

Dermatological Decision Support Systems using CNN for Binary Classification

Rajendra Dev Dondapati

Annamalai University, India | Department of Computer Science & Engineering, Vignan's Institute of Engineering for Women, India
rajendra0511@gmail.com (corresponding author)

Thangaraju Sivaprakasam

Department of Computer Science & Engineering, Annamalai University, India
tsivaprakasam@gmail.com

Kollati Vijaya Kumar

Department of Computer Science & Engineering, Gitam University, India
vjkmr776@gmail.com

Received: 5 March 2024 | Revised: 26 March 2024 | Accepted: 28 March 2024

Licensed under a CC-BY 4.0 license | Copyright (c) by the authors | DOI: <https://doi.org/10.48084/etasr.7173>

ABSTRACT

Skin cancer diagnosis, particularly melanoma detection, is an important healthcare concern worldwide. This study uses the ISIC2017 dataset to evaluate the performance of three deep learning architectures, VGG16, ResNet50, and InceptionV3, for binary classification of skin lesions as benign or malignant. ResNet50 achieved the highest training-set accuracy of 81.1%, but InceptionV3 outperformed the other classifiers in generalization with a validation accuracy of 76.2%. The findings reveal the various strengths and trade-offs of alternative designs, providing important insights for the development of dermatological decision support systems. This study contributes to the progress of automated skin cancer diagnosis and establishes the framework for future studies aimed at improving classification accuracy.

Keywords-ISIC 2017; VGG16; ResNet50; InceptionV3

I. INTRODUCTION

Skin malignancy, such as melanoma, is a major global health concern, with rising incidence rates worldwide. Early and correct diagnosis is critical to improving patient outcomes and lowering mortality rates. Dermatologists frequently use visual inspection and histological examination to diagnose possible skin malignancies, which can be subjective and time-consuming. In recent years, the introduction of deep learning techniques, particularly Convolutional Neural Networks (CNNs), has yielded encouraging results in automating and improving dermatological diagnosis [1]. The ISIC2017 dataset includes a broad collection of skin lesion images and is employed as a benchmark to assess the effectiveness of CNN-based classification algorithms [2]. However, the successful deployment of CNNs for dermatological diagnosis necessitates careful consideration of preprocessing techniques and the selection of appropriate base models. This study provides a comprehensive examination of the efficacy of CNNs, notably VGG-16, ResNet50, and InceptionV3, for the binary classification of skin lesions as benign or seborrheic keratosis.

A novel skin cancer classification model was presented in [4], blending a pre-trained CNN with a feature optimization

method. This study adopted preprocessing methods for image enhancement in addition to a customized dataset derived from the ISIC dataset. This model achieved an accuracy of more than 98% by combining an optimized feature vector with SVM classifiers, showing increased prediction speed and training time efficiency. In [5], the ASCDC-CSODL method was devised, combining deep learning and cat swarm optimization to detect and classify skin cancer. In [6], the BHESKD-ODL model was introduced for skin lesion diagnosis. This model put into service the blockchain technology to ensure the secure storage of medical photos, while homomorphic encryption was implemented to safeguard image security. In [7], a comparative analysis of VGG-16 and multilayer perceptron was performed on the classification of skin cancer in the HAM10000 dataset. In [8], a new framework for the automated diagnosis of melanoma, a highly fatal skin cancer, was presented. The results exhibited a promising advancement in early detection and classification of skin lesions, which could lead to improved patient outcomes and reduced healthcare burden.

In [9], a unique Concatenated Xception-ResNet50 model was presented for skin cancer detection. After rigorous testing following the sliding-window technique, the model achieved a remarkable classification accuracy of 97.8%. In [10], the

efficacy of the refined ResNet50 model was evaluated to accurately identify different skin cancer grades using dermoscopic lesion images. In [11], the focus was on early detection and diagnosis of skin cancer engaging deep learning techniques, including ResNet-50, VGG16, and CNN architectures. This study highlighted the effectiveness of deep learning-based approaches in increasing the accuracy and efficiency of dermatology diagnostics. In [12] deep learning algorithms were applied to classify skin cancer images. This study demonstrated an optimization method, involving transfer learning and pre-trained models. The experiments demonstrated the possibility of achieving approximately 70% accuracy in skin cancer classification, highlighting the importance of model parameters and training constraints in optimizing accuracy and minimizing overfitting, advancing the field of dermatology decision support systems. In [13], a comprehensive approach based on deep learning techniques was presented for skin cancer detection. This model utilized a convolutional autoencoder for feature extraction, a 3D CNN network for classification using transfer learning from an Inception V3-trained model, and data preprocessing techniques to outperform the current state-of-the-art systems, having 0.96 accuracy, 0.97 sensitivity, and 0.97 specificity.

