gray level of "j," and "n" corresponds to the overall number of pixels. The ratio "$n(j)/n$" serves as the Cumulative Distribution Function (CDF) for the pixel value "k". Several reports have shown that CLAHE can enhance contrast by imposing upper limits on pixel intensity values to prevent excessive contrast in uniform regions. CLAHE also restricts improvement by truncating the histogram at pre-established thresholds before computing the CDF, and these predetermined thresholds are denoted as the clip limit. The portion of the histogram above the clip limit is truncated and uniformly distributed across the entire histogram [11].

According to Wang et al. [1], who compared the technique with various modern methods, it excelled in identifying skin cancer compared to others. The Convolutional Neural Networks (CNN) approach achieved an average accuracy of 94.206% in diagnosing skin cancer, surpassing the effectiveness of alternative methods. During data validation, the 5-fold ROC curve and error curve have been reported to represent its superiority and resilience. Whilst Tang et al. [12] also reported an average accuracy of 94.206% for diagnosis, surpassing the performance of Optimization Algorithm-Based Exception Neural Network methods. The 5-fold ROC curve and error curve during data validation revealed its superiority and robustness. Another study, centered on detecting melanoma skin cancer and its preceding stages (common nevus and atypical nevus), introduced methods that combined color, texture, and shape features to extract distinctive attributes from images. By employing CNN and Support Vector Machine (SVM) algorithms, the study identified the type of skin cancer affecting the patient, achieving accuracies of 92% and 95%, respectively [13]. Rajendran & Shanmugam [10] introduced Automated Skin Cancer Detection and Classification using 2.4 Cat Swarm Optimization with a Deep Learning Model and achieved significant outputs with an accuracy of 92.22%.

As indicated in Chakkarapani & Poornapushpakala [14] revealed that segmentation was accomplished using the Double U-Net method on skin lesions. Subsequently, data augmentation was applied, and the detection process was conducted using DMN, with the network finely tuned using a designed CDO. In addition, CDO integrated chronological concepts with the Dingo Optimizer (DOX). This method demonstrated improved results, with superior sensitivity at

0.959, F-measure at 0.908, accuracy at 0.923, and specificity at 0.837. In addition, the findings obtained by Likhar & Ridhorkar [15] showed that the proposed ensemble model, with a specific focus on VGG-16, achieved an impressive average accuracy of 92%. The suggested VGG-16 model had superior performance when compared to VGG-19 and Inception V3 across various critical metrics. The method is significantly better in terms of sensitivity, accuracy, F-Score, specificity, false-positive rate, and precision, making it a promising option for precise and reliable skin cancer detection. The study emphasized the potential of ensemble models in advancing early cancer diagnosis and showed the significant role played by the VGG-16 architecture. These findings offer valuable insights for both the medical community and deep learning practitioners, with the overarching objective of enhancing skin cancer detection methods and ultimately saving lives.

Studies carried out with MobileNetV2 and customized CNN methods were reported to achieve an accuracy of 85% and 95%, respectively. A web application, built using the Python framework, has also been developed to offer a graphical user interface featuring the best-trained model. Through this interface, users can input patient details and upload lesion image. The uploaded image is then classified using the appropriate trained model, predicting whether it is cancerous or non-cancerous. In addition, the web application provides the percentage of cancer affected. Based on the results, the comparison between the two techniques indicates that the customized CNN offers higher accuracy in detecting melanoma [16]. Therefore, this study aimed to address issues related to the swift and precise detection of skin cancer, as well as explore the utilization of Region Growing and RNN-LSTM methods. The primary goals are to 1) develop a system for detecting skin cancer by employing Region Growing and RNN-LSTM techniques, 2) assess the effectiveness of these methods in skin cancer detection process, and 3) assist doctors in diagnosing skin cancer and expediting the treatment process.

2. Related Works

2.1. Image Segmentation

Image segmentation is the first process carried out to enable analysis and processing by a computer through the classification of pixels from an image. The goal of image segmentation is to divide an image into several parts to obtain regions that share similarities based on predefined variables. According to Sharma & Suji [17], there are generally four categories, including 1) Thresholding segmentation, which segments based on Gray level or the intensity value of pixels. The challenge in this method is determining the appropriate Gray level to divide each pixel into two categories, namely dark and bright, 2) Edge Detection, which divides by detecting the edges and grouping them into several parts to represent boundaries between objects, 3) Region Extraction, which divides the entire image into several small parts based on predetermined criteria, usually using similarity in intensity, color, and texture as criteria, and 4) Clustering, which segments the image by classifying patterns or objects into several clusters with similar characteristics [18, 19].

2.2. Harris Corner Detection

Harris Corner Detection is a pre-processing technique to obtain the edges of an image. According to previous studies [20-23], the detection of edges could be carried out by computing using Equation 1:

$$har = \mathrm{de\,t}[\mu(\sigma_I, \sigma_D)] - \alpha\left[\mathrm{trac\,e}\big(\mu(\sigma_I, \sigma_D)\big)^2\right] = g(I_x^2)g\big(I_y^2\big) - \big[g\big(I_xI_y\big)\big]^2 - \alpha\big[g(I_x^2) + g\big(I_y^2\big)\big]^2 \tag{2}$$

$$\mu(\sigma_I, \sigma_D) = \mathrm{g}(\sigma_I) \times \begin{bmatrix} I_x^2(\sigma_D) & I_xI_y(\sigma_D) \\ I_xI_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix} \tag{3}$$

In Equations 2 and 3, derivatives I_x and I_y are computed from image I. I_x^2 and I_y^2 x^2 are the products of the derivatives I_x and I_y, while I_x and I_y are the products of the derivatives of image I_x and $I_{y.}$ $g(\sigma_I)$ is the Gaussian filter. In Equation 1, when the value of "har" is greater than zero, it is considered as a corner. Meanwhile, when its value is less than zero, it is considered an edge.

2.3. Region Growing Segmentation

According to Han et al. [24], the region-growing algorithm commences with an initial segmentation that is not fully defined and endeavors to combine unlabeled pixels into one of the existing regions. The initial region is typically known as the "seed region." The determination of whether a pixel should become a part of a region depends on various fitness functions that indicate the likeness between the region and the pixel under consideration. As proposed in Abualigah et al. [19] and Ram & Padmavathi [20], the sequence in which pixels are processed is established using a global priority queue, which organizes candidate pixels according to their fitness values. This image segmentation algorithm allocates pixels to uniform regions, enhancing accuracy beyond what can be achieved through individual classification. Region growing is a categorization method applied in image segmentation algorithms, usually constructed through an agglomeration process that combines pixels into a region when proximate and possesses similar properties.

2.4. Deep Learning-Based Image Classification

Recurrent Neural Networks (RNN) belongs to the category of artificial neural networks where the connections between nodes create a directed graph capable of handling variable sequences. RNN has evolved into different forms, such as Gated Recurrent Units (GRU) and Long Short-Term Memory Networks (LSTM) to improve the efficiency of the original algorithm. The RNN architecture is composed of an input layer, one or more hidden layers, and an output layer [25-29]. In addition, it has a chain-like structure with recurring modules, functioning as memory to store important information from previous steps. RNN also employs a feedback loop, enabling the neural network to process input sequences. Consequently, the output from the previous step is passed to the network, influencing the subsequent step. Figure 1 illustrates a simplified depiction of how the RNN algorithm operates, featuring a single input unit, one output unit, and recurring hidden units, which can develop into a more intricate network. In this representation, "x_t" represents the input at time step "t", and h_t indicates the output at time step "t" [30].

Figure 1 shows the inner workings of Recurrent Neural Networks (RNNs), offering a concise glimpse into their remarkable ability to comprehend sequential data. Where x_t is the input at time step t and h_t is the hidden state (internal memory) at time step t. Previous studies have successfully implemented RNN in image processing, such as for breast cancer disease detection [31], RNN-LSTM for cervical cancer disease identification [31], and Dense RNN for heart image segmentation [24]. In addition, Wang & Zhang [32] showed that in the common operation of variations of RNN methods, such as LSTM, every cell obtains input from the preceding cell and transmits it to the subsequent cell, as shown in Figure 2.

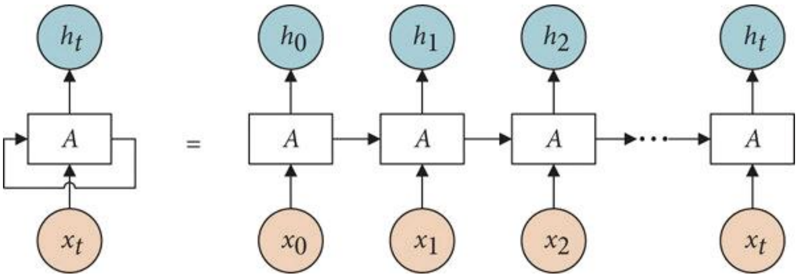


Figure 1. Overview of How RNN Works

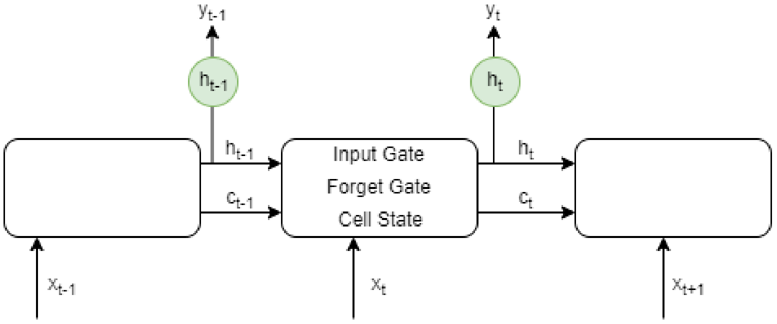


Figure 2. Overview of How LSTM Works

Forget Gate: At each time step, the forget gate determines how much of the previous cell state1 C_{t-1} should be retained or forgotten. The input is the concatenation of the current input x_t and the previous hidden state $h-1$, h_{t-1}. Input Gate: The input gate determines how much of the current input x_t should be added to the cell state $C-1$, C_{t-1}. In addition, the input is the concatenation of x_t and h_{t-1}. Update of Cell State: Based on the values of the forget gate and the input gate as well as a candidate cell state $C\sim t$ calculated using the tanh activation function, the new cell state C_t is computed. This update process allows the LSTM to selectively retain or forget information. According to Roy et al. [29], LSTM has components known as memory cells and gate inputs. A total of four types of gate inputs have been reported, including forget gate, input gate, cell gate, and output gate. The activation function used in these gates is sigmoid, which produces values between 0 and 1. In the forget gate, each input data is processed to decide whether the data should be deleted or stored in memory. The formula used for the forget gate is presented Equation 4:

$$f_t = \sigma\left(W_f.[h_{t-1}, x_t] + b_f\right) \tag{4}$$

Where ft denotes the value of the forget gate at time step t. The forget gate is responsible for determining how much of the information from the previous cell state h_{t-1} should be discarded or forgotten based on the current input xt. $\sigma\sigma$ represents the sigmoid activation function, commonly used in LSTM networks to squish the input values between 0 and 1, thereby providing a probability-like output. W_f, matrix represents the weights associated with the forget gate and is often used to linearly transform the concatenation of the previous hidden state h_{t-1} and the current input x_t. $[h_{t-1}, x_t]$ signifies the concatenation of the previous hidden state h_{t-1} and the current input x_t. These two vectors are concatenated into a single vector before being passed through the forget gate. b_f, represents the bias term associated with the forget gate.

After the forget gate, the next step is the input gate, which consists of two stages. The first stage is determining which values are to be updated using the sigmoid activation function. The second stage comprises the tanh activation function, producing a new vector value to be stored in the memory cell. The formulas used for the input gate are:

$$i_t = \sigma(W_i . [h_{t-1}, x_t] + b_i) \tag{5}$$

$$\hat{c}_t = tanh(W_c . [h_{t-1}, x_t] + b_c) \tag{6}$$

It denotes the value of the input gate at time step t. The input gate controls how much of the new input information x_t should be incorporated into the current cell state $c\ t$. In addition, σ represents the sigmoid activation function, which squashes the input values between 0 and 1. W_i matrix represents the weights associated with the input gate and it has a similar application with the forget gate. The matrix is often used to linearly transform the concatenation of the previous hidden state h_{t-1} and the current input x_t. $[h_{t-1}, x_t]$ signifies the concatenation of the previous hidden state h_{t-1} and the current input x_t, forming a single vector. bibi represents the bias term associated with the input gate, while $\hat{c}_t$ denotes the candidate cell state at time step t and represents the new information that could be added to the cell state C_{t-1} at the current time step. Symbol tanh indicates the hyperbolic tangent activation function, which squashes the input values between -1 and 1, thereby introducing non-linearity to the computation. W_c matrix represents the weights associated with the candidate cell state calculation and is used to linearly transform the concatenation of the previous hidden state h_{t-1} and the current input x_t. Notation $[h_{t-1}, x_t]$ shows the concatenation of the previous hidden state h_{t-1} and the current input x_t, forming a single vector. b_c represents the bias term associated with the candidate cell state calculation.

