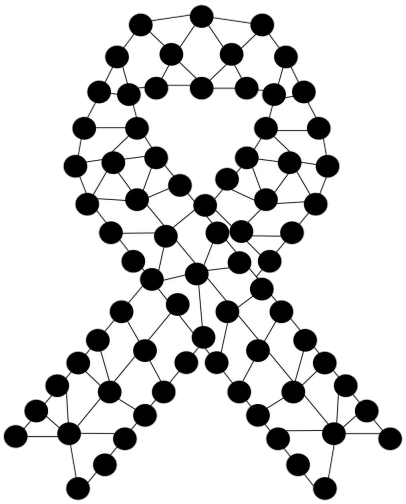
King Saud University
College of Computer and Information Sciences
Computer Science Department

Advanced Skin Cancer Detection Using Deep Learning



Prepared by:

Seba Alhejaili	441200901
Fatimah Alhumaidhi	441200921
Lama Almubarak	441200964
Joud Alismail	441201002
Halah Altammami	441201366

Supervised by: Dr. Mai Alzamel

Research project for the degree of Bachelor in Computer Science
Second Semester 1444/2023

ACKNOWLEDGMENTS

In the name of Allah, the Most Gracious and the Most Merciful, our sincere thanks and gratitude to our supervisor, Dr. Mai Alzamel, for her continuous guidance, encouragement, and support throughout our time as her students. We also would like to thank our examiners, Dr. Sarab Almuhaideb, and Dr. Rasha Aleidan, for their invaluable advice and comments on the earlier version of the research.

We are deeply grateful to our parents for their support, prayers, and endless sacrifices to make us reach this step, the end of this research. In appreciation of our siblings, we thank them for the motivation they gave us every time they saw us working.

Finally, we would like to thank our friends and colleagues for their support, understanding, and belief in us.

English Abstract

Skin cancer is one of the most dangerous forms of cancer. The more the disease progresses, the more the survival rate steeply decreases. In cases of lethal diseases such as Melanoma, diagnosis in the preliminary stages plays a significant role in determining the probability of getting cured. Nonetheless, the detection of skin cancer in the preliminary stages is an arduous and expensive process. Although this type of cancer is visible to the eye, that does not make it easier to diagnose. People with skin lesions are perplexed because they cannot figure out whether this skin lesion is a cancerous tumor or just a normal lesion. Examining all pigmented skin lesions with surgical treatments causes significant soreness and scarring. Because of that, there is a rising need for an automatic and painless skin cancer detection system with high accuracy. Recently, Machine Learning and Deep Learning have demonstrated promising results in prediction and classification, skin cancer detection had been performed exceptionally well by them. This research tackled the problem of detecting skin cancer by building Deep Learning models that can automatically detect skin cancer using the pre-trained models of Convolutional Neural Network (CNN), namely ResNetv2, VGG16, EfficientNet-B5, and EfficientNet-B7, and comparing them with a Machine Learning model, namely the Support Vector Machine (SVM), in order to determine whether or not the sample is cancerous. The EfficientNet-B7 model with data augmentation outperformed the other presented models with an accuracy of 86.91%. This document aims to enrich research in the field of skin cancer detection.

Arabic Abstract

سرطان الجلد هو أحد أخطر أشكال السرطان. كلما تقدم المرض، انخفض معدل البقاء على قيد الحياة بشكل حاد. في حالات الأمراض الفتاكة مثل سرطان الجلد، يلعب التشخيص في المراحل الأولية دورًا هامًّا في تحديد احتمال الشفاء. ومع ذلك، فإن الكشف عن سرطان الجلد في المراحل الأولية عملية شاقة ومكلفة. على الرغم من أن هذا النوع من السرطان مرئي للعين، إلا أن ذلك لا يسهل التشخيص. يشعر الأشخاص الذين يعانون من الآفات الجلدية بالحيرة لأنهم لا يستطيعون معرفة ما إذا كانت هذه الآفة الجلدية ورما سرطانيا أم مجرد آفة طبيعية. يؤدي فحص جميع الآفات الجلدية المصطبغة بالعلاجات الجراحية إلى وجع وتندب كبيرين. وبسبب ذلك، هناك حاجة متزايدة لنظام الكشف التلقائي وغير المؤلم عن سرطان الجلد بدقة عالية. في الآونة الأخيرة، أظهر التعلم الآلي والتعلم العميق نتائجًا واعدة في التنبؤ والتصنيف، وقد تم إجراء الكشف عن سرطان الجلد بشكل جيد للغاية من قبلهم. يتناول البحث مشكلة الكشف عن سرطان الجلد من خلال بناء نماذج التعلم العميق التي يمكنها الكشف تلقائيا عن سرطان الجلد باستخدام المتغيرات المدربة مسبقا للشبكة العصبونية الملتفة (CNN) ، وهي ResNetv2 ، و VGG16 و EfficientNet-B5 و EfficientNet-B7 ، ومقارنتها بنموذج التعلم الآلي، وهو آلة المتجهات الداعمة (SVM) ، من أجل تحديد ما إذا كانت العينة سرطانية أم لا. إذ تفوق نموذج EfficientNet-B7 مع نموذج مولد البيانات (GAN) على النماذج الأخرى المقدمة بدقة 86.91% . سنحاول بإجراء هذا البحث إثراء مجال الكشف عن سرطان الجلد.

Contents

List of Figures

List of Tables

List of Notations

Adam	Adaptive Moment Estimation
ANN	Artificial Neural Network
BGD	Batch Gradient Descent
CNN	Convolutional neural network
DL	Deep Learning
GAN	Generative Adversarial Network
HAM10000	Human Against Machine with 10000 images
ISIC	International Skin Imaging Collaboration
KNN	K-Nearest Neighbor
MBGD	Mini-Batch Gradient Descent
ML	Machine Learning
ReLU	Rectified Linear Unit
ResNet	Residual Neural Network
SGD	Stochastic Gradient Descent
SVM	Support Vector Machine
Tanh	Hyperbolic Tangent

1 Introduction

Skin cancer is a common type of cancer that arises from the skin. The abnormal growth of skin cells causes it to spread, which makes these cells invade other parts of the body [1]. A number of recent studies have revealed that skin cancer is among the most dangerous types of cancer. [2].

Although the skin is an easily accessible site for examination, it is difficult to diagnose skin cancers because many benign lesions visually resemble malignant ones. Consequently, an accurate clinical diagnosis is contingent upon a biopsy and subsequent histopathological examination. Clinicians face the difficulty of determining which and how many skin lesions to biopsisize, typically based on visual inspection and palpation. Despite many efforts, there remains a need for an automated, noninvasive diagnostic method capable of directing biopsies to characterize skin lesions [3].

As with all types of cancer, it has been demonstrated that screening and preliminary detection of skin cancer is the most reliable indicator of complete recovery [4]. Early detection of skin cancer results in a 94% ten-year survival rate [4]. Nevertheless, this survival rate plummets as the cancer advances and reaches later stages. When Melanoma, the deadliest form of skin cancer [5], is diagnosed in its final stage, the ten-year survival rate decreases greatly to 15% [4].

With the new technology, it is possible to detect skin cancer at an early stage [2], where it has been shown by different existing systems that computer vision contributes significantly to medical image diagnosis [2]. Using machine learning and deep learning, clinicians and patients can streamline diagnostic and therapeutic procedures in a single visit [3].

As one of the most threatening forms of cancer, skin cancer detection will be the focus of our research. We will display the performance of the machine learning model, Support Vector Machine (SVM), and the deep learning models, Convolutional Neural Network (CNN), and Convolutional Neural Network with data augmentation using Generative Adversarial Network (GAN).

1.1 Problem Statement

Skin cancer is regarded as one of the cruelest forms of cancer. Multiple factors, including the high cost of the examination, the pain caused by traditional diagnostic methods, the scarring that occurs during the examination process, and the lengthy process of detecting cancer, discourage patients from undergoing early examination. Early detection of skin cancer saves a great deal of time and effort. It also increases the patient's likelihood of recovery. The development of science and technology has facilitated the discovery and classification of diseases. Using machine learning and deep learning, patients are now able to take photographs of their lesions and send them to a specialist doctor for identification and to initiate treatment. For skin cancer detection, we will in this research develop a machine learning model using SVM, and deep learning model using CNNs and GANs.

1.2 Goals and Objectives

We intend in our research to build an efficient and robust deep learning model to classify skin lesions into benign and malignant classes.

Objectives:

In order to achieve our intended goal, the following objectives will be met:

- Reading previous literature on the problem to have a clear view and define the problem's constituents and scope.
- Collecting a public dataset of benign/malignant lesions images.
- Preprocessing the dataset.
- Studying and investigating existing state-of-the-art deep learning classification models in order to choose the most suitable ones to tackle our problem.
- Conducting a set of experiments on the chosen dataset.
- Implementing the proposed models.
- Evaluating our models' performance by conducting multiple performance measurements.
- Comparing our results with a machine learning model, which is the Support Vector Machine (SVM).

1.3 Proposed Solution

This research seeks to identify skin lesions using Computer-Aided Diagnosis (CAD) systems created using a CNN classifier, as they have proven effective for image classification tasks. We will build our system to take the input image, resize it and remove any noise. The model then segments the input image to separate the lesion from the surrounding skin. Finally, the model will classify the lesion into one of two categories: either benign or malignant. We will train our model using a dataset of images taken from the International Image Collaboration Archive (ISIC) [6]. We will use GANs to generate more lesion images in order to overcome any shortage of data. Eventually, the purpose of this research is to develop and train these deep learning models, as well as present measurements that provide a better understanding of the performance of the various approaches on specific medical classification tasks.

1.4 Research Scope

Within the framework of deep learning, the primary focus of our research will be on diagnosing skin cancer from images by detecting signs of skin cancer in the images. The model is supposed to show whether a lesion is benign or malignant. Due to the limited number of datasets available for dark skin tones, the scope of our research is limited to light skin tones, and results may not be promising for those with darker skin [7].

1.5 Research Significance

In recent years, the field of skin cancer detection using deep learning has developed significantly and is still under development and improvement. The outcome of this study, a model to detect skin cancer using deep learning, will enrich this field and provide numerous benefits. First, the developed models will contribute to reducing the time and effort used to detect skin cancer by surgical methods. Second, it will help to avoid the high cost of detecting skin cancer. Finally, it will increase the rate of early detection due to the ease of detection method and its difference from the painful surgical methods that cause scarring and are usually avoided by patients, which increases the recovery rate.

1.6 Ethical and Social Implications

In regards to ethical standards and social implications that we must respect, there is no violation of any of these norms in our research. Recent advances in skin cancer

detection methods have raised a number of new ethical and legal issues. Usually, these methods require a large number of photos of pigmented moles from many individuals. Consequently, the privacy and confidentiality of data owners could be violated, resulting in ethical issues. In order to protect patients from any potential harm, we selected a publicly published dataset [6], which allowed usage and sharing as long as the source was cited.

1.7 Report Organization

The rest of this document is organized into eight chapters. Chapter 2 demonstrates the background including the relevant concepts. Chapter 3 reviews different approaches used in previous works on skin cancer detection. Chapter 4 presents our methodology to tackle the problem. Chapter 5 includes the experimental design. Chapter 6 presents the implementation of our methodology and the limitations and issues that accrued during this phase. Chapter 7 shows the details of the experiments and a discussion of the results. Lastly, chapter 8 concludes the report.

2 Background

This chapter demonstrates the techniques, models, and metrics mentioned and/or used throughout this research, including definitions, explanations, and some illustrative examples.

2.1 Skin Lesion Classification

In most cases, skin lesions are harmless. However, some of them may be cancerous. A variety of types of skin cancer exist, including Melanoma, Basal-Cell Carcinoma (BCC), and Squamous-Cell Carcinoma (SCC) that could be benign or malignant [8][9]. It is therefore necessary to detect them. Using the ABCD rule can help to detect some malignant lesions. In this rule, the letter 'A' refers to "Asymmetry", which means that the lesion is not of a uniform shape. The letter 'B' represents "Border", which indicates that the borders are not well-defined. Usually, cancerous lesions have more than one color, and the letter 'C' refers to "Color", which indicates the color of the lesion. Finally, the letter 'D' stands for "Diameter", which represents a diameter of more than 6 mm, which is typical for Melanoma lesions [10].

2.2 Machine Learning

Machine learning (ML) is a branch of computer science that was invented in 1959 by Arthur Samuel, who described it as a field of study that enables computers to gather knowledge via experience without the need for explicit programming [11]. This term can also refer to the process of solving a practical problem by collecting a dataset and developing a statistical model algorithmically based on that dataset [12]. Due to the advent of ML, computers can now perform tasks like classification, clustering, prediction, pattern recognition, etc [13]. Thefour categories of ML algorithms based on their learning technique are: Supervised Learning, Unsupervised Learning, Semi-Supervised Learning, Reinforcement Learning.

2.2.1 Support Vector Machine

Support Vector Machine (SVM) is a supervised learning algorithm that was invented by Vapnik and Chervonenkis [14]. Statistical learning theory states that SVM is considered one of the most effective ML algorithms for classification and regression. It has been successfully applied to the recognition of images, categorizing texts, diagnosing medical conditions, remote sensing, and classifying motions [15].

This algorithm works according to the principle of margin calculation. Each piece of data is represented as a point in n-dimensional space, and the coordinates of this point are called features. SVM classifies the datasets into classes by trying to find a hyperplane that splits the data as shown in Figure 1. It works by maximizing the distance between the closest data point in both classes and the hyperplane [16].

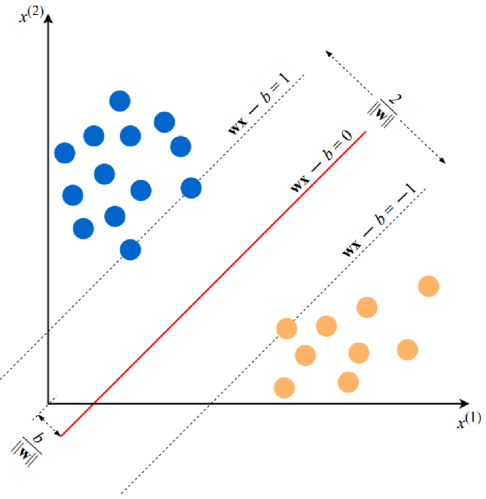


Figure 1: An example of an SVM model for two-dimensional feature vectors [12].

2.2.2 Random Forest

Random Forest is a ML technique that involves training individual classifiers and combining their predictors. It is reserved exclusively for decision tree classifiers and is employed for classification or regression problems. This method generates a set of decision trees from a randomly selected subset of training samples (Figure 2), then regroups the votes from the various decision trees to determine the final class of the test item [17].

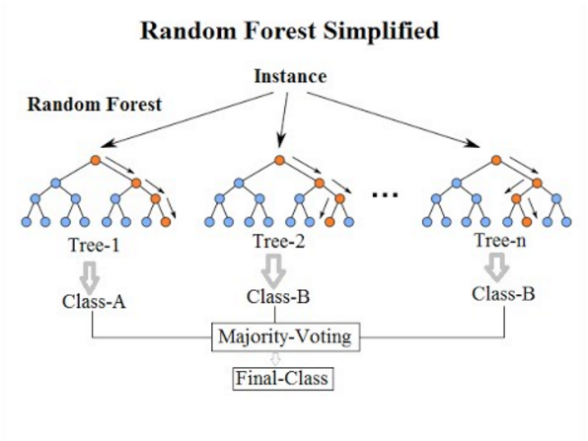


Figure 2: Random Forest algorithm mechanism [17].