This study aims to improve the efficiency and accuracy of skin cancer screenings, enabling earlier detection and intervention and ultimately improving patient outcomes in the face of this global health crisis. The significance of this study is derived from its extensive examination of deep learning architectures for the binary categorization of skin lesions, providing useful insights for the development of dermatological decision support systems. The study's contribution includes setting a framework for better automatic skin cancer identification and categorization by finding strengths and performance gaps between the VGG16, ResNet50, and InceptionV3 models. The objectives of this research include:

- Assess the efficacy of CNNs, specifically VGG-16, ResNet50, and InceptionV3, in binary classifying skin lesions as malignant or benign using the ISIC2017 dataset.
- Investigate how preprocessing strategies, particularly the use of a median filter, affect the performance of CNN-based dermatological decision support systems.
- Compare the accuracy and computational efficiency of several CNN architectures for skin lesion classification.
- Evaluate the robustness of CNNs to variations in picture features, such as texture, form, and size of skin lesions, to establish whether CNN models are suitable for real-world clinical applications.
- Provide doctors with significant insights into the benefits and limits of these CNN architectures for dermatological diagnosis, allowing them to select relevant models for rapid and reliable skin lesion detection [3].

II. PROPOSED MODEL

Figure 1 demonstrates the proposed architecture of the system implemented, which consists of the following:

- Use the ISIC 2017 dataset as an input.
- Dataset Preprocessing: A median filter is employed to reduce noise while keeping edge and texture information in skin lesion photos from the dataset.
- Use the following CNN architectures:
 - VGG-16: selected for its simplicity and efficacy, with fewer layers than more sophisticated systems.
 - ResNet50: chosen due to its deep architecture with residual connections, which allows for deeper network training without the risk of disappearing gradients.
 - InceptionV3: selected for its inception modules, enabling effective feature extraction at different scales by combining numerous filter sizes inside the same layer.
- Binary classification of skin lesions into benign and melanoma categories, with emphasis on the efficacy of various CNN architectures for dermatological decision support systems.
- Metrics for performance evaluation: The classification accuracy, sensitivity, specificity, and processing efficiency of each CNN design are resistant to variations in image qualities, such as the shape, size, and texture of skin lesions.

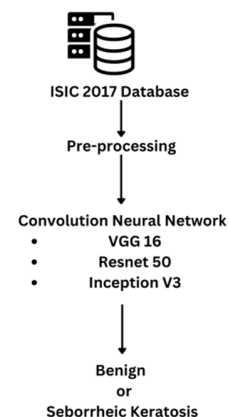


Fig. 1. Proposed architecture.

III. METHODOLOGY

A. Dataset Preparation

The ISIC 2017 dataset is a commonly deployed benchmark dataset in dermatology, particularly for skin lesion classification. This dataset contains a diverse collection of high-resolution dermoscopic images covering a wide range of skin diseases and conditions, providing a complete picture of dermatological abnormalities [14]. One significant feature of the ISIC 2017 dataset is the presence of two separate classes relevant to this research: benign lesions and seborrheic keratosis. These groups represent common skin disorders seen in clinical practice, with benign lesions spanning a wide spectrum of non-cancerous abnormalities. Seborrheic keratosis refers to a benign growth observed on the skin [15].

This study focuses on these two classes to address the challenge of discriminating between them [16].

B. Dataset Preprocessing

Preprocessing is an important step in any machine learning pipeline, especially in medical image analysis, where the quality of input data has a substantial impact on the performance of subsequent models. In the case of the ISIC 2017 dataset, which contains dermoscopic images of skin lesions, median filtering is an appropriate preprocessing strategy. This approach efficiently reduces noise in images, while retaining critical edges and texture information required for accurate lesion categorization. The use of a median filter with a size of three reduces outliers and contributes to the robustness of the preprocessing step [17]. Median filtering is computationally efficient and requires fewer parameters to configure than other algorithms, which makes it an appealing option for large-scale datasets such as ISIC 2017. Its simplicity and proven ability to retain image quality make it a popular choice for medical image analysis applications that require conserving key features while reducing noise. Median filtering eliminates noise and improves the clarity and accuracy of skin lesion images, which makes it easier to comprehend features recovered by CNNs. This preprocessing strategy is consistent with the best practices in medical image analysis, where preserving diagnostic information while minimizing artifacts is critical for reliable and accurate diagnosis [18]. By incorporating these preprocessing steps into the model pipeline, the performance of the CNN model is improved for the binary classification of skin lesions into benign and melanoma categories, resulting in the development of robust dermatological decision support systems with enhanced diagnostic accuracy and clinical utility [19].

The median filter transforms each element (y, i) of the input signal (M) into the equivalent element (n, i) of the output signal. Let l represent the number of elements in the input signal X . For each element y_i in Y , the median filter selects a subset centered on y_i . It contains entries from Y with indices ranging from $y_{(i-n)}$ to $y_{(i+n)}$.

$$M_i = \{y_{i-n}, y_{i-n+1}, \dots, y_{i-1}, y_i, y_{i+1}, \dots, y_{i+n-1}, y_{i+n}\}$$

The median filter is computed using:

$$Y_i = \text{median}(M_i)$$

IV. CONCURRENT NEURAL NETWORKS (CNNS)

This subsection explores the three CNN models used in the model architecture.