Following the input gate phase, the subsequent stage is the cell gate. During this stage, the previous memory cell's value is substituted with a new value formed by combining the values from both the forget gate and the input gate. The cell gate is computed using the following Equation:

$$c_t = f_t \times c_{t-1} + i_t \times \hat{c}_t \tag{7}$$

In the final stage, the output gate determines which memory cell values should be released. This is achieved by applying the sigmoid activation function, and then the selected value is integrated into the memory cell using the tanh activation function. Subsequently, the product of these two values is computed to produce the output value. The equations used for the output gate are as follows:

$$o_t = \sigma(W_o . [h_{t-1}, x_t] + b_o) \tag{8}$$

$$h_t = o_t \tan\mathrm{h}(c_t) \tag{9}$$

2.5. Segmentation Evaluation

Measuring the quality of segmentation algorithms is an essential step in this study. The evaluation of segmentation algorithms is often carried out using several predefined metrics, such as Precision (P), Recall (R), and Accuracy (ACC). These metrics are selected due to their ability to provide relevant insights into the algorithm's capacity to accurately identify foreground pixels [33, 34]. In the evaluation process, the segmentation algorithm is compared with a known ground truth mask. The Precision metric measures how well the algorithm can correctly identify foreground pixels, while the Recall metric shows the algorithm's ability to recognize actual foreground pixels. In addition, the Accuracy metric provides an overall view of the algorithm's correctness in separating foreground and background pixels. To calculate Accuracy, Precision, and Recall, the following equations are used:

$$Precision = \frac{TP}{TP+FP} \tag{10}$$

$$Recall = \frac{TP}{TP+FN} \tag{11}$$

$$Accuracy = \frac{TP + TN}{TP+TN+FP+FN} \tag{12}$$

In this context, TP (True Positive) is the count of accurately detected foreground pixels, TN (True Negative) corresponds to the number of background pixels correctly identified as such, FP (False Positive) indicates the number of background pixels erroneously categorized as foreground, and FN (False Negative) represents the number of foreground pixels that were mistakenly recognized as background.

2.6. Deep Learning Method Evaluation

According to Behura [35], deep learning models can be assessed for their effectiveness using various metrics, such as accuracy, precision, and recall. These metrics offer an evaluation of the model's capacity to make correct classifications in alignment with real data and labels. The equations to compute these evaluation metrics are illustrated in Equations 13 to 15. Accuracy quantifies the proportion of correctly identified samples by the model in comparison to the total number of samples.

$$Accuracy = (TP + TN) / (TP + FN + FP + TN) \times 100\% \tag{13}$$

Precision is described as the ratio of true positive samples that the model correctly identifies as positive, relative to the overall number of positive samples that the model predicts.

$$Precision = TP / (TP + FP) \times 100\% \tag{14}$$

Recall is defined as the proportion of the total number of true positive samples correctly identified as positive by the model compared to the total number of actual positive samples.

$$Recall = TP / (TP + FN) \times 100\% \tag{15}$$

In the domain of deep learning models, the metrics for evaluation heavily rely on the values of TP (True Positive), TN (True Negative), FP (False Positive), and FN (False Negative). TP represents the number of positive samples that are accurately predicted, TN corresponds to the count of negative samples correctly predicted, FP indicates the number of negative samples that are incorrectly classified as positive, and FN signifies the count of positive samples that are inaccurately classified as negative. The outcomes of this assessment serve as a gauge of the deep learning model's capabilities. Higher values for accuracy, precision, and recall reflect the model's superior performance in accurately and consistently executing classification tasks [36, 37].

2.7. Imbalanced Dataset

Several studies have used public datasets that often have imbalances in the number of data in each directory or class. This can cause issues in classification, as classes with more data appear more frequently during predictions compared to others with fewer data. According to Lu et al. [38] and Zhang et al. [39], there are two ways to handle imbalanced datasets, namely oversampling and under sampling. Oversampling comprises adding or duplicating samples in the class with fewer data than the class with the most data. The goal is to ensure that the class initially having fewer data can have an equivalent amount of data to the other. Meanwhile, under sampling comprises removing data from the class with the most data to ensure that the data quantity in the class is balanced with the class with the least data. The difference between oversampling and under sampling is presented in Figure 3.

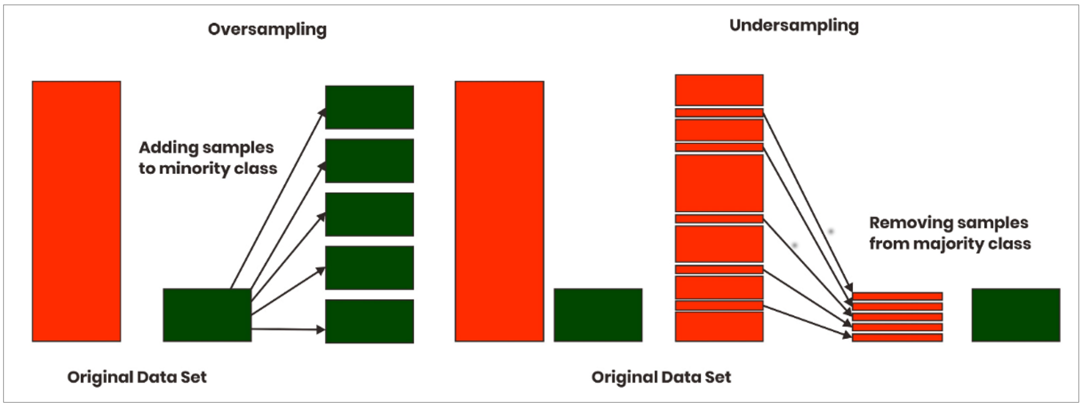

Oversampling
Undersampling
Adding samples to minority class
Removing samples from majority class
Original Data Set
Original Data Set

Figure 3. Overview of Oversampling and Under sampling k Mechanisms

In this subsection, previous studies were discussed with the aim of addressing the gaps in existing literature. Shilpa Kamdi [40] used Region Growing to detect skin lesions. The study employed three pre-processing methods before segmentation with Region Growing. The combination of grayscale image pre-processing and Region Growing yielded a high accuracy of 95% for skin lesion detection. In addition, the study suggested the application of the same technique to other skin diseases. Implementation from Ali et al. [41] that can be adopted includes determining the seed based on the Harris Corner Detection algorithm, where the seed is selected from the edge with the highest intensity value to be used as the initial seed for the Region Growing segmentation process.

In the study by Gulzar & Khan [42], segmentation of skin cancer lesions was performed using the GrabCut method combined with pre-processing techniques, such as Corner Borders Removal, Hairs Removal, and Image Enhancement. The accuracy achieved was 77% on the PH2 dataset. The study recommended trying other methods as GrabCut resulted in over-segmentation in the segmented image. For skin cancer detection, Wang & Zhang [32] utilized the RNN method for classification and K-mean clustering for segmentation, with a high accuracy of 93%. Pre-processing techniques included median filtering before clustering data with K-mean clustering, but a significant limitation was the small dataset of only 83 images.

A study was conducted by Vishnu Priya et al. [43] to detect cervical cancer based on colposcopy images using the RNN-LSTM method for classification. Before training the data, image pre-processing with Histogram Equalization and median filtering was performed for better image quality, with an accuracy of 66%. Due to the relatively low accuracy, the study suggested the use of alternative methods for both classification and data pre-processing. The application of appropriate image pre-processing techniques could affect the accuracy of skin cancer detection, as reported by Imtiaz et al. [8]. CLAHE pre-processing was applied before training the data, leading to an accuracy of 87.99%. Implementing CLAHE improved image contrast, leading to good accuracy with the CNN classification method. Under sampling was performed in this study to balance data from each class in the dataset.

The combination of CLAHE and MSRCR (Multiscale Retinex with Color Restoration) image pre-processing techniques with VGG-16 for predicting skin cancer achieved high accuracy in Hayati et al. [44]. The study reported accuracy rates of 92.6% for CLAHE + VGG-16 and 91.9% for MSRCR + VGG-16. Imtiaz et al. [45] introduced a two-step approach for segmenting skin lesions in images. The first step comprised detecting Harris corners, which were salient points in the image, and the second step used Region Growing to separate the lesion from the rest of the image. The results showed a high segmentation accuracy of 95%, indicating the effectiveness of the method for accurately delineating skin lesions in images. While the paper showed the method's strength in achieving high accuracy, it did not specify its weaknesses or limitations. A more comprehensive analysis of potential drawbacks could provide a more balanced assessment of its practical applicability in medical imaging scenarios.

It is referred to Gowthami & Sneha [46], who introduced a technique for identifying melanoma by employing Recurrent Neural Networks (RNNs). The paper took advantage of the suitability of RNNs for processing sequential data, making them particularly beneficial for tasks comprising data with temporal or sequential characteristics. A significant accomplishment was reported, achieving a classification accuracy of 93%. This high level of accuracy showed the efficacy of the RNN-based approach in differentiating between melanoma and non-melanoma cases, which was of significant importance. In addition, this was because it could have a profound impact on patient outcomes through early and precise melanoma detection. The authors also maintained transparency regarding the limited number of classes and the size of the training dataset, providing readers with a clear understanding of the study's scope and context. Acknowledging these limitations was vital for a fair assessment of the method's applicability. However, a primary limitation of the study was the restricted size of the training dataset. Deep learning models, such as RNNs, heavily relied on having a substantial and diverse dataset to perform robustly. A limited dataset often led to overfitting and could constrain the model's capacity to generalize to a wider range of real-world melanoma cases. To validate the model's performance in practical scenarios, further studies with a more extensive and diverse dataset were advantageous.

As mentioned by Asyhar et al. [47], who focused on using LSTM (Long Short-Term Memory) algorithms for the classification of cervical cancer with colposcopy data. The study used a combination of image enhancement techniques, including Histogram Equalization and Median Filter, followed by an RNN-LSTM (Recurrent Neural Network—Long Short-Term Memory) algorithm. This approach combined image processing and deep learning techniques, with an accuracy of 66%. Therefore, the method achieved 66% accuracy in distinguishing between different cervical cancer classes using colposcopy data. The strength of the study lies in its application of image enhancement techniques, such as Histogram Equalization and Median Filter. These techniques could help improve the quality and clarity of the colposcopy image, which was important for accurate classification. However, a significant weakness was the relatively low classification accuracy of 66%. The accuracy level was not sufficient for reliable cervical cancer classification in a medical context. The paper did not also contain any efforts or strategies to improve this accuracy, which was a notable limitation. In summary, the studies presented an approach for cervical cancer classification using colposcopy data, including image enhancement techniques and LSTM-based deep learning. While the application of enhancement techniques was a strength, the primary weakness was the relatively low classification accuracy of 66%, which limited the practical utility of the method in a medical setting. Further studies and efforts to improve accuracy were beneficial for enhancing the effectiveness of the approach.

It stated by Jain et al. [48] that focused on using a combination of Contrast Limited Adaptive Histogram Equalization (CLAHE), Convolutional Neural Networks (CNN), and under-sampling techniques for the recognition of skin cancer. The study used a combination of techniques, including CLAHE for image enhancement, CNN for deep learning-based classification, and under-sampling for addressing class imbalance issues in the dataset. This holistic approach integrated image processing and deep learning methods for skin cancer recognition, with a classification accuracy of 87%. This indicated that their approach achieved an 87% accuracy in correctly classifying different types of skin cancer. The strengths of the study lied in its utilization of multiple techniques, including CLAHE for image enhancement, CNN for deep learning, and under-sampling to address class imbalance. These techniques collectively contributed to the success of the approach, leading to an 87% accuracy. However, it did not specify any weaknesses or limitations, which could make it challenging to understand potential areas for improvement or further studies. In summary, the study presented an approach for skin cancer recognition that combined CLAHE for image enhancement, CNN for deep learning, and under-sampling to address class imbalance issues. The primary strength was the high classification accuracy of 87%, indicating the method's effectiveness. However, the absence of specified weaknesses or limitations could limit a comprehensive assessment of the method's applicability and areas for potential refinement.

As confirmed by Ray et al. [49], which analyzed the impact of Image Enhancement Methods on Early Skin Cancer Detection. The study used VGG-16, CLAHE, and MSRCR techniques, with a high classification accuracy of 92.6%. Significant strengths of the study included the utilization of state-of-the-art methods, such as VGG-16, CLAHE, and MSRCR, which contributed to the high classification accuracy. One limitation was its focus on a binary classification task with only two classes, potentially limiting the generalizability of the findings to more diverse skin conditions.

3. Material and Methods

The software and hardware used in this study for data collection, processing, analysis, and presentation include Windows 10 Home Operating System (64-bit), Python Programming Language, and Visual Studio Code. Hardware. Intel I5 Gen 6 Processor. 8 GB RAM, and 200GB SSD. Dermoscopy images were the most common method used by experts for initiating skin disease analysis. The materials used were from the publicly available PH2 Dataset, consisting of dermoscopy images of skin cancer categorized into three types, namely Normal, Malignant, and Benign. The dataset comprised image files in .bmp format and could be obtained from the link: *https://www.fc.up.pt/addi/ph2%20database.html*. In addition, it also consisted of 200 dermoscopy images and 200 corresponding Ground Truth data, categorized as 80 normal skin, 40 melanoma, and 80 benign cases.

