A Novel end-to-end Deep Convolutional Neural Network based Skin Lesion Classification Framework

Razia Sulthana A
Department of Computing and Mathematical Sciences, University of Greenwich, Old Royal Naval College London, United Kingdom
razia.sulthana@greenwich.ac.uk

Vinay Chamola
Department of EEE and APPCAIR BITS-Pilani, Pilani Campus India
vinay.chamola@pilani.bits-pilani.ac.in

Amir Hussain
Director, Centre for AI and Robotics (CAIR) School of Computing, Edinburgh Napier University Scotland, United Kingdom
A.Hussain@napier.ac.uk

Zain Hussain
Edinburgh Medical School College of Medicine and Veterinary Medicine, University of Edinburgh Edinburgh, United Kingdom
zain.hussain@ed.ac.uk

Faisal Albalwy
Department of Computer Science College of Computer Science and Engineering Taibah University, Madinah Saudi Arabia
Division of Informatics, Imaging and Data Sciences, School of Health Sciences The University of Manchester, Oxford Road, Manchester, United Kingdom
faisal.albalwy@manchester.ac.uk

December 30, 2023

Abstract

Background: Skin diseases are reported to contribute 1.79% of the global burden of disease. The accurate diagnosis of specific skin diseases is known to be a challenging task due, in part, to variations in skin tone, texture, body hair, etc.

Classification of skin lesions using machine learning is a demanding task, due to the varying shapes, sizes, colors, and vague boundaries of some lesions. The use of deep learning for the classification of skin lesion images has been shown to help diagnose the disease at its early stages. Recent studies have demonstrated that these models perform well in skin detection tasks, with high accuracy and efficiency.

Objective: Our paper proposes an end-to-end framework for skin lesion classification, and our contributions are two-fold. Firstly, two fundamentally different algorithms are proposed for segmenting and extracting features from images during image preprocessing. Secondly, we present a deep convolutional neural network model, S-MobileNet that aims to classify 7 different types of skin lesions.

Methods: We used the HAM10000 dataset, which consists of 10000 dermatoscopic images from different populations and is publicly available through the International Skin Imaging Collaboration (ISIC) Archive. The image data was preprocessed to make it suitable for modeling. Exploratory data analysis (EDA) was performed to understand various attributes and their relationships within the dataset. A modified version of a Gaussian filtering algorithm and SFTA was applied for image segmentation and feature extraction. The processed dataset was then fed into the S-MobileNet model. This model was designed to be lightweight and was analysed in three dimensions: using the Relu Activation function, the Mish activation function, and applying compression at intermediary layers. In addition, an alternative approach for compressing layers in the S-MobileNet architecture was applied to ensure a lightweight model that does not compromise on performance.

Results: The model was trained using several experiments and assessed using various performance measures, including, loss, accuracy, precision, and the F1-score. Our results demonstrate an improvement in model performance when applying a preprocessing technique. The Mish activation function was shown to outperform Relu. Further, the classification accuracy of the compressed S-MobileNet was shown to outperform S-MobileNet.

Conclusions: To conclude, our findings have shown that our proposed deep learning-based S-MobileNet model is the optimal approach for classifying skin lesion images in the HAM10000 dataset. In the future, our approach could be adapted and applied to other datasets, and validated to develop a skin lesion framework that can be utilised in real-time.

Keywords: Skin lesion, Image Segmentation, Classification, Deep learning, Convolution Neural Network, MobileNet

1 Introduction

The skin is the body's largest organ and consists of three layers: the epidermis (outermost layer), dermis, and hypodermis (innermost layer). Some common skin diseases include acne, eczema, psoriasis, lesions, and skin cancer [1]. Skin lesions are unusual patches or bumps on the skin and can be categorized into 3 classes: those formed by fluids, those which are solid-like masses in the skin, and those which are flat, like rashes. Manual detection and classification of skin lesions is challenging because of the varying shapes, sizes, colors, and vague boundaries of lesions. There are a number of studies in the literature on image segmentation using deep learning (DL) algorithms and artificial intelligence (AI) approach, and transfer learning, to detect and classify skin lesions in a timely manner, before they become fatal [2], [3].

This paper aims to build an end-to-end deep convolutional neural network (D-CNN) framework to classify skin lesion images. The proposed S-MobileNet D-CNN model is a modified version of the MobileNet architecture [4]. The clinical image dataset used in this work is HAM10000 [5]. The skin lesion images in the dataset are distributed among 7 classes: Melanocytic nevi, Melanoma, Benign keratosis-like lesions, Basal cell carcinoma, Actinic keratoses, Vascular lesions, and Dermatofibroma. The skin lesion images were pre-processed before being fed into the neural network (NN) model. Image preprocessing techniques viz. image segmentation and feature extraction were carried out using the proposed algorithms to identify the latent clean image from the hidden layers. Exploratory data analysis (EDA) and hypothesis formulation was carried out to obtain a better understanding of the data. An additional analysis of the relationship between various attributes was conducted to improve prediction. The dataset is split into train and test sets for training and validation in the ratio of 80:20. The S-MobileNet model is built, trained, and evaluated using various performance measures like accuracy, precision, and F1 score. The main contributions of the paper include

1. Two new algorithms are proposed for image segmentation and feature extraction respectively. The former segments images by analysing the pixels by constructing a threshold and is more effective in removing noise from the image. The latter is a modified version of the Segmentation-based Fractal (SFTA) that determines the texture pattern from the image.

2. A D-CNN S-MobileNet is built to extract low-level features of the image and to automatically classify skin lesion images into 7 classes of disease.

3. The proposed S-MobileNet model is fine-tuned and analysed using the Relu and Mish Activation functions. Hyperparameters are fine-tuned to improve the model's performance.

4. A lightweight S-MobileNet model is built by altering the architecture of the model and by compressing the intermediary layers to enhance the classification performance.

The paper begins with a general introduction about the types of skin cancer, an explanation of the problem statement, and the main contributions. In section 2, a literature survey is presented with a detailed analysis of existing machine learning approaches and image preprocessing techniques. Section 3 details the algorithm used for image segmentation and feature extraction, methodology, and architecture of the proposed S-MobileNet. Section 4 describes the implementation setup, the experiments, and the corresponding results. Finally, the paper is concluded in section 5.

2 Background

A number of custom-made models are proposed by researchers in recent years in relation to skin lesion classification and prediction. Some of them related to the proposed problem statement are briefed in Table 1. The Table aggregates the algorithms and performance metrics used in the latest years to classify skin lesion images.

2.1 Machine Learning approaches

Machine learning algorithms are used for image classification across a number of applications [2]. A specific focus of this section is on image-based methods of classifying skin diseases. In [6], as an initial preprocessing step, the image data is rescaled, resized, and then classified using Naive Bayes, k-Nearest Neighbour (KNN), Support Vector Machines (SVM), Neural Network(NN). In spite of the high classification accuracy of the model, prediction accuracy is a major concern. Self diagnosis of skin diseases is introduced in [7] which uses image transformation techniques like Discrete Cosine Transform (DCT), Discrete Wavelet Transform (DWT), and Singular Value Decomposition (SVD). A comparison of all the image analysis techniques is made and an ensemble transformation technique is proposed by combining all three. This approach is found to be faster in diagnosing the skin disease.

A semi-supervised Computer-Aided System (CAD) [8] for Psoriasis image classification uses both unsupervised and supervised image classification techniques. It builds a dictionary for sparse image classification using aggregation methods deployed over local features in an image. Multi-class machine learning classification techniques like Random Forest (RF), SVM, and AlexNet are applied for severity score calculation. A detailed analysis of pre-trained networks is applied in [9] for recognizing 20 skin abnormalities. The performance of the different models is compared by generating the confusion matrix and accuracy. However, the highest accuracy score is generated by an ensemble model. Inspired by this work, the author in [10] used the median filtration technique to remove the noise from the image. Following this sobel edge detection is applied to detect the edges of the images. The result shows an increase in entropy value. Yet, ways to extract the discriminative features from the image still remain fuzzy.

A mass of varying architectures of Convolution Neural Network (CNN) [11–15] is applied in many research articles for skin image segmentation and classification. The segmentation techniques, in general, help to minimize the distortions from the images and improves the accuracy of classification. Although the impact of segmentation in the classification of images has been explored partially in the literature work cited above, there is still a lot of room for further research. D-CNN networks are also applied for building learning models in [16–18] to classify skin lesion images. Gradient boosting is applied to classify around 300 heterogeneous skin lesions into 5 categories in [16] and the results of the machine learning algorithm are intervened by human interest and are cross analysed to produce an accuracy of 82.95%. Five custom-designed deep learning CNN models are proposed in by varying the convolution blocks, pooling blocks, and dropout blocks. In addition, the model is tuned to bring the optimal results by modifying the activation functions, and tuning the hyperparameters and so there is limited evidence for considering this model as the de-facto standard.

Melanoma-affected skin lesion images are classified in multi-stage by analyzing their pixels at the fine level using enhanced encoder-decoder feature map [18]. This approach compares and classifies images in real-time using three segmentation techniques with a minimal number of training parameters and resources. A real-time algorithm using Generative Adversarial Network (GAN) to detect melanin and sebum from skin images is proposed in [19]. In the first step, grayscale images are converted into black and white and enhanced before being passed on to the UNet architecture. Failing to clearly identify melanin and sebum in the same image is one drawback of this approach.

The Lyme skin infection is identified using the skin images taken from the EM image dataset [20] and applied to the HAM10000 dataset using the transfer learning approach in [21]. This study applies twenty-three well-known CNN architectures and confirms a lightweight CNN model to be very effective and useful in classifying the images. Additionally, the extended study in [22] by the same author proposes a customised ResNet for classifying the skin images. In a way similar to this, the proposed method creates a lightweight, customised S-MobileNet to classify the skin lesion images in HAM10000.

Another article, [23] introduces a customised CNN with compression complexity pooling as compared to the conventional pooling technique. The pooling technique extracts the spatial features from the image by generating relatively complex feature maps. The experimental results show the results of object detection with a number of cropped and resized CNNs. Hence, in order to create a model that is appropriate for the given dataset in the domain or across domains, the convolution, pooling, and flattening layers of CNN and the operations carried out in them can be modified in correspondence with the application. Similarly, the layers of the proposed S-MobileNet are pruned, customised and the activation functions are modified in accordance to attain the proposed objective.

2.2 Image Segmentation and Feature Extraction algorithms

The classification of skin images is been automated in recent years and a number of researchers have proposed Automated Classification Methods (ACM) [31]. In general, a study of skin image analysis includes preprocessing of images; image segmentation; image feature extraction, and image classification. Image preprocessing [32] comprises a variety of procedures: Downsampling, space transformations, contrast adjustments, normalization, and artifact removal [33]. The accuracy of image classification totally

Table 1: Literature related to the proposed problem statement

Ref	Year	Objective	Algorithms used	Performance metrics
[7]	2017	Classification of Skin diseases	DCT, DWT, SVD	Accuracy
[8]	2018	Comparative study of the proposed system with AlexNet and other CNN models	RF, SVM, Boosting	F1 Score
[9]	2018	Propose a computer vision approach to differentiate and recognize 20 skin abnormalities with increased accuracy	Inception_v3, MobileNet, Resnet, xception, RF, and LR	Accuracy
[10]	2018	Apply segmentation and filtering techniques to classify skin diseases. Provides a visualization of the skin images for improved identification and classification of skin lesions	Sobel Edge Detection, Median Filtering	Entropy
[11]	2018	Proposes a deep learning model called FrCN for skin lesion segmentation analysis	Full Resolution Convolutional Networks	Jaccard Index, Accuracy
[12]	2019	Classification of skin images	k-means Clustering, Morphology-based image segmentation	Signal to Noise Ratio
[13]	2019	Classification of skin images	RF, Naive Bayes, LR, Kernel SVM, and CNN	Accuracy and Error rate
[14]	2019	Classification of skin images	CNN and SVM	Accuracy
[15]	2019	Classification of skin images	CNN with regularization algorithms like Lasso	Accuracy and ROC curve
[16]	2019	Combines human intelligence with Artificial Intelligence to classify skin images	Deep learning, CNN	Accuracy
[17]	2019	Classification of Skin diseases	Deep learning model	ROC Curve
[18]	2020	Classification of skin images and build an automatic grouping system of the skin diseases	CNN	Accuracy
[24]	2020	Builds a system based on D-CNN with a supervised encoder-decoder network to recognize and differentiate between melanoma and non-melanoma lesions.	D-CNN	Accuracy
[25]	2020	Creates a web application for diagnosis of skin lesion	D-CNN	Accuracy
[26]	2020	Build a Deep Learning model for classification of skin images	Mask R-CNN and DeeplabV3+	Sensitivity and Specificity
[27]	2020	Build a deep learning model with an ensemble model with U-Net and ResNet	CNN and Transfer Learning	Jaccard Index, Accuracy
[28]	2021	Investigates a number of deep learning and transfer learning models in classification of skin diseases	7 layered deep CNN	Accuracy
[29]	2021	Classification of skin diseases and a comparative study of the proposed D-CNN with other transfer learning models	D-CNN	Accuracy
[30]	2021	Uses Deep Learning and Artificial Intelligence to build an automated system for skin disease	AI and CNN	Accuracy

depends upon the algorithms used in these stages.