2.2.3 Extreme Gradient Boosting Algorithm

Before defining the Extreme Gradient Boosting Algorithm (XGBoost), a related concept, Boosting, must be clarified. Boosting is an ML technique that is applied in classification and regression problems. It generates a weak learner at each step and adds them to the model as a whole. When the direction of the gradient of the loss function is used to determine the weak learner, it is called a Gradient Boosting Machine (GBM). The primary distinction between RF and GBM is that in RF, trees are constructed independently of one another, whereas GBM adds a new tree to complement already constructed trees [18]. The XGBoost algorithm is a scalable tree-boosting ML system [18] that is extensively used by data scientists to achieve state-of-the-art results on many challenges [19].

2.3 Artificial Neural Networks

An Artificial Neural Network (ANN) is a machine learning method inspired by how biological neural networks work in the brain [20]. The first wave of interest in neural networks (known as Parallel Distributed Processing or Connectionist Models) appeared after the introduction of simplified neurons in 1943 by McCulloch and Pitts [21]. ANN is motivated by the enormously distributed parallel computation in the brain, which allows it to be successful at complex control and recognition/classification tasks [22]. The biological neural network that manages those tasks can be mathematically modeled by a weighted, directed graph of greatly interconnected nodes (neurons) [22]. Commonly, ANNs are structured in layers (Figure 3) composed of interconnected nodes. These nodes contain an activation function. The input layer presents a pattern (data) to the network, followed by communication to one or more hidden layers. The actual processing within the hidden layers is done by utilizing a system of weighted connections. Then, the hidden layers are connected to an output layer, where the output is the answer [23].

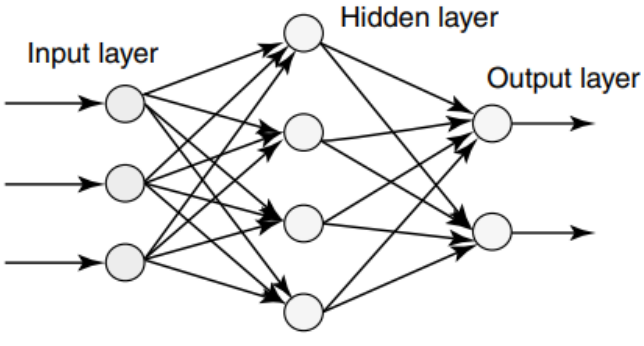


Figure 3: Architecture of ANN [21].

The learning process of an ANN involves updating the connection strength (weight) of a node (neuron) [24]. One of the ways to ensure lower error rates is by tuning the weights properly because it makes the model reliable and increases its generalization [25]. An ANN learns based on the difference between predicted and true values, and adjusts its weight, producing a close to true output [24]. In neural networks, Back-Propagation is the process of fine-tuning the weights based on error rates [25]. The name came from the idea of sending information back into the network when the expected output is different from the actual output.

2.3.1 Activation Functions

Activation Functions are essential in an ANN because it helps in learning and understanding non-linear and complicated mappings between the inputs and corresponding outputs. The prediction accuracy of the neural network is dependent on the number of layers used, and a more critical factor that makes a difference is the activation function used. In a neural network, the net inputs are the most important units, which are processed and changed into an output result known as the Unit's Activation by applying the activation function, also called the Transfer Function or Threshold Function, which is a scalar to scalar transformation [26].

2.3.1.1 Sigmoid

The sigmoid function is a mathematical function commonly used in neural networks trained using back-propagation algorithms for binary classification [26][27]. Sigmoid is a non-linear function because its S-shaped curve is limited between 0 and 1 (Figure 4), and because it is not symmetric about zero, all output values have the same sign [26] [28]. While they are not identical, Sigmoid functions are very similar to biological neuron input-output relationships [28]. By using it, we provide a limit on the output value of neurons. Here is how the sigmoid function can be expressed:

$$f(x) = \frac{1}{1 + e^{-x}}$$

In this equation, $f(x)$ represents sigmoid activation function, x is the net input value, and e is euler's constant [27].

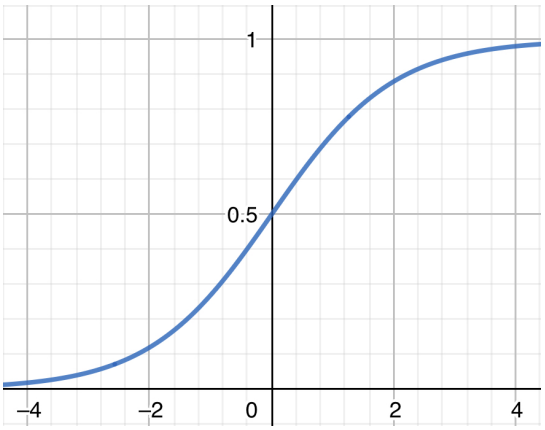


Figure 4: Sigmoid function plot.

2.3.1.2 Hyperbolic Tangent Function

The Hyperbolic Tangent Function, also called the Tanh Function, shares a lot of similarities with the Sigmoid function, which is why they are commonly used [26]. However, Tanh functions are preferred due to their limit, which lies between -1 and as shown in Figure 5. They also differ in symmetry, as Tanh is symmetric about the origin, resulting in output values with different signs [26][29]. However, both Tanh and Sigmoid functions are non-linear with S-shaped curves [30]. The formula below describes Tanh:

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

In this case, $f(x)$ is the hyperbolic tangent function of the net input x, and e is an euler's constant [29].

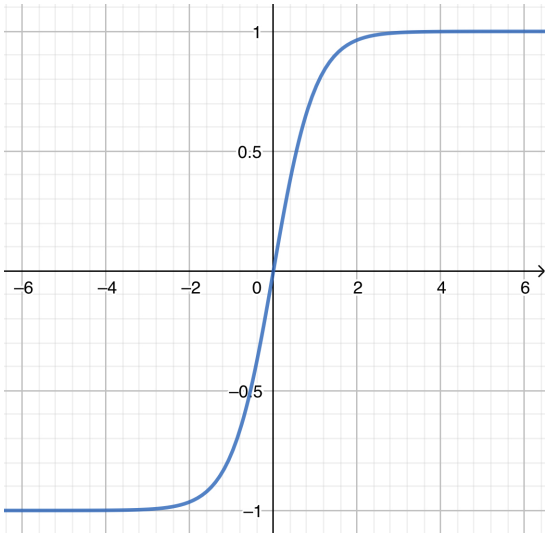


Figure 5: Tanh plot.

2.3.1.3 Rectified Linear Units Function

Rectified Linear Units Function (ReLU) is a non-linear activation function [26]. ReLU has become very popular in recent years. It is used to avoid and rectifies the vanishing gradient problem. ReLU is used in hidden layers of ANNs, especially in CNN. Compared to Sigmoid and Tanh, ReLU is less computationally expensive [31]. It is defined mathematically as:

$$f(x) = max(0, x)$$

In the ReLU function, all the neurons are not active simultaneously, making it more efficient than other functions. This means the neuron will deactivate when the output is zero [26]. As illustrated in Figure 6, the gradient is 0 for negative inputs, indicating that the weights will not get modified during the descent. The neurons that enter this state will then stop responding to variations in inputs, and they will be stuck forever in an inactive state and "dying"; this is known as the "Dying ReLU" problem [31].

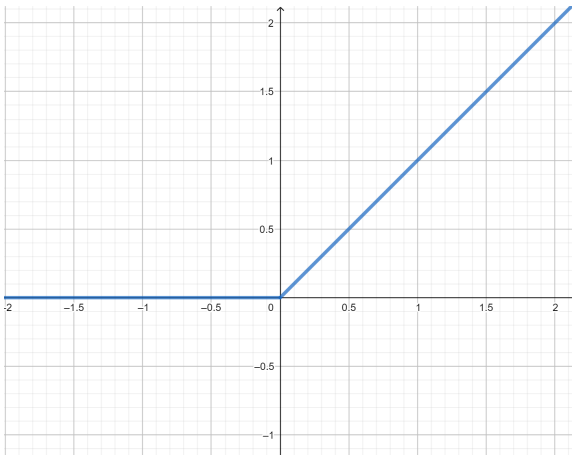


Figure 6: ReLU plot.

2.3.2 Optimization

The primary objective of ML is to minimize the Cost Function, which is the difference between the predicted output and the actual output. When the cost function is minimized, predictions get closer to the correct answers, which leads to optimization. Optimization is the key component that helps a model to train better during back-propagation by minimizing loss error [25].

Recently, several back-propagation optimization algorithms have been used. In this subsection, we will mention some of them.

2.3.2.1 Gradient Descent

Gradient Descent (GD) is an iterative algorithm that is one of the most widely known optimization algorithms and the most common method for optimizing neural networks [32] [33]. Its objective is to minimize a cost function. GD is When the cost function converges to a minimum value, the iterations stop and the cost function is no longer reduced. The approach of GD optimization is used in the Back-propagation algorithm wherein the gradient of the loss function is computed to adjust the weight of neurons. There are three different types of this method, which are: Stochastic Gradient Descent (SGD), Batch Gradient Descent (BGD), and Mini Batch Gradient Descent (MBGD) [32].

In BGD, the error is computed for every example within the training dataset, but the model will be updated only after completing the evaluation of all the training examples. The main advantage of the BGD algorithm is its efficiency of computation, where it produces a stable convergence and a stable error gradient. Nevertheless, the disadvantage of BGD is that the stable error gradients can sometimes lead to a model that can't achieve its optimal state of convergence. Furthermore, the algorithm requires the entire training dataset to be in memory and available [32].

In contrast to BGD, SGD within the dataset calculates the error for each training example, and parameters are updated for each training example. For the specific problem, this could result in SGD being faster than BGD. One of the advantages of SDG is that it provides a detailed rate of improvement due to frequent updates (Figure 7). However, frequent updates are more computationally expensive than the BGD method. The frequency of those updates can also produce noisy gradients, causing the error rate to fluctuate instead of decreasing slowly [32].

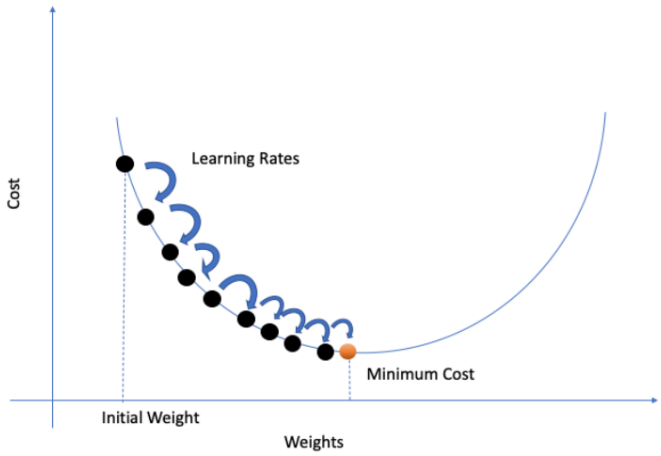


Figure 7: Stochastic Gradient Descent [34].

MBGD is obtained by combining the concepts of SGD and BGD. In this approach,

the training dataset is split into small batches. For each of the batches, an update is performed. Therefore, it creates a balance between the tenacity of SGD and the effectiveness of BGD. A neural network can be trained with this approach, so it is primarily used in DL [32].

2.3.2.2 Root Mean Square Propagation

Root Mean Square Propagation (RMSProp) is a well-known algorithm for training adaptive stochastic Deep Neural Networks (DNNs). The gradient is accumulated by the RMSProp algorithm by modifying Adagrad. The average of gradients is exponentially weighted. RMSProp discards historical and retains only current gradient knowledge [35].

2.3.2.3 Adaptive Moment Estimation

Adaptive Moment Estimation (Adam) is an optimization algorithm of type SGD. For each parameter, Adam calculates adaptable learning rates. In neural networks, there are numerous step-size strategies, but this is one of the most prevalent. The name is a reference to Adaptive Moments. Adam reduces computing costs, requires less memory execution, and is gradient diagonal rescaling invariant [35].

2.4 Deep Learning

Deep learning (DL) uses neural networks with multiple hidden layers; hence the word deep. One of the most significant potentials of DL is its capability of feature representation learning, which contrasts with ML approaches that require tailor-made algorithms to extract features. Based on the available data and the task expected, feature representation learning makes DL networks able to build the right set of features. The result of this ability is that DL has shown remarkable accuracy in a variety of applications that were previously considered complex or impossible [36].

2.4.1 Convolutional Neural Networks

A Convolutional Neural Network (CNN) is a deep neural network that has an architecture designed for image analysis that is trained through the process of supervised learning [37]. The basic composition of CNN architecture can be divided into five layers, which are: input layer, convolution layer, pooling layer, fully connected layer, and output layer. In the input layer, the input raw dataset will be directly input to the input layer. One image is inputted by its pixel value into the input layer. Convolutional layer uses a filter to extract features from input and produce a

feature map that provides a summary of the features that have been identified in the input [38]. Pooling layer (Down-sampling layer), its main task is to finish the second extraction of the feature data followed by the convolution layer. The fully connected layer is responsible for making all the feature maps connected together as input. This layer integrates and normalizes the abstracted features of the previous convolutions to give a probability for abundant conditions. Eventually, the number of neurons in the output layer is set according to the required conditions. If the classification is required, the number of neurons is generally related to the number of categories to be classified [39]. CNN uses labeled data such as dermoscopic images with their corresponding diagnoses or ground truths to determine a relationship between the input data and the labels. In such a manner, CNN can apply learned operations to unknown images and classify them using the extracted features [37]. Clinical dermatology and dermatopathology diagnoses largely through the recognition of visual patterns. Therefore, using CNN could be helpful in developing additional and/or improved clinically meaningful databases [40]. Figure 8 demonstrates the five layers of CNN.

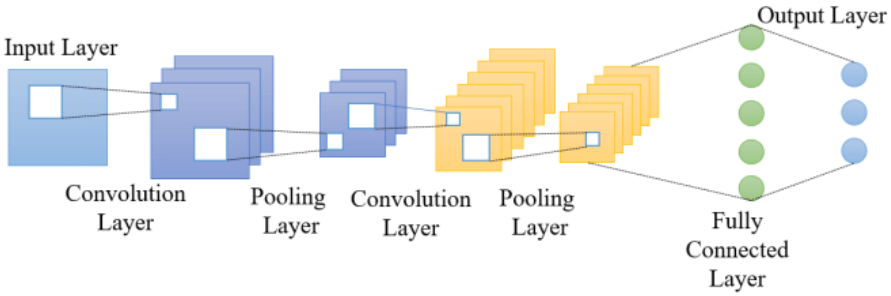


Figure 8: Basic architecture of CNN [39].

2.4.2 Residual Neural Networks

As layers increase in CNN, a problem called the Vanishing/Exploding Gradient occurs. This problem causes the gradient to be zero or too large. Hence, when the number of layers increases, the training and test error rate increases as well [41].

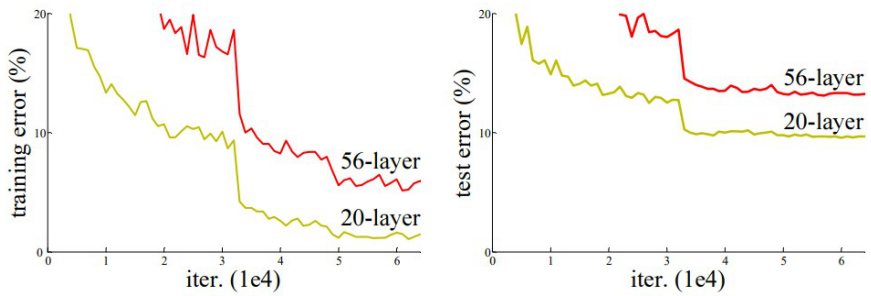


Figure 9: Comparison of 20-layer versus 56-layer architecture [41].