A. VGG-16

VGG-16 has a simple architecture, which includes stacked convolutional layers with modest 3×3 filters and max-pooling layers, followed by fully linked layers at the end. This simplicity makes it easier to understand and interpret the learned features, which is critical in medical applications where interpretability is required for clinical approval. It has been trained on large-scale datasets, such as ImageNet, allowing it to capture generic features that can then be fine-tuned for specialized applications, such as skin lesion detection [20]. VGG-16 has exhibited a comparable performance in a variety

of image classification benchmarks. Algorithm 1 demonstrates the VGG-16 framework employed and Figure 2 the model summary.

ALGORITHM 1: TRAINING VGG-16 MODEL

```

Require:
df: DataFrame with filenames and labels
training_epochs: Number of epochs for training
image_dir: Directory that contains images
batch_size: Batch size
1. Load the pre-trained VGG16 base model with ImageNet weights
2. Freeze base model layers to avoid retraining
3. Create model architecture for binary classification
3.1: Add GlobalAveragePooling2D layer after the base model
3.2: Add a dense layer with sigmoid activation for binary classification
4. Compile the model using the Adam optimizer with binary cross-entropy loss
5. Set up ImageDataGenerator with a preprocessing function for VGG16 and validation split
6: Configure training and validation data generators
6.1: Configure generators to import photos from DataFrame and adjust target and batch sizes
6.2: Divide data into training and validation subsets according to the validation split
7. Train the model
7.1: Iterate across epochs
7.1.1: Iterate across batches in the training generator
7.1.1.1: Load batches of photos and labels from the training generator
7.1.1.2: Train the model in batches using the fit() function
7.1.2: Evaluate model performance on validation data using validation generator
7.1.3: Update model weights according to training progress
8: End the training loop

```

Model: "sequential_4"

Layer (type)	Output Shape	Param #
vgg16 (Functional)	(None, 7, 7, 512)	14714688
global_average_pooling2d_4 (GlobalAveragePooling2D)	(None, 512)	0
dense_4	(None, 1)	513
Total params: 14715201 (56.13 MB)		
Trainable params: 513 (2.00 KB)		
Non-trainable params: 14714688 (56.13 MB)		

Fig. 2. VGG-16 model summary.

The VGG16 architecture is made up of multiple convolutional layers, followed by fully connected layers. Let K be the input picture to the network's fully connected layer before the softmax activation.

- Convolutional Layers: Let M^l represent the filters (weights) of the l -th convolutional layer, and x^l represent the biases. Z denotes the output of the previous layer. The output from the l -th convolutional layer is calculated as:

$$K^l = \text{ReLU}(M^l * Z^{l-1} + b^l)$$

- Max Pooling Layers: After each convolutional block, max pooling is used to downsample the feature maps. Let g be the size of the maximum pooling filter. The output of the max pooling layer is calculated as:

$$Z^l = \text{MaxPool}(Z^{l-1}, g)$$

- Fully Connected Layers: After numerous convolutional and max pooling layers, the feature maps are flattened before being input into fully connected layers. Let M^{gc} represent the fully connected layer's weights, and b^{gc} represent the biases [21]. The output of the completely connected layer is calculated as follows:

$$Z^{gc} = \text{ReLU}(M^{gc}.Z^l + b^{gc})$$

- Output Layer: To determine the output probabilities for each class, a softmax activation function is applied after the final fully connected layer.
- Let M^{out} represent the output layer's weights, and b^{out} represent the biases. \hat{Y} represents the predicted probability. The output of the final layer is calculated as follows:

$$\hat{Y} = \text{Softmax}(M^{out}.Z^{gc} + b^{out})$$

B. Resnet50

ResNet50 is flexible due to its deep design with residual connections, which allows for the training of deeper networks while avoiding the vanishing gradient problem. The remaining connections enable direct gradient propagation during training, reducing deterioration in deeper networks and allowing more effective feature learning [22]. ResNet50 has also proven its high performance in a variety of image classification tasks, including medical imaging, which aims to binary classify skin lesions as benign or seborrheic keratosis. Algorithm 2 demonstrates the Resnet50 framework employed and Figure 3 depicts the model summary. The ResNet50 architecture incorporates the concept of residual blocks, which feature skip connections (also known as identity shortcuts), allowing for the training of very deep neural networks without vanishing gradients [23]. Figure 4 shows the internal layers of the ResNet50 framework.