In the process of segmentation, the infected area is extracted from the dermoscopy image [27] and the segmentation process is carried out in three ways: Pixel-based segmentation, Region-based segmentation, Edge-based segmentation [34, 35]. In addition to the aforementioned methods, clustering-based segmentation and threshold-based segmentation are also proposed in literature [36]. The study of inter-relation between pixels in the Region Of Interest (ROI) of an image, facilitates proper segmentation. The authors in [37] apply edge-based segmentation to find the rapid change in the intensity of the pixels in an image. The color, texture, and contours of the image are figured during the segmentation process. While another segmentation approach that analyses every pixel and classifies each of them to a specific class label named semantic segmentation is applied in [38]. Performance is measured in ResNet with the VGG model. The model appears to be too complex with large parameters that consume more time and memory.

Feature extraction is the process of converting the image into numerical values. It analyzes the color, texture, shape, and other qualities of the image. Numerous feature extraction methods is been explored in literature like Gray Level Co-occurrence Matrix (GLCM), Local Binary Patterns, Bag of features, etc [39]. The extracted features are subjected to correlation analysis, homogeneity, and entropy analysis and further transformed in [40] for increasing the predicting accuracy. In another relevant article [41] an UNet CNN architecture is proposed over the ISIC 2018 skin lesion dataset that fuses image segmentation with feature maps for segmentation. A five cross-validation is applied and the performance is measured. Alternatively, the model is characterized as computationally complex because of the large number of parameters.

According to one of the related works, [20] which investigates the effect of frequency bias in CNN image classification, a number of challenges prevent CNN from accurately extracting the features when used alone. So it employs the use of the Gaussian kernel function and suggests using feature discrimination in addition to CNN. Following this direction, the suggested work applies a segmentation method (modified Gaussian filtering) and feature extraction method (modified version of Segmentation-based Fractal (SFTA)) before moving on to the CNN model, which significantly increases accuracy as shown in 4. Image histogram, filtration and k-NN classifier is applied in on images of human finger that is injured. Images of injured finger captured by mobile camera is processed after 60h, 160hr and 450 hrs of injury. This research work consumes minimal cost in capturing the image and processing it in real time.

The challenges in segmenting lesion in skin images is detailed in [43] which notifies that the color, texture, shape, hairs, veins and light reflections might add noise or decrease the segmentation accuracy. The author applies the superpixel segmentation, L2 normalization and captures the variations in the superpixels using autoencoders. Given that L1 norm is quite robust than L2 norm pruning [44], we apply L1 norm pruning while designing the CNN model in our proposed system.

An end to end CNN is built using image segmentation techniques for edge prediction in [45] to classify the skin lesion images. The deep CNN is customised and integrated with modules to identify and highlight lesion boundaries for effective segmentation. Edge detection being a pivotal element, in the proposed system, we use homogenity predicates that distinguishes the change in the pixel color and gradients across the edges and effectively segments the lesion. Section 3 discusses in detail the proposed system.

2.3 Dataset

Human Against Medicine with 10000 training images (HAM10000) [46] is an archive of dermoscopic images from varying populations across the world Fig.1. The HAM10000 Dataset is cleaned to remove the ambiguous images as some of the images are similar but shown in different magnifications and angles. Around 50% of the image, lesions are taken from histopathology reports and from expert's microscopical examination. The images in the dataset are tracked using the metadata file. The HAM10000 dataset is used for skin lesion classification in [47], [48], [49], [50], [51] and the performance of the proposed Deep CNN framework is compared with the aforementioned state of the art approaches.

The dataset includes images from 7 different categories of skin disease. In order to avoid bias, an equal number of images are taken from each of the 7 classes using a random

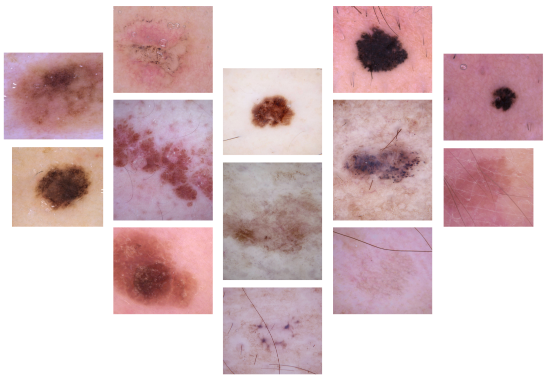


Figure 1: Images from Dataset

image generator.

3 Materials and Methods

Machine Vision is generally categorized into low-level and high-level vision. In the low-level vision, image processing operations are performed to produce another new image with minimal noise and edges being enhanced. On the other hand, high-level vision tries to perform object recognition and scene interpretation. Both of these are connected using the segmentation process. The block architecture of the end-to-end Deep CNN is shown in Fig. 2. The first block, displays the input, an image of the skin, is followed by two blocks for image segmentation and feature extraction. These are followed by the blocks of the customised S-MobileNet. Each block of customised S-MobileNet indicates a layer in the CNN architecture and the two coloured (yellow and green) block signifies the depthwise and pointwise convolution operations in the CNN layer. Also mentioned are the layer's filters and pruning percentages.

3.1 Proposed Image Segmentation Technique

As the first step in the proposed system, a modified version of the Gaussian Filtering [52] algorithm is used for pixel-level segmentation of images. An ideal segmentor segments

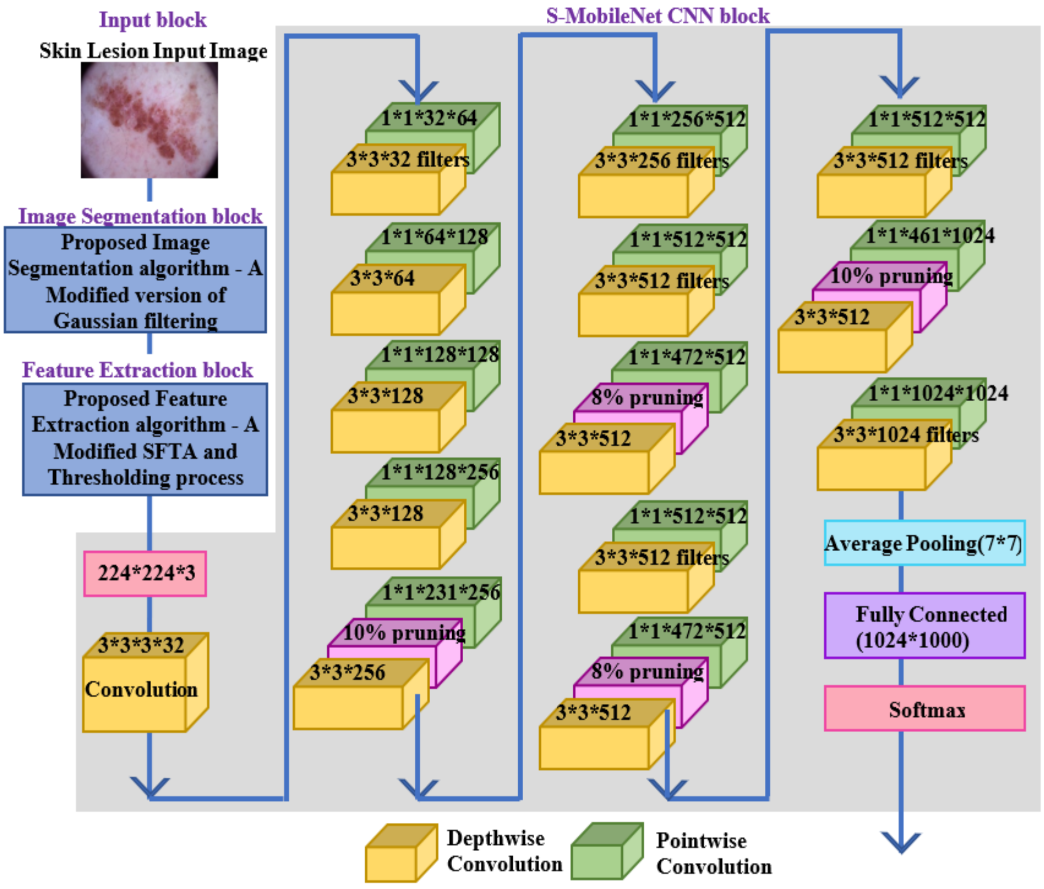


Figure 2: Architecture of the proposed end-to-end Deep CNN framework

regions that are more uniform in texture, and homogeneous in gray tone, the boundaries of the segmented image would be smooth and there would be a significant difference in values between pixels of adjacent regions. Contemporary Gaussian filtering is effective at eliminating noise from images, and it outperforms median filtering in terms of effectiveness. In [53], the author compares Gaussian filtering, median filtering, and denoise autoencoding to three performance measures, such as Normalization Mean Square Error, Structure Similarity, and Peak Signal to Noise Ratio, demonstrating that a Gaussian filter produces better results in a shorter period of time than two other filtering methods.

A sample of the result of the gaussian filtered images is shown in Fig.3.

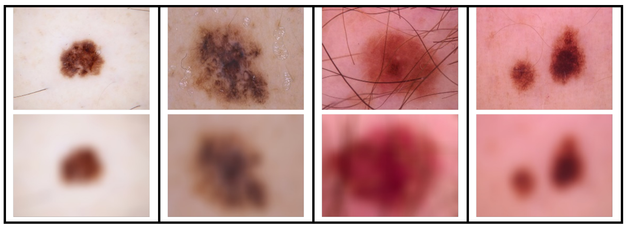


Figure 3: Original image and Gaussian filtered image

Let A be some sample collection of pixels and *N()* be the homogeneity predicate on the connected pixels. The homogeneity predicate specifies the property that the pixels are homogenous or uniform. This property is set to true for regions that are similar in color or edge gradient. Mathematically the initial setup of pixel arrangements within regions can be represented below:

1. The segmentation of A is simply the partitioning of the image into regions $\{Re_1, Re_2, \ldots., Re_x\}$ s.t ${}^{x}_{i=1} Re_i = A$ wherein $Re_i \bigcap Re_b = \emptyset \ \forall \ i \neq b$. he homogenity predicate satisfies the condition $Prob(Re_i) = TRUE \ \forall \ i$. n addition, it is essential that the homogenity predicate satisfies rob($\mathrm{Re}_i \cup Re_b) = FALSE, \ \forall \ Re_i \ being \ near \ to \ R_b$ nd $(\mathrm{Re}_i \supset \mathrm{Re}_b) \wedge (\mathrm{Re}_b \neq \emptyset) \wedge (Prob(\mathrm{Re}_i) = TRUE) \implies Prob(\mathrm{Re}_b) = TRUE$

Image segmentation is an ad hoc property and is applied based on certain requirements like the nature of the image, size of the region, etc. Often, the segmenter is applied at the cost of other properties of the image. Disturbances like noise shatter the uniformity in the image and fragment the segmentation results. Especially in large regions, noise considerably disturbs the segmentation result. The fourth indices mentioned above states that a large region is considered to be uniform or noise-free if its subsets are uniform.

The proposed segmentation technique applies a modified version of the Gaussian filtering algorithm for pixel-level segmentation. In this approach, the pixels are classified based on the gray levels and uniformity of the pixels. Owing to the fact that gray-level images are very supportive [54] in image classification and hence segmentation, the skin images are converted to gray tones to improve the accuracy of the model. The properties of grayscale images: hue, saturation, and brightness enhance the correctness of classification as compared to RGB.

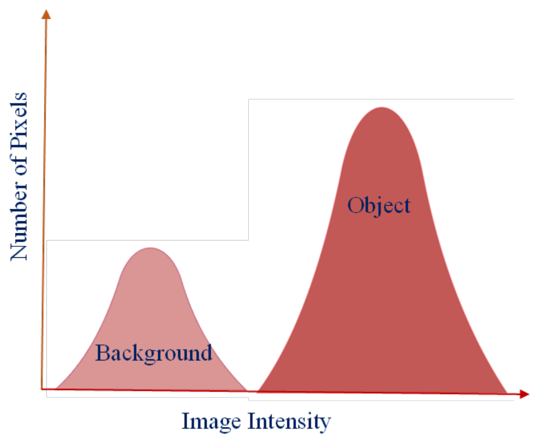


Figure 4: Gaussian filtering threshold level

A specified threshold value captures the line of separation between the two modes: the gray level of the objects and the gray level of the background pixels. The modified Gaussian filtering algorithms roots in the Bayes algorithm. It analyzes the pixel density of the object pixels in the foreground as well as the background pixels. Let there be an image with object level and background level classified into dominant modes.

The threshold $u\ (a, b)$ of the image $v\ (a, b)$ is given as $(a, b) = 1\ if\ (a, b) > Thresholdif\ (a, b) \leq threshold$

$Thisresultantwouldbeabinaryimageandasampleofitisshownin Fig.3.$

Algorithm 1 details the proposed segmentation approach. It constructs the histogram to classify the pixels between the object and the background of the image. More importantly, an important feature of this proposed algorithm is that the histogram is smoothened using the moving average as the smoothing function to minimize the error while interpreting the results. The resultant of the algorithm would be a smoothened histogram suitable for finding the threshold value. The central theme of classification is to apply the threshold value in classifying the pixels (explained in Algorithm 2). Algorithm 2 identifies the deep valley in the histogram and picks every pixel p and marks it to either of the two classes. The segmented skin lesion image is passed on to the next level for feature extraction.