The PH2 dataset used in this study only contained 200 images, while the datasets used in HAM10000 had 10,000 data, and ISIC 2019 comprised 20,771 data. The substantial difference in the dataset size allowed other studies to have more data for LSTM to learn from, reducing the risk of overfitting even with large batch processes. Moreover, the HAM10000 dataset had only 2 classes (benign and malignant), making classification easier. The ISIC 2019 dataset had 8 classes, but none represented healthy skin or minimal pigment nevi (moles) compared to those in the PH2 dataset.

3.1. Evaluation Method

The evaluation results of the methods were compared between LSTM and Region Growing + LSTM in terms of time, accuracy, and loss during both training and testing phases. In this study, integrated LSTM and Region Growing method [24, 26, 30, 50] were utilized. The method employed was evaluated by calculating accuracy, precision, and recall.

3.2. The Proposed Methods Stages

This section explained the stages conducted in the study. Initially, from skin cancer image data, the image extraction process was performed to obtain color or shape features. Subsequently, the extracted features were subjected to data pre-processing, including resizing and conversion of the RGB color image to grayscale. Image enhancement was then carried out using median filters, CLAHE, or both. Edge detection was performed using Harris Corner Detection, where the edges with the highest pixel intensity were designated as the initial seed for the Region Growing segmentation process. The segmented image was utilized for training data with RNN-LSTM. The final step comprised evaluating the method to calculate the accuracy of the algorithm. A clearer overview of the proposed methods in this study is presented in Figure 4.

Image quality enhancement was conducted before segmentation and classification using the median filter and CLAHE methods. Experiments were carried out using each method individually and their combination to achieve the best results. In addition, median filtering was applied to reduce noise in the image, while CLAHE was used to enhance the contrast values, making the sample sharper. Image segmentation was performed using the Region Growing method. The Region Growing process began by selecting an initial pixel that belonged to the desired region. This pixel could be selected manually or determined computationally. Subsequently, neighbors of this initial pixel were analyzed to determine their homogeneity or similarity with the initial pixel. The segmentation process was executed automatically, but the initial seed/pixel must be manually determined. According to Leiter et al. [5], the initial seed could be obtained automatically based on pixel intensity values from the image edges obtained using the Harris Corner Detection method. Therefore, edge detection using the Harris Corner Detection method was necessary before the segmentation process.

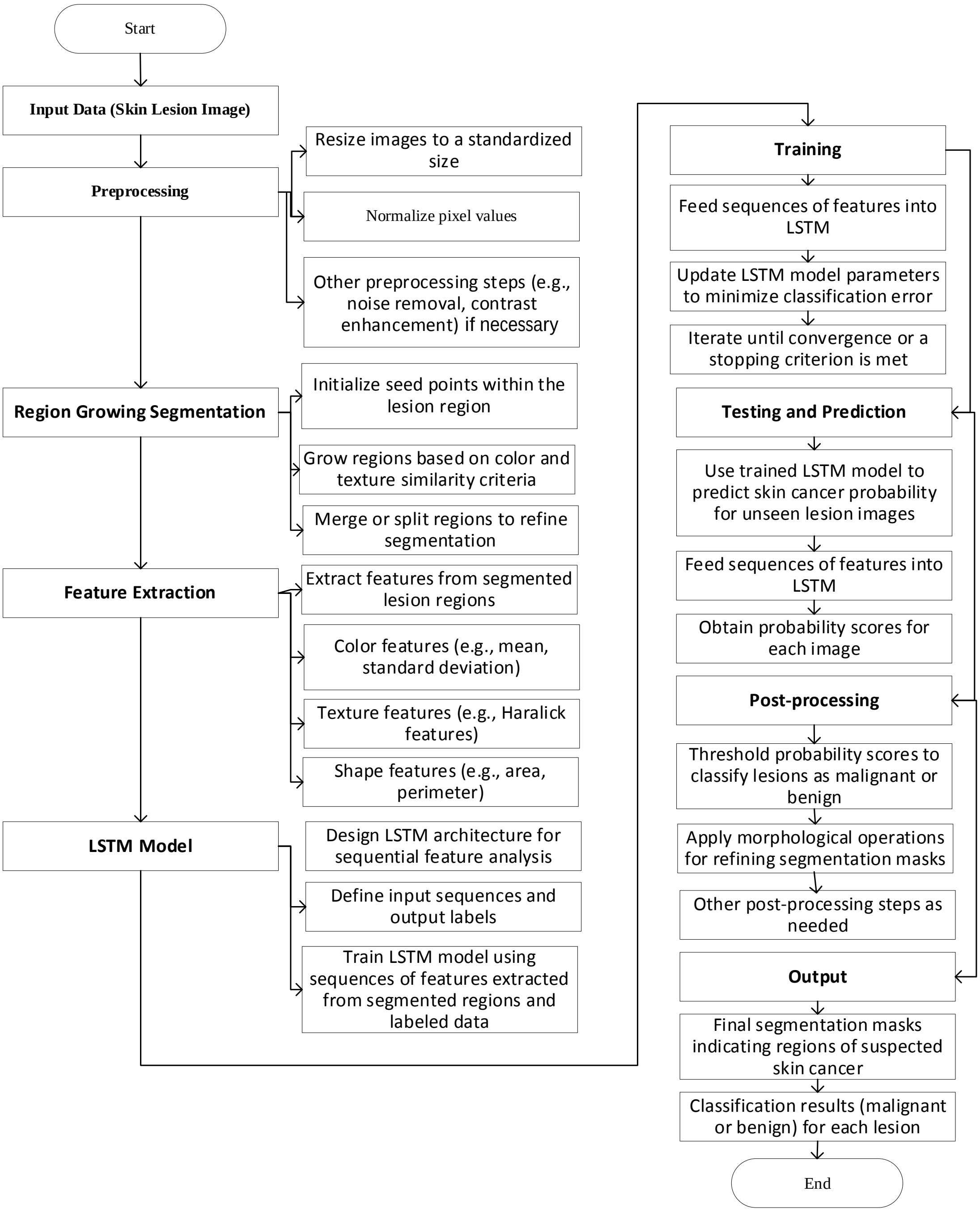

Start
Input Data (Skin Lesion Image)
Preprocessing
Resize images to a standardized size
Normalize pixel values
Other preprocessing steps (e.g., noise removal, contrast enhancement) if necessary
Region Growing Segmentation
Initialize seed points within the lesion region
Grow regions based on color and texture similarity criteria
Merge or split regions to refine segmentation
Feature Extraction
Extract features from segmented lesion regions
Color features (e.g., mean, standard deviation)
Texture features (e.g., Haralick features)
Shape features (e.g., area, perimeter)
LSTM Model
Design LSTM architecture for sequential feature analysis
Define input sequences and output labels
Train LSTM model using sequences of features extracted from segmented regions and labeled data
Training
Feed sequences of features into LSTM
Update LSTM model parameters to minimize classification error
Iterate until convergence or a stopping criterion is met
Testing and Prediction
Use trained LSTM model to predict skin cancer probability for unseen lesion images
Feed sequences of features into LSTM
Obtain probability scores for each image
Post-processing
Threshold probability scores to classify lesions as malignant or benign
Apply morphological operations for refining segmentation masks
Other post-processing steps as needed
Output
Final segmentation masks indicating regions of suspected skin cancer
Classification results (malignant or benign) for each lesion
End

Figure 4. The proposed methods, Region Growing -LSTM

This study also utilized the Recurrent Neural Network (RNN) algorithm, which fell under the artificial neural network category. Within RNN, the interconnections among nodes in each layer formed a directed graph that processed sequential variables. RNN comprised different variations, such as GRU (Gated Recurrent Units) and LSTM (Long Short-Term Memory Network), which aimed to enhance the effectiveness of the RNN algorithm. The RNN structure consisted of an input layer, one or more hidden layers, and an output layer.

4. Results and Discussion

4.1. Segmentation Results of the Region Growing Method

Before conducting experiments on skin cancer detection with the LSTM method, a series of experiments were performed to test the effectiveness and reliability of Region Growing algorithm in image segmentation. The segmentation results obtained were used for training the LSTM data. Region Growing technique is a segmentation approach based on a seed point. To initiate the segmentation process with this method, the determination of a seed point was necessary. In this study, the Harris cornering detection method was applied to obtain the seed point based on the angle with the highest intensity value. After the seed point was established, segmentation with Region Growing was executed. An overview of Region Growing segmentation experiment process is presented in Figure 5.

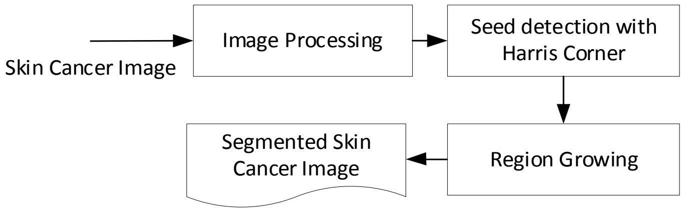

Skin Cancer Image
Image Processing
Seed detection with Harris Corner
Region Growing
Segmented Skin Cancer Image

Figure 5. Flowchart of the Region Growing

The pseudocode for the Harris corner detection and Region Growing segmentation algorithms is presented in Algorithm 1.

Algorithm 1: ***Initial Seed with Harris Corner detection***

```
Input: Dermoscopy image
If image not Grayscale
convert RGB image into Grayscale.
Use the Harris corner detector to extract corners of the skin lesion.
Select a seed from the detected corners:
If only one corner is detected, select it as the initial seed.
Else, for many detected corners, do the following:
Store the detected corners in an array.
Select the highest intensity of the array as the initial seed.
Pass the seed to region growing segmentation algorithm.
Output: Segmented Skin Lesion
```

Insert the Harris corner detection process to obtain corner points. Set the seed point based on the corner point with the highest intensity value. Perform Region Growing segmentation using the determined seed point. In this experiment, the Region Growing algorithm was evaluated based on its ability to effectively segment skin cancer images. The results obtained were expected to serve as the foundation for the subsequent experiments using the LSTM method for disease detection. The segmentation process comprised determining the initial seed point using Harris corner detection, which was crucial for Region Growing technique. The detailed flowchart of Region Growing experiment is presented in Figure 5.

The pseudocode for the Harris corner detection and Region Growing segmentation is shown in Algorithm 1, where the initial seed point was determined using the detected corners. This initial seed point was essential for the subsequent Region Growing segmentation process. The first stage in the Region Growing segmentation experiment was to test the influence of image size and neighbors on segmentation accuracy. From the experiment results, for an image size of 256×256 and neighbors of 4×4, an accuracy of 64.43%, precision of 73.31%, and recall of 69.87% were obtained with an average execution time of 25.72 seconds. For an image size of 200×200 and neighbors of 4×4, the accuracy was 63.62%, precision was 72.47%, and recall was 71.78% with an average execution time of 11.63 seconds. For an image size of 128×128 and neighbors of 4×4, the accuracy was 63.31%, precision was 63.31%, and recall was 72.61% with an average execution time of 3.05 seconds.

For an image size of 256×256 and neighbors of 8x8, the accuracy was 65.59%, precision was 72.67%, and recall was 75.64% with an average execution time of 41.73 seconds. For an image size of 200×200 and neighbors of 8x8, the accuracy was 65.49%, precision was 71.98%, and recall was 75.79% with an average execution time of 16.84 seconds. For an image size of 128×128 and neighbors of 8x8, the accuracy was 65.28%, precision was 65.28%, and recall was 78.55% with an average execution time of 3.75 seconds. The summarized results of the experiment on the influence of image size and neighbors on segmentation accuracy are presented in Table 1.

Table 1. Experiment Results for Image Size and Neighbours

Image Size	Region Growing	Image/s	Avg. Accuracy	Avg. Precision	Avg. Recall
256×256	4×4	25.72	64.43%	73.31%	69.87%
200×200	4×4	11.63	63.62%	72.47%	71.78%
128×128	4×4	3.05	63.31%	63.31%	72.61%
256×256	8×8	41.73	65.59%	72.67%	75,64%
200×200	8×8	16.84	65.49%	71.98%	75.79%
128×128	8×8	3.75	65.28%	65.28%	78.55%

Based on the obtained results, several considerations could be drawn. Region Growing technique with neighbors 8x8 and an image size of 256×256 performed well with high accuracy, precision, and recall. However, the longer execution time could be a drawback in real-time applications or when processing a large number of images. Region Growing technique with neighbors 4×4 and an image size of 128×128 had a fast execution time and provided competitive results in terms of accuracy, precision, and recall. This could be a better choice in cases where execution speed was a crucial consideration. Region Growing technique with neighbors 8x8 and an image size of 128×128 also produced good results with accuracy, precision, and recall comparable to larger image sizes (256×256), but with a faster execution time. Based on these results, this study used an image size of 128×128 and neighbors 8x8 with an accuracy of 65.28%, precision of 65.28%, and recall of 78.55%.