In Figure 9, it is obvious that a 56-layer CNN gives a higher error rate on both training and testing dataset than a 20-layer CNN architecture. After analyzing more error rates, the authors in [41] were able to reach the conclusion that it is caused by the vanishing/exploding gradient. A Residual Neural Network (ResNet) was proposed in 2015 by researchers at Microsoft Research for this purpose. To make the problem of the vanishing/exploding gradient solvable, ResNets architecture introduced the concept called Residual Blocks. In this network, a technique called Shortcut Connections (Figure 10) is used. This technique connects activation $\mathcal{F}(x)$ of a layer to further layers by skipping some layers in between, which forms a residual block $\mathcal{H}(x)$. ResNets are made by stacking these residual blocks together. The approach behind this network is that the network fits the residual mapping instead of layers learning the underlying mapping. Thus, instead of saying $\mathcal{H}(x)$, initial mapping, let the network fit: $\mathcal{F}(x) := \mathcal{H}(x) - x$ which gives $\mathcal{H}(x) := \mathcal{F}(x) + x$.

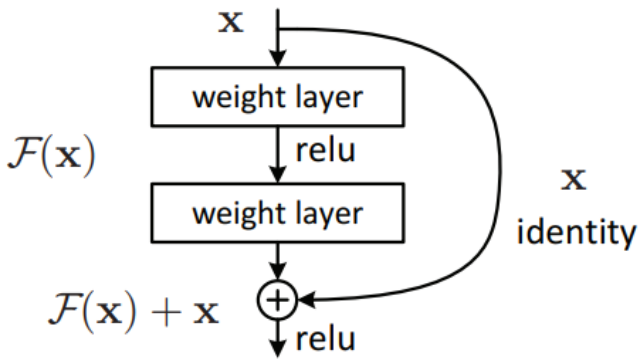


Figure 10: Representation of a residual block with a shortcut connection [41].

The idea of using shortcut connections is that if any layer hurt the performance of architecture, then it will be skipped by regularization. In this way, we can train very deep neural networks without the problems caused by vanishing/exploding gradients [41].

2.4.3 Generative Adversarial Networks

A network called Generative Adversarial Network (GAN) was first introduced in a 2014 paper by Goodfellow et al. [42]. GANs are semi-supervised DL models that consist of two parts: a generative model and an adversary model. The generative model generates samples by passing random noise through a multilayer network, which then sends the output to the adversary model. The adversary model is a discriminative model that learns to determine whether a sample is from the model distribution or the data distribution [42]. Figure 11 shows a high-level visualization

of the GAN structure. This framework was based on the two-player minimax game with the following objective function:

$$\min_G \max_D V(G,D) = \mathbb{E}_{x\sim pdata(x)}[\log D(x)] + \mathbb{E}_{z\sim pz(z)}[\log(1 - D(G(z)))]$$

In this equation, the generator (G) aims to minimize correct labels for samples from its distribution z, while the discriminator (D) strives to maximize the probability of giving the proper label to both training instances x and samples from the generator $G(z)$. GANs showed great success in image generation tasks. One example would be the StyleGAN used in [43], which has produced state-of-the-art outcomes for data-driven, unconditional generative image synthesis.

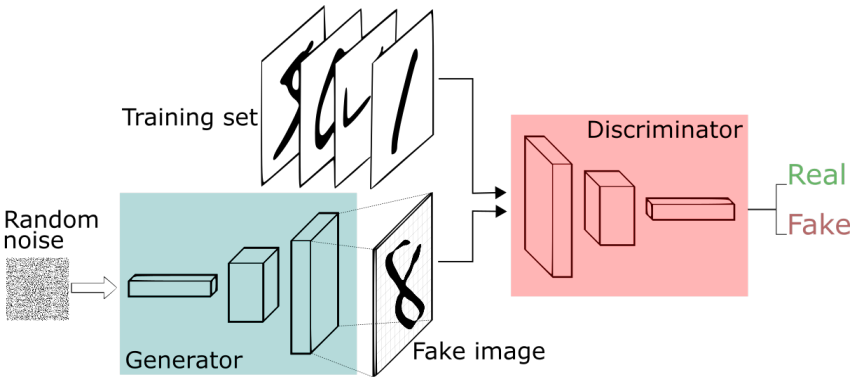


Figure 11: Structure of GAN [44].

2.4.4 Transfer Learning

To solve a new task, a human learner is able to transfer relevant knowledge from previous experiences. In contrast, most ML algorithms are designed to address isolated problems. In order to change this, we have to explore the role of Transfer Learning. Through transfer learning, knowledge is transferred from one or more source tasks to improve learning in a related target task [45]. Figure 12 below illustrates some intuitive examples of transfer learning, which are based on the ability of human beings to transfer knowledge across domains. The purpose of transfer learning is to improve performance in a new domain or reduce the required number of labeled examples. [46]. In general, we will be more successful at mastering a new task if it is closely related to our previous experience [45]. Besides, the similarities between domains do not always facilitate learning because sometimes the similarities may be misleading [46].

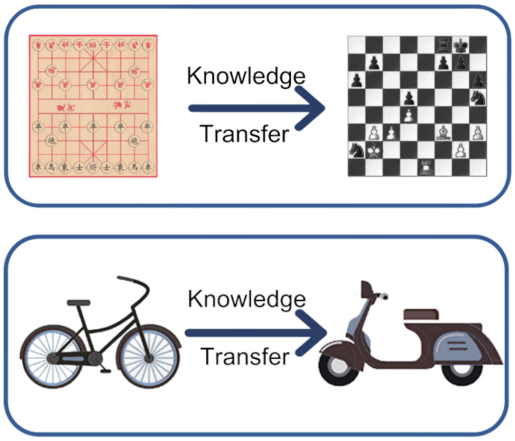


Figure 12: Intuitive examples about Transfer Learning [46].

There are several examples of transfer learning. In the following, we will demonstrate some of them:

I. MobileNet

MobileNet is a lightweight class of Deep Convolutional Neural Networks (DCNNs) with streamlined architectures [47][48]. The size and performance of these networks are vastly superior to those of many other popular models [47]. This technology is used in a variety of fields, including object detection, recognition, mobile, and embedded vision applications [47][49]. However, it has limited memory, energy, and power limits hardware deployment. Moreover, when the number of parameters and the model size gets reduced, the overall accuracy of the model will decrease [49].

II. Inception

While scaling up the network, the inception module significantly improves computational efficiency. This module makes excellent use of its internal architecture for two reasons. First, it increases network width and depth to improve overall performance because a bigger size means a larger number of parameters, which causes overfitting in deep networks, leveraging sparsity even within convolutions to perform better results. Therefore filter-level sparsity blocks are introduced in the inception module. Second, reduce input dimensions and eliminate computational bottlenecks [50].

III. Visual Geometry Group

Visual Geometry Group, also known as VGG, is a pre-trained CNN model. It was first introduced in a paper titled "Very Deep Convolutional Networks for Large-Scale Image Recognition" [51]. The architecture of VGG is consisting of 16 convolutional layers, each layer has a 3x3 filter size, followed by a pooling layer, a final dense network with two hidden layers, and lastly the output layer [51]. This network was

trained on over one million images and gained popularity after it finished first in the ImageNet Challenge in 2014.

IV. EfficientNets

EfficientNets refer to a family of scaling techniques that uniformly scale all depth/width/resolution dimensions using a simple and highly effective compound coefficient. In scaling up MobileNets and ResNet, EfficientNets demonstrated their efficacy. The creators of EfficientNets achieved considerably higher accuracy and efficiency than previous ConvNets [52].

2.5 Image Processing

An image can be thought of as a two-variable function $f(x, y)$, with x and y being plane coordinates, and the range of f at any given pair of coordinates (x, y) is referred to as the intensity or gray level of the image at that location. Images are considered digital when x, y, and the range values of f are all finite, discrete quantities with a unique position and value, these components are known as pixels. Digital image processing is the practice of employing a digital computer to process digital pictures [53].

2.5.1 Image Denoising

Image Denoising is the process of reducing noise in an image. One way of applying denoising techniques is for image smoothing and filtering. An example of image smoothing is the Gaussian Filter, which is defined as follows:

$$g(x, y) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{d^2}{2\sigma^2}}$$

Where $d = \sqrt{(x - x_c)^2 + (y - y_c)^2}$ is the distance of the neighborhood pixel (x, y) from the center pixel (x_c, y_c), and σ denotes the standard deviation of the distribution, which assumed to have 0 mean [54]. One intuitive filter named Median Filter is a smoothing filter that uses the median value of a pixel's neighborhood to be replaced by it [54]. Another filter called Morphological Transformations uses the concept of "Erosion" and "Dilation" to filter noise in images. Erosion and Dilation are two basic morphological operators, the former diminishes the foreground of the object's borders, and the latter dilates it. Other types called Opening (erosion followed by dilation) and Closing (dilation followed by erosion) are also very common [55].

In Figure 13, Ebrahimi et al. [56] used Morphological Transformations to detect hair in a lesion image, interpolated hair pixels with neighboring pixels, and applied a median filter for image smoothing.

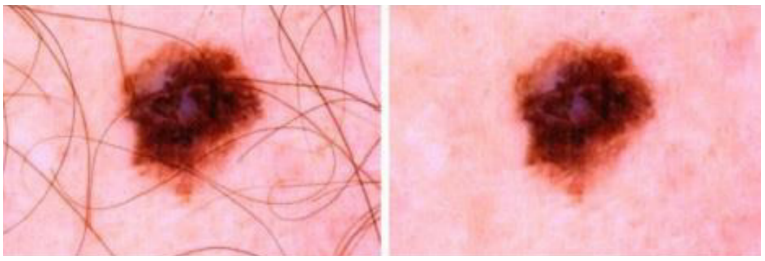


Figure 13: Hair removal from lesion image [56].

2.5.2 Image Segmentation

Segmentation of an image is the process of separating parts of it into a set of regions in order to represent meaningful areas in it. A region may be defined as a group of pixels with border and shape [54]. Some well-known general algorithms for performing segmentation include region growers, clustering algorithms, and edge detectors [54].

A region grower starts at a position in the image, and grows by comparing pixels until they are too dissimilar using some statistical test, one technique proposed by haralick assumes that a region is a set of connected pixels with the same population mean and variance [54].

Clustering is the process of partitioning a set of values into subsets known as Clusters. The process of clustering can be done using algorithms like the iterative K-means method, where K is the number of clusters, or in our case, regions in an image [54].

An intermediate representation for the image region called a Mask can be used for further processing of the segmented image. In this representation, each pixel in the image can be assigned a unique value (or label) representing its region [54]. In Figure 14, we can see an instance of a skin lesion image with its mask. The image is segmented into two parts: the lesion and the background. The image was taken from the International Skin Image Collaboration Archive (ISIC) [6].

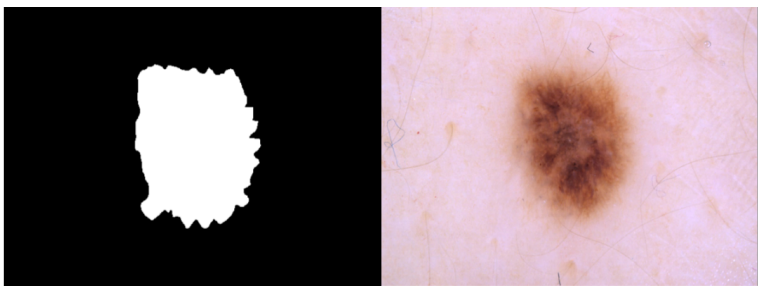


Figure 14: Skin lesion image with its mask [6].

An example of an effective algorithm to do segmentation, especially for medical purposes, is the U-Net algorithm. It is a neural network architecture designed primarily for image segmentation. A U-Net architecture consists of two paths. First, a contracting path, also known as an encoder or analysis path, provides classification information like a regular convolution network. Second is the expansion path, also called the decoder or the synthesizer. It consists of up-convolutions and concatenations of contracting path features. The creation of highly detailed segmentation maps is what makes U-Net particularly useful in medical imaging. It is also much faster to train U-Net due to its context-based learning capability [57].

2.5.3 Color Spaces

The use of color in image processing is driven by two main reasons. First, color is a strong differentiator that makes identifying and removing objects from a scene easier. Second, compared to merely a few dozen shades of gray, humans can distinguish thousands of distinct color shades [53]. There are many different color spaces, each of which has a specific color coordinate system. The original theory of Thomas Young (1802) according to which: every color may be represented by combining a suitable selection of three main colors is what gave rise to the development of digital image color representations [55].

Red, green, and blue are the three so-called primary colors that the human eye perceives as varying combinations [53]. These three colors represent the primary spectral components of the RGB model. This model is based on a subspace of a 3-dimensional Cartesian coordinate system. In this subspace, the primary and secondary RGB colors represent three corners of the cube, black represents the origin, and white is the corner farthest from the origin. The grayscale in this system is the subspace centered at the origin (0,0,0), and extended to the white corner (1,1,1) [53] as seen in Figure 15. An RGB system represented by 8-bits has 256^3 or 16,777,216 color [58].

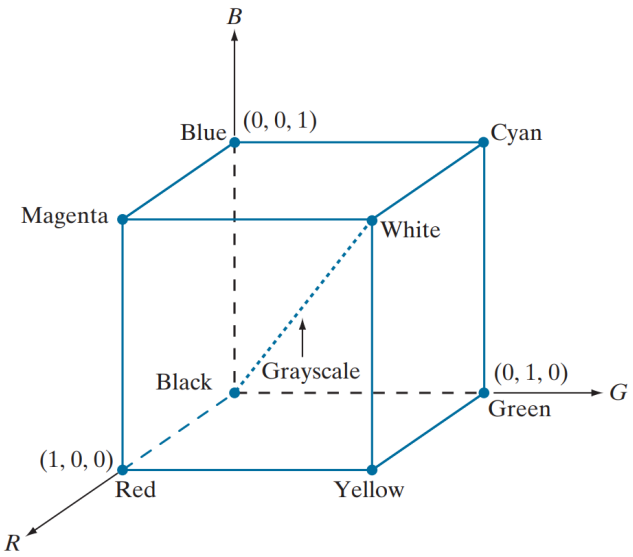


Figure 15: Schematic of the RGB color cube [53].

Red, green, and blue values for color description can be difficult for humans to comprehend because we are typically used to expressing colors by values like hue, saturation, and brightness. While saturation indicates how much a pure color has been diluted by white light, hue is a color property that defines a pure color. Brightness is a purely subjective term that is very hard to quantify. It is one of the fundamental elements in expressing color sense. It conveys the achromatic idea of intensity [53]. Intensity (gray level) is a greatly helpful term to describe achromatic pictures, which is the reason of being HSI a model that focuses on Hue, Saturation, and Intensity, HSI also can be referred to as HSV using the term ”Value” instead of ”Intensity” [54]. The HSI model benefits from decoupling the color and grayscale data in an image, which makes it appropriate for many of the existing grayscale approaches [53].

2.6 Performance Measurements

There are many ways to evaluate an algorithm’s performance in DL classification based on several different features. Most of the performance measurements focus on the classifier’s capability to predict correctly. Table 1 represents the confusion matrix for a binary classifier. It is a table that summarizes how successful the classification model is at predicting examples belonging to various classes. In addition, it can help to determine mistake patterns [12]. True Positive (TP) refers to the condition while the data is positive and correctly predicted as positive. False Negative (FN) refers to the situation where the data is positive but incorrectly detected as negative. False Positive (FP) represents the condition where the data is negative but incorrectly

detected as positive. Lastly, True Negative (TN) indicates the condition where the data is negative and correctly predicted as negative [59].