3.2 Proposed Feature Extraction technique

The critical aspect of image preprocessing is feature extraction. Analyzing the texture of the skin lesion images gives a better understanding of whether the infected region is swollen/bulged or is built with dead cells or has been rugged. Many researchers have been predominantly using feature extraction in medical images for understanding the patterns in an image. Nevertheless, the same feature extraction techniques cannot be applied to all kinds of images, such as character recognition or object detection, since each of them is unique. In the proposed work, a modified version of Segmentation-based Fractal (SFTA) [55] is applied to break the components of the image into smaller fractions and to determine the texture and other patterns in them.

As stated earlier, the texture underpins the semantic nature of the image and hence classifying images considering the surrounding texture would yield greater accuracy. Especially when it comes to skin images, the hair over the skin is always an obstacle in detecting its texture. Regardless of it, the proposed approach is efficient in capturing the granularity structure of the image.

The binary version of the grayscale image is represented as I_{binary} which is obtained by applying the threshold $u(a, b)$ on the image (from Section 3.1). As the first step, a binary threshold image filter is applied to the image with two threshold values as input. Let $threshold_{lower}$ and $threshold_{upper}$ be the lower and upper threshold values. The threshold values are chosen between [0.0,1.0] and are very influential in extracting patterns from the image. There is limited evidence from the existing work that a randomly chosen threshold would increase the efficiency of feature extraction. Nevertheless, in this

Algorithm 1 Proposed Segmentation Algorithm - Building the Histogram - Part 1

Assumption: The density function of object level pixels and background level pixels are gaussian in nature.

```
1. Construction of the histogram (H) to differentiate the density of the pixels in the object vs the pixels in the background of the image
2. Let mean of the histogram be $H_\mu$, standard deviation of the histogram be $H_\sigma$ and number of chosen gray levels or gray level resolution be $G\_L$, given that $G\_L = 2^{bpp}$ where bpp is the number of $\frac{bits}{pixel}$
$H_\mu = \frac{1}{X}\sum H(i) * i$
$H_\sigma = \sqrt{\frac{1}{X} \sum H(i) * \ (i - H_\mu)^2}$
Here, $H(i) \longrightarrow Histogram\ for\ gray\ level\ i$
$X \longrightarrow Number\ of\ pixels\ in\ window$
Alternatively $X$ can take up values either 0 or 1 representing number of pixels. If $X$ takes up a value 0 in the window, it leads to undefined situation and hence a non negative constant $c$ is added to $H_\sigma$
$H_\sigma = \sqrt{\frac{1}{X} \sum H(i) * \ (i - H_\mu)^2} + c$
3. Minimize the sum of the square of the offset of the below equation in against to the $H(i)$ by altering the parameters included in it.
$h(i) = \frac{N_1}{H_{\sigma 1}} e^{-\frac{(i-H_{\mu 1})^2}{2H_{\sigma 1}^2}} + \frac{N_2}{H_{\sigma 2}} e^{-\frac{(i-H_{\mu 2})^2}{2H_{\sigma 2}^2}}$
4. The next step is to alter the bins of the histogram and smoothen it using the below equation. The smoothened histogram is analysed to find the deep valley ($d\_v$) and that is considered to be the threshold to partition the histogram.
Whilst the smoothening can be done in two steps:
(i)By using an moving average function $W_F = \frac{1}{(2M+1)^2}$
$H'(i) = \frac{1}{2M+1}\sum_{k=1}^{2M+1} H\,(i)$
(ii)By using local weighted average
$H^i(i) = \frac{H(i-2)+2H(i-1)+3H(i)+2H(i+1)+H(i+2)}{constant}$
5. The $d\_v$ in the histogram is taken up for dividing the $H(i)$ into two histograms. The initial values of the parameters are given below:
$X_1 = \sum_{i=1}^{d\_v} H(i)$

$X_2 = \sum_{i=d\_v+1}^{G\_L} H(i)$

$H_{\mu 1} = \frac{1}{X_1}\sum_{i=1}^{d\_v} H(i) * i$

$H_{\mu 2} = \frac{1}{X_2}\sum_{i=d\_v+1}^{G\_L} H(i) * i$

$H_{\sigma 1} = \sqrt{\frac{1}{X_1}\sum_{i=1}^{d\_v} H(i)(i - H_{\mu 1})^2}$

$H_{\sigma 2} = \sqrt{\frac{1}{X_2}\sum_{i=d\_v+1}^{G\_L} H(i)(i - H_{\mu 2})^2}$
```

Algorithm 2 Proposed Segmentation Algorithm - Classifying the Pixels - Part 2

```
1. Deepest valley calculation in histogram
$min \sum_{i=1}^{G\_L} [h(i) - H(i)]^2$
2. For i being the deepest valley
$value = |h(i) - H(i)|$
LABEL: Value Calculation:
$left\_value = |h(i-1) - H(i-1)|$
$right\_value = |h(i+1) - H(i+1)|$

  if $left\_value \leq value$ then
      $deepest\ value\ at\ i-1$
  else if $right\_value \leq val$ then
      $deepest\ value\ at\ i+1$
  else
      $deepest\ value\ is\ at\ i$
  end if
For any change in $value$ repeat to calculate new values for $N_1$, $N_2$, $H_{\mu1}$, $H_{\mu2}$, $H_{\sigma1}$, $H_{\sigma2}$
using Algorithm 1 and reestimate Value Calculation.
3. Let $p$ be a random gray pixel taken from the image. The pixel is allotted to the
object if
$\frac{N_1}{H_{\sigma1}} e^{-\frac{(p-H_{\mu1})^2}{2H_{\sigma1}^2}} > \frac{N_2}{H_{\sigma2}} e^{-\frac{(p-H_{\mu2})^2}{2H_{\sigma2}^2}}$
4. The threshold value is defined when
$\frac{N_1}{H_{\sigma1}} e^{-\frac{(thresholdvalue-H_{\mu1})^2}{2H_{\sigma1}^2}} = \frac{N_2}{H_{\sigma2}} e^{-\frac{(thresholdvalue-H_{\mu2})^2}{2H_{\sigma2}^2}}$
where both the error equation are equal.
5. And threshold value is supposed to satisfy
$V = \left(\frac{1}{H_{\sigma1}^2} - \frac{1}{H_{\sigma2}^2}\right)$
$W = \left(\frac{H_{\mu2}}{H_{\sigma2}^2} - \frac{H_{\mu1}}{H_{\sigma1}^2}\right)$
$Y = \left(\frac{H_{\mu1}^2}{H_{\sigma1}^2} - \frac{H_{\mu2}^2}{H_{\sigma2}^2}\right)$
$Z = 2ln\frac{X_2 H_{\sigma1}}{X_1 H_{\sigma2}}$
$V * \text{thresholdvalue}^2 + 2 * W * thresholdvalue + Y + Z = 0$
```

work, the threshold values are chosen from the histogram that is designed in Algorithm 1 which improves the accuracy of the model. In addition, two intensity values are chosen for classifying the pixels in the image. The intensity values range between [0,255]. Let the intensity value of the pixel be v_1 and v_2. Let the range of chosen intensity values be $1, 2, 3, 4, 5, ..., g_l$. Let p be the pixel taken randomly from the binary image I_{binary}. The pixel is allotted to the values based on the following condition:

$$p_v = \begin{cases} \mathrm{v}_1 & p_{threshold} < \text{threshold}_{lower} \\ \mathrm{v}_2 & \text{threshold}_{lower} \leq p_{threshold} \leq \text{threshold}_{upper} \\ \mathrm{v}_1 & p_{threshold} > \text{threshold}_{upper} \end{cases}$$

The I_{binary} is partitioned into smaller portions (i_{binary}) and the threshold filter is applied to individual ones. The contiguous pairs of threshold $Threshold_{pairs}$ are chosen randomly during this thresholding process and $\{threshold_{lower}, threshold_{upper}\} \in Threshold_{pairs}$. Say the thresholding process is done t times over i_{binary} images, the number of resultant binary images would be $2 * t * g_l$. The $\bigcup i_{binary}$ will be the final image. Another important aspect of the proposed work is that if thresholds are chosen from a histogram built from Section 3.1 and lying in the midrange of gray level intensity $1, 2, 3, 4, 5, ..., g_l$, the feature extraction is further enhanced. It is undoubtedly possible to choose threshold values in pairs to extract features from certain regions of an image such as the middle portion or left corner which is difficult to extract using a single threshold.

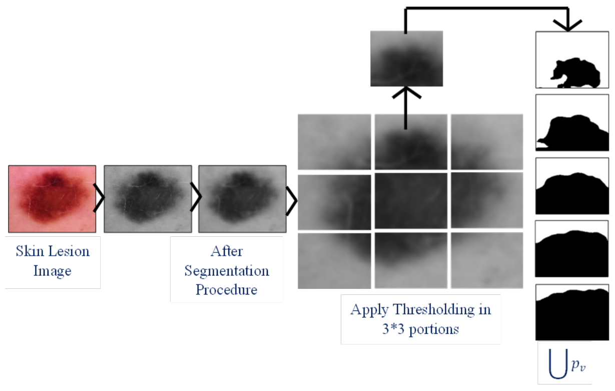

Skin Lesion Image
After Segmentation Procedure
Apply Thresholding in 3*3 portions
$\bigcup p_v$

Figure 5: Feature Extraction - Thresholding

The Fig. 5 illustrates the feature extraction process. The image is divided into smaller portions, with each portion being thresholded separately using multiple threshold pairs.

Once the decomposition and thresholding process is over, the extracted feature vectors are structured to generate the fractal dimensions of the image. The final bordered output image $\Delta_{binary}(a, b)$ is computed after combining the individual threshold images ($\bigcup i_{binary}$).

$$\Delta_{binary}(a, b) = \begin{cases} 1 & \exists(a', b') \in t(a, b) : I_{binary}(a', b') = 0 \wedge \, I_{binary}(a, b) = 1 \\ 0 & otherwise \end{cases}$$

In the above equation $t(a, b)$ is the number of times the thresholding process is executed using the pairs of threshold and the set of pixels that are interconnected in t times of execution is mentioned as $t[(a, b)]$. Using this approach, a resultant image is generated that shows mean gray level, highlighted features, and boundaries that correlate with each other to help identify patterns with minimal error.

3.3 Proposed CNN Architecture

Computer Vision has become increasingly popular with the use of CNN. Nevertheless, modern CNN is becoming more complex and deeper as they strive to improve accuracy. CNNs are a class of neural networks that uses grid-like topology to process data and contains three layers, namely convolution, pooling, and fully connected layer. As part of CNN, the convolution layer is responsible for computation. The input to the CNN is an image (feature vector) and the parameters of this layer are a set of filters (kernels). The kernels convolve over the feature vectors to produce feature maps (activation maps). This layer keeps intact the spatial relation between the pixels in feature vectors. The feature maps are then passed to the pooling layer. The pooling layer reduces the representation size of the feature map and hence minimizes computation cost. Pooling techniques in CNN include max pooling, min pooling, and average pooling. A number of convolution and pooling layers are stacked one above the other to achieve accuracy. The reduced feature maps from the pooling layer are flattened and passed to the fully connected layer which is one-dimensional in nature. A number of such fully connected layers may be stacked and following which Softmax or another classifier is applied for classification.

There are CNN models raised in literature like VGG Net [56], Alex Net [57], and

Google Net [58]. These models are trained with ImageNet Large-Scale Visual Recognition (ILSVRC) [59] and are available as pre-trained models. Often a number of custom-made models are developed in literature [60]. A few of the ways of modifying/building a pre-trained network would be by compressing or shrinking the layers, factorizing the operations, adding dropouts in them in accordance with the requirements of the developer, the nature of the input images, and the tradeoff between latency and accuracy. The proposed approach builds a modified MobileNet model, S-MobileNet for image classification. The MobileNet is a deep learning model with the unique characteristics of being small, showing low latency, and consuming little power. It produces higher accuracy than other deep CNN models when it comes to categorizing images, identifying objects in images, and segmenting images. Despite its low parameters, MobileNet has no latency excuse compared with other CNN models. Unlike CNN, MobileNet uses Depthwise Separable Convolution which makes it faster with fewer parameters than CNN.

3.3.1 S-MobileNet Model

This section discusses the S-MobileNet CNN architecture for image classification. The proposed S-MobileNet architecture is very efficient in classifying images since the hyper-parameters of the model are fine-tuned for producing results in low latency. The hyperparameters of a CNN model include modifying the kernel dimension, varying the number of kernels, changing the stride length, etc. In addition, the classic MobileNet architecture is shrunk to increase the accuracy of the proposed model (S-MobileNet CNN). S-MobileNet is applied over the processed images $\Delta_{binary}(a, b)$ generated after image segmentation and feature extraction. The processed dataset contains 10000 images which are split into a training dataset and a test dataset. One of the main features of MobileNet is that they apply DepthWise Separable Convolution in lieu of normal convolution operation.