The next stage of the segmentation experiment was to test the influence of pre-processing on the segmentation accuracy of Region Growing technique. Pre-processing techniques to be tested included converting images to grayscale, dark corner removal, median filter, and Contrast Limited Adaptive Histogram Equalization (CLAHE). In this study, images from the dataset were converted to grayscale. According to Leiter et al. [5], segmentation on a grayscale image yielded higher segmentation results compared to a color image.

The first pre-processing technique to be tested was the application of the Corner Borders Removal module. This was necessary because the PH2 dataset contained objects that significantly disturbed the segmentation process, specifically a black background in the corners of the dermoscopy image. To address this challenge, the study cropped the image by cutting 20 pixels on each side (top, bottom, left, and right). After cropping, the image size was returned to its original size, namely 128×128. This technique had been applied in Roy et al. [29] and was effective in reducing detection errors. The pseudocode algorithm for Corner Borders Removal is presented in Algorithm 2:

Algorithm 2: ***Corner Borders Removal***

```
Input: Dermoscopy image & size pixels to cut
Get the original image dimensions
    height, width = get dimensions(image)
    Calculate the new dimensions after cropping
    new height = height - 2 * pixels_to_cut
    new_width = width - 2 * pixels_to_cut
    Initialize a new image with the new dimensions
    Corner less Image = create_image(new_height, new_width)
Output: Cornerless Image
```

The results of cropping an image using the Corner Borders Removal algorithm is presented in Figure 6.

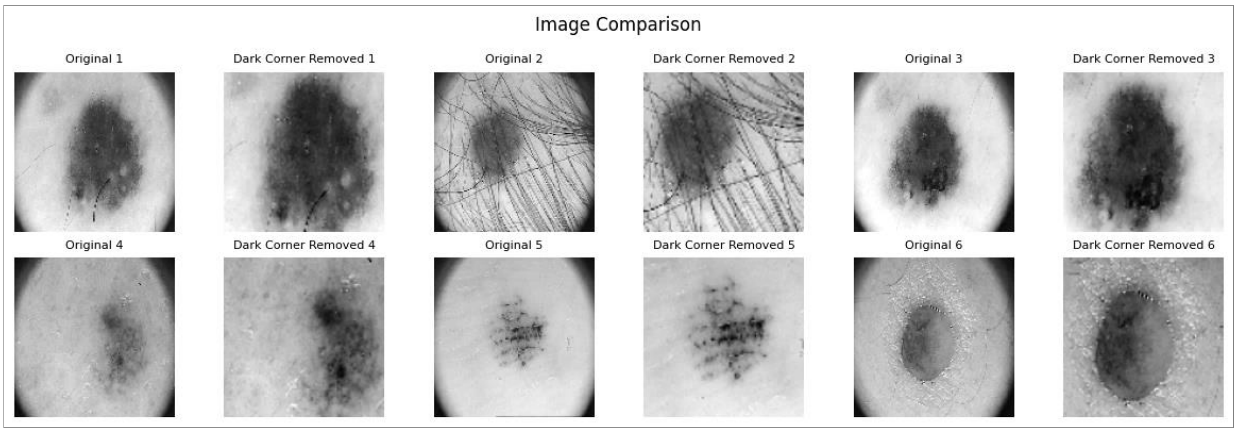


Figure 6. Dataset Image Before and After Cropping

After obtaining the dataset image processed with dark corner removal, the next step was to perform segmentation with this data. In addition, the segmentation was also performed with the original image. Testing was conducted by taking 40 images from the dataset with image size 128×128 and neighbour size 8×8. The segmentation results for the original image yielded an accuracy of 61.9%, precision of 80%, and recall of 67.6%. Meanwhile, the findings for the cropped image provided an accuracy of 65.97%, precision of 63.94%, and recall of 84.44%. Based on these results, the dark corner removal pre-processing technique was used. Figure 7 shows a comparison of segmentation results between the original and cropped image.

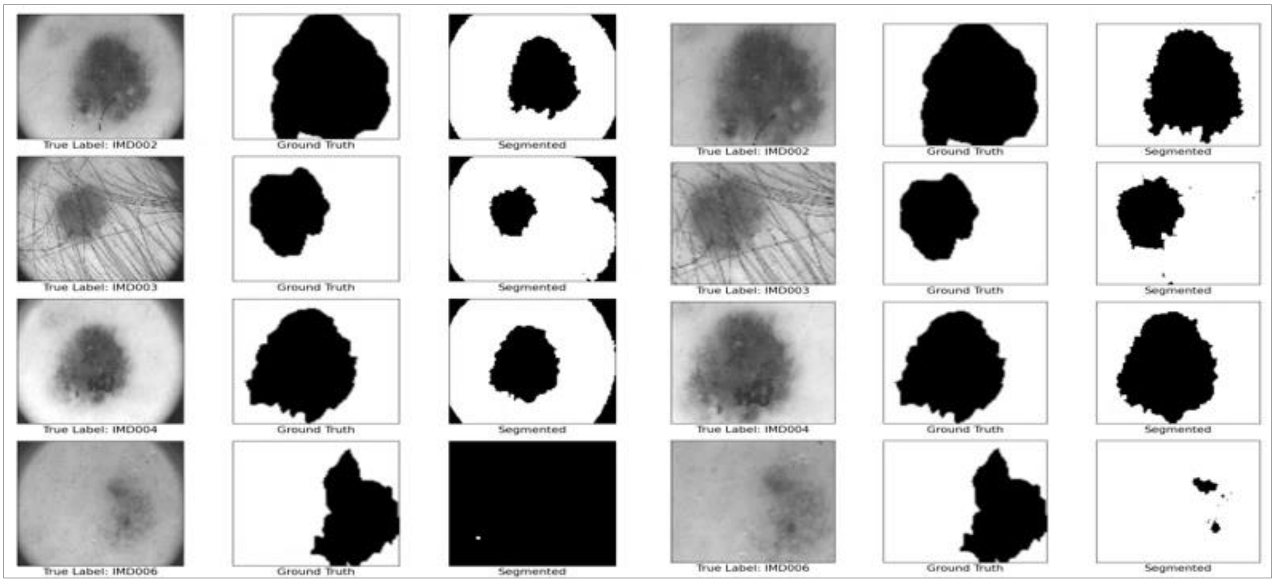


Figure 7. Segmentation Results of Original and Dark Corner Removal Image

The next pre-processing technique to be tested was the median filter and CLAHE. The implementation of the median filter was carried out using the median Blur module from the OpenCV2 library, while the create CLAHE module from the OpenCV2 library was used for CLAHE. For CLAHE implementation, values for the tile size parameter for histogram equalization and the clip limit to prevent excessive contrast were needed. Based on Greff et al. [30], the parameter sizes for CLAHE used in this study are tiles 8 to achieve sharper contrast enhancement and a clip limit size of 1% as a limit for contrast. The use of these parameters helped Greff et al. [30] to achieve a high classification accuracy. The results of pre-processing with the median filter, CLAHE, and their combination are shown in Figures 8 to 10.

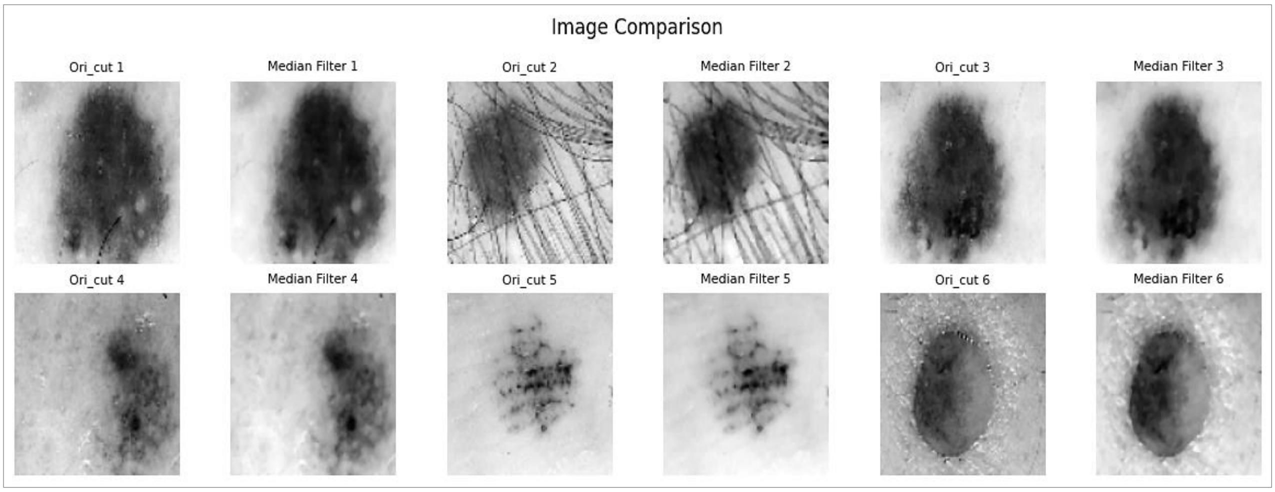


Figure 8. Result of Median Filter Pre-processing

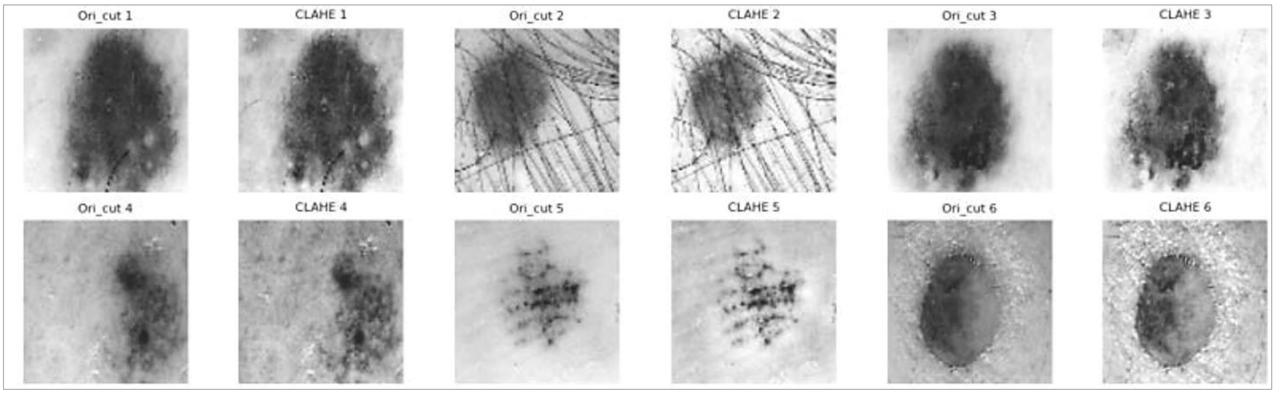


Figure 9. Result of CLAHE Pre-processing

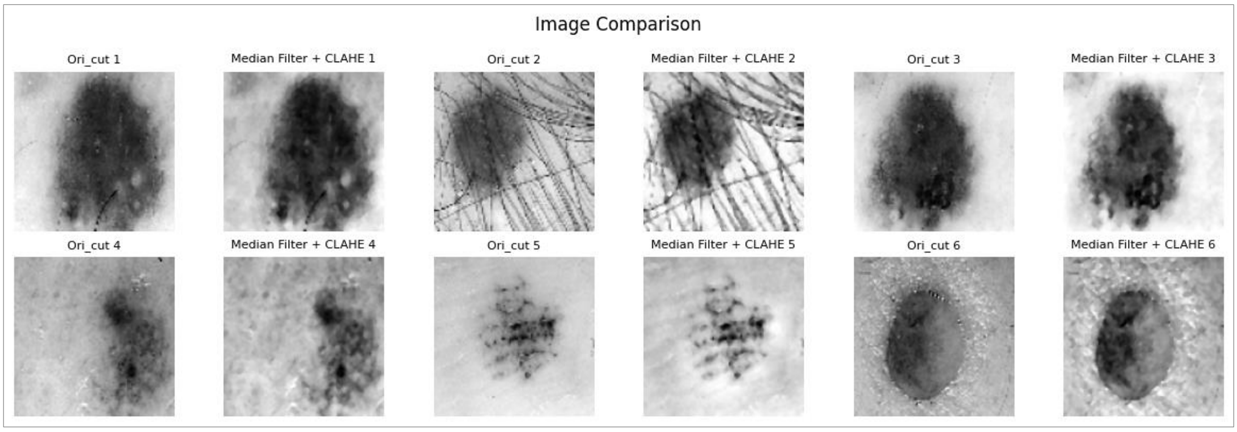


Figure 10. Result of Median Filter + CLAHE Pre-processing

The pre-processed image was tested for segmentation using Region Growing technique. The segmentation results for median filtered pre-processing yielded an accuracy of 65.41%, precision of 64.58%, and recall of 84.05%. The findings for CLAHE pre-processing gave an accuracy of 60.65%, precision of 62.24%, and recall of 77.17%. Meanwhile, the combination of both methods yielded an accuracy of 70.19%, precision of 68.042%, and recall of 83.67%. Figures 11 and 12 provide a comparison of the results for each segmentation.