	Positive	Negative
Positive	TP	FN
Negative	FP	TN

Table 1: Confusion Matrix.

Accuracy is the ratio of the sample's correct predictions to the total number of predictions. [60]. Moreover, it is a helpful metric when errors in predicting all classes are equally important [12]. It is given by:

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

Precision and **Recall** are the most frequently used metrics for assessing a model. The precision determines the number of predicted positive instances correctly classified by the algorithm [60]. It is the ratio of correct positive predictions to the overall number of positive predictions as presented in the equation:

$$\textbf{Precision} = \frac{TP}{TP + FP}$$

Recall is the ratio of correct positive predictions to the overall number of positive examples in the test set [12]:

$$\textbf{Recall} = \frac{TP}{TP + FN}$$

As the equation shows, F1 Score represents the harmonic mean of precision and recall. A high value of F-measure indicates that both precision and recall are reasonably high [60].

$$\textbf{F1 Score} = \frac{2(Recall * Precision)}{Recall + Precision}$$

3 Literature Review

A growing number of studies are being conducted using ML and DL to detect skin cancer in recent years. This chapter aims to provide a summary of some previous works that were studied, to show their models, datasets, techniques, and performance.

A 2019 paper by Al-masni et al. [61] proposed a model for lesion segmentation using a Full-resolution Convolutional Network (FrCN). The segmented images were classified using a Deep Residual Network, namely ResNet-50. The authors used the International Skin Image Collaboration (ISIC) 2017 dataset for training and testing, and used rotation transformation on training data to augment more images. In addition, they used transfer learning and fine-tuning to address the shortcomings of their models. The segmenter model achieved 94.03% accuracy, while the classifier achieved 81.57% accuracy, and 0.7575 on the F1 Score. The classifier was compared to two other classifiers that did not use segmentation to demonstrate the superior performance of the model.

Daghrir et al. [62] (2020) used a CNN and two ML classifiers, KNN and SVM to be trained on a set of features describing the borders, texture, and color of a skin lesion. These features represented the ABCD from the ABCDE sign, The paper employed a color enhancement technique known as Gaussian (DOG) for pre-processing and noise reduction in order to remove hair from lesion images, as well as Morphological Active Contours without Edges (MorphACWE) for lesion segmentation. The paper chose 640 lesion images from the ISIC Archive for models' training. The models' accuracy results were 57.3% for the KNN model, 71.8% for the SVM model, 85.5% for the CNN model, and 88.4% on majority voting.

Sarkar et al. [63] (2019) proposed a Deep Depthwise Separable Residual Convolutional Network model. The model has been trained and validated using a subset of the ISIC dataset, and has been tested on 3 other datasets (PH2, DermIS, and MED-NODE). The authors used multiple pre-processing methods such as image denoising using Non-local Means denoising technique, image enhancement using CLAHE-DWT algorithm, and selecting multiple color channels. The channels selected were RGB color space channels, saturation channel of the HSV color space, b* channel of the CIELAB color space, and finally the inverted grey scale color space channel. The paper demonstrated the model's performance using various kernel sizes, with the best performance being 99.49% accuracy and a 0.9948 F1 Score on the ISIC dataset using a 4x4 kernel and 4 residual blocks. The authors performed a comparative analysis in the paper.

Garg et al. proposed in [64] a system using CNNs to identify skin cancer and

categorize it. A MNIST HAM-10000 dataset was used for skin cancer images. For the classification task, the authors trained a CNN network to classify the given images in the dataset in their respective classes. Moreover, transfer learning was used as well by applying models such as ResNet and VGG16 to increase the classification accuracy of the images, and to compare the accuracy of the model with that of the proposed DL model. Furthermore, to accomplish the optimization purpose, the authors used Adam optimizer. They found that the ResNet model which is pre-trained in ImageNet dataset could be beneficent, and found that it is the successful classification of cancer lesions in the HAM1000 dataset. Also, they mentioned that learning algorithms such as Random Forest, XGBoost, and SVMs were not very effective for classification tasks, and do not give promising results, where the accuracy of Random Forest was 65.9%, XGBoost was 65.15%, and SVM was 65.86%. On the other hand, their CNN model gave a weighted average precision of 0.88, a weighted recall average of 0.74, and a weighted F1 Score of 0.77. The transfer learning approach applied using VGG model gave an accuracy of 78%, whereas the ResNet model gave an accuracy of 90.5%,

In [65], Nataha et al. developed a skin cancer detection CNN model that can classify skin cancer types and detect them early. The model was trained and tested on the state-of-art CNNs, namely Inception V3, ResNet50, VGG16, MobileNet, and InceptionResnet to perform a seven-class classification of the skin lesion images. For the dataset, there were two datasets used to develop their research, which are: ISIC 2018, and ISIC 2019. They combined the two datasets and used them as one dataset. The researchers mentioned that using one metric in a classification problem such as accuracy cannot help in evaluating the complete model efficiency effectively. Hence, they measured the accuracy, precision, recall, F1 Score, and support for every class of skin lesion disease. The results of the average metrics achieved by all final CNNs were as follows. In ResNet, researchers got an accuracy of 85%, a precision of 0.86, a recall of 0.85, and an F1 Score of 0.85. In MobileNet, researchers got an accuracy of 85%, a precision of 0.86, a recall of 0.85, an F1 Score of 0.85, and support of 6944. In VGG16, researchers got an accuracy of 87%, a precision of 0.87, a recall of 0.87, and an F1 Score of 0.87. In Inception V3, researchers got an accuracy of 90%, a precision of 0.90, a recall of 0.90, and an F1 Score of 0.90. Lastly, in InceptionResnet, researchers got an accuracy of 91%, a precision of 0.91, a recall of 0.91, and an F1 Score of 0.91.

In [66], Nur Fu'adah et al. built a system that used a CNN with three hidden layers, 3×3 filter sizes, and output channels of 16, 32, and 64, respectively, to automatically diagnose skin cancer and benign tumor lesions. The used dataset in this study is an augmentation of the ISIC dataset composed of four classes: Dermatofibroma, Nevus Pigments, Squamous-Cell Carcinoma (SCC), and Melanoma. There were 4,000 images. the authors used 3000 training images and 1000 validation

images. These images were trained using the CNN model with several optimization techniques, such as SGD, RMSprop, Adam, and Nadam optimizer with a learning rate of 0.001 and using loss categorical cross-entropy. Their results showed that the Adam optimizer provides the highest performance, with an accuracy value of 99%, a loss of 0.0346, and precision, recall, and F1 Score values of 0.91. Moreover, they noted that the proposed model has promising usage as an existing tool for medical personnel.

Albahar et al. proposed a new classification model in [67] that used 2-layer CNNs with a novel regularizer technique to classify skin lesions as benign or malignant. For training and validation, they used the ISIC dataset, which contains 23906 images. They divided it into 3 parts, and used 5600 images for the training set and 2400 for the validation set in each part. They achieved an average accuracy of 97.49%. A regularization technique can be used in many different ways to control the complexity of classifiers. [67] proposed a novel regularizer based on the standard deviation of the classifier's weight matrix. They were reducing the complexity by penalizing the dispersion of the weight matrix values. Finding an acceptable value for it was a challenging task since it was a continuous variable that was expensive and time-consuming to reach. Additionally, they demonstrated that their model performed better than existing algorithms and could be applied to assist medical practitioners in classifying skin lesions.

According to [68], using five steps, they applied a CNN approach to approximately 23907 images collected from the ISIC archive. The first step was pre-processing the data. The authors saved the pre-processed file as a second step which saves each of the pre-processed images in a record along with their classes. In the third step, they feed the pre-processed data to the CNN. Step four was training their model up to 200 times. Eventually, step five was saving the model for other testing purposes. In the last step, the model is used to predict the images that might contain benign or malignant lesions images. In this research, Hasan et al. [68] used three layers. The first layer is the input layer on which the data sets are trained. Input layers collect data delivered and add weight to them that goes to hidden layers. Neurons in the hidden layer separate features from the data to find patterns. The pattern is then used to select appropriate classes based on output layers. As a final step, binary classification was used to determine classes 1 and 0. There are no harmful cells in class 0, and there are cancerous cells in class 1. After they applied all their steps, the result with the standard evaluation metrics were 0.84 for recall, 0.8325 for precision, 0.8325 for F1 Score, and 89.5% for accuracy.

Mustafa et al. [69] proposed research for skin cancer detection using color space. It is used by experimenting with luminance to improve the visualization for GrabCut

segmentation accuracy, by extracting corner and geometric features that are used to train the SVM model. Many research works have been done using image processing and computer vision to detect skin cancer, and in most cases, the ABCDE rule has been applied. In their research, they took advantage of asymmetry and boundary rules (AB) to extract features to identify melanoma. Their first step in pre-processing is to enhance the image by modifying the contrast and brightness (reducing the brightness of the light). The HSV color space is better for image processing. Because it splits color data (Chroma) from intensity or lighting (Luma), the next step is color space transformation (HSV). Afterward, they used the GrabCut technique for Segmentation derived from graph cuts. Then they extracted external pixels, known as recognized backgrounds, while the internal pixels were identified as unknown. They created multivariate Gaussian Mixture Models (GMM) from extracted information to determine if the unknown pixels represent either background or foreground. After that, the segmentation results are used to identify the largest contour to derive its features: Geometric Circularity and Harris Corner Detection. They used an SVM model in the classification to train and predict if an image is positive or negative based on the extracted features. Their result shows that the separation using SVM can find the optimum plane to separate the two classes from the trained samples and predict the testing samples. Their results were 0.80 for Accuracy, 0.8621 for precision, and 0.7143 for recall.

In [70], Rozaoana et al. presented a CNN system with three main components: feature extraction, detection, and classification. Over 25,000 images were used in the analysis. By enhancing the output to solve the overfitting problem, a data augmentation technique was used to train CNN just-in-time to avoid distortion and maintain the original consistency of input and output data. Several choices for image augmentation were provided to choose values from different sizes. In this way, the model was able to display images in a variety of ways, which maximized its utility. The convolution, activation, and max-pooling layers were used concurrently in this method (CNN). After that, parallel layers were connected at the function level. Thereafter, the flattened features had two layers of Multi-Layer Perceptrons (MLP), and to prevent overfitting, the number of neurons in each layer was altered. Lastly, the final layer was the layer responsible for classification in addition to the softmax layer. Class activation maps are generated for those layers. The function of the created class is to translate the classification paired to the final convolution layer. Based on a comparison of their results of CNN and other models such as VGG-16, and VGG-19, they determined that CNN is the most reliable model with 79.45% accuracy, 76.17% precision, 78.15% recall, and 76.92% F1 Score, which has the highest score among the models.

Lack of data is one of the primary factors holding back the development of DL techniques for cancer detection. Sedigh et al. [71] tackled this problem using augmented images to train CNN models. The paper used less than a hundred images from the ISIC Archive to train a GAN for this task. According to [71], the performance of the CNN algorithm for cancer detection increased by nearly 17% in terms of accuracy, and the F1 Score showed a growth of 0.2 when using the augmented images for CNN training.

Ref	Year	Dataset	Model(s)	Accuracy	Precision	Recall	F1 Score
[61]	2019	ISIC 2017	ResNet50	81.57%	ε	ε	0.7575
[62]	2020	ISIC 2019	Ensembling (KNN, SVM, CNN)	88.4%	ε	ε	ε
[63]	2019	ISIC Archive	Deep Depthwise Separable Residual Convolutional Network	99.49%	0.9965	0.9931	0.9948
[64]	2019	MNIST HAM10000	ResNet	90.5%	ε	ε	ε
			VGG16	78%	ε	ε	ε
			Self-made CNN	ε	0.88	0.74	0.77
			Random Forest	65.9%	ε	ε	ε
			XGBoost	65.15%	ε	ε	ε
			SVM	65.86%	ε	ε	ε
[65]	2020	HAM10000 ISIC 2019	ResNet50	85%	0.86	0.85	0.85
			MobileNet	85%	0.86	0.85	0.85
			VGG16	87%	0.87	0.87	0.87
			Inception V3	90%	0.90	0.90	0.90
			Inception Resnet	91%	0.91	0.91	0.91
[66]	2020	ISIC Archive	CNN	99%	0.91	0.91	0.91
[67]	2022	ISIC Archive	CNN	97.49%	ε	ε	ε
[68]	2019	HAM10000	CNN	89.5%	0.8325	0.84	0.8325
[69]	2017	DermQuest	SVM	80%	0.86	0.71	ε
[70]	2020	ISIC 2019	CNN	79.45%	0.76	0.78	0.77
[71]	2019	ISIC Archive & Augmented Images	CNN	71%	ε	ε	0.7

Table 2: Summary of the literature review. Unmentioned performance metrics were denoted with ε.

4 Methodology

After evaluating the related work, the most promising models will be implemented, and a study will be conducted to address the challenge of detecting skin cancer from lesion images. Our analysis has established that CNN and SVM are the most appropriate models for this challenge. To evaluate their performance, we will first preprocess the lesion images, and then we will segment them. After that, we will augment the malignant lesions images using GAN and evaluate the performance of the CNN and the SVM models before and after adding augmented data.

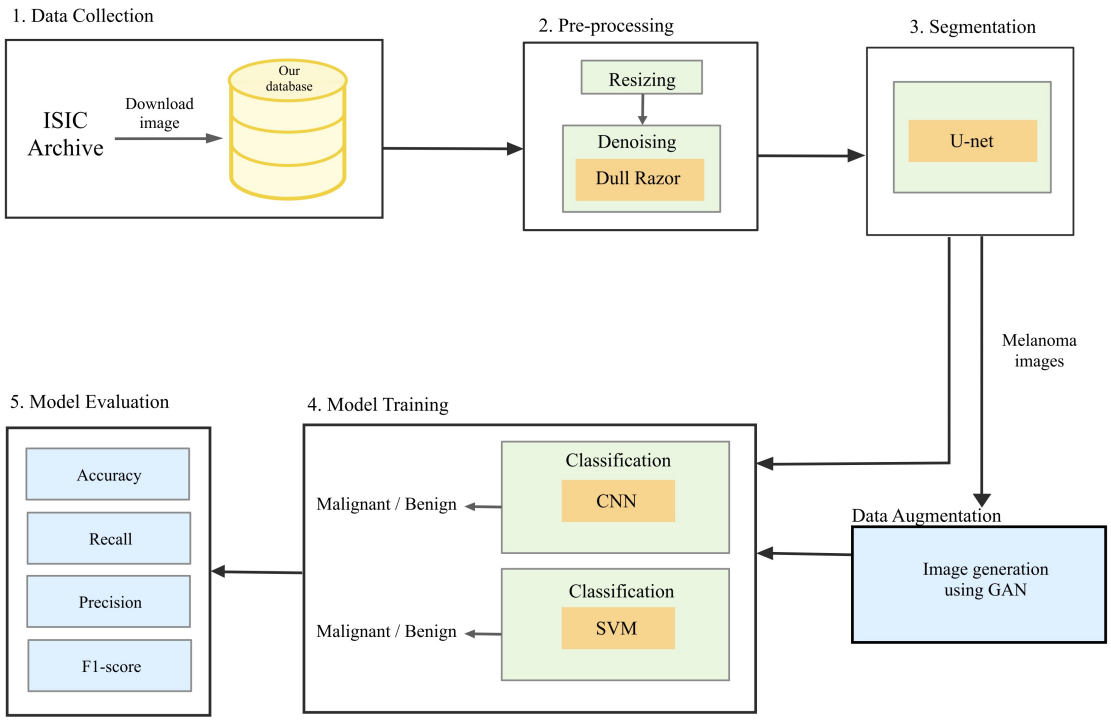


Figure 16: Graphical representation of the metholoogy.