Depthwise Separable Convolution: The Depthwise Separable Convolution operates in two phases: Depthwise Convolution and Pointwise Convolution. They perform the filtration operation and combination operation respectively. The standard convolution applies convolution operation across all channels whereas the depthwise convolution applies across one channel at a time. Channels are one of the parameters of input image (feature vector), say for example, if an RGB image is passed as input, the number of channels would be 3. The Fig. 6 shows the regular convolution operation that takes place between input images of dimension (I1*I2*A) where I1 and I2 are its dimensions, A is the number of input channels with kernels, each of dimension (K1*K2*A) where K1 and K2 are kernels dimensions, A is the width of kernel and there are B such kernels. This operation consumes a large number of multiplication operations as the kernels convolve over the input image, resulting in a huge cost.

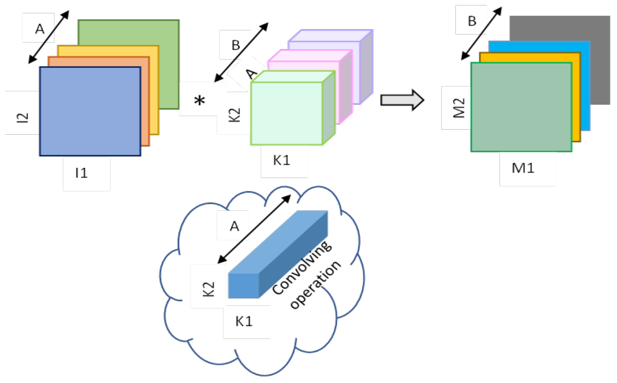


Figure 6: A Regular Convolution Operation

A single convolving operation of kernel over an image takes (K1*K2*A) operations and the complete convolving operation of a kernel over the image to produce a M1*M1 feature map would take (M1*M2*K1*K2*A). And for each of the B kernels it takes (B*M1*M2*K1*K2*A).

However, when it comes to Depthwise Separable Convolution, it's going to be a different case. The Fig. 7 shows the operation taking place in two stages. In the first stage, a depthwise convolution operation takes place between input images of dimension (I1*I2*A) with kernels of dimension (K1*K2*1) and there are A such kernels. The convolution operation takes place between the input image channels and kernels in a 1:1 ratio, unlike the conventional convolution where the kernel convolves over all the channels. In this case, depthwise convolution requires the same number of kernels as the number of input channels. The number of multiplication operation that takes place in one convolution operation would be (K1*K2*1) since it's one dimensional. When this convolves over the one channel of input image it takes (M1*M2*K1*K2*1). When A such filters are applied to A channels it takes (A*M1*M2*K1*K2*1).

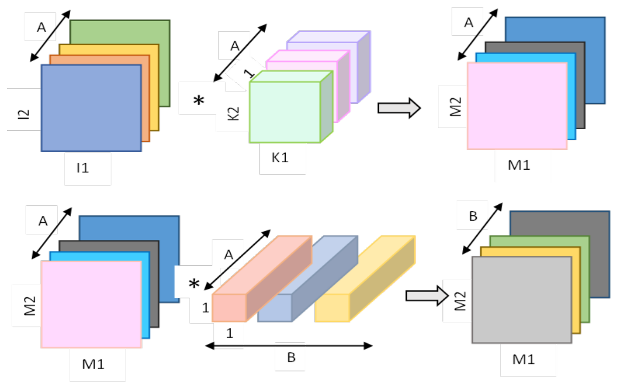


Figure 7: Depthwise Separable Convolution

Moving on to the second stage which is pointwise convolution. Each of the B filters of dimension (1*1*A) convolves the input channels (M1*M2*A) (A is the depth of the input volume) to produce an output tensor of dimension (M1*M2*B). The filters are called KPC (Kernel Point Convolution) filters as their dimension is 1*1 and suited for pointwise convolution. For one instance of convolution operation between one KPC filter with the input channel of depth A, it would take (A) multiplications. The entire convolution of the KPC filter over input volume would take (M1*M2*A). And for B such channels it would be (B*M1*M2*A). The total number of multiplications in depthwise separable convolution would be (A*M1*M2*K1*K2*1)+ (B*M1*M2*A) $\rightarrow$ (A*M1*M2)(K1*K2+B). On analysis, it's seen that the number of multiplication in conventional convolution is 9 times more than depthwise separable convolution.

S-MobileNet Architecture: It can be seen from [61] that MobileNet models are trained with RGB images and hence requires 3 channels as input. However, the dataset that we obtained after image segmentation and feature extraction include grayscale images of size (450*650). While it is possible to convert the grayscale images to RGB and feed them to the S-MobileNet model, but does so at the expense of losing a tremendous amount of information in the first layer of the convolution process. Hence, the grayscale image is reduced to (224*224) and is repeated 3 times to produce an input tensor of dimension (224*224*3). The Fig. 8 represents the standard convolution layer, the MobileNet convolution layer, and the S-MobileNet Convolution Layer.

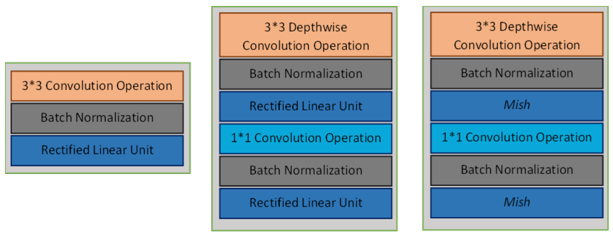


Figure 8: Layers in Conventional Convolution Layer, Depthwise Separable Convolution Layer and proposed S-MobileNet model

The proposed S-MobileNet model applies Mish as the activation which is very efficient compared to the regular activation function (Relu) applied in the other convolution layer as shown in Fig. 8. The activation function (Transfer function) defines the output that has to be generated at the end of every node/layer based on the input values provided to the node/layer. Very importantly they introduce non-linearity in the output which is an important factor to learn complex patterns in the input image.

$$f(x) = \begin{cases} 1 & x >= 0 \\ 0 & x < 0 \end{cases}$$

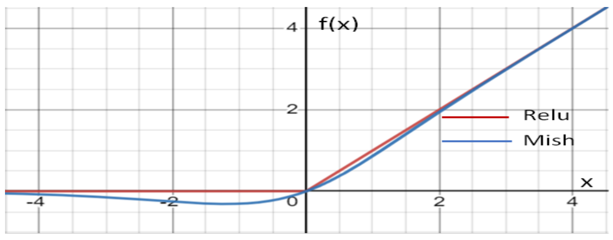


Figure 9: Activation function: Relu and Mish

The Relu activation function being non-linear in nature activates specific neurons at the output leading to convergence of gradient to global minima. Though being an efficient function, during training the model fails to activate certain neurons as their weight gets diminished during backpropagation, and at one point it causes the neurons to die. This is called the dying-Relu problem.

On the other hand, the Mish activation function outperforms Relu (Fig. 9), Leaky Relu, and other activation functions on several benchmark applications like ResNet, DarkNet, etc [62]. The Mish activation function is a non-monotonic activation function which produces a smooth and continuous output.

$$f(x) = x * tanh(softplus(x))$$

$$f(x) = x * tanh(ln(1 + e^x))$$

One of the remarkable characteristics of Mish is that it reduces overfitting and overcomes the dying Relu problem by preserving a very small amount of negative weights and permits information flow during backpropagation. Table 2 describes the S-MobileNet architecture in levels, mentioning the operation, strides, filter dimension, and tensor output dimension.

1. The S-MobileNet CNN architecture starts with the initial convolution layer {level 1} which takes up the input volume of dimension (width-224*height-224*depth-3) and 32 kernels with each filter of dimension (3*3*3). The convolving operation with a stride of 2 produces an output tensor of dimension (112*112*32). The dimension of the output tensor is calculated based on the following formula, $((widthofimage - filter\_dimension + 2 * padded\_pixel)/stride) + 1$. In our case, there are no padded pixels and hence it's 0. Thus $((224 - 3 + 2 * 0)/2) + 1 \rightarrow 112$. The input to the next convolution layer will be (112*112*number of filters i.e. 32).

2. This is followed by a sequence of depthwise convolution and pointwise convolution operation in four iterations {level 2-level 5} with depthwise filters of dimension (3*3) and pointwise filters of dimension (1*1) with (32-64, 64-128, 128-128, 128-256) number of kernels respectively.

3. Layer 6 in the regular MobileNet architecture operates with the depthwise filter (3*3) and pointwise filter (1*1) but with a kernel count of (256-256) respectively. On the contrary, in the proposed S-MobileNet model, the convolution layer in level 6 is compressed to produce an output tensor of reduced dimension. In this layer 10% of the

filters are reduced during the depthwise convolution operation i.e. $\lfloor \frac{10*256}{100} \rfloor = 25$ i.e. we discard 25 filters in this layer. This reduces the depth of the output tensor from 256 to 231. The output tensor of dimension 28*28*231 is subjected to pointwise convolution with 256 filters to produce an output of dimension 28*28*256.

4. In level 7, depthwise convolution applies 256 kernel filters of dimension (3*3) with a stride of 2 and generates a downsampled image of dimension 14*14*256. This is followed by 512 pointwise filters to produce an output tensor 14*14*512.

5. In Level 8, both depthwise and pointwise convolutions are applied with a stride of 1 and kernel count of 512 and 512 in each respectively, generating an output tensor of depth 512 with dimension 14*14*512. As the dimension of the tensor is high, compression is applied in the next layer to reduce the number of parameters.

Level	Operation	Stride	Filter Dimension	No:of filters	Compression	Input dimension
1	Convolution	2	3*3*3	32	-	$224 \times 224 \times 3$
	Depthwise convolution	1	3*3	32	-	(Downsampled image) 112*112*32
	Pointwise convolution	1	1*1*32	64	-	112*112*32
	Depthwise convolution	2	3*3	64	-	112*112*64
	Pointwise convolution	1	1*1*64	128	-	(Downsampled image) 56*56*64
	Depthwise convolution	1	3*3	128	-	56*56*128
	Pointwise convolution	1	1*1*128	128	-	56*56*128
	Depthwise convolution	2	3*3	128	-	56*56*128
	Pointwise convolution	1	1*1*128	256	-	(Downsampled image) 28*28*128
	Depthwise convolution	1	3*3	256	10% reduction i.e. Approx 25 filters (256-25=231)	28*28*256-
	Pointwise convolution	1	1*1*231	256	-	28*28*231
	Depthwise convolution	2	3*3	256	-	28*28*256
	Pointwise convolution	1	1*1*256	512	-	(Downsampled image) 14*14*256
	Depthwise convolution	1	3*3	512	-	14*14*512
	Pointwise convolution	1	1*1*512	512	-	14*14*512
	Depthwise convolution	1	3*3	512	8% reduction i.e. Approx 40 filters.(512-40=472)	14*14*512
	Pointwise convolution	1	1*1*472	512	-	14*14*472
	Depthwise convolution	1	3*3	512	-	14*14*512
	Pointwise convolution	1	1*1*512	512	-	14*14*512
	Depthwise convolution	1	3*3	512	8% reduction i.e. Approx 40 filters. (512-40=472)	14*14*512
	Pointwise convolution	1	1*1*472	512	-	14*14*472
	Depthwise convolution	1	3*3	512	-	14*14*512
	Pointwise convolution	1	1*1*512	512	-	14*14*512
	Depthwise convolution	2	3*3	512	10% reduction i.e. Approx 51 filters. (512-51=461)	14*14*512
	Pointwise convolution	1	1*1*461	1024	-	7*7*461
	Depthwise convolution	2	3*3	1024	With stride value of 2 and to produce an output dimension equal to input, padding is introduced	7*7*1024
	Pointwise convolution	1	1*1*1024	1024	-	7*7*1024
15	Average Pooling	1	7*7	-	-	1*1*1024
16	Fully connected	1	1024*1000	-	-	1*1*1024
17	Softmax	1	Classifies into 7 classes	-	-	1*1*1000

Table 2: S-MobileNet Architecture - Layers

6. The depthwise convolution layer in level 9 is compressed by 8% and a total of 40 filters are reduced ($\lfloor \frac{8*512}{100} \rfloor = 40$). The depth of the output tensor is reduced from 512 to 472. The output tensor (14*14*472) is passed to a pointwise convolution operation with 512 filters to produce a resultant tensor of dimension (14*14*512).

7. Level 10 shows no change. But again in level 11, there is a compression of 8% reducing approximately 40 filters. This is followed by level 12 with no change. And again in level 13, there is compression of 10% and 51 ($\lfloor \frac{10*512}{100} \rfloor = 51$) filters are removed. A stride length of 2 is applied and produces an output tensor of dimension (7*7*461). This is followed by pointwise convolution with a 1024 filter and thus raising the depth of the output tensor to 1024 (7*7*1024).

8. Padding is introduced in level 14 which applies a stride of 2 over the input tensor of dimension (7*7*1024) and produces an output of the same dimension.