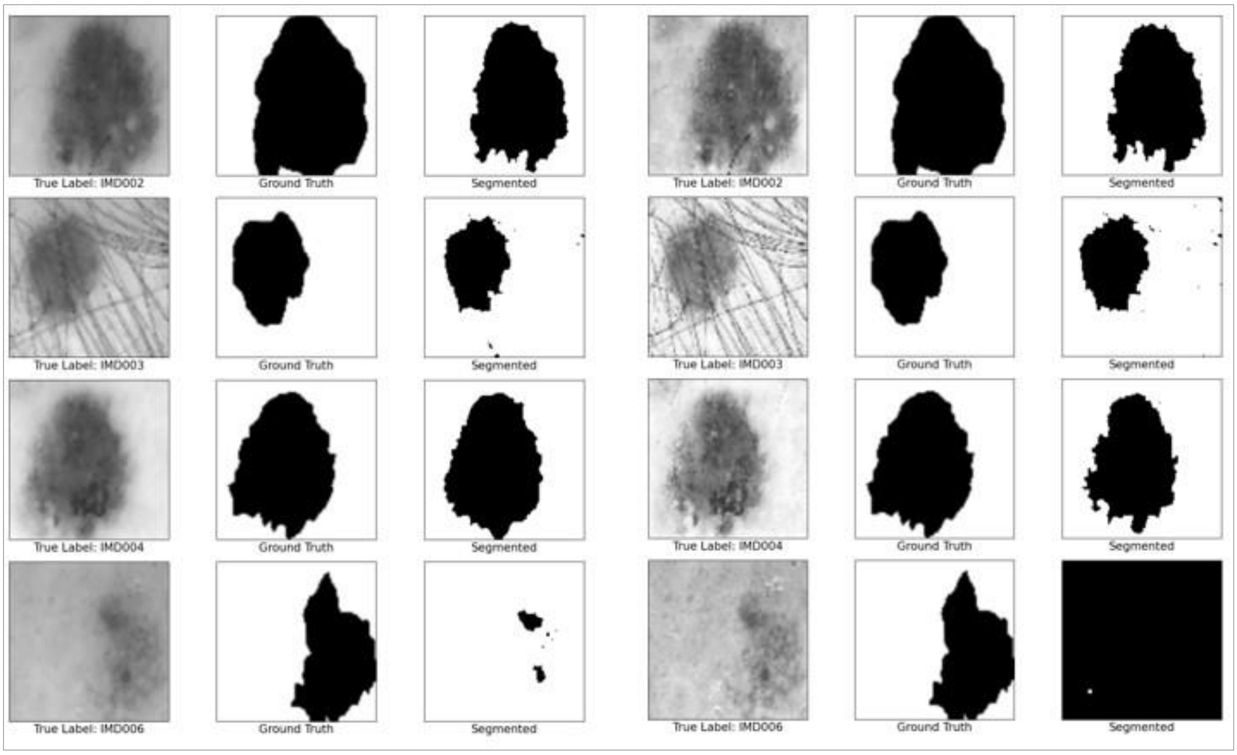


Figure 11. Segmentation Results of Median Filtered (Left) and CLAHE (Right)

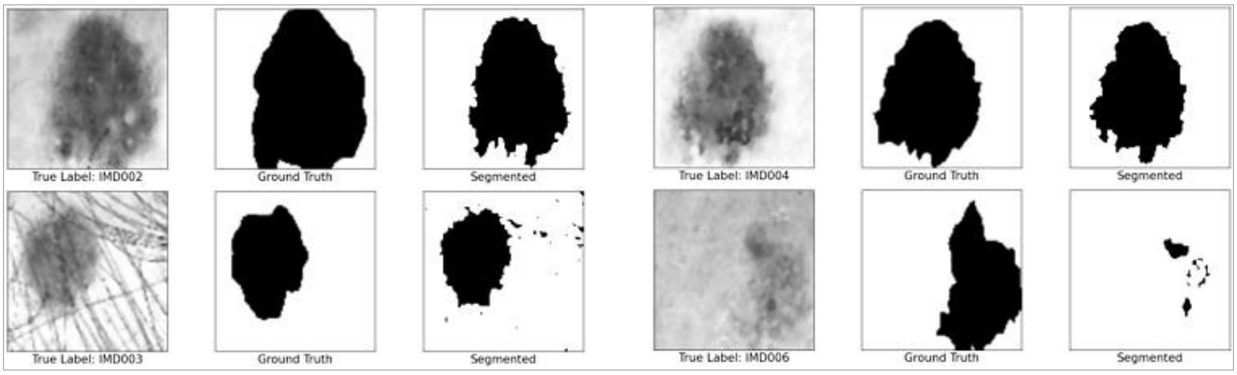


Figure 12. Segmentation Results of Median Filtered (Left) and CLAHE (Right)

Based on Region Growing segmentation results in Table 2, the best processing technique was the combination of Dark Corner Removal, Median Filtered, and CLAHE. Therefore, the combination of these three pre-processing techniques was used in this study, and its segmentation results were used for training the LSTM model.

Table 2. Experiment Results of Pre-processing and Region Growing Segmentation

No.	Processing Methods	Accuracy	Precision	Recall
1	No-preprocessing	61,94%	80,64%	67,64%
2	Dark Corner Removal	65,98%	63,94%	84,45%
3	Dark Corner Removal + Median Filtered	65,42%	64,59%	84,05%
4	Dark Corner Removal + CLAHE	60,65%	62,24%	77,17%
5	Dark Corner Removal + Median Filtered + CLAHE	70,20%	68,04%	83,68%

4.2. Results of the LSTM Method

In the experiment on skin cancer detection using the LSTM method, several steps were used. First, oversampling was performed to balance the dataset. Second, the dataset was divided into three parts, namely training, testing, and validation. Third, data pre-processing was carried out, including normalization, resizing, and data pre-processing, followed by experiments to determine the optimal image size. Hyperparameter adjustments were made to find the right model structure. Subsequently, epoch testing was performed to obtain an appropriate number of epochs to prevent overfitting. Finally, before prediction, training was conducted using both training and validation data, and the model was evaluated for accuracy, precision, and recall. By following these steps, the LSTM experiment for skin cancer detection could be conducted systematically, aiming to produce an accurate prediction model. Figure 13 provides more detailed information about the LSTM experiment process.

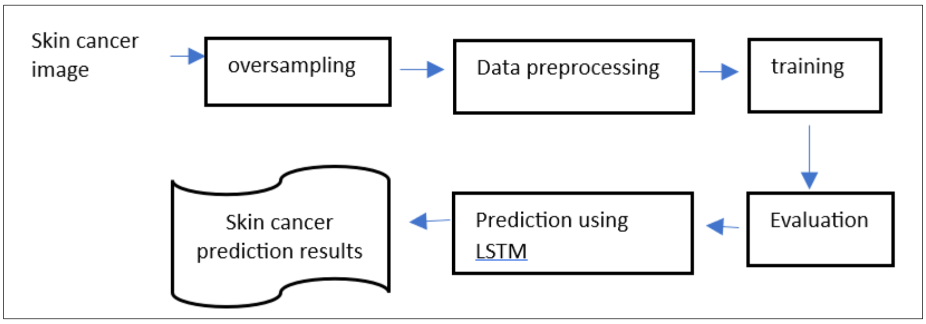
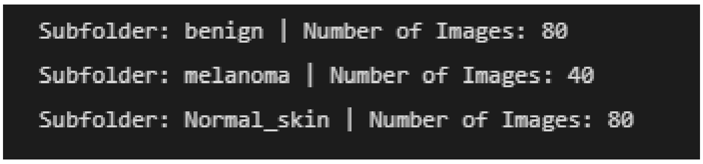


Figure 13. LST Experiment Flowchart

Before advancing to the prediction phase using LSTM, the initial step comprised performing oversampling. The dataset was then partitioned into training, validation, and testing sets using the splitfolders Python library. In addition, due to the imbalanced nature of the dataset employed, as depicted in Figure 14, it was essential to adjust the quantity of data instances in each class utilizing oversampling or undersampling methods. In this study, the oversampling strategy was utilized, in line with the findings of Limanto et al. [51], suggesting that oversampling techniques offered superior outcomes compared to undersampling. Consequently, the oversampling technique was implemented for this study.



```
Subfolder: benign | Number of Images: 80
Subfolder: melanoma | Number of Images: 40
Subfolder: Normal_skin | Number of Images: 80
```

Figure 14. Imbalanced Dataset

During oversampling, this study generated data in the minority class by providing data until it reached the same quantity as the majority class. To obtain better results, the duplicated data were first transformed through the process of image augmentation (Figure 15). This augmentation was randomly performed and could include horizontal or vertical image flipping, image rotation (between -15° to 15°), or enlargement (10%-25%). The pseudocode for image augmentation is provided in Algorithm 3.

Algorithm 3: ***Image Augmentation***

```
Input: Dermoscopy image & random Action type
if augmentation_type is equal to Augmentation.ZOOM:
        zoom_factor = random value between 0.10 and 0.25
        zoomed_width = image.shape[1] multiplied by zoom_factor
        zoomed_height = image.shape[0] multiplied by zoom_factor
        image = resize image to (zoomed_width, zoomed_height)
if augmentation_type is equal to Augmentation.FLIP:
        flip_code = random value of 0 or 1
        image = flip image horizontally or vertically using flip_code
if augmentation_type is equal to Augmentation.ROTATE:
        angle = random value between -15 and 15 degrees
        height, width = image size
rotation_matrix = get rotation matrix with angle and rotation center at (width/2, height/2)
        image = warpAffine image using rotation_matrix and size (width, height)
end if
Output: Augmentated Dermoscopy Image
```

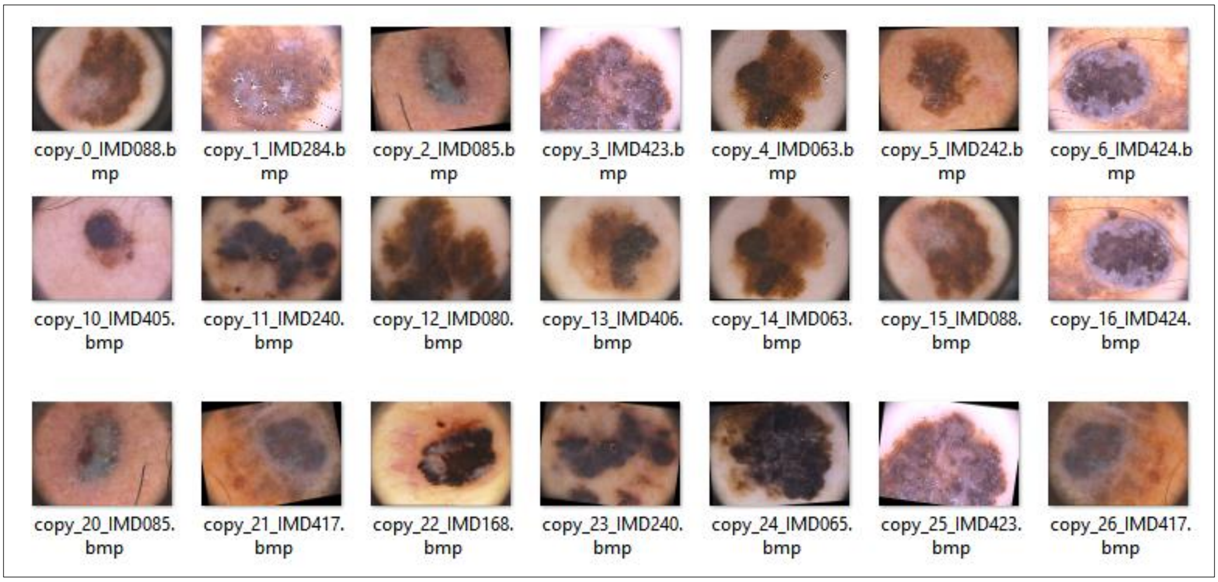
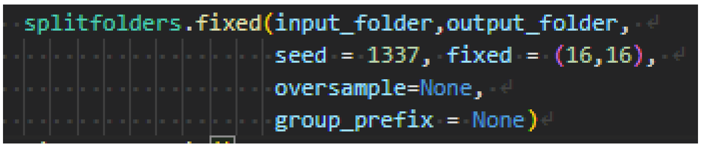


Figure 15. Result of Oversampling and Image Augmentation

From the oversampling process, the minority class data (melanoma) had the same amount of data as the other classes. The next step was to divide the data in the dataset to be used for training, validation, and testing using the splitfolders library. This study used a total of 16 data for validation, 16 data for testing each class, and the rest for training. The initialization of the splitfolders module is presented in Figure 16.



```
splitfolders.fixed(input_folder,output_folder,
                   seed = 1337, fixed = (16,16),
                   oversample=None,
                   group_prefix = None)
```

Figure 16. Split folder module for training data division

The dataset was partitioned into training, validation, and testing sets using the splitfolders procedure. The outcome of this operation revealed that each class in the training dataset now contained 48 data points, leading to a total of 144 data points overall. In addition, there were 16 data points allocated to both the validation and testing datasets. Figure 16 presents the results of the splitfolders process, demonstrating that the number of images in each class had been equalized, eliminating any class having more images than the others. In Figure 16, the left part showed the splitting results for the training data with a total of 144 images. The middle part showed the splitting and oversampling results for the testing data with a total of 48 images, and the right part revealed the splitting and oversampling results for the validation data with a total of 48 images.