- Identity shortcut: Connects the input directly before traversing other levels. The residual function (F) is implemented by the layers in the residual block. The weights of the layers (W) are denoted by $[l, i]$. The addition operation combines the shortcut connection with the residual function output.

$$Y^{l+1} = F(Y^l + \{W^{l,i}\}) + Y^l$$

- The residual function is commonly composed of convolutional layers, batch normalization, and ReLU activations. Let $W[l, i]$ represent the weights of the i^{th} layer in the residual block. L is the number of layers in the residual block and BatchNorm. BatchNorm refers to the batch normalization operation.

$$F(Y^l, \{W^{l,i}\}) = W^{l,L}. \text{ReLU}(W^{l,L-1}. \text{BatchNorm}(W^{l,L-2}. \text{ReLU}(\dots)))$$

- Output Layer: After passing through many residual blocks, the final output of the ResNet50 architecture is created by applying a global average pooling operation followed by a fully connected layer with softmax activation [24].

$$\hat{Y} = \text{Softmax}(W^{out}. \text{GlobalAvgPool}(Y^l) + b^{out})$$

ALGORITHM 2: TRAINING RESNET50 MODEL

```

Require:
df: DataFrame with filenames and labels
image_dir: Directory with images
Batch size and epochs used for training
1. Load ImageNet weights into the pre-trained ResNet50 base model:
   base_model ← ResNet50(weights='imagenet',
                           include_top=False, input_shape=(224,224,3))
2: Base model layers should be frozen by setting layer to prevent retraining. For every layer in base_model, trainable to false
3: Create a model architecture for binary classification
3.1: Add GlobalAveragePooling2D layer after the base model: model ← Sequential([base_model, GlobalAveragePooling2D()])
3.2: Add a dense layer with sigmoid activation for binary classification using
   model.add(Dense(units=1,activation='sigmoid'))
4: Compile the model with the Adam optimizer with binary cross-entropy loss:
   model.Compile(optimizer=Adam(learning_rate=0.0001), loss='binary_crossentropy', metrics=['accuracy'])
5: Initialize ImageDataGenerator with ResNet50's preprocessing function and validation split:
   datagen ← ImageDataGenerator(preprocessing_function=preprocess_input; validation_split=0.2)
6: Configure training and validation data generators
6.1: Configure generators to load pictures from DataFrame, and adjust target and batch sizes:
   train_generator →
     datagen.flow_from_dataframe(dataframe=df,
                               directory=image_dir, x_col='filename',
                               y_col='label', target_size=(224, 224),
                               batch_size=batch_size, class_mode='binary',
                               subset='training')
   validation_generator →
     datagen.flow_from_dataframe(dataframe=df,
                               directory=image_dir, x_col='filename',
                               y_col='label', target_size=(224,224),
                               batch_size=batch_size, class_mode='binary',
                               subset='validation')
7. Train the model.
7.1: Iterate across epochs:
7.1.1: Iterate across batches in the training generator.
7.1.1.1: Load images and labels from the training generator: (X_train, y_train) ← train_generator.next()
7.1.1.2: Train the model in batches using the fit() function: model.fit(X_train, y_train; epochs=1, verbose=0)
7.1.2: Use the validation generator to evaluate model performance on validation data
   Val_loss, val_accuracy Use model.evaluate(validation_generator).
7.1.3: Update model weights as training progresses: (No explicit steps in pseudocode)
8: End the training loop

```

Model: "sequential_2"

Layer (type)	Output Shape	Param #
resnet50 (Functional)	(None, 7, 7, 2048)	23587712
global_average_pooling2d_2 (GlobalAveragePooling2D)	(None, 2048)	0
dense_2 (Dense)	(None, 1)	2049

Total params: 23589761 (89.99 MB)

Trainable params: 2049 (8.00 KB)

Non-trainable params: 23587712 (89.98 MB)

Fig. 3. Resnet50 model summary.

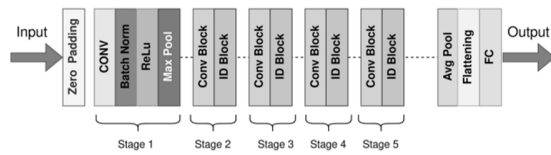


Fig. 4. Resnet50 Architecture.

C. InceptionV3

InceptionV3 includes several filter sizes inside each layer, permitting effective feature extraction at varied scales. This allows it to collect both local and global data effectively, which is very useful for identifying skin lesions of various textures, forms, and sizes. The InceptionV3 has been pre-trained on a large-scale dataset (ImageNet), giving it a comprehension of the visual features that can be refined for skin lesion classification tasks [25]. Its demonstrated effectiveness in picture classification tasks, along with its ability to capture multi-scale characteristics, makes it a significant addition. Algorithm 3 displays the InceptionV3 framework employed and Figure 5 shows its model summary [26].