9. Nearing the end of the convolution layer, the average pooling operation is carried out with a 7*7 sliding window and downsamples the tensor to a dimension of (1*1*1024)

10. The fully connected layer flattens into a layer of 1000 pixels and a softmax classifier is applied for classifying them into 7 classes.

As mentioned before, Mish is the activation function that is applied in all the layers of depthwise and pointwise convolution. On top, of all the layers of execution, dropout is enabled to be True. During training, dropout ensures to prevent overfitting by dropping certain neurons. And hence the model acts as an ensemble model and the prediction value is by default averaged in each layer. In the network architecture the information from the previous layers (I_j) is multiplied with the link weights (W_{ij}) and the output neuron (O_i) aggregates them as shown below [63]:

$O_i = \sum_{j=1}^{N} W_{ij} \, I_j$

while the standard dropout applies bernoulli function

$O_i = \frac{1}{u} \sum_{j=1}^{N} W_{ij} \, (\alpha_j * I_j) \, , \, \alpha_j \; Bernoulli(u)$

to minimize the number of neurons in intermediary layers.

Compressing the S-MobileNet: Compressing the CNN network is an efficient approach since it reduces the number of parameters in compressed layers and thereby the total parameter count of the model. Although this approach produces a low latency and high-speed network, applying compression in the initial layers leads to the loss of a huge amount of pixel information. Hence, in the S-MobileNet model, they are applied in intermediary layers. One of the key aspects of S-MobileNet is that the compression is applied four times with enough spacing between layers and not between successive layers. Compression when applied in successive layers will decrease the patterns of lost information in the first layer over the further layers. In S-MobileNet, compression is applied in levels 6, 9, 11, and 13 and there is enough spacing between compressed layers to restore the patterns between the lost features.

Compression is often referred to as the pruning of filters with sparse information that is not significant in changing the final decision at the output. In the S-MobileNet CNN, L1 norm pruning is applied to cut down the insignificant filters in levels 6, 9, 11, and 13 of the model. A CNN network

$\left\{N^i \in \mathbb{R}^{I_i * O_i * C_W * C_H}\right\},\ 1 \leq i \leq L$

is defined with parameters N_i, weight matrix of connections in layer i; L, number of layers; I_i and O_i, Number of input channels and output channels in layer i respectively; C_W and C_H, represents the height and width of the input channel respectively. In our case $C_W = C_H$, as the input channel is a square of dimension 224*224 and varies in each layer retaining the square property. The number of computational operations in a convolution layer is given by $I_i * O_i * W * H * w_{i+1} * h_{i+1}$. After applying the L1 norm, the number of computational operations is $I_i * W * H * w_{i+1} * h_{i+1}$. Here $w_i * h_i$ and $w_{i+1} * h_{i+1}$ are the input feature size and output feature size respectively. The number of computations will considerably reduce as the pruning is applied in the successive layers. During network training, emphasis is provided on minimizing the loss function. The minimization objective of the loss function is represented as

$\hat{\theta} = \underset{W}{\min}\ \sum_{(I_i,O_i)} \theta\left(W^T I_i - O_i\right)^2 + \lambda\left|W\right|$

The function $\theta()$ is the squared loss function that sums the square of the difference between the predicted and actual values. The λ controls the degree of sparsity of the weight matrix W. The L1 norm is penalizing the filters that have a small magnitude and in parallel keeps track of the optimization function $\hat{\theta}$. In addition, it regulates the tradeoff between the loss function, regularization parameters, and the weight matrix.

S-MobileNet hyperparameters: The two specific hyperparameters of MobileNet: width multiplier, α, and resolution multiplier, ρ. The parameter α takes any value from [0:1] and for every layer with I_i number of input channels and O_i number of output channels, it becomes $\alpha * I_i$ and $\alpha * O_i$. The tradeoff between latency and speed of small networks is decided by the width multiplier and reduces the complexity by α^2. The resolution multiplier ρ takes values between [0:1] and each layer's internal representation parameter is reduced by ρ. Similar as α, ρ reduces the computational complexity by ρ^2. The TensorFlow version used in the proposed model is a stable release 2.10.0 using Python language. The S-MobileNet model is executed using three optimizers individually: Adam, RMSProp, and SGD with stochastic gradient descent and with Nesterov Momentum. With this proposed model, data overfitting is minimized to a greater extent. The model is trained varying the epochs and learning rate. In the next section, we will look in detail at the experiments and results. In a similar way, the authors in [64] compare and analyze a number of optimizers, among which are SGD, Adam, and FastAdaBelief, and demonstrate that one outperforms the rest.

4 Experiments and Results

The HAM10000 archive [46] of dermoscopic images is subjected to a segmentation experiment. The Dice and Jaccard coefficients are used to measure the performance of modified Gaussian filtering and the standard Gaussian filtering approach. The Dice coefficient measured between the segmented image (S) and the ground truth Image (G) is $Dice(S, G) = \frac{2*(S \cap G)}{|S|+|G|}$. The Dice scores images on a scale of 0 to 1, with higher scores indicating more accurate segmentation. Table 3 shows the Dice score obtained by executing the proposed segmentation algorithm discussed in Section 3.1 and standard Gaussian filter on random 25 samples.

Image	Proposed segmentation discussed (in Algorithm 1 and 2)	Gaussian Filter
ISIC_25773	**0.951**	0.922
ISIC_26757	**0.741**	0.644
ISIC_27086	0.764	0.799
ISIC_24748	**0.658**	0.534
ISIC_28064	**0.786**	0.781
ISIC_26461	**0.859**	0.754
ISIC_27179	**0.986**	0.785
ISIC_27408	**0.939**	0.855
ISIC_27766	**0.842**	0.743
ISIC_27139	0.828	0.876
ISIC_27936	0.816	0.821
ISIC_24815	**0.958**	0.789
ISIC_27421	0.607	0.654
ISIC_24408	**0.793**	0.655
ISIC_25207	**0.749**	0.549
ISIC_28886	**0.824**	0.801
ISIC_26944	0.802	0.987
ISIC_27872	**0.970**	0.921
ISIC_28649	**0.706**	0.692
ISIC_27022	**0.972**	0.901
ISIC_27581	**0.784**	0.692
ISIC_25439	**0.961**	0.894
ISIC_26900	0.745	0.769
ISIC_26060	**0.853**	0.847
ISIC_25281	0.862	0.987
ISIC_27339	**0.949**	0.901
ISIC_26515	**0.772**	0.701
ISIC_27616	**0.969**	0.899
ISIC_26741	0.754	0.799
ISIC_25902	**0.967**	0.934
Average	**0.839**	**0.796**

Table 3: Dice Score of Proposed Segmentation method and standard Gaussian filter approach

Block of 500 random images from shuffled HAM dataset	Proposed segmentation discussed (in Algorithm 1 and 2)	Gaussian Filter
Block 1	**0.989**	0.876
Block 2	**0.976**	0.705
Block 3	**0.806**	0.721
Block 4	**0.989**	0.789
Block 5	0.873	0.994
Block 6	**0.790**	0.598
Block 7	**0.976**	0.980
Block 8	0.885	0.899
Block 9	0.787	0.843
Block 10	**0.989**	0.768
Block 11	0.985	0.874
Block 12	**0.878**	0.872
Block 13	0.791	0.874
Block 14	**0.979**	0.923
Block 15	**0.874**	0.685
Block 16	**0.840**	0.743
Block 17	**0.975**	0.839
Block 18	0.777	0.854
Block 19	**0.850**	0.765
Block 20	**0.974**	0.896
Average	**0.899**	**0.825**

Table 4: Dice Score of HAM 10000 images in 20 blocks, each of 500 images with Proposed Segmentation method against standard Gaussian filter approach

The HAM10000 dataset is shuffled and 20 blocks of random 500 images are subjected to the proposed segmentation method and standard Gaussian filter approach. The average dice score of each block is recorded in Table 4 and is shown in Figure 10.

Following the image segmentation, feature extraction is applied over the images using the modified version of Segmentation-based Fractal (SFTA). Several iterations of this modified version are executed using pairs of thresholds selected from the histogram (detailed in **??**) that is generated during the segmentation phase. Figure 11 shows the results of traditional feature selection and the proposed SFTA model. The proposed model extracts the infected region and the nearby segments that are risk at of infection. Many recent works on feature extraction apply to label the image and to locate the region by dividing them into smaller portions and analysing them individually. However, in SFTA

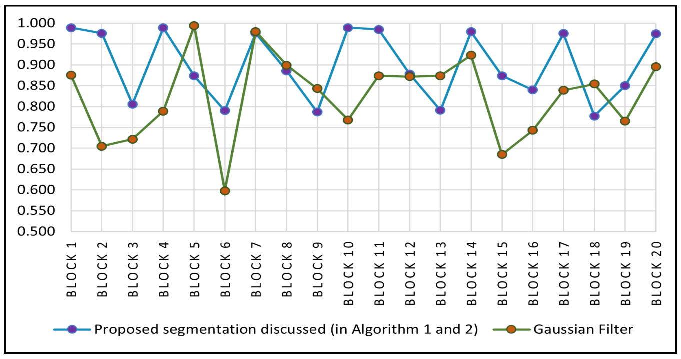


Figure 10: Dice score plot with reference to Table 4

the feature vector is applied in 8 thresholds and its classification accuracy is recorded [65].

The feature extraction process using labeling has a couple of drawbacks. It consumes more time and fails to extract new features when added to the image. And in most the cases, this approach prioritizes features with more unique values than those with redundant values. As can be seen that the feature extraction approach that is obtained by modifying the SFTA algorithm produces better results in classifying the images using S-MobileNet architecture.

4.1 Performance metrics - S-MobileNet model

The segmented images are subjected to S-MobileNet CNN architecture. This section details the model parameters and tabulates the performance of the model by fine-tuning its hyperparameters. As an initial step, the dataset is split in an 80:20 train-test split ratio and the model is trained with 8000 images each of size (224*224) from the HAM dataset in many epochs. The model is designed with layers as mentioned in Table 2 and the learning rate is set to be 0.01 initially. The model is executed in 5 folds with 20% of images in each fold. During the execution of the model in multiple iterations, the number of epochs was varied and the performance was studied. It was found in the initial epochs that the model learned the parameters and after 15 epochs the results were found to be stable and minute changes were recorded. The performance metrics for evaluating the model are the training loss, testing loss, training accuracy, testing accuracy, precision, and F1-score. The model is executed using three optimizers Adam, RMSProp, and SGD.

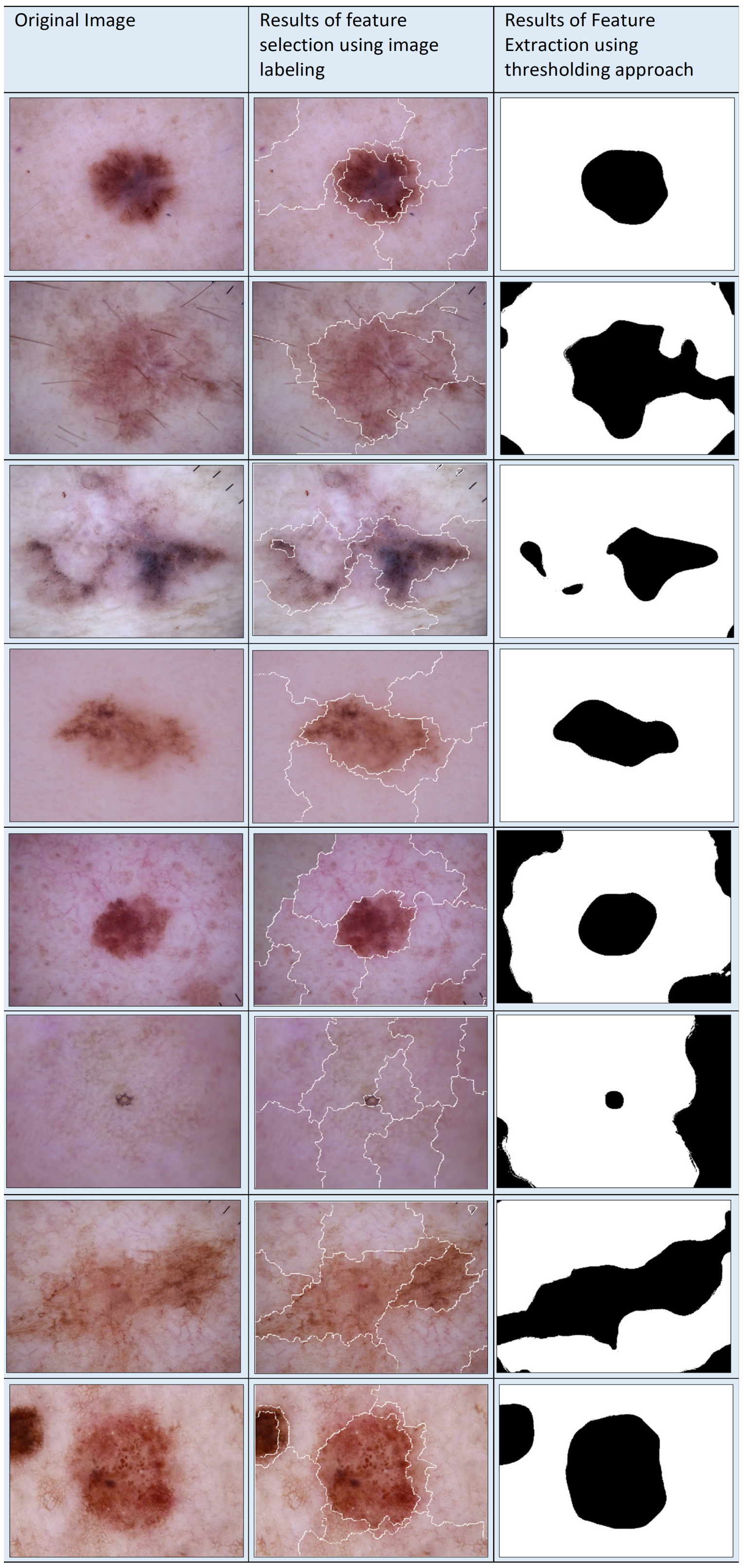

Original Image
Results of feature selection using image labeling
Results of Feature Extraction using thresholding approach

Figure 11: Feature extraction results of modified SFTA algorithm

Categorical Cross Entropy loss: Being a multiclass classification, the loss function used is Categorical Cross Entropy loss (Softmax loss).

$$Loss = -\sum_i^C G_i log\left(func\,(SC)_i\right)$$

$$func(SC)_i = \frac{e^{\mathrm{SC}_i}}{\sum_k^C e^{\mathrm{SC}_k}}$$

Here C is the number of classes and in the proposed model its 7, G_i is the ground truth value of each class i, $func(SC)_i$ is the S-MobileNet score for each class i. The aforementioned loss function is used in both training and testing of the S-MobileNet model.