After the directories of each class were balanced, the process of pre-processing dermoscopy image was initiated. First, the dermoscopy image was converted from RGB to grayscale using the OpenCV library and the imread() function.

The image was resized using the resize() function in OpenCV to speed up the deep learning process. However, a small experiment was needed to determine whether the image size affected the accuracy of the deep learning model.

The experimental results could be seen in Table 3, which presented the findings of experiments to determine the appropriate image size using a simple deep learning model with 1 LSTM layer. Training and validation were performed for 20 epochs with a batch size of 32. Epochs were the number of iterations performed by the deep learning model, and it was crucial to pay attention to the execution time. In this study, the same number of epochs was used to compare the results with validity. The results showed that the validation accuracy did not differ significantly, but the time required for training and validation was greatly reduced. A previous study by Sherstinsky [31] also showed that the image size did not always impact the accuracy of the deep learning model, and execution time tended to increase with larger image sizes. Based on this experiment, it could be concluded that an image size of 128×128 was most suitable.

Table 3. Experiment Results Image Size

No.	Image Size	Result				
		Training		Validation		Execution Time
		Accuracy	Loss	Accuracy	Loss	
1	256 × 256	0.9721	0.6528	0.9520	0.9333	26.4 second
2	200 × 200	0.9756	0.6042	0.9437	0.8840	19.1 second
3	128 × 128	0.9701	0.7372	0.9541	0.8515	15.9 second
	Average	0.9626	0.6647	0.9493	0.8896	0.46 seconds

The image was converted into NumPy arrays using the NumPy library's array () function. The last step was the color normalization of image from the range of 0-255 to the range of 0-1. The results of data pre-processing could be seen in Figure 17, which showed an example of a previously processed leaf image entering the deep learning model.

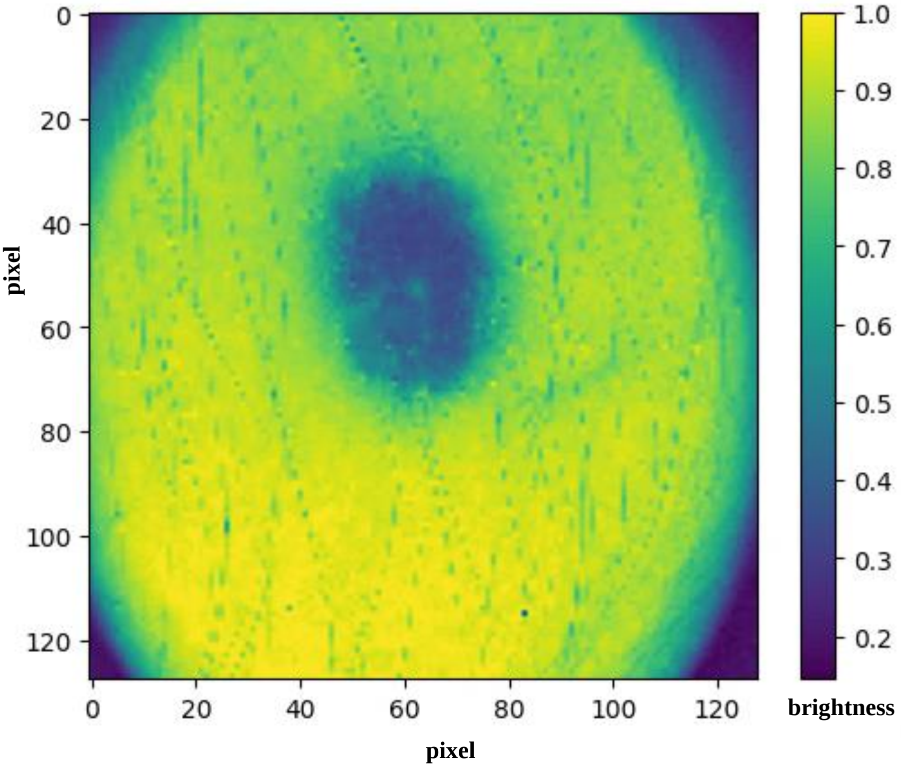
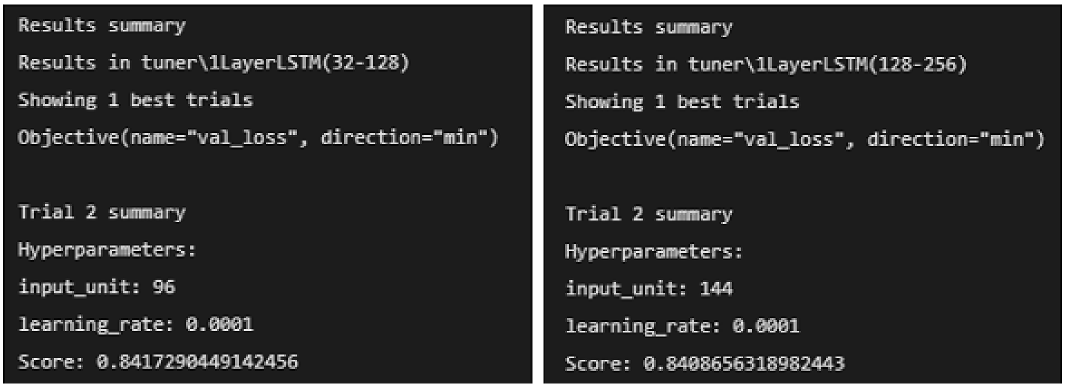


Figure 17. Pre-processing Results

Experiments were conducted to determine the structure of the deep learning model. In this stage, a comparison was made between two models, namely 2-layer LSTM and 1-layer LSTM. In addition, hyperparameter tuning was performed using the Keras Tuner library. In addition, Keras Tuner was employed with a random search algorithm to find the optimal parameters for the LSTM deep learning model. The parameters tuned included the number of neurons in the LSTM layer and the suitable learning rate. The hyperparameter tuning process was divided into two parts, namely from the node value 32 to 128, as well as from 128 to 256. For the learning rate, the tuner was given options of 0.01, 0.001, and 0.0001. The tuning process aimed to reduce the validation loss variable. Moreover, the early stopping technique was utilized to halt the tuning process after the validation loss did not decrease for 10 epochs. Each tuning iteration was carried out for 100 epochs, and the entire tuning process was repeated three times. The total number of tuning iterations performed was 5 times.

The results of the experiments using a 1-layer LSTM are presented in Figure 18. The left side displayed the results of the first-layer experiment, where the best model with a search for node numbers between 32 and 128 produced the lowest validation loss score of 0.8417290449142456. The optimal LSTM node value was 96, with a learning rate of 0.0001. The right side showed the best model from the second-layer experiment, with a search for node numbers between 128 and 256. The result was the lowest validation loss score of 0.8408656318982443. In this experiment, the best LSTM node value was 144, with a learning rate of 0.0001.



```
Results summary
Results in tuner\1LayerLSTM(32-128)
Showing 1 best trials
Objective(name="val_loss", direction="min")

Trial 2 summary
Hyperparameters:
input_unit: 96
learning_rate: 0.0001
Score: 0.8417290449142456
```

```
Results summary
Results in tuner\1LayerLSTM(128-256)
Showing 1 best trials
Objective(name="val_loss", direction="min")

Trial 2 summary
Hyperparameters:
input_unit: 144
learning_rate: 0.0001
Score: 0.8408656318982443
```

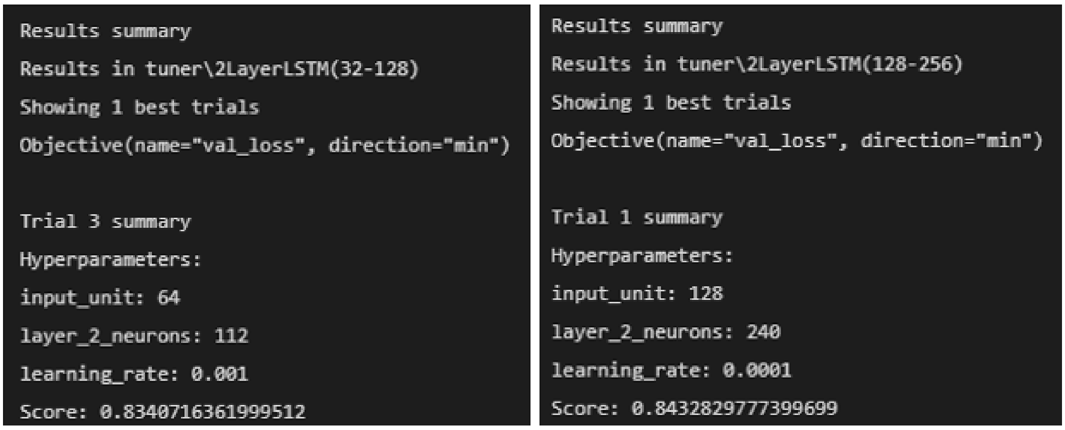
Figure 18. Results of 1-layer LSTM Experiments

Experiments were conducted to determine the best validation loss value for a model with two LSTM layers. The experimental process was similar to the previous experiment for a one-layer LSTM model, but the difference was in the number of LSTM layers used. In addition, the process was divided into two parts, namely finding the number of LSTM nodes between 32 and 128 as well as between 128 and 256. The outcomes of the experiments comprising the two-layer LSTM models are depicted in Figure 19. On the left side, the results of the first experiment revealed that the optimal model was achieved with 64 nodes in the first-layer LSTM, 112 nodes in the second-layer LSTM, and a learning rate of 0.001. The lowest validation loss attained in this scenario was 0.8340716361999512. On the right side, the results of the second experiment showed that the best model was obtained with 128 nodes in the first-layer LSTM, 240 nodes in the second-layer LSTM, and a learning rate of 0.0001, leading to the lowest validation loss of 0.8432829777399699. A summary of the findings from the hyperparameter tuning experiments is presented in Table 4.

Table 4. Hyperparameter Tuning Experiments

No.	$\sum Layer$	Nodes range	$\sum Node$		Result	
			Node layer 1	Node layer 2	Learning rate	Validation loss
1	1	32-128	96	-	0.0001	0. 841729
2		128-256	144	-	0.0001	0. 840865
3	2	32-128	64	112	0.001	0. 834071
4		128-256	128	240	0.0001	0. 843282

Based on the hyperparameter tuning results in Table 4, the best model structure was obtained, consisting of 2 LSTM layers with 64 nodes in layer 1 and 112 nodes in layer 2. Therefore, the training model used in this study is presented in Figure 20.



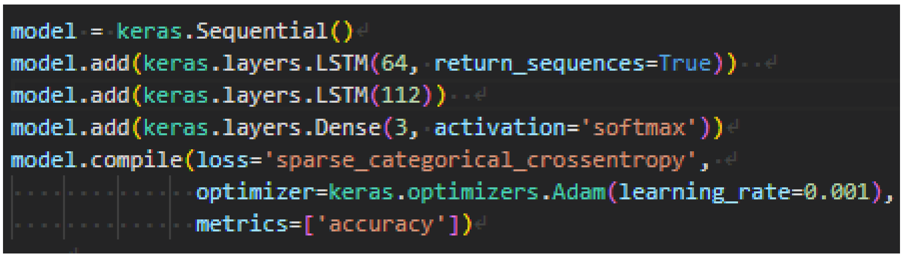
```
Results summary
Results in tuner\2LayerLSTM(32-128)
Showing 1 best trials
Objective(name="val_loss", direction="min")

Trial 3 summary
Hyperparameters:
input_unit: 64
layer_2_neurons: 112
learning_rate: 0.001
Score: 0.8340716361999512
```

```
Results summary
Results in tuner\2LayerLSTM(128-256)
Showing 1 best trials
Objective(name="val_loss", direction="min")

Trial 1 summary
Hyperparameters:
input_unit: 128
layer_2_neurons: 240
learning_rate: 0.0001
Score: 0.8432829777399699
```

Figure 19. Results of 2-layer LSTM Experiments



```
model = keras.Sequential()
model.add(keras.layers.LSTM(64, return_sequences=True))
model.add(keras.layers.LSTM(112))
model.add(keras.layers.Dense(3, activation='softmax'))
model.compile(loss='sparse_categorical_crossentropy',
              optimizer=keras.optimizers.Adam(learning_rate=0.001),
              metrics=['accuracy'])
```

Figure 20. 2-layer LSTM Training Model

The next step was determining the number of epochs to be used to run the model. The model was tested 10 times using 200 epochs, and accuracy and loss plots were obtained based on the number of epochs. The plots for the ^{2}nd, ^{5}th, and ^{10}th tests are presented in Figure 21.