ALGORITHM 3: INCEPTIONV3 FRAMEWORK

```

Require:
df: DataFrame with filenames and labels
image_dir: Directory with images
Batch size and epochs used for training
1. Load the pre-trained InceptionV3 base model with ImageNet
weights: base_model ← InceptionV3(weights='imagenet',
include_top=False, input_shape=(224, 224, 3))
2. To avoid retraining, freeze base model layers by setting
layer.trainable to false for each layer in base_model
3. Create a model architecture for binary classification
3.1: Add GlobalAveragePooling2D layer after the base model:
model ← Sequential([base_model,
GlobalAveragePooling2D()]).
3.2: Add a dense layer with sigmoid activation for binary
classification. model.add(Dense(units=1,
activation='sigmoid'))
4. Compile the model with the Adam optimizer with binary
cross-entropy loss: model.Compile(optimizer=
Adam(learning_rate=0.0001),
loss='binary_crossentropy',
metrics=['accuracy'])
5. Set up ImageDataGenerator with preprocessing function for
InceptionV3 and validation split:
datagen → ImageDataGenerator(
preprocessing_function=preprocess_input,
validation_split=0.2).
6. Configure training and validation data generators.
6.1: Configure generators to load pictures from DataFrame,
and adjust target and batch sizes:
train_generator → datagen.flow_from_dataframe(
dataframe=df, directory=image_dir,
x_col='filename', y_col='label', target_size=(224,
224), batch_size=batch_size, class_mode='binary',
subset='training')
7. Train the model
7.1: Iterate across epochs:
7.1.1: Iterate across batches in the training generator.
7.1.1.1: Load images and labels from the training
generator: (X_train, y_train) ←
train_generator.next().
7.1.1.2: Train the model in batches using the fit()
function: model.fit(X_train,y_train;
epochs=1,verbose=0)
7.1.2: Use the validation generator to evaluate model
performance on validation data
Val_loss, val_accuracy Use
model.evaluate(validation_generator)
7.1.3: Update model weights as training progresses
(No explicit steps in pseudocode)
8: End the training loop

```

Model: "sequential_3"

Layer (type)	Output Shape	Param #
inception_v3 (Functional)	(None, 8, 8, 2048)	21802784
global_average_pooling2d_3 (GlobalAveragePooling2D)	(None, 2048)	0
dense_3 (Dense)	(None, 1)	2049
Total params: 21804833 (83.18 MB)		
Trainable params: 2049 (8.00 KB)		
Non-trainable params: 21802784 (83.17 MB)		

Fig. 5. InceptionV3 model summary.

Convolutions are conducted in an inception module using filters of various sizes (for example, 1×1 , 3×3 , and 5×5) [27]. Let the weights of the filters for each size be represented by W . The output feature maps for each filter size are indicated as Y , respectively. The output feature maps are calculated as follows:

$$Y_{1 \times 1} = \text{ReLU}(W_{1 \times 1} * X)$$

$$Y_{3 \times 3} = \text{ReLU}(W_{3 \times 3} * X)$$

$$Y_{5 \times 5} = \text{ReLU}(W_{5 \times 5} * X)$$

Convolutions with varied filter sizes produce feature maps, which are then concatenated along the channel dimension to create the final output map Y . The concatenation operation is as follows:

$$Y = \text{Concat}(Y_{1 \times 1}, Y_{3 \times 3}, Y_{5 \times 5})$$

D. Evaluation Metrics

1) Training Accuracy

Accuracy represents the proportion of properly predicted cases in the validation dataset, measuring the model's overall performance on previously unknown data and emphasizing its ability to categorize skin lesions reliably.

$$\text{Training Accuracy} = \frac{TP+TN}{N}$$

where TP denotes true positives, TN denotes true negatives, and N is the number of samples in the training dataset

2) Training Loss

Training loss indicates the difference between expected and actual values during the training phase. This helps to understand how well the model fits the training data and guides tweaks to improve the architecture's learning process.

$$\text{Training Loss} = \frac{1}{N} \sum_{i=1}^N \text{CrossEntropy}(\hat{Y}_i, Y_i)$$

where N is the number of samples in the dataset, \hat{Y} denotes the predicted labels, and Y denotes the true labels.

3) Validation Accuracy

Validation accuracy represents the proportion of properly predicted cases in the validation dataset, providing a measure of the model's overall performance on previously unknown data and emphasizing its ability to reliably categorize skin lesions [28].

$$Validation\ Accuracy = \frac{TP+TN}{N}$$

where TP and TN denote true positives and negatives, and N is the number of samples in the validation set.

4) Validation Loss

Validation loss gauges the model's generalization power by calculating the difference between predicted and actual values on unseen validation data, offering insights into whether the model is overfitting or underfitting.

$$Validation\ Loss = \frac{1}{N} \sum_{i=1}^N CrossEntropy(\hat{Y}_i, Y_i)$$

where N is the number of samples in the validation set, \hat{Y} denotes the predicted labels and Y denotes the true labels.

E. Classification

The binary classification system intends to accurately classify skin lesions as benign or seborrheic keratosis. A robust dermatological decision support system was created employing the cutting-edge deep learning architectures VGG16, ResNet50, and InceptionV3. Each model's performance was thoroughly evaluated through training on the ISIC2017 dataset and using rigorous assessment criteria, such as validation loss, training loss, validation accuracy, and training accuracy. This binary classification has the potential to diagnose skin cancer and improve early identification, resulting in better patient outcomes and addressing the global health burden that skin cancer presents.