Accuracy: The accuracy of the S-MobileNet is calculated using

$$Accuracy = \frac{T_P+T_N}{T_P+T_N+F_P+F_N}$$

Here T_P refers to true positive, i.e. the number of positive images correctly predicted; T_N refers to true negative, i.e. the number of negative images correctly predicted; F_P refers to false positive, i.e. the number of positive images incorrectly predicted; F_N refers to false negative, i.e. the number of negative images incorrectly predicted. The higher the value of T_P, T_N, the higher the accuracy. The metric accuracy defines the performance of any network.

Precision: The precision is calculated using

$$Precision = \frac{T_P}{T_P+F_P}$$

and this metric defines the accuracy of the model in identifying a sample as positive. The precision of a model increases in two cases: either when T_P is high or when F_P is low.

F1 Score: This score decides the overall performance of the model. There is a fine line difference between accuracy and F1-score. Accuracy is primarily concerned with predicting the positive samples, while F1-score addresses the behavior of the model toward negative samples as well.

4.2 S-MobileNet model result analysis

In the initial go, the model is designed and tested by applying Mish and Relu activation functions and by executing it with and without compression when the Mish activation function is applied. The effectiveness of applying the Mish Activation function and compression is shown in the experimental results. Firstly, the experiments are performed with a batch size of 32 and the images are passed to the S-MobileNet model without applying the modified segmentation and feature extraction procedure. In contrast, the performance of the model is also recorded after applying segmentation and feature extraction. Secondly, the model is executed with three different optimizers. Thirdly, each of the optimizer's results is recorded with the Relu activation function and Mish Activation function. In the fourth and final step, the model is evaluated both with and without layer compression. The words compression and pruning are used interchangeably in the coming sections. All of their results are shown in Table 5 and Table 6.

Optimizers The state-of-the-art of deep learning libraries the gradient descent algorithms (Optimizers). They are coded as a black box with their strength and weakness. The performance of the gradient descent algorithm varies for different applications and can be fine-tuned. Gradient descent algorithms are used to train the CNN model. Being an optimization algorithm the objective of gradient descent is to minimize the cost function and reach the global minima. This is achieved by adjusting the learning rate. The author in [66] has detailed different optimization algorithms on gradient descent. Adaptive Moment Estimation (Adam) is a gradient descent algorithm that keeps track of the exponentially decaying average of past gradients. Adam is a slight variation of another gradient descent algorithm named momentum. Adam faces the problem of diminishing learning which is overcome by Root Mean Square Propagation (RMSProp) and concludes that the learning rate of 0.001 will produce optimal results. Stochastic Gradient Descent (SGD) produces better results with large datasets. Following many trials of other gradient descent algorithms, the above three algorithms were chosen.

Table 5 shows the results of the model without segmentation and feature extraction applied to the dataset. Trial analysis is made with Relu and Mish activation function.

Method	Batch Size	Optimizer	Training loss	Test loss	Training Accuracy	Test Accuracy	Precision	F1 score
S-MobileNet using Relu	32	Adam	0.19905	0.19714	0.93042	0.88386	0.96958	0.93358
S-MobileNet using Mish without pruning	32	Adam	0.19190	0.19103	0.94182	0.86156	0.96973	0.94341
S-MobileNet using Mish with pruning of layers	32	Adam	0.17845	0.17281	0.94188	0.91272	0.97469	0.95828
S-MobileNet using Relu	32	RMSProp	0.17630	0.20839	0.80410	0.89891	0.89364	0.87904
S-MobileNet using Mish without pruning	32	RMSProp	0.16598	0.20116	0.86561	0.90188	0.89382	0.88662
S-MobileNet using Mish with pruning of layers	32	RMSProp	0.16169	0.19905	0.91159	0.91109	0.92594	0.92333
S-MobileNet using Relu	32	SGD	0.15895	0.23190	0.95078	0.86411	0.91791	0.92120
S-MobileNet using Mish without pruning	32	SGD	0.15600	0.17845	0.95079	0.93265	0.92477	0.90309
S-MobileNet using Mish with pruning of layers	32	SGD	0.14910	0.17630	0.97757	0.93222	0.95588	0.93567

Table 5: Performance evaluation of the proposed S-MobileNet model without applying Segmentation and Feature extraction mehods, with different optimizers, Relu and Mish activation function, with and without pruning intermediary layers

Across all three optimizers,

1. Mish activation function shows lower training and testing loss than relu.
2. Mish activation function shows an increased training and testing accuracy than relu.
3. Mish activation function shows an increased precision and F1 score than relu.
4. Mish activation function along with pruning layers shows lower training and testing loss than just applying Mish.
5. Mish activation function along with pruning layers shows higher training and test accuracy than just applying Mish except for SGD optimizer.
6. Mish activation function along with pruning layers shows higher Precision than just applying Mish.
7. Mish activation function along with pruning layers shows higher Accuracy than just applying Mish.

Method	Batch Size	Optimizer	Training loss	Test loss	Training Accuracy	Test Accuracy	Precision	F1 score
S-MobileNet using Relu	32	Adam	0.16177	0.19061	0.96650	0.95991	0.96809	0.93367
S-MobileNet using Mish without pruning	32	Adam	0.15722	0.18617	0.96542	0.94289	0.97896	0.95330
S-MobileNet using Mish with pruning of layers	32	Adam	0.15763	0.17607	0.96604	0.95032	0.97392	0.95838
S-MobileNet using Relu	32	RMSProp	0.13723	0.17346	0.87666	0.86086	0.89902	0.89653
S-MobileNet using Mish without pruning	32	RMSProp	0.13498	0.17175	0.87639	0.89797	0.91053	0.91029
S-MobileNet using Mish with pruning of layers	32	RMSProp	0.12739	0.16891	0.91619	0.93680	0.94757	0.95288
S-MobileNet using Relu	32	SGD	0.15603	0.16823	0.96340	0.91702	0.92162	0.92859
S-MobileNet using Mish without pruning	32	SGD	0.15473	0.17177	0.98158	0.94489	0.94751	0.94592
S-MobileNet using Mish with pruning of layers	32	SGD	0.14154	0.16093	0.98345	0.98154	0.96233	0.94593

Table 6: Performance evaluation of the proposed S-MobileNet model after applying Segmentation and Feature extraction mehods, with different optimizers, Relu and Mish activation function, with and without pruning intermediary layers

8. Adam optimizer produces a high precision and F1 score for image classification when compared to other two, whereas the Accuracy score of SGD is slightly higher than other two optimizers.

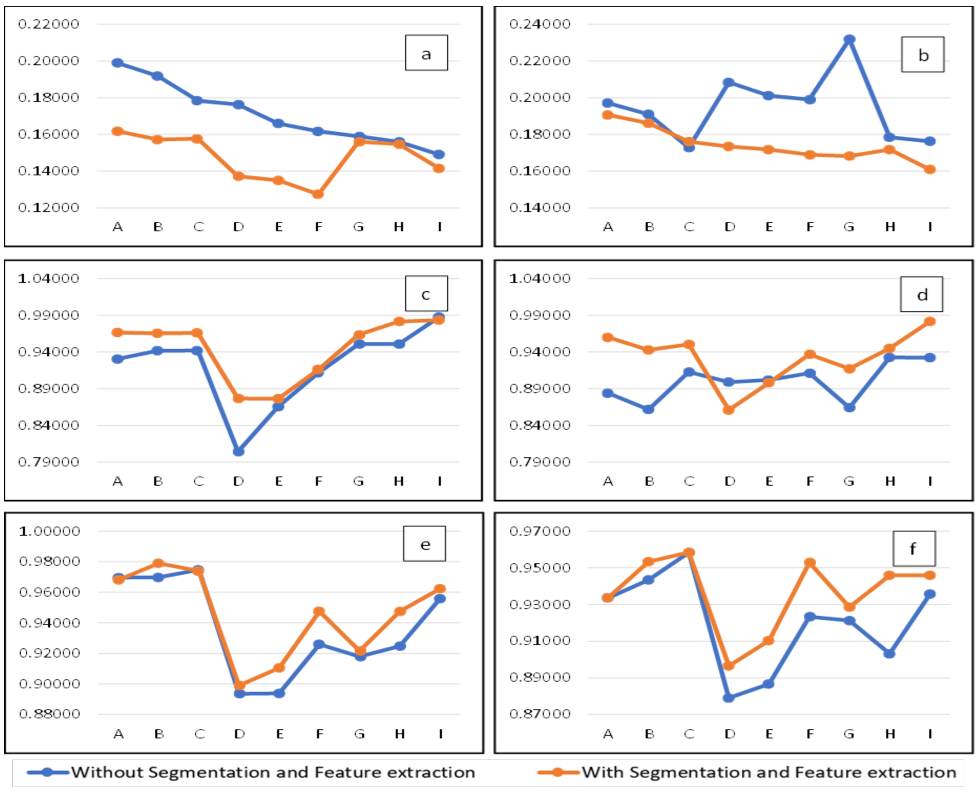
Following the passing of the processed dataset after segmentation and feature extraction to the S-MobileNet model the results are recorded in Table 6. A couple of differences are found in the behavior of the optimizers as compared to Table 5.

The comparative results/inferences obtained by applying segmentation and feature extration on S-MobileNet are listed below:

1. All the optimizers produce lower training and testing loss for applying Mish activation function than compared to Relu except the test loss of SGD which shows a slighly higher value for Mish than for Relu.

2. The training and test accuracy of Adam optimizer shows a slightly lower value for Mish than for Relu. A slighly exceptional case.

3. The training accuracy of RMSProp shows lower accuracy for Mish than Relu. But the test accuracy is ideal with higher value for Mish than for Relu.



A → Relu, Batch Size-32, Optimizer-Adam
B → Mish without pruning, Batch Size-32, Optimizer-Adam
C → Mish with pruning of layers, Batch Size-32, Optimizer-Adam

D → Relu, Batch Size-32, Optimizer-RMSProp
E → Mish without pruning, Batch Size-32, Optimizer-RMSProp
F → Mish with pruning of layers, Batch Size-32, Optimizer-RMSProp

G → Relu, Batch Size-32, Optimizer-SGD
H → Mish without pruning, Batch Size-32, Optimizer- SGD
I→ Mish with pruning of layers, Batch Size-32, Optimizer- SGD

Figure 12: Result Analysis (a) Training loss of S-MobileNet model (b) Test loss of S-MobileNet model (c) Training Accuracy of S-MobileNet model (d) Testing Accuracy of S-MobileNet model (e) Precision of S-MobileNet model (f) F1 score of S-MobileNet model

4. For the SGD optimizer, the accuracy for training and testing is higher with Mish activation function than with Relu.

5. The precision and F1 score of all the three optimizers shows a higher value for Mish than for Relu.

6. A notable drop in training and testing loss values is found after pruning layers for all the three optimizers except the training loss of Adam which is slightly higher.

7. For all the optimizers, a notable analysis is the accuracy, precision and F1-score of the model after undergoing pruning is higher than that without pruning.

8. Thus it can be concluded that pruning layers produces higher accuracy irrespective of optimizer. Having many layers in the model not always guarantees higher accuracy.

On comparison of Tables 5 and 6, the processed dataset, regardless of the optimizer used, produced higher accuracy, precision, and f1 score than the original dataset. Figure 12 illustrates the performance of the S-MobileNet model.