4.1 Preprocessing

In this section, we describe the preprocessing steps we employed in our study to denoise the skin lesion dataset.

4.1.1 Desnoising

The problem of removing hair from images has been addressed by the development of various algorithms, one of which is the algorithm presented in the work by lee et al. [72]. In this research, we introduce the DullRazor algorithm, which is a

promising solution to the issue of hair detection and removal.

4.1.2 Segmentation

Many known techniques tackle the problem of image segmentation, and U-Net will be our choice for this task, since we saw in 2.5.2 that it handles complex structures and shapes, and it is used for medical image segmentation and other tasks where high accuracy is required. Figure 17 below shows a graphical representation of U-Net architecture.

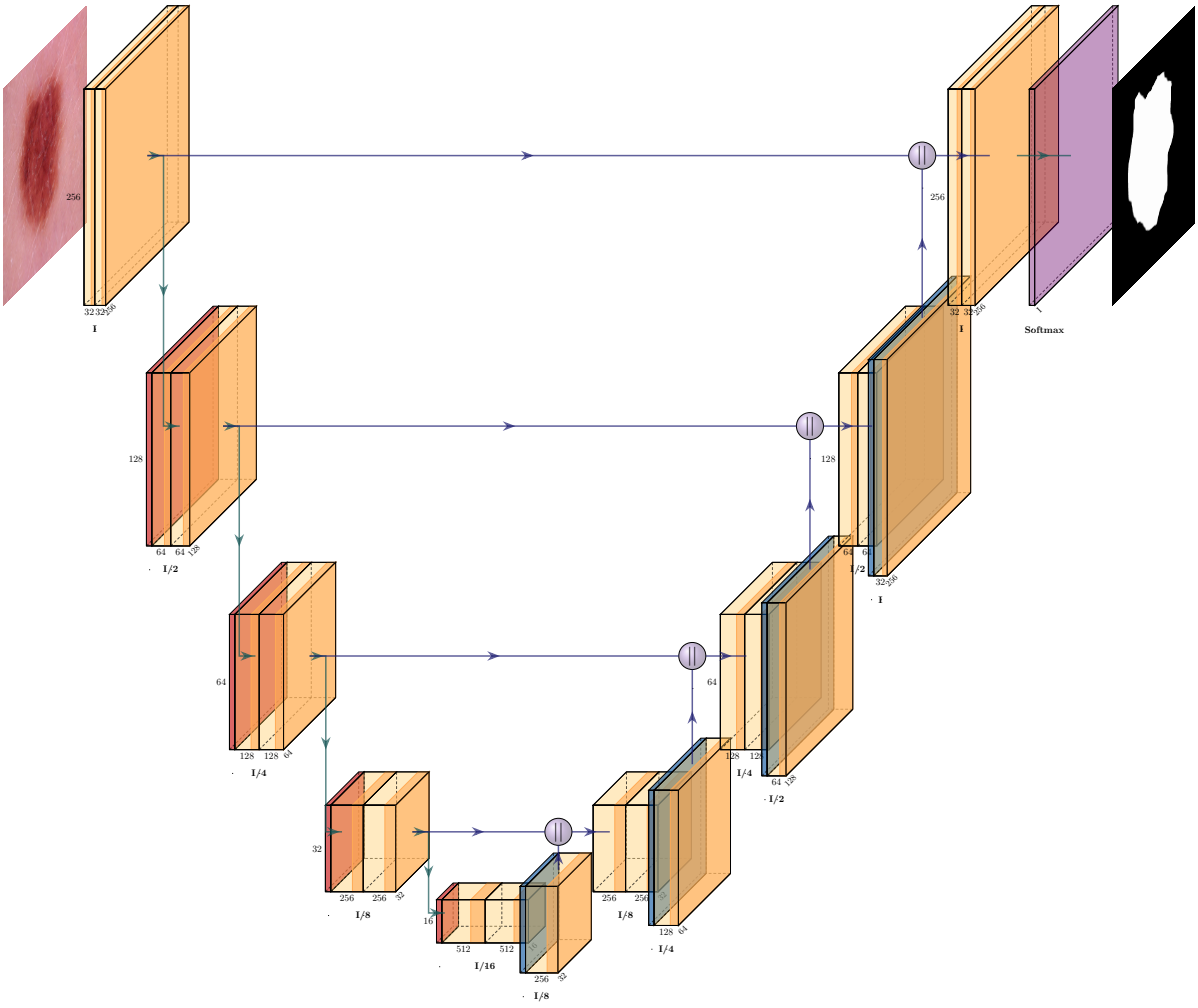


Figure 17: U-Net architecture.

4.1.3 Agumentation

For data augmentation, we will use GAN, a class of DL models used for image generation and augmentation. They consist of two neural networks, a generator, and a discriminator. The generator creates new images, while the discriminator evaluates

the authenticity of the generated images and compares them to the real images in the training dataset, as seen in 2.4.3. Figure 18 shows a graphical representation of the GAN architecture.

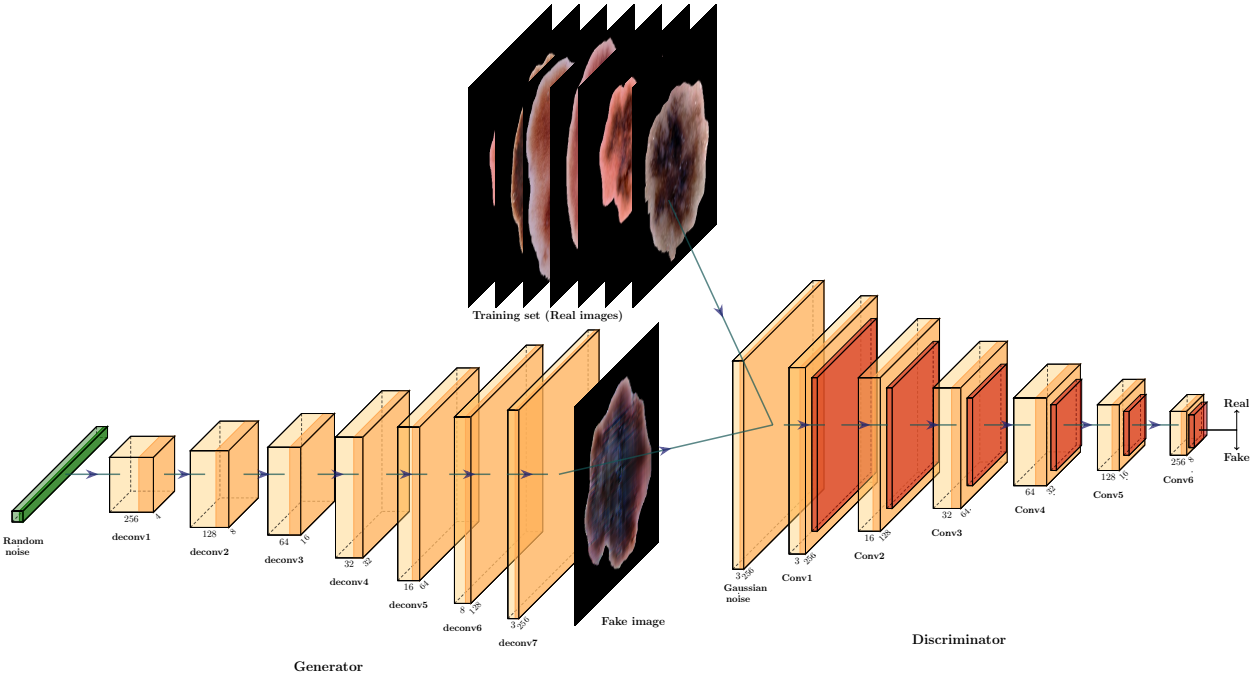


Figure 18: GAN architecture.

4.2 CNN Classifiers

For the classification task, we will use the following four CNN pre-trained models, which are ResNet50V2, VGG16, EfficientNet-B5, and EfficientNet-B7. These models provide diverse architectures, computational requirements, and performance characteristics, making them suitable for our task. The figures below show the architecture for each pre-trained model, and the tables containing additional final layers for each model.

4.2.1 ResNet50V02

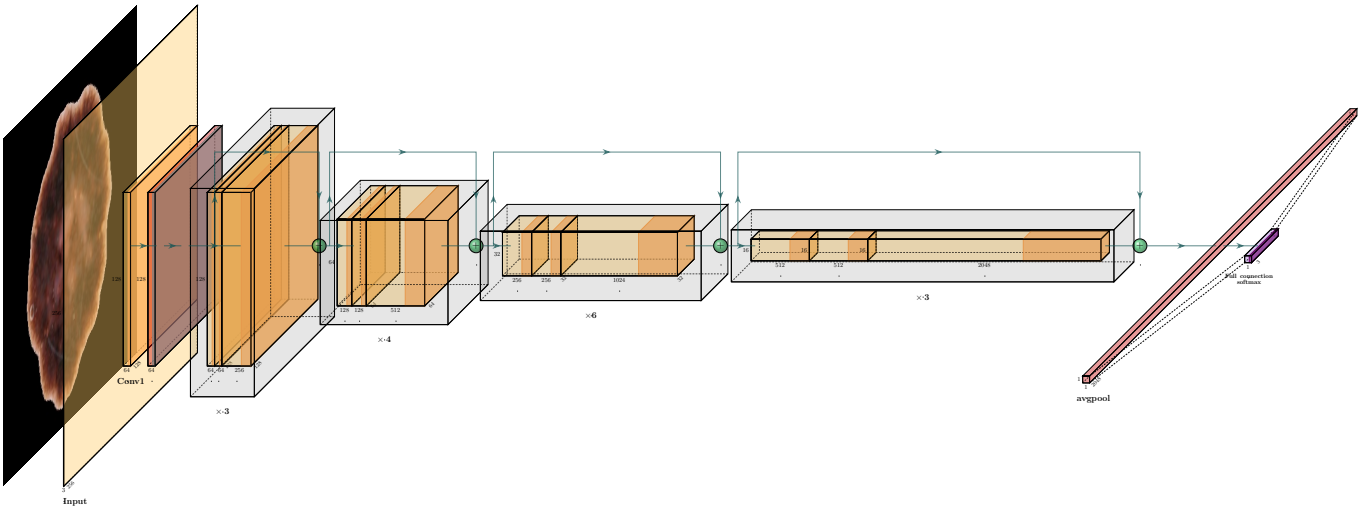


Figure 19: ResNet50V2 architecture.

Layer (type)	Output Shape	Parameter
Input image	(None, 256, 256, 3)	0
Resnet50V2 (Functional)	(None, 2048)	23564800
Top flatten (Flatten)	(None, 2048)	0
Dense 8 (Dense)	(None, 4096)	8392704
Dropout 8 (Dropout)	(None, 4096)	0
Dense 9 (Dense)	(None, 32)	131104
Dropout 9 (Dropout)	(None, 32)	0
Output layer (Dense)	(None, 1)	33

Table 3: Details of ResNet50V2 model proposed.

4.2.2 VGG16

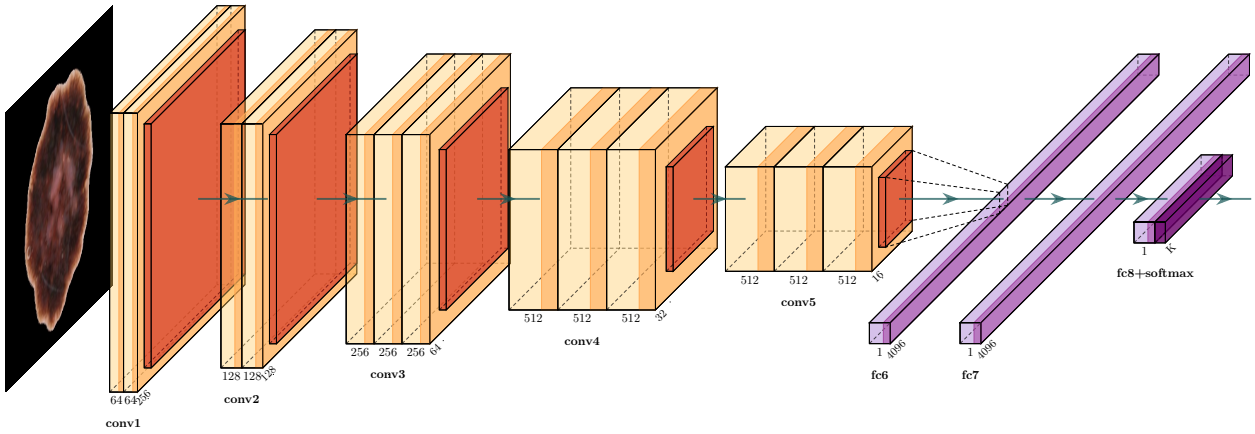


Figure 20: VGG16 architecture.

Layer (type)	Output Shape	Parameter
Input image	(None, 256, 256, 3)	0
VGG16 (Functional)	(None, 512)	14714688
Top flatten (Flatten)	(None, 512)	0
Dense 6 (Dense)	(None, 4096)	2101248
Dropout 6 (Dropout)	(None, 4096)	0
Dense 7 (Dense)	(None, 64)	262208
Dropout 7 (Dropout)	(None, 64)	0
Output layer (Dense)	(None, 1)	65

Table 4: Details of VGG16 model proposed.

4.2.3 EfficientNet

The EfficientNet is widely acknowledged as one of the largest and most intricate models in DL, rendering its representation is often challenging. Nonetheless, a simple way to represent it is by dividing it into units and showcasing each unit in a sub-block. Subsequently, these sub-blocks are employed to represent different versions of the EfficientNet [73].

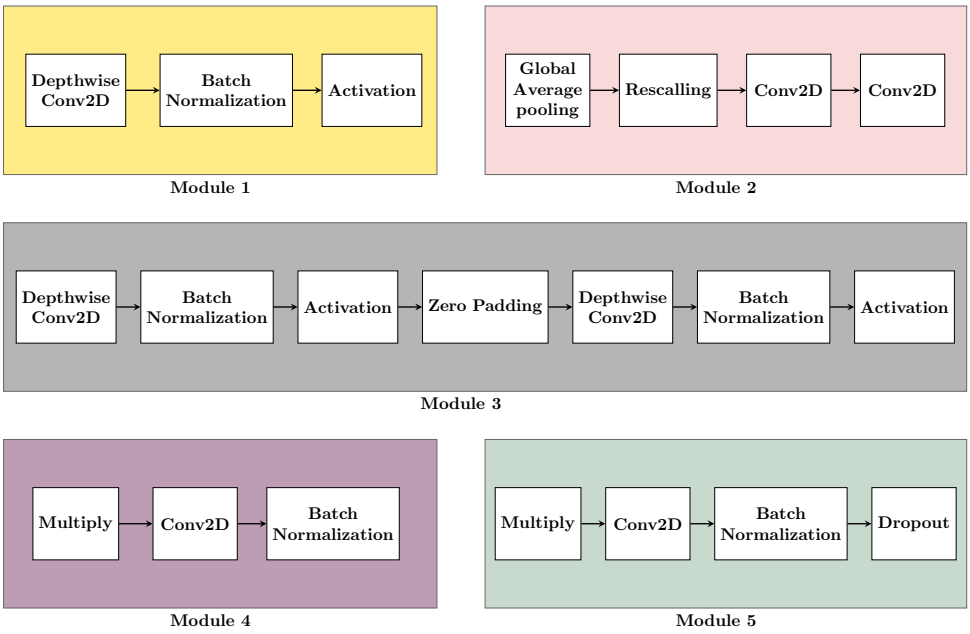


Figure 21: EfficientNet Modules architecture.