V. EXPERIMENTAL RESULTS

A median filter transforms a picture by replacing each pixel's value with the median value of its neighbors within a predetermined window size. This method effectively eliminates noise while preserving image borders, resulting in smoother textures and increased clarity. Figure 6 demonstrates some randomly chosen images from the dataset, comparing the visuals before and after preprocessing the dataset.

TABLE I. EVALUATION METRICS

Models	Training accuracy	Training loss	Validation accuracy	Validation loss
VGG16	0.751	0.626	0.712	1.043
Resnet50	0.811	0.424	0.756	0.556
InceptionV3	0.800	0.458	0.762	0.528

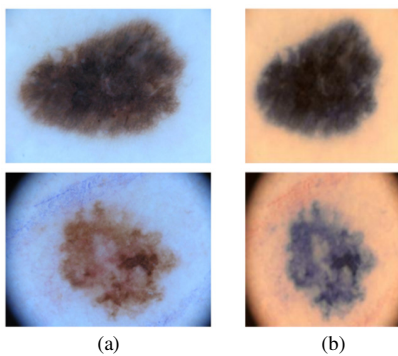


Fig. 6. Visuals of the dataset before and after preprocessing: (a) Raw dataset, (b) After applying the median filter.

Table II portrays the evaluation metric scores obtained by the models. ResNet50 had the highest training accuracy (0.811) of the three models, indicating that it could properly categorize a considerable proportion of training data instances. ResNet50 also had the lowest training loss (0.424), suggesting that the predictions were made with few errors during the training phase. However, in terms of validation measures, InceptionV3 outperformed the other models, with the highest validation accuracy (0.762) and the lowest validation loss (0.528), manifesting stronger generalizability on unseen data. Although VGG16 performed lower than ResNet50 and InceptionV3, its validation measures still show reasonable accuracy and loss values. These results reveal that ResNet50 excels in training performance, whereas InceptionV3 displays higher generalization performance on validation data, thus demonstrating various capabilities across the models.

TABLE II. IMAGE CLASSIFICATION WITH PREDICTION ACCURACIES

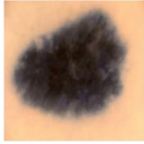
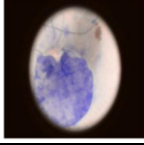
Image	Classification	Accuracy Probability
	Benign	0.9414
	Benign	0.9667

TABLE III. MODEL PARAMETERS

Resnet50 Model		
Layer Name	Output Shape	Param #
conv1.weight	[64, 3, 7, 7]	9408
bn1.weight	[64]	64
bn1.bias	[64]	64
layer1.0.conv1.weight	[64, 64, 1, 1]	4096
layer1.0.bn1.weight	[64]	64
fc.weight	[1000, 2048]	2048000
fc.bias	[1000]	1000
Total Trainable Params: 25557032		
InceptionV3 Base Model		
Conv2d_1a_3x3.conv.weight	[32, 3, 3, 3]	864
Conv2d_1a_3x3.bn.weight	[32]	32
Conv2d_1a_3x3.bn.bias	[32]	32
Conv2d_2a_3x3.conv.weight	[32, 32, 3, 3]	9216
Conv2d_2a_3x3.bn.weight	[32]	32
Total Trainable Params: 27161264		
VGG16 Model		
features.0.weight	[64, 3, 3, 3]	1728
features.0.bias	[64]	64
features.2.weight	[64, 64, 3, 3]	36864
features.2.bias	[64]	64
features.5.weight	[128, 64, 3, 3]	73728
classifier.6.weight	[1000, 4096]	4096000
classifier.6.bias	[1000]	1000
Total Trainable Params: 138357544		

Table III depicts the prediction of the proposed model on images, classifying them into benign or seborrheic keratosis, along with the prediction probability that showcases the

trustability of the model. The probabilities greater than 90% assure that the model performs sufficiently well and can be further integrated into real-life applications of biomedical imaging. Table III presents the parameter list of the three trained models used in the system.

VI. FUTURE SCOPE

The study findings suggest various avenues for future research. Investigating ensemble methods that incorporate the strengths of numerous architectures, such as VGG16, ResNet50, and InceptionV3, has the potential to boost classification performance even further. Furthermore, studying the effects of multiple preprocessing procedures and data augmentation strategies on model performance can improve the resilience of dermatological decision support systems. Broadening the analysis to incorporate more diverse datasets and clinical data, namely patient demographics and lesion characteristics, could provide more detailed insights into skin cancer classification. Explainability techniques to comprehend model predictions and understand the elements that drive categorization judgments will ameliorate clinician adoption and trust in automated diagnosis systems.