4.3 Pruning layers in S-MobileNet

The layers of S-MobileNet at Level 6, 9, 11, 13 are pruned using L1-Norm values.

Level 6 of the S-MobileNet mentioned in Table 2 has 256 filters. The L1 norm values of each filter are generated and are plotted in Figure 13-Diagram(A). The L1-norm value of 0.1 is chosen as the threshold and this contributes to 10% of the filters at this layer, around 25 in count, being pruned. Secondly, it is applied at Level 9 with 512 filters. The L1-norm cutoff value is 0.09 (Figure (13)-Diagram(B)) and around 40 filters are pruned. Thirdly at Level 11, the L1 norm threshold is fixed at 0.1, and around 40 filters are pruned (Figure (13)-Diagram(C)). And finally, at Level 13, the L1 norm threshold is 0.11, and around 51 filters out of 512 are pruned (Figure (13)-Diagram(D)).

After many trials, the threshold values are selected in a way that effectively increases the classification accuracy of the model. Also, the levels for pruning are chosen with

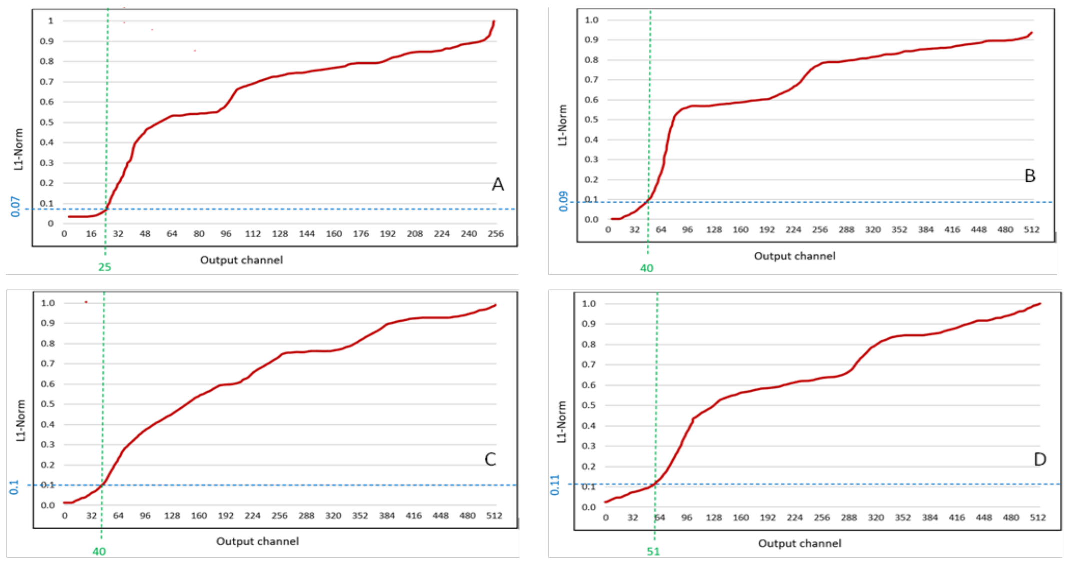


Figure 13: L1 norm compression value (A) Level 6 in S-MobileNet architecture-25 filters pruned with L1Norm cutoff-0.07 (B) Level 9 in S-MobileNet architecture-40 filters pruned with L1Norm cutoff-0.09 (C) Level 11 in S-MobileNet architecture-40 filters pruned with L1Norm cutoff-0.1 (D) Level 13 in S-MobileNet architecture-51 filters pruned with L1Norm cutoff-0.11.

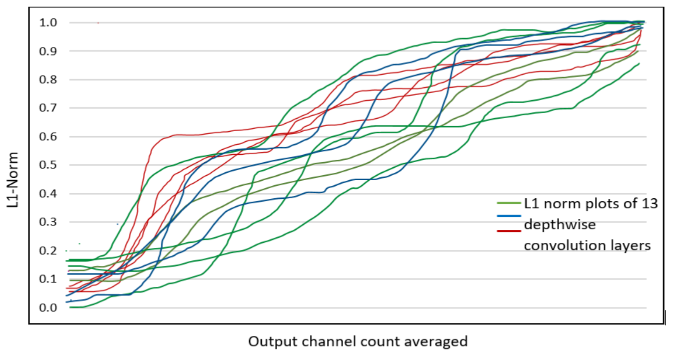


Figure 14: L1 norm values

enough gaps between them to avoid loss of data. The L1 norm values of all the 13 depthwise convolution layers are shown in Figure 14.

Pruning is only applied at only 4 levels in the proposed model since more pruning will have a significant negative influence on accuracy. A detailed result analysis of the drop in filter percentage and the associated increase in other performance metrics is discussed below:

1. In levels 6, 9, 11, and 13, there are a total of 1792 filters. Of these, 156 filters are pruned. Approximately 8% of the filters are pruned in these four levels. The complete S-MobileNet architecture is made of 10944 filters and 156 filters are pruned, which is approximately 1.4%.

2. The training accuracy of Adam, RMSProp, and SGD is raised by 0.06%, 4.5%, and 0.19% respectively. The testing accuracy of Adam, RMSProp, and SGD is raised by 0.78%, 4.32%, and 3.87% respectively. Precision has decreased by 0.51% for the Adam optimizer while rising by 4.06% and 1.56% for the RMSProp and SGD optimizer respectively. The F1 score shows a raise in 0.53%, 4.67%, and 0.0009% increase for Adam, RMSProp, and SGD respectively. Figure 14 shows the percentage of variation in all these performance metrics with and without applying the pruning process.

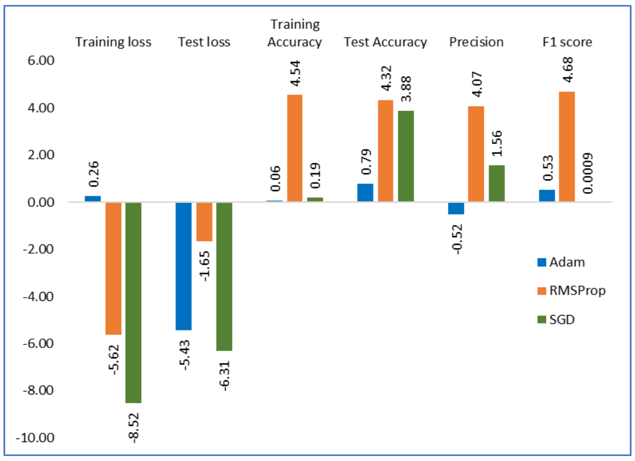


Figure 15: Percentage change in performance metrics after applying pruning

From Figure 15, it's seen that there is a significant drop in the training and testing loss percentage and an overall raise in other performance metrics. The proposed S-MobileNet model works efficiently in classifying the images into 7 classes with high accuracy and

Ref	Model	Accuracy
[47]	MobileNet	80.14%
	Modified MobileNet without data up sampling and data augmentation method	83.93%
	Modified MobileNet with data up sampling and data augmentation method	83.23%
[48]	Custom built model	88.57%
[49]	Inception	87.70%
	Xception	86.20%
	Inception ResNet	87.80%
	ResNet	85.20%
	DenseNet	88.20%
	Ensemble	85.20%
[50]	SVM	89.80%
[51]	ShuffleNet	76.83%
	Wide- ShuffleNet	77.88%
	Entropy-based Weighting and First-order Cumulative Moment (EW-FCM) + ShuffleNet	83.66%
	Entropy-based Weighting and First-order Cumulative Moment (EW-FCM) + wide -ShuffleNet	84.80%
	Entropy-based Weighting and First-order Cumulative Moment (EW-FCM) + EfficientNet-B0	85.50%
S-MobileNet	Without segmentation and feature extraction	97.757%
	With segmentation and feature extraction	98.345%

Table 7: Performance comparison of S-MobileNet to MobileNet and other existing algorithms executed over HAM10000

minimal loss.

The performance of the S-MobileNet framework is compared to the MobileNet benchmark algorithm and other existing algorithms executed over the HAM10000 dataset in Table 7. Compared to other models/algorithms/CNN frameworks, S-MobileNet performs better. A notable feature in the proposed model is the pruning which makes it lightweight and other preprocessing techniques which boost its performance. Due to the uncertainty about the GPU, processor, hardware power, or other external factors under which literature algorithms are executed, it would not be possible to compare or analyze the latency of the algorithms or the time is taken to execute them with the proposed ones.

Although the S-MobileNet CNN framework has achieved promising performance on the HAM10000 dataset, future enhancements may be applying S-MobileNet on other datasets like ISBI 2016 challenge dataset for skin lesion analysis towards melanoma detection [67], PAD-UFES-20 skin lesion dataset [68], PH2 database [69] or other real-time datasets to make it acceptable as a global framework on skin lesion analysis. Another interesting challenge would be to identify and validate over-segmented images, as well as

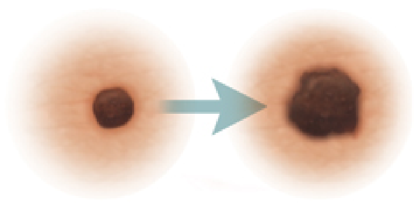


Figure 16: Skin lesion size progression

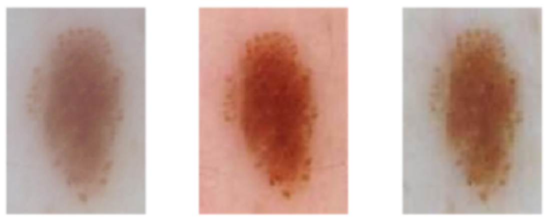


Figure 17: Change in skin lesion color with time

to control thresholds based on brightness or contrast.

The societal benefits of automated processing of skin images can help in early detection of skin diseases like cancer/melanoma etc, reduces misdiagnoses of skin diseases, earlier the prediction reduces the healthcare cost, early predicts the change in the size of the skin lesion (Fig. 16) and the color change (Fig. 17), and promotion of human well being.

5 Conclusion

In this paper, we proposed an end-to-end deep CNN based skin lesion classification framework. Images from the HAM10000 dataset were preprocessed using the proposed image segmentation and feature extraction algorithm, and fed into our customised S-MobileNet CNN model for classification. The S-MobileNet CNN model was fed the raw dataset in the first phase, and the processed dataset in the second, and a comparative study is performed. S-MobileNet CNN model was trained in either case and hyperparameters were fine-tuned to ensure higher accuracy in classification. The layers of the S-MobileNet are custom-made and analysed by applying the Mish activation function. The performance of the S-MobileNet model with the Mish activation function was compared with the contemporary Relu activation function. Further, we compressed/pruned S-MobileNet to develop a lightweight model. The filters in the four intermediary layers of the S-MobileNet model were compressed using the L1-norm CNN compression technique. Overall, 156 filters were pruned out of a total of 10944, in S-MobileNet. The performance of the model is evaluated in four dimensions: with and without passing preprocessed data; Relu activation function vs Mish activation function; mish activation function with and without applying compression and across three CNN optimizers, namely Adam, RMSProp, and SGD. Our results demonstrated that using processed data in the model results in improved performance. The Mish activation function was shown to outperform the Relu activation function, and pruning specific layers was also shown to improve model performance. To conclude, our proposed model demonstrated a higher classification accuracy compared to benchmark approaches, whilst still being lightweight.

Acknowledgement

The authors extend their appreciation to the Deputyship for Research & Innovation, Ministry of Education in Saudi Arabia for funding this research work through the project number 445-9-512.

References

[1] T. L. Diepgen and V. Mahler, "The epidemiology of skin cancer," *British Journal of Dermatology*, vol. 146, pp. 1–6, 2002.

[2] A. R. Sulthana, M. Gupta, S. Subramanian, and S. Mirza, "Improvising the performance of image-based recommendation system using convolution neural networks and deep learning," *Soft Computing*, vol. 24, no. 19, pp. 14 531–14 544, 2020.

[3] J. Premaladha and K. Ravichandran, "Novel approaches for diagnosing melanoma skin lesions through supervised and deep learning algorithms," *Journal of medical systems*, vol. 40, no. 4, pp. 1–12, 2016.

[4] P. N. Srinivasu, J. G. SivaSai, M. F. Ijaz, A. K. Bhoi, W. Kim, and J. J. Kang, "Classification of skin disease using deep learning neural networks with mobilenet v2 and lstm," *Sensors*, vol. 21, no. 8, p. 2852, 2021.

[5] P. Tschandl, C. Rosendahl, and H. Kittler, "The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions," *Scientific data*, vol. 5, no. 1, pp. 1–9, 2018.

[6] F. Tushabe, E. Mwebaze, and F. Kiwanuka, "An image-based diagnosis of virus and bacterial skin infections," in *The International Conference on Complications in Interventional Radiology*, 2011, pp. 1–7.

[7] A. Ajith, V. Goel, P. Vazirani, and M. M. Roja, "Digital dermatology: Skin disease detection model using image processing," in *2017 International Conference on Intelligent Computing and Control Systems (ICICCS)*. IEEE, 2017, pp. 168–173.

[8] Y. George, M. Aldeen, and R. Garnavi, "Psoriasis image representation using patch-based dictionary learning for erythema severity scoring," *Computerized Medical Imaging and Graphics*, vol. 66, pp. 44–55, 2018.