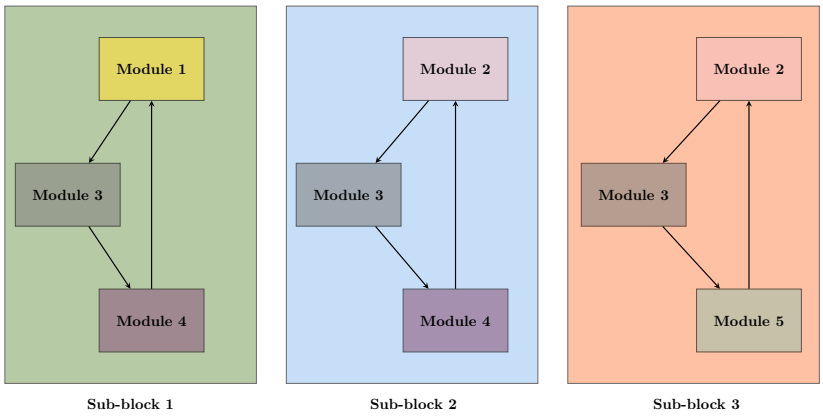


Figure 22: EfficientNet sub-blocks architecture.

4.2.3.1 EfficientNetB5

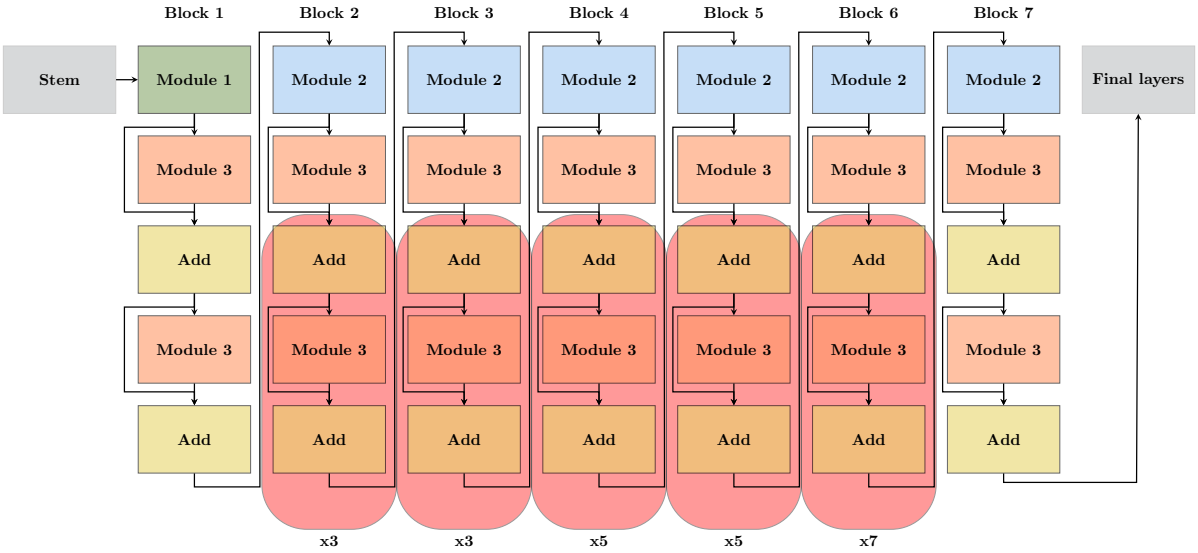


Figure 23: EfficientNet-B5 architecture.

Layer (type)	Output Shape	Parameter
Input image	(None, 256, 256, 3)	0
EfficientNet-B5 (Functional)	(None, 2048)	28513527
Top flatten (Flatten)	(None, 2048)	0
Dense 12 (Dense)	(None, 256)	524544
Dropout 12 (Dropout)	(None, 256)	0
Dense 13 (Dense)	(None, 32)	8224
Dropout 13 (Dropout)	(None, 32)	0
Output layer (Dense)	(None, 1)	33

Table 5: Details of EfficientNet-B5 model proposed.

4.2.3.2 EfficientNetB7

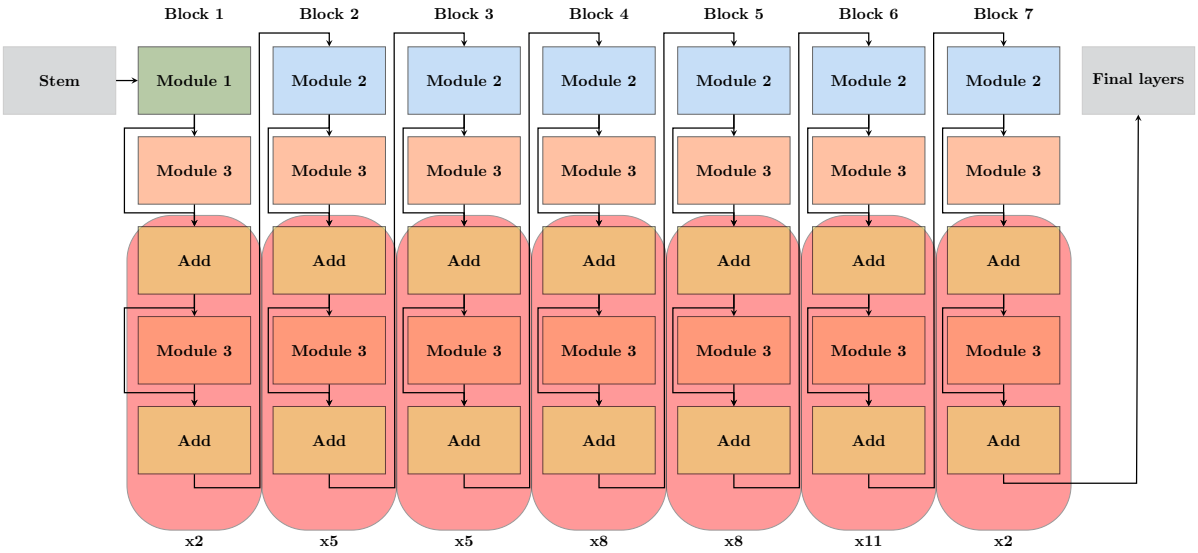


Figure 24: EfficientNet-B7 architecture.

Layer (type)	Output Shape	Parameter
Input image	(None, 256, 256, 3)	0
EfficientNet-B7 (Functional)	(None, 2560)	64097687
Top flatten (Flatten)	(None, 2560)	0
Dense 14 (Dense)	(None, 512)	1311232
Dropout 14 (Dropout)	(None, 512)	0
Dense 15 (Dense)	(None, 256)	131328
Dropout 15 (Dropout)	(None, 256)	0
Output layer (Dense)	(None, 1)	257

Table 6: Details of EfficientNet-B7 model proposed.

4.3 SVM Classifier with Feature Extraction

The technique of feature extraction will be employed for the SVM. This method proves to be efficient in decreasing the dimensionality of large datasets, particularly when utilized in conjunction with SVM algorithms, which require a significant amount of computational resources [74]. By reducing the number of features in the dataset, we can make the training process more efficient and manageable. The following features will be extracted from lesion images for skin cancer classification:

- **Area and Perimeter of the lesion**: These features provide valuable insights into the characteristics of a skin lesion and can be used to distinguish between

various types of skin lesions. The area represents the two-dimensional extent of the lesion, while the perimeter indicates the length of the boundary of the lesion. Together, these measurements provide a comprehensive picture of the size and shape of a skin lesion.

- **RGB and HSV**: These features will be used to differentiate between different types of skin lesions based on their color characteristics. RGB values can be used to determine the exact color of a lesion. HSV, on the other hand, can be useful in analyzing the different aspects of the lesion's color. By using both RGB and HSV features, a more comprehensive understanding of the lesion's color can be obtained, which can be useful in differentiating between different types of skin lesions.
- **Mean and Standard Deviation of RGB and HSV**: These features provide information about the distribution of color intensity values in the image and can be used to distinguish between subtle differences in color. By measuring the mean and standard deviation of the RGB and HSV channels, we can quantify the distribution of color information in the image and use that information to differentiate between different types of skin lesions. This information can be especially useful in cases where the differences in color between lesions are subtle, but still important to consider in making a diagnosis.

5 Experimental Design

5.1 Dataset

In order to ensure good generalization in our models, validate their efficiency, and avoid overtraining them, appropriate data splitting was used. To split the dataset into different sets, there are several techniques, and we chose the Hold-out technique in this research. Hold-out is a popular technique because of its easiness and efficiency. It divides the dataset T of size n into three subsets, training T_{tr}, validation T_v, and testing T_t of sizes n_{tr}, n_v and n_t consecutively. This technique has the advantage of allowing the proportion of each of these three data subsets to be varied. The model is trained on the training subset T_{tr}, while the validation subset T_v is used periodically to assess the model's performance during training to prevent overtraining. The training is terminated when the performance on T_v is satisfactory or when it ceases to improve. The testing subset T_t is then utilized to obtain a reliable estimate of the models' performance [75].

In this research, the dataset used was collected from the ISIC Archive [6]. We split the dataset as follows: 80% for the training set, 10% for the validation set, and 10% for the test set. Dataset's Images Ground-Truth was based on each lesion's medical diagnosis. A total of 20,092 images were used. 11063 images of benign types, and 8732 images of malignant types.

Following the pre-processing of the dataset, starting with the segmentation task, 10,015 of the 20,092 images were provided with a mask and were used to train a U-Net model to segment the rest of the images. After segmenting all the images in the dataset, in image generation task, 647 images from the malignant types were used to train a GAN model. The model generated 1299 new images to make the total number of malignant class images 10031.

5.2 Experiments

In this step, we will experiment with the models mentioned in the methodology: The CNN model, CNN with data augmentation using GAN model, and SVM model. Four different tests will be conducted.

I. CNN Classifiers Test

We will conduct the first test by taking a subset of images and preprocess them by resizing them, and then denoise them using the DullRazor technique. A U-Net network will be used to output a mask for each image, and the mask will be applied to the images to remove the background, and finally, the output dataset will be used

to train the pre-trained CNN classifiers.

II. CNN Classifiers with Data Augmentation using GAN

For the second test, we shall use the same aforementioned process provided that we will add an extra step after the pre-processing. We will take the pre-processed malignant lesions images to generate more data using GAN, and then we will train, test and evaluate our pre-trained CNN models again.

III. SVM Classifier

In this test, we will implement an SVM model using our pre-processed and segmented images to compare the results with our CNN models before the data augmentation step.

IV. SVM Classifier with Data Augmentation using GAN

Finally, the same previously mentioned SVM model will be used, and the pre-processed malignant lesions images will be trained to generate more data using GAN, and the results in this test will be compared with the results in the second test.

6 Implementation Details

6.1 Implementation Environment

In order to implement our experiments, we have chosen Python as it is the most suitable programming language for ML and DL experiments. It provides a variety of libraries to support the implementation process. As implementing these experiments consumes a lot of hardware resources, Google Colaboratory was our choice to run them. The reason for this is that it provides cloud-based resources like GPUs. We used several libraries, including TensorFlow's Keras, OpenCV, Scikit-Learn, and SciPy.

Keras is a Python-based DL API that runs on top of TensorFlow and provides a high-level interface for developing DL neural networks [76].

OpenCV is an open-source Python library used for image processing and computer vision. Many algorithms are available that can be used to identify objects, detect and recognize faces, and more [77].

Scikit-Learn is an open-source Python library for ML and statistical modeling. Among its features are algorithms for classification, regression, clustering, and dimension reduction, as well as unsupervised and supervised learning techniques [78].

SciPy is a library that takes on standard problems, and mathematical computations, using arrays and manipulations [79].

6.2 Preprocessing

In order to provide a clear understanding of our models workings, we will provide a pseudocode of algorithms used through out this experiment. Our experiment with the DullRazor algorithm have produced intriguing results, which demonstrate its effectiveness in detecting and removing hair from images.

Algorithm 1: DullRazor

Input : Image Array

Output: Image Array After Applying DullRazor Algorithm

$grayimage = \textbf{RGB2GRAY}(image)$

$blackhat = \textbf{MorphologyEx}(grayimage, MORPH\_BLACKHAT, filter_{9\times9})$

$blurredimage = \textbf{GaussianBlur}(blackhat, filter_{3\times3})$

$mask = \textbf{BinaryThreshold}(blurredimage, threshold = 15, maxval = 255)$

$output = \textbf{Inpaint}(image, mask, inpaintRadius = 6, INPAINT\_TELEA)$

The algorithm was implemented using OpenCV library. Figure 25 shows a sample of images processed by the DullRazor algorithm.

Figure 25: Sample output of the DullRazor.

6.3 Segmentation

The below algorithm shows a pseudocode for U-Net algorithm witch used in segmentation task.

Algorithm 2: U-Net Model Training Function

Input : DIR "Dataset directory"
Output: masks for images with no available mask
(trainingData, trainingMasks), (validationData, validationMasks) ←
 loadDataset(DIR=DIR)
model ← getUNet()
callbacks ← CSVLogger and ModelCheckpoint callbacks
model.fit(trainingData, trainingMasks, batchSize=32, epochs=30,
 validationData=(validationData, validationMasks), callbacks=callbacks)
images ← loadDatasetWithNoMask()
masks ← empty set
for *image* ∈ *images* **do**
    prediction ← *model*.predict(image)
    mask ← applyOtsuThreshold(prediction)
    append mask to masks
end

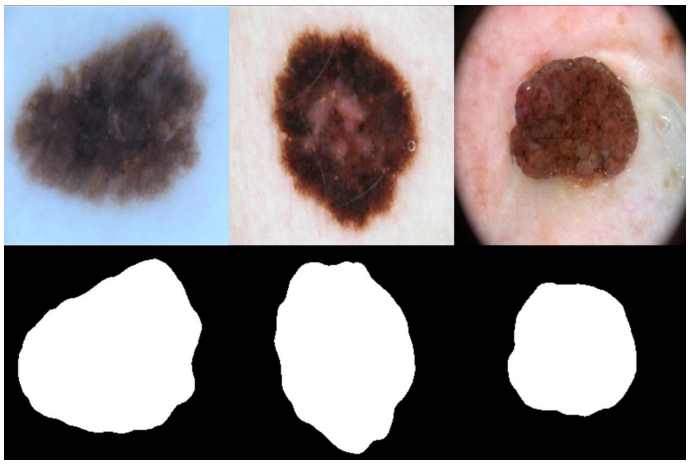


Figure 26: Sample output of U-Net image segmentation.

6.4 Augmentation

The pseudocode below represents a GAN algorithm witch used in data augmentation task.