VII. CONCLUSION

This study delves into a detailed evaluation of the VGG16, ResNet50, and InceptionV3 architectures for binary categorization of skin lesions as benign or malignant. During experiments on the ISIC 2017 dataset, notable strengths and performance disparities were discovered between models. Although ResNet50 had excellent training accuracy and low training loss, InceptionV3 had better generalization ability, with higher validation accuracy and lower validation loss. VGG16 also performed reasonably across measures. These findings emphasize the necessity for selecting appropriate architectures based on individual requirements, as well as the potential of deep learning models for dermatological diagnostics. The novelty of this study stems from its complete evaluation of deep learning architectures, notably VGG16, ResNet50, and InceptionV3, for the binary classification of skin lesions. The study demonstrates their particular strengths and performance discrepancies on the ISIC 2017 dataset, emphasizing the significance of selecting optimal architectures depending on specific requirements.

REFERENCES

- [1] U. B. Ansari and T. Sarode, "Skin Cancer Detection Using Image Processing," *International Research Journal of Engineering and Technology*, vol. 4, no. 4, pp. 2875–2882, Apr. 2017.
- [2] B. Cassidy, C. Kendrick, A. Brodzicki, J. Jaworek-Korjakowska, and M. H. Yap, "Analysis of the ISIC image datasets: Usage, benchmarks and recommendations," *Medical Image Analysis*, vol. 75, Jan. 2022, Art. no. 102305, <https://doi.org/10.1016/j.media.2021.102305>.
- [3] A. Yilmaz, M. Kalebasi, Y. Samoylenko, M. E. Guvenilir, and H. Uvet, "Benchmarking of Lightweight Deep Learning Architectures for Skin Cancer Classification using ISIC 2017 Dataset." arXiv, Oct. 23, 2021, <https://doi.org/10.48550/arXiv.2110.12270>.
- [4] T. Imran, A. S. Alghamdi, and M. S. Alkathiri, "Enhanced Skin Cancer Classification using Deep Learning and Nature-based Feature Optimization," *Engineering, Technology & Applied Science Research*, vol. 14, no. 1, pp. 12702–12710, Feb. 2024, <https://doi.org/10.48084/etasr.6604>.
- [5] V. A. Rajendran and S. Shanmugam, "Automated Skin Cancer Detection and Classification using Cat Swarm Optimization with a Deep Learning Model," *Engineering, Technology & Applied Science Research*, vol. 14, no. 1, pp. 12734–12739, Feb. 2024, <https://doi.org/10.48084/etasr.6681>.
- [6] K. Rajeshkumar, C. Ananth, and N. Mohananthini, "Blockchain-Assisted Homomorphic Encryption Approach for Skin Lesion Diagnosis using Optimal Deep Learning Model," *Engineering, Technology & Applied Science Research*, vol. 13, no. 3, pp. 10978–10983, Jun. 2023, <https://doi.org/10.48084/etasr.5594>.
- [7] Y. Jusman, I. M. Firdiantika, D. A. Dharmawan, and K. Purwanto, "Performance of Multi Layer Perceptron and Deep Neural Networks in Skin Cancer Classification," in *2021 IEEE 3rd Global Conference on Life Sciences and Technologies (LifeTech)*, Nara, Japan, Mar. 2021, pp. 534–538, <https://doi.org/10.1109/LifeTech52111.2021.9391876>.
- [8] H. Tabrizchi, S. Parvizpour, and J. Razmara, "An Improved VGG Model for Skin Cancer Detection," *Neural Processing Letters*, vol. 55, no. 4, pp. 3715–3732, Aug. 2023, <https://doi.org/10.1007/s11063-022-10927-1>.
- [9] A. Panthakkan, S. M. Anzar, S. Jamal, and W. Mansoor, "Concatenated Xception-ResNet50 — A novel hybrid approach for accurate skin cancer prediction," *Computers in Biology and Medicine*, vol. 150, Nov. 2022, Art. no. 106170, <https://doi.org/10.1016/j.combiomed.2022.106170>.
- [10] S. ElGhany, M. Ibraheem, M. Alruwaili, and M. Elmogy, "Diagnosis of Various Skin Cancer Lesions Based on Fine-Tuned ResNet50 Deep Network," *Computers, Materials & Continua*, vol. 68, no. 1, pp. 117–135, 2021, <https://doi.org/10.32604/cmc.2021.016102>.
- [11] A. Mehra, A. Bhati, A. Kumar, and R. Malhotra, "Skin Cancer Classification Through Transfer Learning Using ResNet-50," in *Emerging Technologies in Data Mining and Information Security*, Singapore, 2021, pp. 55–62, https://doi.org/10.1007/978-981-33-4367-2_6.
- [12] T. Cao, "Skin cancer image classification optimization through transfer learning with Tensorflow and InceptionV3," in *International Conference on Mechatronics Engineering and Artificial Intelligence (MEAI 2022)*, Mar. 2023, vol. 12596, pp. 449–458, <https://doi.org/10.1117/12.2672699>.
- [13] C. P. Manju, J. P. Jeslin, and V. Vinitha, "Computer-Aided Detection of Skin Cancer Detection from Lesion Images Via Deep Learning Techniques: 3d CNN Integrated Inception V3 Networks," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 11, no. 9s, pp. 550–562, Jul. 2023.
- [14] D. Gutman *et al.*, "Skin Lesion Analysis toward Melanoma Detection: A Challenge at the International Symposium on Biomedical Imaging (ISBI) 2016, hosted by the International Skin Imaging Collaboration (ISIC)." arXiv, May 04, 2016, <https://doi.org/10.48550/arXiv.1605.01397>.
- [15] A. Naeem, M. S. Farooq, A. Khelifi, and A. Abid, "Malignant Melanoma Classification Using Deep Learning: Datasets, Performance Measurements, Challenges and Opportunities," *IEEE Access*, vol. 8, pp. 110575–110597, 2020, <https://doi.org/10.1109/ACCESS.2020.3001507>.
- [16] U. Leiter, U. Keim, and C. Garbe, "Epidemiology of Skin Cancer: Update 2019," in *Sunlight, Vitamin D and Skin Cancer*, J. Reichrath, Ed. Cham, Switzerland: Springer International Publishing, 2020, pp. 123–139.
- [17] A. Bommert, X. Sun, B. Bischl, J. Rahnenführer, and M. Lang, "Benchmark for filter methods for feature selection in high-dimensional classification data," *Computational Statistics & Data Analysis*, vol. 143, Mar. 2020, Art. no. 106839, <https://doi.org/10.1016/j.csda.2019.106839>.
- [18] J. Astola and P. Kuosmanen, *Fundamentals of Nonlinear Digital Filtering*. Boca Raton, FL, USA: CRC Press, 2020.
- [19] A. Selveia, V. N. Prakash, N. Saravanan, B. Jawahar, and V. Karthick, "Skin Lesion Detection Using Feature Extraction Approach," *Annals of the Romanian Society for Cell Biology*, pp. 3939–3951, Apr. 2021.
- [20] V. Anand, S. Gupta, A. Altameem, S. R. Nayak, R. C. Poonia, and A. K. J. Saudagar, "An Enhanced Transfer Learning Based Classification for Diagnosis of Skin Cancer," *Diagnostics*, vol. 12, no. 7, Jul. 2022, Art. no. 1628, <https://doi.org/10.3390/diagnostics12071628>.