[9] S. K. Patnaik, M. S. Sidhu, Y. Gehlot, B. Sharma, and P. Muthu, "Automated skin disease identification using deep learning algorithm," *Biomedical & Pharmacology Journal*, vol. 11, no. 3, p. 1429, 2018.

[10] N. Mittal, S. Tanwar, and S. K. Khatri, "Identification & enhancement of different skin lesion images by segmentation techniques," in *2017 6th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO)*. IEEE, 2017, pp. 609–614.

[11] M. A. Al-Masni, M. A. Al-Antari, M.-T. Choi, S.-M. Han, and T.-S. Kim, "Skin lesion segmentation in dermoscopy images via deep full resolution convolutional networks," *Computer methods and programs in biomedicine*, vol. 162, pp. 221–231, 2018.

[12] K. Roy, S. S. Chaudhuri, S. Ghosh, S. K. Dutta, P. Chakraborty, and R. Sarkar, "Skin disease detection based on different segmentation techniques," in *2019 International Conference on Opto-Electronics and Applied Optics (Optronix)*. IEEE, 2019, pp. 1–5.

[13] S. Bhadula, S. Sharma, P. Juyal, and C. Kulshrestha, "Machine learning algorithms based skin disease detection," *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, vol. 9, no. 2, pp. 4044–4049, 2019.

[14] N. S. A. ALEnezi, "A method of skin disease detection using image processing and machine learning," *Procedia Computer Science*, vol. 163, pp. 85–92, 2019.

[15] M. A. Albahar, "Skin lesion classification using convolutional neural network with novel regularizer," *IEEE Access*, vol. 7, pp. 38 306–38 313, 2019.

[16] A. Hekler, J. S. Utikal, A. H. Enk, A. Hauschild, M. Weichenthal, R. C. Maron, C. Berking, S. Haferkamp, J. Klode, D. Schadendorf *et al.*, "Superior skin cancer classification by the combination of human and artificial intelligence," *European Journal of Cancer*, vol. 120, pp. 114–121, 2019.

[17] M. A. Kadampur and S. Al Riyaee, "Skin cancer detection: Applying a deep learning based model driven architecture in the cloud for classifying dermal cell images," *Informatics in Medicine Unlocked*, vol. 18, p. 100282, 2020.

[18] T. Shanthi, R. Sabeenian, and R. Anand, "Automatic diagnosis of skin diseases using convolution neural network," *Microprocessors and Microsystems*, vol. 76, p. 103074, 2020.

[19] L. Hu, Q. Chen, L. Qiao, L. Du, and R. Ye, "Automatic detection of melanins and sebums from skin images using a generative adversarial network," *Cognitive Computation*, vol. 14, no. 5, pp. 1599–1608, 2022.

[20] Z. Lin, Y. Gao, and J. Sang, "Investigating and explaining the frequency bias in image classification*," *arXiv preprint arXiv:2205.03154*, 2022.

[21] S. I. Hossain, "Early diagnosis of lyme disease by recognizing erythema migrans skin lesion from images utilizing deep learning techniques."

[22] S. I. Hossain, J. d. G. de Herve, M. S. Hassan, D. Martineau, E. Petrosyan, V. Corbin, J. Beytout, I. Lebert, J. Durand, I. Carravieri *et al.*, "Exploring convolutional neural

networks with transfer learning for diagnosing Lyme disease from skin lesion images," *Computer Methods and Programs in Biomedicine*, vol. 215, p. 106624, 2022.

[23] S. Yu, D. Lee, and H. Yu, "Convolutional neural networks with compression complexity pooling for out-of-distribution image detection," in *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, 2021, pp. 2435–2441.

[24] A. A. Adegun and S. Viriri, "Deep learning-based system for automatic melanoma detection," *IEEE Access*, vol. 8, pp. 7160–7172, 2019.

[25] A. K. Waweru, K. Ahmed, Y. Miao, and P. Kawan, "Deep learning in skin lesion analysis towards cancer detection," in *2020 24th International Conference Information Visualisation (IV)*. IEEE, 2020, pp. 740–745.

[26] M. Goyal, A. Oakley, P. Bansal, D. Dancey, and M. H. Yap, "Skin lesion segmentation in dermoscopic images with ensemble deep learning methods," *IEEE Access*, vol. 8, pp. 4171–4181, 2019.

[27] K. Zafar, S. O. Gilani, A. Waris, A. Ahmed, M. Jamil, M. N. Khan, and A. Sohail Kashif, "Skin lesion segmentation from dermoscopic images using convolutional neural network," *Sensors*, vol. 20, no. 6, p. 1601, 2020.

[28] Q. Abbas, F. Ramzan, and M. U. Ghani, "Acral melanoma detection using dermoscopic images and convolutional neural networks," *Visual Computing for Industry, Biomedicine, and Art*, vol. 4, no. 1, pp. 1–12, 2021.

[29] M. S. Ali, M. S. Miah, J. Haque, M. M. Rahman, and M. K. Islam, "An enhanced technique of skin cancer classification using deep convolutional neural network with transfer learning models," *Machine Learning with Applications*, vol. 5, p. 100036, 2021.

[30] F. W. Alsaade, T. H. Aldhyani, and M. H. Al-Adhaileh, "Developing a recognition system for diagnosing melanoma skin lesions using artificial intelligence algorithms," *Computational and mathematical methods in medicine*, vol. 2021, 2021.

[31] D. A. Okuboyejo, O. O. Olugbara, and S. A. Odunaike, "Automating skin disease diagnosis using image classification," in *proceedings of the world congress on engineering and computer science*, vol. 2, 2013, pp. 850–854.

[32] D. Lu and Q. Weng, "A survey of image classification methods and techniques for improving classification performance," *International journal of Remote sensing*, vol. 28, no. 5, pp. 823–870, 2007.

[33] M. Goyal, A. Oakley, P. Bansal, D. Dancey, and M. H. Yap, "Skin lesion segmentation in dermoscopic images with ensemble deep learning methods," *IEEE Access*, vol. 8, pp. 4171–4181, 2019.

[34] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention.* Springer, 2015, pp. 234–241.

[35] L. Cai, J. Gao, and D. Zhao, "A review of the application of deep learning in medical image classification and segmentation," *Annals of translational medicine*, vol. 8, no. 11, 2020.

[36] A. Adegun and S. Viriri, "Deep learning techniques for skin lesion analysis and melanoma cancer detection: a survey of state-of-the-art," *Artificial Intelligence Review*, vol. 54, no. 2, pp. 811–841, 2021.

[37] C. A. Z. Barcelos and V. Pires, "An automatic based nonlinear diffusion equations scheme for skin lesion segmentation," *Applied Mathematics and Computation*, vol. 215, no. 1, pp. 251–261, 2009.

[38] Y.-N. Wang, X. Tian, and G. Zhong, "FFNet: Feature Fusion Network for Few-shot Semantic Segmentation," *Cognitive Computation*, vol. 14, no. 2, pp. 875–886, 2022.

[39] N. Xu and C. Li, "Image feature extraction in detection technology of breast tumor," *Journal of King Saud University-Science*, vol. 32, no. 3, pp. 2170–2175, 2020.

[40] S. Chatterjee, D. Dey, S. Munshi, and S. Gorai, "Extraction of features from cross correlation in space and frequency domains for classification of skin lesions," *Biomedical Signal Processing and Control*, vol. 53, p. 101581, 2019.

[41] A. Iqbal, M. Sharif, M. A. Khan, W. Nisar, and M. Alhaisoni, "Ff-unet: a U-shaped deep convolutional neural network for multimodal biomedical image segmentation," *Cognitive Computation*, vol. 14, no. 4, pp. 1287–1302, 2022.

[42] A. Glowacz and Z. Glowacz, "Recognition of images of finger skin with application of histogram, image filtration and K-NN classifier," *Biocybernetics and biomedical engineering*, vol. 36, no. 1, pp. 95–101, 2016.

[43] Z. Zhang, S. Ye, Z. Liu, H. Wang, and W. Ding, "Deep Hyperspherical Clustering for Skin Lesion Medical Image Segmentation," *IEEE Journal of Biomedical and Health Informatics*, 2023.

[44] J. Brownlee, "Basics of linear algebra for machine learning," *Machine Learning Mastery*, 2018.

[45] X. He, Y. Wang, S. Zhao, and X. Chen, "Joint segmentation and classification of skin lesions via a multi-task learning convolutional neural network," *Expert Systems with Applications*, p. 120174, 2023.

[46] P. Tschandl, "The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions," 2018. [Online]. Available: https://doi.org/10.7910/DVN/DBW86T

[47] W. Sae-Lim, W. Wettayaprasit, and P. Aiyarak, "Convolutional neural networks using mobilenet for skin lesion classification," in *2019 16th international joint conference on computer science and software engineering (JCSSE)*. IEEE, 2019, pp. 242–247.

[48] S. Nasiri, M. Jung, J. Helsper, and M. Fathi, "Deep-class at isic machine learning challenge 2018," *arXiv preprint arXiv:1807.08993*, 2018.

[49] N. Heller, E. Bussmann, A. Shah, J. Dean, and N. Papanikolopoulos, “Computer aided diagnosis of skin lesions from morphological features,” 2018.

[50] M. A. Khan, M. Y. Javed, M. Sharif, T. Saba, and A. Rehman, “Multi-model deep neural network based features extraction and optimal selection approach for skin lesion classification,” in *2019 international conference on computer and information sciences (ICCIS)*. IEEE, 2019, pp. 1–7.

[51] L. Hoang, S.-H. Lee, E.-J. Lee, and K.-R. Kwon, “Multiclass Skin Lesion Classification Using a Novel Lightweight Deep Learning Framework for Smart Healthcare,” *Applied Sciences*, vol. 12, no. 5, p. 2677, 2022.

[52] J. P. D’Haeyer, “Gaussian filtering of images: A regularization approach,” *Signal Processing*, vol. 18, no. 2, pp. 169–181, 1989.

[53] A. Kumar and S. S. Sodhi, “Comparative analysis of gaussian filter, median filter and denoise autoenocoder,” in *2020 7th International Conference on Computing for Sustainable Global Development (INDIACom)*. IEEE, 2020, pp. 45–51.

[54] H. Fitriyah and R. C. Wihandika, “An analysis of rgb, hue and grayscale under various illuminations,” in *2018 International Conference on Sustainable Information Engineering and Technology (SIET)*. IEEE, 2018, pp. 38–41.

[55] F. Al-Areqi and M. Z. Konyar, “Effectiveness evaluation of different feature extraction methods for classification of covid-19 from computed tomography images: A high accuracy classification study,” *Biomedical Signal Processing and Control*, vol. 76, p. 103662, 2022.

[56] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[57] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[58] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[59] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[60] R. S. A. Kareem, A. G. Ramanjineyulu, R. Rajan, R. Setiawan, D. K. Sharma, M. K. Gupta, H. Joshi, A. Kumar, H. Harikrishnan, and S. Sengan, "Multilabel land cover aerial image classification using convolutional neural networks," *Arabian Journal of Geosciences*, vol. 14, no. 17, pp. 1–18, 2021.

[61] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[62] D. Misra, "Mish: A self regularized non-monotonic neural activation function," *arXiv preprint arXiv:1908.08681*, vol. 4, no. 2, pp. 10–48 550, 2019.

[63] S. Cai, Y. Shu, G. Chen, B. C. Ooi, W. Wang, and M. Zhang, "Effective and efficient dropout for deep convolutional neural networks," *arXiv preprint arXiv:1904.03392*, 2019.

[64] Y. Zhou, K. Huang, C. Cheng, X. Wang, A. Hussain, and X. Liu, "FastAdaBelief: improving convergence rate for belief-based adaptive optimizers by exploiting strong convexity," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[65] A. F. Costa, G. Humpire-Mamani, and A. J. M. Traina, "An efficient algorithm for fractal analysis of textures," in *2012 25th SIBGRAPI Conference on Graphics, Patterns and Images*. IEEE, 2012, pp. 39–46.

[66] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[67] D. Gutman, N. C. Codella, E. Celebi, B. Helba, M. Marchetti, N. Mishra, and A. Halpern, "Skin lesion analysis toward melanoma detection: A challenge at the international symposium on biomedical imaging (isbi) 2016, hosted by the international skin imaging collaboration (isic)," *arXiv preprint arXiv:1605.01397*, 2016.

[68] A. G. Pacheco, G. R. Lima, A. S. Salomão, B. Krohling, I. P. Biral, G. G. de Angelo, F. C. Alves Jr, J. G. Esgario, A. C. Simora, P. B. Castro *et al.*, "Pad-ufes-20: A skin lesion dataset composed of patient data and clinical images collected from smartphones," *Data in brief*, vol. 32, p. 106221, 2020.

[69] T. Mendonça, P. M. Ferreira, J. S. Marques, A. R. Marcal, and J. Rozeira, "Ph 2-a dermoscopic image database for research and benchmarking," in *2013 35th annual international conference of the IEEE engineering in medicine and biology society (EMBC)*. IEEE, 2013, pp. 5437–5440.