From the plot in the test results at epoch 200, the average validation loss increased after the 25th epoch. The increase in validation loss was attributed to overfitting. Based on the test results at epoch 200, the selected number of epochs for this study was 25. After determining the number of epochs and the structure of the deep learning model, the next step comprised the training phase using both the training and validation data to generate a model for predicting the testing data. The model resulting from the training phase needed to be evaluated to obtain its accuracy, precision, and recall rates. The outcomes from the LSTM model included 54.1% for accuracy, 57.8% for precision, and 54.1% for recall.

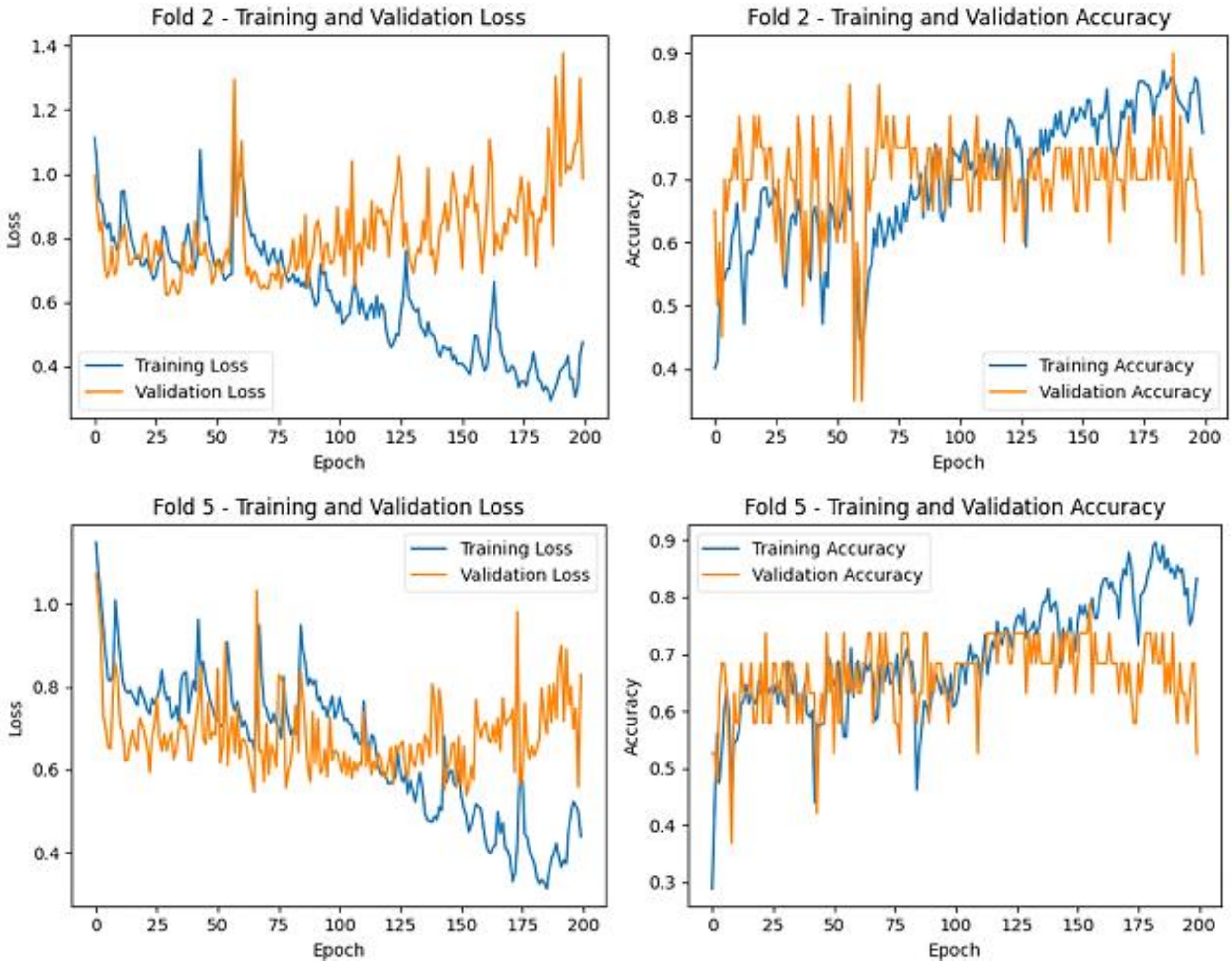


Figure 21. Plot of Accuracy and Loss for Testing at Epoch 200

4.3. Results of the LSTM Method and Region Growing

In the experiment on skin cancer detection using LSTM and Region Growing, the parameters for image size, model, and epochs used were the same as those employed in the implementation of the LSTM method. The key difference lied in the utilization of segmented data for training, validation, and testing, achieved through Region Growing. The segmented data was trained with a two-layer LSTM model (64 and 122) for 25 epochs. The trained model was then evaluated to obtain accuracy, precision, and recall values. The expectation was that the use of segmented datasets could enhance prediction accuracy compared to the utilization of LSTM method alone. Figure 22 presents detailed information about the overall process of the LSTM and Region Growing experiments.

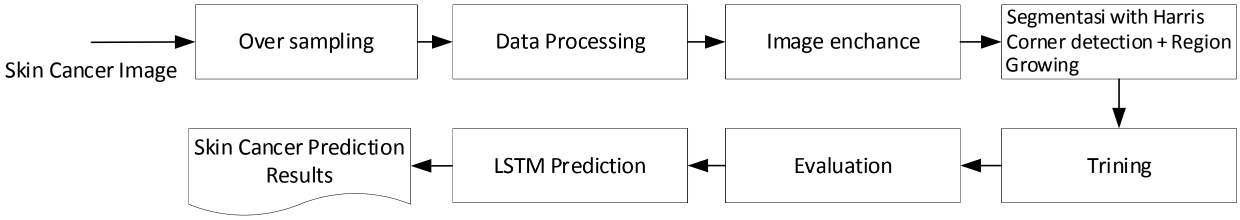


Figure 22. Flowchart of the LSTM and Region Growing Experiment

In the oversampling process, the implementation followed the same approach as described in subsection 4.1 Results of the LSTM Method. After oversampling the data, the same preprocessing, image enhancement, and segmentation processes were applied as outlined in subsection 4.1 Results of the Region Growing Method. Following segmentation, the dataset was trained using a two-layer LSTM model (64 and 122) for 25 epochs. The prediction process was conducted for individual data points and for all data within the testing dataset. The prediction results could be seen in Figure 23, illustrating an example prediction for a dermoscopy image of skin affected by benign skin cancer. When the model successfully predicted that the dermoscopy image was affected by the disease, recommendations for the patient were displayed below the prediction results. The prediction process for the entire dataset in the testing phase occurred for approximately 3.5 seconds.

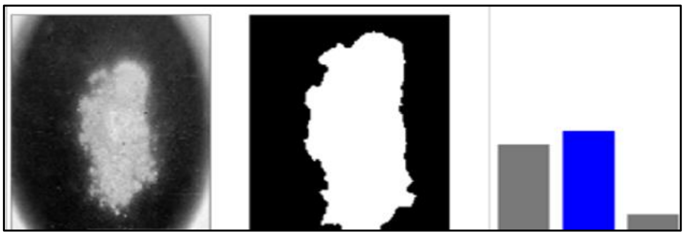


Figure 23. Prediction results for one data point using LSTM and Region Growing

4.4. Results Analysis and Algorithm Comparison

The comparison table for segmentation methods in this study and relevant previous studies are presented in Table 5. Despite the use of dark corner removal and Region Growing segmentation aided by image enhancement using Median filter & CLAHE, as well as Harris Corner Detection for seed determination, accurate segmentation results similar to the findings in Leiter et al. [5], which applied Harris Corner Detection + Region Growing + Post Processing filling and dilation, could not be achieved due to the use of different image sizes. The first factor was that image in this study was smaller, preventing the use of the post-processing filling and dilation technique as in Leiter et al. [5]. This study could not use image with a size of 523 × 382 pixels due to the computational limitations of the tool used.

The second factor was that the initial seed from the detection results of corners using Harris Corner Detection sometimes inaccurately targeted lesions or wounds in dermoscopy image. This could be caused by interfering objects in the image, such as hair or remaining dark corners even after removal by the dark corner removal module. In addition, in image with vague skin lesions, accuracy remained challenging, even with enhanced contrast using CLAHE. Other preprocessing techniques were required to eliminate unwanted objects in the segmentation process, as well as other image enhancement techniques to improve details in image with thin or vague skin lesions.

From the results of the training experiments using the LSTM method with Region Growing + LSTM, the application of segmentation before training could improve the accuracy of the created model. An accuracy improvement of 20.9% was obtained after applying segmentation to the dataset. The increase in accuracy was also accompanied by improvements in precision and recall. This occurred because segmentation aided the training process by providing image that was more focused on skin lesions and eliminating unnecessary objects. Table 2 showed the comparison of the evaluation results of the LSTM method with Region Growing + LSTM.

The classification accuracy rate of 96.26% generated from Region Growing + LSTM in this study was relatively good when compared to the results of previous studies, such as those in previous studies [6, 15, 30].

Table 5. Comparison methods for the detection of skin cancer

Methods	Accuracy (%)
CNN [1]	94.206
Optimization Algorithm-Based Exception Neural Network [12]	92.22
SVM [13]	92
CNN [13]	95
Dingo Optimizer (DOX) [14]	92.3
VGG-16 [15]	92
MobileNetV2 and customized CNN [16]	85
The proposed methods -Region Growing-LSTM	96.62
Proposed method- LSTM solely	84

Based on Table 5, the proposed method utilizing Region Growing-LSTM achieved the highest accuracy at 96.62%. This method combined Region Growing, a technique for segmenting images, with LSTM, a type of recurrent neural network known for its ability to process sequential data. The high accuracy suggested that the integrated approach effectively captured the complex patterns present in skin cancer images, leading to improved detection performance. Meanwhile, the proposed method employing LSTM solely achieved an accuracy of 84%, which was lower compared to the Region Growing-LSTM technique. This indicated that while LSTM could be effective for processing sequential data, combining it with Region Growing led to better results in the context of skin cancer detection.

Among the other methods, CNNs demonstrated consistent performance, with reported accuracy ranging from 94.206% to 95%. SVM, optimization algorithm-based neural networks, and VGG-16 also exhibited competitive accuracies in the range of 92% to 92.3%. However, MobileNetV2 and customized CNNs showed a slightly lower accuracy of 85%. In conclusion, the integration of Region Growing with LSTM in the proposed method proved to be a promising approach for skin cancer detection, offering higher accuracy compared to other methods evaluated in the study. This showed the importance of incorporating diverse methodologies and leveraging their complementary strengths to enhance detection performance in medical image analysis tasks.

5. Conclusion

In conclusion, Region Growing segmentation on the grayscale image of the PH2 dataset using pre-processing techniques such as dark corner removal, median filter, and CLAHE achieved an accuracy of 70.20% with a segmentation processing time of 4 seconds per image. Region Growing + LSTM algorithm demonstrates superior accuracy at 96.62%, outperforming the LSTM algorithm with an accuracy of 84%. While Region Growing + LSTM proved to be more accurate in skin cancer image detection, the trade-off with increased time demands must be considered. Future studies could focus on refining efficiency without compromising accuracy, potentially advancing the application of automated methods for early melanoma detection and contributing to improved outcomes in management.

Based on the results, future studies are advised to optimize pre-processing for improved accuracy in Region Growing segmentation, either by removing objects, such as hair, or optimizing the threshold for Region Growing. The use of better tools for the segmentation process with Region growing on high-quality images is also advised. Post-processing techniques after segmentation, including feature extraction, image masking, or other image enhancement methods, could also be applied.

6. Declarations

6.1. Author Contributions

Conceptualization, S.Y.I. and R.Y.; methodology, S.Y.I.; software, R.Y.; validation, S.Y.I., M.S.H., and R.Y.; formal analysis, S.Y.I.; investigation, M.S.H.; resources, R.Y.; data curation, R.Y.; writing—original draft preparation, S.Y.I.; writing—review and editing, D.A.D.; visualization, R.Y.; project administration, N.P.; funding acquisition, N.P. and D.A.D. All authors have read and agreed to the published version of the manuscript.

6.2. Data Availability Statement

The data presented in this study are available in the publicly accessible PH2 Dataset.

6.3. Funding

This work was supported by Institute Informatics and Business Darmajaya, Indonesia and the INTI International University, Malaysia.

6.4. Institutional Review Board Statement

Not applicable.

6.5. Informed Consent Statement

Not applicable.

6.6. Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

7. References

[1] Wang, H., Shen, Z., Zhang, Z., Xu, Z., Li, S., Jiao, S., & Lei, Y. (2021). Improvement of region-merging image segmentation accuracy using multiple merging criteria. Remote Sensing, 13(14), 2782. doi:10.3390/rs13142782.

[2] Dai, A., & Kim, S. J. (2024). Systemic calcineurin inhibitors tacrolimus and voclosporin: A review of off-label dermatologic uses. Journal of the American Academy of Dermatology, 90(2), 358–367. doi:10.1016/j.jaad.2023.05.074.