- [21] H. Qassim, A. Verma, and D. Feinzimer, "Compressed residual-VGG16 CNN model for big data places image recognition," in *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, NV, USA, Jan. 2018, pp. 169–175, <https://doi.org/10.1109/CCWC.2018.8301729>.
- [22] T. Emar, H. M. Afify, F. H. Ismail, and A. E. Hassanien, "A Modified Inception-v4 for Imbalanced Skin Cancer Classification Dataset," in *2019 14th International Conference on Computer Engineering and Systems (ICCES)*, Cairo, Egypt, Dec. 2019, pp. 28–33, <https://doi.org/10.1109/ICCES48960.2019.9068110>.
- [23] S. Likhitha and R. Baskar, "Skin Cancer Segmentation Using R-CNN Comparing with Inception V3 for Better Accuracy," in *2022 11th International Conference on System Modeling & Advancement in Research Trends (SMART)*, Moradabad, India, Dec. 2022, pp. 1293–1297, <https://doi.org/10.1109/SMART55829.2022.10047686>.
- [24] Md. A. Mamun, Md. S. Kabir, M. Akter, and M. S. Uddin, "Recognition of human skin diseases using inception-V3 with transfer learning," *International Journal of Information Technology*, vol. 14, no. 6, pp. 3145–3154, Oct. 2022, <https://doi.org/10.1007/s41870-022-01050-4>.
- [25] O. R. Devi, S. Aarathi, and O. Sirisha, "An Efficient Approach for classification of skin cancer lesions using Inception V3," *NeuroQuantology*, vol. 20, no. 10, pp. 7361–7371, Aug. 2022.
- [26] M. Attique Khan, M. Sharif, T. Akram, S. Kadry, and C.-H. Hsu, "A two-stream deep neural network-based intelligent system for complex skin cancer types classification," *International Journal of Intelligent Systems*, vol. 37, no. 12, pp. 10621–10649, 2022, <https://doi.org/10.1002/int.22691>.
- [27] A. Lembhe, P. Motarwar, R. Patil, and S. Elias, "Enhancement in Skin Cancer Detection using Image Super Resolution and Convolutional Neural Network," *Procedia Computer Science*, vol. 218, pp. 164–173, Jan. 2023, <https://doi.org/10.1016/j.procs.2022.12.412>.
- [28] E. D. Pulakos, "A comparison of rater training programs: Error training and accuracy training," *Journal of Applied Psychology*, vol. 69, no. 4, pp. 581–588, 1984, <https://doi.org/10.1037/0021-9010.69.4.581>.