Algorithm 3: Minibatch Stochastic Gradient Descent training of Generative Adversarial Nets [42]

for *number of training iterations* **do**
  for *k steps* **do**
    Sample minibatch of m noise samples $z^{(1)}, ..., z^{(m)}$ from noise prior $p_g(Z)$.
    Sample minibatch of m examples $x^{(1)}, ..., x^{(m)}$ from data generating distribution $p_{data}(x)$.
    Update the discriminator by ascending its stochastic gradient:
    $\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} [\log D(x^{(i)}) + \log(1 - D(G(z^{(i)})))]$
  end
  Sample minibatch of m noise samples $z^{(1)}, ..., z^{(m)}$ from noise prior $p_g(Z)$.
  Update the generator by descending its stochastic gradient:
  $\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log(1 - D(G(z^{(i)})))$
end

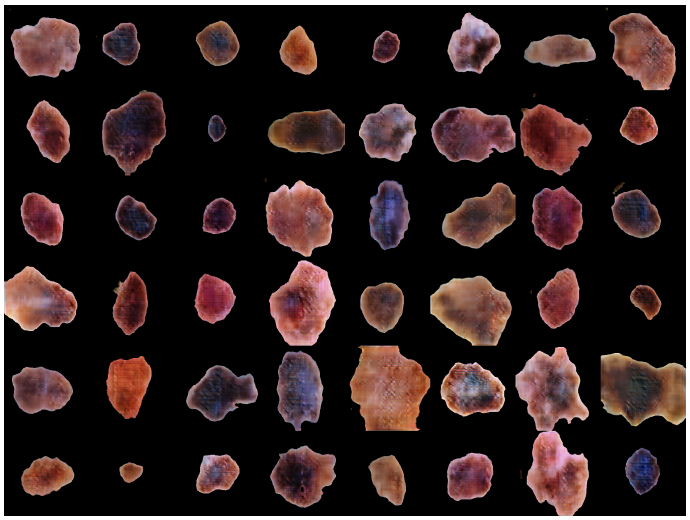


Figure 27: Sample output of the GAN model.

6.5 Classification

6.5.1 SVM

In the image classification task using SVM, the model is available in Scikit-learn, we trained the model as shown in the algorithm below.

Algorithm 4: SVM

Input : Images' extracted features
Output: Trained SVM model
dataset ← Images Features
train, test ← SplitData(dataset)
dataset ← StandardScalar(dataset)
model = SVM()
model.fit(train, trainLabels)
model.evaluate(test, testLabels)
Save the model and its accuracy report

6.5.2 CNN

For an image classification task using CNN, the four pre-trained models are available in tensorflow.keras.applications module, models were trained using a training loop as shown in the pseudocode algorithm 5.

Algorithm 5: Models Training Loop

```
Input  : preTrainedModels a dictionary containing transfer learning models
         as keys and model parameters as value
Output : Models' testing results
results ← empty DataFrame
for preTrainedModel ∈ preTrainedModels dictionary do
    for dataset ∈ datasets do
        parameters ← preTrainedModels[preTrainedModel]
        epochs ← parameters.epochs
        train, validate, test ← getDataset(DIR=dataset,
         BATCHSIZE=parameters.batchSize)
        model ← getModel(preTrainedModel,
         learningRate=parameters.learningRate, pooling=parameters.pooling,
         dropout=parameters.dropout, denseSize=parameters.denseSize)
        callbacks ← modelCallbacks()
        model.train(train, validate, callbacks=callbacks)
        result ← model.evaluate(test, batchSize=parameters.batchSize)
        get evaluation result and store it as a dictionary
        calculate F1 Score using recall and precision
        add Model and dataset names, and F1 Score to the result dictionary
        create a DataFrame from the result dictionary
        append the DataFrame to the results DataFrame
    end
end
```

6.6 Evaluation methods

In order to make results clear and accurate as possible, in the classification task, all the models had been evaluated using the four performance measurements mentioned in the Background, which are: Accuracy, Precision, Recall, and F1 Score.

6.7 Implementation Issues

The implementation of large-scale DL models often involves various hardware and resource management challenges. These challenges can be particularly pronounced when using limited resources such as those available through Google Colaboratory. During the course of our experiment, we encountered several significant issues related to GPU and storage space management. These issues required

careful planning and resource allocation to overcome. Despite these challenges, we were able to successfully implement the DL model and obtain meaningful results. However, it is important to note that proper hardware and resource management is crucial for the successful deployment of DL models, especially when working with limited resources.

7 Results and Discussion

In this chapter, we present our models' test result and provide a detailed analysis of the performance metrics of our trained models.

7.1 Parameters Tuning

Hyperparameter tuning is one of the fundamental ways to enhance the performance of ML models. During the learning process of a model, hyperparameters are passed in order to make corrections or adjustments. Diverse data patterns may necessitate distinct constraints, weights, or learning rates for the same ML model. These kinds of measurements are referred to as hyperparameters [80].

7.1.1 SVM Tuning

We tuned the hyperparameters of our SVM model by using Grid Search. Grid Search, also known as the Estimator, is a tuning technique aiming to calculate hyperparameters' optimal values. It is an exhaustive search that is performed on specific parameter values of a specific model [81].

The ranges of values to tune the hyperparameter in SVM are as follows:

- C: [0.001, 0.01, 0.1, 1, 10, 100, 1000]
- kernel: [linear, poly, rbf, sigmoid]
- gamma: [0.0001, 0.001, 0.01, 0.1, 1, 10, scale, auto].

7.1.2 SVM Tuning Results

After performing the grid search, the optimal hyperparameters for our SVM model were found to be C = 1000, gamma = auto, and kernel = rbf. This indicates that a high value of C, the radial basis function (RBF) kernel, and an automatic setting for gamma were the best choices for our problem.

7.1.3 CNN Tuning

To tune the hyperparameters of our DL models, we used KerasTuner, which is a scalable, easy-to-use hyperparameter optimization framework that alleviates the difficulties associated with hyperparameter search. KerasTuner comes with Bayesian Optimization, Hyperband, and Random Search algorithms, and is designed to be extensible so that researchers can experiment with new search algorithms [82].

The hyperparameters we tuned are:

1. Dropout
2. Dense layers sizes
3. Learning Rate
4. Pooling type

Each hyperparameter has a range of values to make the tuner decide which value is more suitable, the ranges of values are as follows:

- Dropout 1: Range [0, 0.8]
- Dsense layer 1: [4096, 512, 256]
- Dropout 2: Range [0, 0.6]
- Dsense layer 2: [128, 64, 32]
- Learning Rate: Range [1e-4, 1e-7]
- Pooling type: [Average, Max]

7.1.4 CNN Tuning Results

Using the above mentioned processing techniques and model parameter tuning libraries. We present the results of the models on the test set using two datasets, where we added augmentation data in one, demonstrating the effectiveness of the proposed approach.

Model	**Dropouts**	**Dense layers**	**Learning Rate**	**Pooling type**
EfficientNetB5	[.4, .2]	[256, 32]	1e-05	Average pooling
EfficientNetB7	[.8, .6]	[512, 256]	7e-5	Max pooling
ResNet50V2	[.3, .1]	[4096, 32]	9e-06	Average pooling
VGG16	[.3, .0]	[4096, 64]	5e-06	Max pooling

Table 7: Models Configurations.

7.2 Results

As the primary objective of our research was to investigate the effectiveness of DL algorithms in accurately classifying skin lesions as either malignant or benign. In this chapter, we provide a detailed analysis of the performance metrics of our trained models and compare them with existing methods.

Model	Dataset	Accuracy	Recall	Precision	F1 Score
EfficientNet-B7	Segmented	0.856219	0.882151	0.805643	0.842163
EfficientNet-B7	**Segmented and augmented**	**0.869159**	**0.862550**	**0.859127**	**0.860835**
EfficientNet-B5	Segmented	0.843781	0.737986	0.883562	0.804239
EfficientNet-B5	**Segmented and augmented**	**0.855140**	**0.797809**	**0.882159**	**0.837866**
ResNet50V2	Segmented	0.831343	0.779176	0.823458	0.800705
ResNet50V2	**Segmented and augmented**	**0.852804**	**0.833665**	**0.849746**	**0.841629**
VGG16	Segmented	0.840796	0.779176	0.842822	0.809750
VGG16	**Segmented and augmented**	**0.841121**	**0.795817**	**0.855460**	**0.824561**

Table 8: CNN Models testing results.

Figure 28: EfficientNet-B5

Figure 29: EfficientNet-B5 with GAN

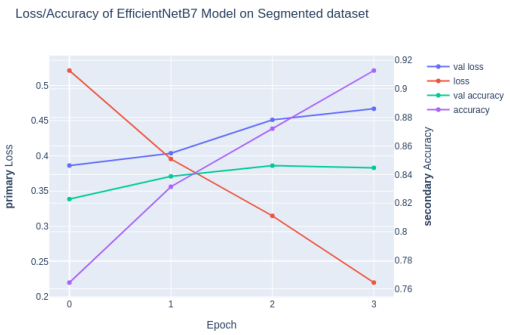


Figure 30: EfficientNet-B7

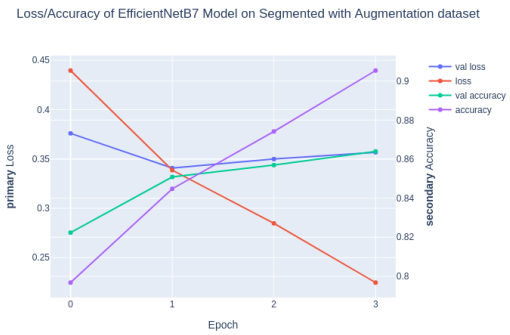


Figure 31: EfficientNet-B7 with GAN

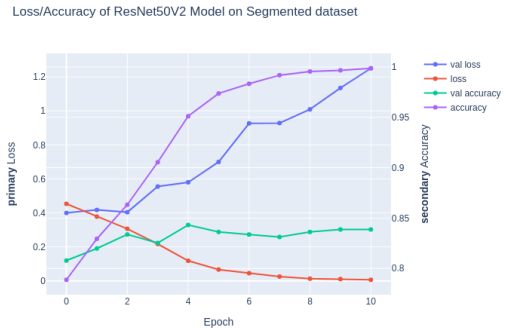


Figure 32: ResNet50V2

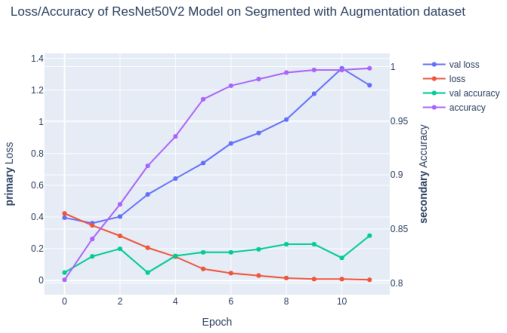


Figure 33: ResNet50V2 with GAN

Figure 34: VGG16

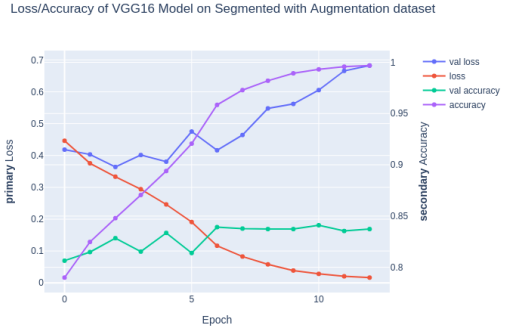


Figure 35: VGG16 with GAN

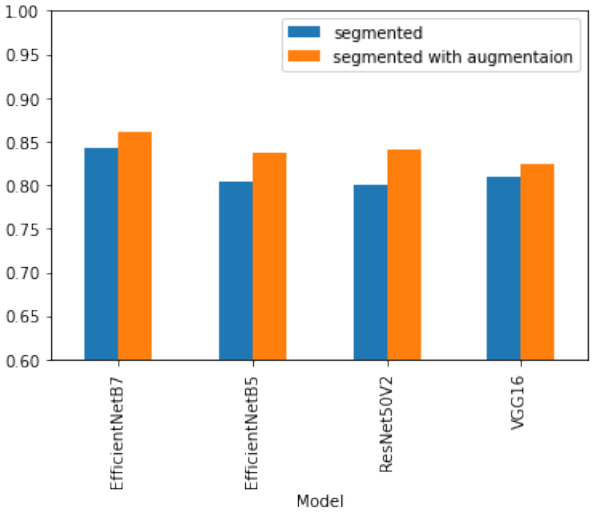


Figure 36: Comparing Models' F1 Scores.

Dataset	Accuracy	Recall	Precision	F1 Score
Segmented	0.748693	0.657175	0.738796	0.695600
Segmented and augmented	**0.757419**	**0.716915**	**0.754450**	**0.735204**

Table 9: SVM testing results.

7.3 Performance Analysis

The following table (Table 10) shows the time complexity for SVM model and for each of the CNN models using datasets and without augmented data. Additionally, we used Google's colab that have over 500 GPUs to speed up the run.

Model	Dataset	Training Time
EfficientNetB7	Segmented	1253 s
EfficientNetB7	**Segmented and augmented**	**1325 s**
EfficientNetB5	Segmented	6003 s
EfficientNetB5	**Segmented and augmented**	**6368 s**
ResNet50V2	Segmented	787 s
ResNet50V2	**Segmented and augmented**	**909 s**
VGG16	Segmented	915s
VGG16	**Segmented and augmented**	**918 s**
SVM	Segmented	3983 s
SVM	**Segmented and augmented**	**5186 s**

Table 10: Models' Training Time

7.4 Discussion

Related works show that DL models outperform shallow models, such as SVM, in skin cancer detection, Our training results demonstrate that as well. The deep models achieved significantly higher accuracy rates in classifying skin lesions as malignant or benign, indicating the superiority of DL techniques for this task.

As Sedigh et al. [71] mentioned in their work about the good performance models could achieve using GAN, we observed that image augmentation using GAN generally improved the performance of our models, particularly in terms of accuracy and recall. As recall is an important metric in automated skin lesion

detection, our findings suggest that GAN-based image augmentation techniques can be a valuable tool in improving the detection of malignant skin lesions. This is particularly important as the detection of malignant lesions is more critical than that of benign ones, as early detection of malignant skin cancer can significantly improve patient outcomes.

The model with the highest performance was EfficientNet-B7. This model was not used in the literature review, which encouraged us to train it and evaluate its performance to enrich the field of skin cancer with DL.

Overall, our results highlight the potential of DL techniques and image augmentation methods for skin cancer detection and pave the way for further research in this area.

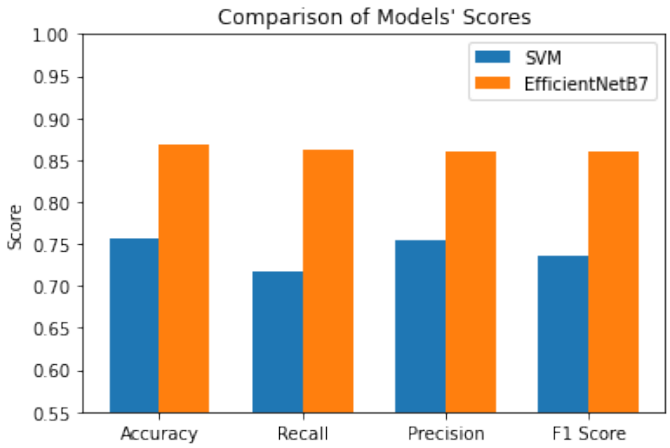


Figure 37: Comparison between best cnn model and best SVM model.