[3] Dildar, M., Akram, S., Irfan, M., Khan, H. U., Ramzan, M., Mahmood, A. R., Alsaiari, S. A., Saeed, A. H. M., Alraddadi, M. O., & Mahnashi, M. H. (2021). Skin cancer detection: A review using deep learning techniques. International Journal of Environmental Research and Public Health, 18(10), 5479. doi:10.3390/ijerph18105479.

[4] Hamilton, N. A., Pantelic, R. S., Hanson, K., & Teasdale, R. D. (2007). Fast automated cell phenotype image classification. BMC Bioinformatics, 8. doi:10.1186/1471-2105-8-110.

[5] Leiter, U., Keim, U., & Garbe, C. (2020). Epidemiology of skin cancer: Update 2019. In Advances in Experimental Medicine and Biology, 1268, 123–139. doi:10.1007/978-3-030-46227-7_6.

[6] Stevenson, M. L., Glazer, A. M., Cohen, D. E., Rigel, D. S., & Rieder, E. A. (2017). Frequency of total body skin examinations among US dermatologists. Journal of the American Academy of Dermatology, 76(2), 343–344. doi:10.1016/j.jaad.2016.09.017.

[7] Blackledge, J. M. (2005). Digital Image Processing: Mathematical and Computational Methods. In Digital Image Processing: Mathematical and Computational Methods, 1–797. doi:10.1533/9780857099464.

[8] Imtiaz, I., Ahmed, I., Jeon, G., & Muramatsu, S. (2021). An Efficient Image Processing and Machine Learning based Technique for Skin Lesion Segmentation and Classification. 2021 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA ASC 2021 - Proceedings, December, 1499–1505.

[9] Arnal Barbedo, J. G. (2013). Digital image processing techniques for detecting, quantifying and classifying plant diseases. SpringerPlus, 2(1), 660. doi:10.1186/2193-1801-2-660.

[10] Rajendran, V. A., & Shanmugam, S. (2024). Automated Skin Cancer Detection and Classification using Cat Swarm Optimization with a Deep Learning Model. Engineering, Technology and Applied Science Research, 14(1), 12734–12739. doi:10.48084/etasr.6681.

[11] Kuran, U., & Kuran, E. C. (2021). Parameter selection for CLAHE using multi-objective cuckoo search algorithm for image contrast enhancement. Intelligent Systems with Applications, 12. doi:10.1016/j.iswa.2021.200051.

[12] Tang, X., & Rashid Sheykhahmad, F. (2024). Boosted dipper throated optimization algorithm-based Xception neural network for skin cancer diagnosis: An optimal approach. Heliyon, 10(5), e26415. doi:10.1016/j.heliyon.2024.e26415.

[13] Shinde, P., & Ingle, Y. (2024). Skin Cancer Detection: A Review Using Machine Learning Techniques. Asian Journal of Research in Computer Science, 17(2), 15–26. doi:10.9734/ajrcos/2024/v17i2416.

[14] Chakkarapani, V., & Poornapushpakala, S. (2024). Chronological Dingo Optimizer-based Deep Maxout Network for skin cancer detection and skin lesion segmentation using Double U-Net. Multimedia Tools and Applications, 83, 71235–71263. doi:10.1007/s11042-024-18229-5.

[15] Likhar, K., & Ridhorkar, S. (2024). Enhancing Skin Cancer Detection: A Comparative Analysis of Models with VGG-16, VGG-19, and Inception V3. International Journal of Intelligent Systems and Applications in Engineering, 12(10s), 502–514.

[16] Moturi, D., Surapaneni, R. K., & Avanigadda, V. S. G. (2024). Developing an efficient method for melanoma detection using CNN techniques. Journal of the Egyptian National Cancer Institute, 36(1). doi:10.1186/s43046-024-00210-w.

[17] Sharma, P., & Suji, J. (2016). A Review on Image Segmentation with its Clustering Techniques. International Journal of Signal Processing, Image Processing and Pattern Recognition, 9(5), 209–218. doi:10.14257/ijsip.2016.9.5.18.

[18] Zhao, S., Gu, C., Yu, J., Akashi, T., & Zhang, C. (2024). TriClick: Interactive Dermoscopic Image Segmentation with Triangle Map. IEEJ Transactions on Electrical and Electronic Engineering, 19(5), 733–744. doi:10.1002/tee.24020.

[19] Abualigah, L., Al-Okbi, N. K., Awwad, E. M., Sharaf, M., & Daoud, M. S. (2024). Correction to: Boosted Aquila Arithmetic Optimization Algorithm for multi-level thresholding image segmentation. Evolving Systems, 15(4), 1399-1427. doi:10.1007/s12530-024-09576-7.

[20] Ram, P., & Padmavathi, S. (2017). Analysis of Harris corner detection for color images. In International Conference on Signal Processing, Communication, Power and Embedded System, SCOPES 2016 - Proceedings. doi:10.1109/SCOPES.2016.7955862.

[21] Haggui, O., Tadonki, C., Lacassagne, L., Sayadi, F., & Ouni, B. (2018). Harris corner detection on a NUMA manycore. Future Generation Computer Systems, 88, 442–452. doi:10.1016/j.future.2018.01.048.

[22] Zhou, P., Zhang, H., & Liang, W. (2023). Research on hybrid intrusion detection based on improved Harris Hawk optimization algorithm. Connection Science, 35(1), 2195595. doi:10.1080/09540091.2023.2195595.

[23] Li, Y. B., & Li, J. J. (2011). Harris corner detection algorithm based on improved contourlet transform. Procedia Engineering, 15, 2239–2243. doi:10.1016/j.proeng.2011.08.419.

[24] Han, X., Wang, X., Leng, Y., & Zhou, W. (2021). A plane extraction approach in inverse depth images based on region-growing. Sensors (Switzerland), 21(4), 1–15. doi:10.3390/s21041141.

[25] Adams, R., & Bischof, L. (1994). Seeded Region Growing. IEEE Transactions on Pattern Analysis and Machine Intelligence, 16(6), 641–647. doi:10.1109/34.295913.

[26] Poux, F., Mattes, C., Selman, Z., & Kobbelt, L. (2022). Automatic region-growing system for the segmentation of large point clouds. Automation in Construction, 138. doi:10.1016/j.autcon.2022.104250.

[27] Arjun, K. P., & Kumar, K. S. (2022). A combined approach of VGG 16 and LSTM transfer learning technique for skin melanoma classification. International Journal of Health Sciences, 13504–13516. doi:10.53730/ijhs.v6ns1.8378.

[28] Jiang, L., Sun, X., Mercaldo, F., & Santone, A. (2020). DECAB-LSTM: Deep Contextualized Attentional Bidirectional LSTM for cancer hallmark classification. Knowledge-Based Systems, 210. doi:10.1016/j.knosys.2020.106486.

[29] Roy, S. S., Awad, A. I., Amare, L. A., Erkihun, M. T., & Anas, M. (2022). Multimodel Phishing URL Detection Using LSTM, Bidirectional LSTM, and GRU Models. Future Internet, 14(11). doi:10.3390/fi14110340.

[30] Greff, K., Srivastava, R. K., Koutnik, J., Steunebrink, B. R., & Schmidhuber, J. (2017). LSTM: A Search Space Odyssey. IEEE Transactions on Neural Networks and Learning Systems, 28(10), 2222–2232. doi:10.1109/TNNLS.2016.2582924.

[31] Sherstinsky, A. (2020). Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network. Physica D: Nonlinear Phenomena, 404. doi:10.1016/j.physd.2019.132306.

[32] Wang, Y., & Zhang, W. (2021). A Dense RNN for Sequential Four-Chamber View Left Ventricle Wall Segmentation and Cardiac State Estimation. Frontiers in Bioengineering and Biotechnology, 9(August), 1–8. doi:10.3389/fbioe.2021.696227.

[33] Müller, D., Hartmann, D., Meyer, P., Auer, F., Soto-Rey, I., & Kramer, F. (2022). MISeval: A Metric Library for Medical Image Segmentation Evaluation. In Studies in Health Technology and Informatics, 294, 33–37. doi:10.3233/SHTI220391.

[34] Li, J., Udupa, J. K., Tong, Y., Wang, L., & Torigian, D. A. (2021). Segmentation evaluation with sparse ground truth data: Simulating true segmentations as perfect/imperfect as those generated by humans. Medical Image Analysis, 69. doi:10.1016/j.media.2021.101980.

[35] Behura, A. (2021). Congruence of Deep Learning in Medical Image Processing: Future Prospects and Challenges. Studies in Computational Intelligence, 936, 197–221. doi:10.1007/978-981-33-4698-7_10.

[36] Singh, C., Nischitha, Shetty, S. S., Bekal, A., Bhat, S., & Badiger, M. (2024). Deep Learning Analysis for Skin Cancer Detection. Lecture Notes in Electrical Engineering: Vol. 1062 LNEE, 159–171. doi:10.1007/978-981-99-4444-6_12.

[37] Obaid, A. M., Shawkat, A. S., & Abdulhussein, N. S. (2024). A powerful deep learning method for skin cancer detection. Journal of Autonomous Intelligence, 7(1), 1156. doi:10.32629/jai.v7i1.1156.

[38] Lu, Y., Cheung, Y. M., & Tang, Y. Y. (2020). Bayes Imbalance Impact Index: A Measure of Class Imbalanced Data Set for Classification Problem. IEEE Transactions on Neural Networks and Learning Systems, 31(9), 3525–3539. doi:10.1109/TNNLS.2019.2944962.

[39] Zhang, X., Li, R., Zhang, B., Yang, Y., Guo, J., & Ji, X. (2019). An instance-based learning recommendation algorithm of imbalance handling methods. Applied Mathematics and Computation, 351, 204–218. doi:10.1016/j.amc.2018.12.020.

[40] Shilpa Kamdi, R. (2012). Image Segmentation and Region Growing Algorithm. International Journal of Computer Technology and Electronics Engineering, 2(1), 103–107.

[41] Ali, E., Ullah Khan, E., Zarrar Mahmudi, E., & Ullah, R. (2016). A Comparison of FAST, SURF, Eigen, Harris, and MSER Features. International Journal of Computer Engineering and Information Technology, 8(6), 100–105.

[42] Gulzar, Y., & Khan, S. A. (2022). Skin Lesion Segmentation Based on Vision Transformers and Convolutional Neural Networks—A Comparative Study. Applied Sciences (Switzerland), 12(12), 5590. doi:10.3390/app12125990.

[43] Vishnu Priya, A., Singh, H. K., Siva Chaitanya Prasad, M., & Jai Siva Sai, G. (2023). RNN-LSTM Based Deep Learning Model for Tor Traffic Classification. Cyber-Physical Systems, 9(1), 25–42. doi:10.1080/23335777.2021.1924284.

[44] Hayati, M., Muchtar, K., Roslidar, Maulina, N., Syamsuddin, I., Elwirehardja, G. N., & Pardamean, B. (2022). Impact of CLAHE-based image enhancement for diabetic retinopathy classification through deep learning. Procedia Computer Science, 216, 57–66. doi:10.1016/j.procs.2022.12.111.

[45] Imtiaz, I., Ahmed, I., Ahmad, M., Ullah, K., Adnan, A., & Ahmad, M. (2019). Segmentation of Skin Lesion Using Harris Corner Detection and Region Growing. 2019 IEEE 10th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference, UEMCON 2019, 0614–0619. doi:10.1109/UEMCON47517.2019.8993034.

[46] Gowthami, V., & Sneha, G. (2021). Melanoma Detection Using Recurrent Neural Network. Lecture Notes in Electrical Engineering, 700, 1563–1573. doi:10.1007/978-981-15-8221-9_146.

[47] Asyhar, A. H., Foeady, A. Z., Thohir, M., Arifin, A. Z., Haq, D. Z., & Novitasari, D. C. R. (2020). Implementation LSTM Algorithm for Cervical Cancer using Colposcopy Data. 2020 International Conference on Artificial Intelligence in Information and Communication, ICAIIC 2020, 485–489. doi:10.1109/ICAIIC48513.2020.9065068.

[48] Jain, A., Mittal, N., & Nain, S. (2023). CNN-based Recognition of Skin Cancer Using Contrast Limited Adaptive Histogram Equalization. Lecture Notes in Networks and Systems: Vol. 617 LNNS, 667–678. doi:10.1007/978-981-19-9512-5_61.

[49] Ray, P. J., Priya, S., & Kumar, T. A. (2015). Nuclear segmentation for skin cancer diagnosis from histopathological images. Global Conference on Communication Technologies, GCCT 2015, 397–401. doi:10.1109/GCCT.2015.7342692.

[50] Patra, A., Behera, S. K., & Barpanda, N. K. (2022). Hybrid deep CNN-LSTM network for breast histopathological image classification. Onkologia i Radioterapia, 16(9), 12–15.

[51] Limanto, S., Buliali, J. L., & Saikhu, A. (2024). GLoW SMOTE-D: Oversampling Technique to Improve Prediction Model Performance of Students Failure in Courses. IEEE Access, 12, 8889–8901. doi:10.1109/ACCESS.2024.3351569.