8 Conclusion

As research has recently gravitated toward DL algorithms for image recognition, the automated detection of skin cancer using these DL algorithms can be used by patients and physicians to more easily and accurately diagnose skin cancer. The majority of the papers we found in the field of skin cancer detection focused on applying DL algorithms to improve the process of detecting skin cancer since it yields promising results quickly, and it is an expensive and difficult process if the patient does it without automation. Moreover, we discovered a dearth of research focusing on shallow learning algorithms. In contrast, models created using DL were common. DL can serve as a standard for skin cancer detection by assisting medical professionals. In the field of skin cancer detection, there are currently few large datasets for skin cancer imaging that can be used for supervised learning, which is the reason for the interest and application of data augmentation in prior work. In this research, we tackled the problem of detecting skin cancer by segmenting the tumors from skin lesions in the images using U-Net, generating more images to deal with the lack of large datasets using GAN, implementing the pre-trained CNN models: ResNetV2, VGG16, EfficientNet B5, and EfficientNet B7, and compared their performance with SVM. After presenting and analyzing the models' performance, we found that EfficientNet-B7 has the highest accuracy which is 86.91%. Our study showed that image augmentation using GAN models can help complex DL models' accuracy in detecting malignant skin lesions, it also highlights the need for larger and more diverse datasets to further improve the performance of DL algorithms in this domain. Future research should focus on addressing these limitations and explore the potential of other DL techniques, such as attention-based models, to improve skin cancer detection. Additionally, there is a need to conduct clinical studies to evaluate the real-world effectiveness of these algorithms and their impact on patient outcomes. Overall, we believe that continued research in this area has the potential to make a significant impact on the field of dermatology and ultimately improve patient care.

References

1. Narayanamurth *et al.*, "Skin cancer detection using non-invasive techniques," *RSC advances*, vol. 8, no. 49, pp. 28095–28130, 2018.

2. S. Jain, N. Pise, *et al.*, "Computer aided melanoma skin cancer detection using image processing," *Procedia Computer Science*, vol. 48, pp. 735–740, 2015.

3. C. A. Lieber, S. K. Majumder, D. D. Billheimer, D. L. Ellis, and A. Mahadevan-Jansen, "Raman microspectroscopy for skin cancer detection in vitro," *Journal of biomedical optics*, vol. 13, no. 2, p. 024013, 2009.

4. P. Dubal, S. Bhatt, C. Joglekar, and S. Patil, "Skin cancer detection and classification," in *2017 6th international conference on electrical engineering and informatics (ICEEI)*, pp. 1–6, IEEE, 2017.

5. G. P. Guy Jr, C. C. Thomas, T. Thompson, M. Watson, G. M. Massetti, and L. C. Richardson, "Vital signs: melanoma incidence and mortality trends and projections—united states, 1982–2030," *Morbidity and mortality weekly report*, vol. 64, no. 21, p. 591, 2015.

6. The International Skin Imaging Collaboration (ISIC), "Melanoma Project." `https://www.isic-archive.com/`, accessed 2021-10-01.

7. Rotemberg *et al.*, "A patient-centric dataset of images and metadata for identifying melanomas using clinical context," *Scientific data*, vol. 8, no. 1, pp. 1–8, 2021.

8. B. K. Armstrong and A. Kricker, "The epidemiology of UV induced skin cancer," *J. Photochem. Photobiol. B*, vol. 63, pp. 8–18, Oct. 2001.

9. B. Rao *et al.*, "Can early malignant melanoma be differentiated from atypical melanocytic nevi by in vivo techniques? part i. clinical and dermoscopic characteristics," *Skin Research and Technology*, vol. 3, no. 1, pp. 8–14, 1997.

10. D. N. H. Thanh, V. B. S. Prasath, L. M. Hieu, and N. N. Hien, "Melanoma skin cancer detection method based on adaptive principal curvature, colour normalisation and feature extraction with the ABCD rule," *J. Digit. Imaging*, vol. 33, pp. 574–585, June 2020.

11. A. L. Samuel, "Some studies in machine learning using the game of checkers," *IBM J. Res. Dev.*, vol. 3, pp. 210–229, July 1959.

12. A. Burkov, *The hundred-page machine learning book*. Andriy Burkov, Jan. 2019.

13. F. Zantalis, G. Koulouras, S. Karabetsos, and D. Kandris, "A review of machine learning and IoT in smart transportation," *Future internet*, vol. 11, p. 94, Apr. 2019.

14. S. Ghosh, A. Dasgupta, and A. Swetapadma, "A study on support vector machine based linear and non-linear pattern classification," in *2019 International Conference on Intelligent Sustainable Systems (ICISS)*, IEEE, Feb. 2019.

15. R. Yuan, Z. Li, X. Guan, and L. Xu, "An SVM-based machine learning method for accurate internet traffic classification," *Inf. Syst. Front.*, vol. 12, pp. 149–156, Apr. 2010.

16. J. Alzubi, A. Nayyar, and A. Kumar, "Machine learning from theory to algorithms: An overview," *J. Phys. Conf. Ser.*, vol. 1142, p. 012012, Nov. 2018.

17. M. Azhari, A. Alaoui, Z. Achraoui, B. Ettaki, and J. Zerouaoui, "Adaptation of the random forest method: solving the problem of pulsar search," in *Proceedings of the 4th International Conference on Smart City Applications*, pp. 1–6, 2019.

18. B. Pan, "Application of xgboost algorithm in hourly pm2. 5 concentration prediction," in *IOP conference series: earth and environmental science*, vol. 113, p. 012127, IOP publishing, 2018.

19. T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.

20. I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

21. A. Abraham, "Artificial neural networks," *Handbook of measuring system design*, 2005.

22. M. H. Hassoun *et al.*, *Fundamentals of artificial neural networks*. MIT press, 1995.

23. R. Bartzatt, "Determination of dermal permeability coefficient (kp) by utilizing multiple descriptors in artificial neural network analysis and multiple regression analysis," *J Sci Res Rep*, vol. 3, pp. 2884–2899, 2014.

24. S.-H. Han, K. W. Kim, S. Kim, and Y. C. Youn, "Artificial neural network: understanding the basic concepts without mathematics," *Dementia and Neurocognitive Disorders*, vol. 17, no. 3, pp. 83–89, 2018.

25. P. Gnanasekaran *et al.*, "Ground water level estimator with back propagation neural network classification using machine learning approach," *Ground Water Level Estimator With Back Propagation Neural Network Classification Using Machine Learning Approach (June 29, 2022)*, 2022.

26. S. Sharma, S. Sharma, and A. Athaiya, "Activation functions in neural networks," *towards data science*, vol. 6, no. 12, pp. 310–316, 2017.

27. T. M. Jamel and B. M. Khammas, "Implementation of a sigmoid activation function for neural network using fpga," in *13th Scientific Conference of Al-Ma'moon University College*, vol. 13, 2012.

28. S. A. Balaji and K. Baskaran, "Design and development of artificial neural networking (ann) system using sigmoid activation function to predict annual rice production in tamilnadu," *arXiv preprint arXiv:1303.1913*, 2013.

29. F. M. Shakiba and M. Zhou, "Novel analog implementation of a hyperbolic tangent neuron in artificial neural networks," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 11, pp. 10856–10867, 2020.

30. J. Shamsi, A. Amirsoleimani, S. Mirzakuchaki, A. Ahmade, S. Alirezaee, and M. Ahmadi, "Hyperbolic tangent passive resistive-type neuron," in *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 581–584, IEEE, 2015.

31. J. Feng and S. Lu, "Performance analysis of various activation functions in artificial neural networks," in *Journal of physics: conference series*, vol. 1237, p. 022030, IOP Publishing, 2019.

32. S. Ray, "A quick review of machine learning algorithms," in *2019 International conference on machine learning, big data, cloud and parallel computing (COMITCon)*, pp. 35–39, IEEE, 2019.

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

34. B. Ghosh, I. K. Dutta, A. Carlson, M. Totaro, and M. Bayoumi, "An empirical analysis of generative adversarial network training times with varying batch sizes," in *2020 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pp. 0643–0648, IEEE, 2020.

35. S. H. Haji and A. M. Abdulazeez, "Comparison of optimization techniques based on gradient descent algorithm: A review," *PalArch's Journal of Archaeology of Egypt/Egyptology*, vol. 18, no. 4, pp. 2715–2743, 2021.

36. R. Senthilnathan, "Deep learning in vision-based automated inspection: Current state and future prospects," *Machine Learning in Industry*, pp. 159–175, 2022.

37. Haggenmüller *et al.*, "Skin cancer classification via convolutional neural networks: systematic review of studies involving human experts," *Eur. J. Cancer*, vol. 156, pp. 202–216, Oct. 2021.

38. J. Brownlee, "How do convolutional layers work in deep learning neural networks," *Machine Learning Mastery*, vol. 17, 2019.

39. H. Gu, Y. Wang, S. Hong, and G. Gui, "Blind channel identification aided generalized automatic modulation recognition based on deep learning," *IEEE Access*, vol. 7, pp. 110722–110729, 2019.

40. Tschandl *et al.*, "Expert-level diagnosis of nonpigmented skin cancer by combined convolutional neural networks," *JAMA Dermatol.*, vol. 155, pp. 58–65, Jan. 2019.

41. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

42. Goodfellow *et al.*, "Generative adversarial networks," *arXiv e-prints*, June 2014.

43. T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of StyleGAN," *arXiv e-prints*, Dec. 2019.

44. T. S. Silva, "A short introduction to generative adversarial networks," *https://sthalles.github.io*, 2017.

45. L. Torrey and J. Shavlik, "Transfer learning," in *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pp. 242–264, IGI global, 2010.

46. Zhuang *et al.*, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2020.

47. S. Ghoury, C. Sungur, and A. Durdu, "Real-time diseases detection of grape and grape leaves using faster r-cnn and ssd mobilenet architectures," in *International conference on advanced technologies, computer engineering and science (ICATCES 2019)*, pp. 39–44, 2019.

48. Howard *et al.*, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

49. D. Sinha and M. El-Sharkawy, "Thin mobilenet: An enhanced mobilenet architecture," in *2019 IEEE 10th annual ubiquitous computing, electronics & mobile communication conference (UEMCON)*, pp. 0280–0285, IEEE, 2019.

50. S. Pouyanfar, S.-C. Chen, and M.-L. Shyu, "An efficient deep residual-inception network for multimedia classification," in *2017 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 373–378, IEEE, 2017.

51. K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014.

52. M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," 2019.

53. R. C. Gonzalez and R. E. Woods, *Digital image processing*. Upper Saddle River, NJ: Pearson, 2 ed., Nov. 2001.

54. G. Stockman and L. G. Shapiro, *Computer Vision*. Upper Saddle River, NJ: Pearson, Jan. 2001.

55. A. K. Jain, *Fundamentals of digital image processing*. Upper Saddle River, NJ: Pearson, Sept. 1988.

56. S. M. Seyyed Ebrahimi, H. Pourghassem, and M. Ashourian, "Lesion detection in dermoscopy images using sarsa reinforcement algorithm," in *2010 17th Iranian Conference of Biomedical Engineering (ICBME)*, IEEE, Nov. 2010.

57. N. Siddique, S. Paheding, C. P. Elkin, and V. Devabhaktuni, "U-net and its variants for medical image segmentation: A review of theory and applications," *Ieee Access*, vol. 9, pp. 82031–82057, 2021.

58. T. Acharya and A. Ray, *Image Processing: Principles and Applications*. Wiley, 2005.

59. Y. N. Fu'adah, N. C. Pratiwi, M. A. Pramudito, and N. Ibrahim, "Convolutional neural network (cnn) for automatic skin cancer classification system," in *IOP Conference Series: Materials Science and Engineering*, vol. 982, p. 012005, IOP Publishing, 2020.

60. T. M. Alam, M. Mushtaq, K. Shaukat, I. A. Hameed, M. Umer Sarwar, and S. Luo, "A novel method for performance measurement of public educational institutions using machine learning models," *Applied Sciences*, vol. 11, no. 19, 2021.

61. M. A. Al-masni, M. A. Al-antari, H. M. Park, N. H. Park, and T.-S. Kim, "A deep learning model integrating FrCN and residual convolutional networks for skin lesion segmentation and classification," in *2019 IEEE Eurasia Conference on Biomedical Engineering, Healthcare and Sustainability (ECBIOS)*, IEEE, May 2019.

62. J. Daghrir, L. Tlig, M. Bouchouicha, and M. Sayadi, "Melanoma skin cancer detection using deep learning and classical machine learning techniques: A hybrid approach," in *2020 5th International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*, IEEE, Sept. 2020.

63. R. Sarkar, C. C. Chatterjee, and A. Hazra, "Diagnosis of melanoma from dermoscopic images using a deep depthwise separable residual convolutional network," *IET Image Process.*, vol. 13, pp. 2130–2142, Oct. 2019.

64. R. Garg, S. Maheshwari, and A. Shukla, "Decision support system for detection and classification of skin cancer using CNN," *arXiv e-prints*, Dec. 2019.

65. H. Nahata and S. P. Singh, "Deep learning solutions for skin cancer detection and diagnosis," in *Learning and Analytics in Intelligent Systems*, pp. 159–182, Cham: Springer International Publishing, 2020.

66. Y. N. Fu'adah, N. K. C. Pratiwi, M. A. Pramudito, and N. Ibrahim, "Convolutional neural network (CNN) for automatic skin cancer classification system," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 982, p. 012005, Dec. 2020.

67. M. A. Albahar, "Skin lesion classification using convolutional neural network with novel regularizer," *IEEE Access*, vol. 7, pp. 38306–38313, 2019.

68. M. Hasan, S. D. Barman, S. Islam, and A. W. Reza, "Skin cancer detection using convolutional neural network," in *Proceedings of the 2019 5th International Conference on Computing and Artificial Intelligence - ICCAI '19*, (New York, New York, USA), ACM Press, 2019.

69. S. Mustafa, A. B. Dauda, and M. Dauda, "Image processing and SVM classification for melanoma detection," in *2017 International Conference on Computing Networking and Informatics (ICCNI)*, IEEE, Oct. 2017.

70. N. Rezaoana, M. S. Hossain, and K. Andersson, "Detection and classification of skin cancer by using a parallel CNN model," in *2020 IEEE International Women in Engineering (WIE) Conference on Electrical and Computer Engineering (WIECONECE)*, IEEE, Dec. 2020.

71. P. Sedigh, R. Sadeghian, and M. T. Masouleh, "Generating synthetic medical images by using GAN to improve CNN performance in skin cancer classification," in *2019 7th International Conference on Robotics and Mechatronics (ICRoM)*, IEEE, Nov. 2019.

72. T. Lee, V. Ng, R. Gallagher, A. Coldman, and D. McLean, "Dullrazor®: A software approach to hair removal from images," *Computers in Biology and Medicine*, vol. 27, no. 6, pp. 533–543, 1997.

73. V. Agarwal, "Complete architectural details of all efficientnet models," *Medium Toward Data Science*, 2020.

74. Z. M. Hira and D. F. Gillies, "A review of feature selection and feature extraction methods applied on microarray data," *Advances in bioinformatics*, vol. 2015, 2015.

75. Z. Reitermanova *et al.*, "Data splitting," in *WDS*, vol. 10, pp. 31–36, Matfyzpress Prague, 2010.

76. N. K. Manaswi and N. K. Manaswi, "Understanding and working with keras," *Deep learning with applications using Python: Chatbots and face, object, and speech recognition with TensorFlow and Keras*, pp. 31–43, 2018.

77. G. Bradski and A. Kaehler, *Learning OpenCV: Computer vision with the OpenCV library*. " O'Reilly Media, Inc.", 2008.

78. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

79. E. Bressert, "Scipy and numpy: an overview for developers," 2012.

80. M. Z. A. Pon and K. P. KK, "Hyperparameter tuning of deep learning models in keras," *Sparklinglight Transactions on Artificial Intelligence and Quantum Computing (STAIQC)*, vol. 1, no. 1, pp. 36–40, 2021.

81. F. Malik, "What is grid search," *Medium, FinTechExplained*, 2020.

82. F. Chollet *et al.*, "Keras," 2015.