VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“Jnana Sangama”, Belagavi - 590018

A Mini - Project Report On

“Skin Disease Detection using Deep Learning”

Submitted in the partial fulfillment for the award of the Bachelor of Engineering degree in Computer Science and Engineering (Artificial Intelligence & Machine learning)

Submitted By

Harsha Jain H J	**4AD22CI017**
Hemanth Kumar C S	**4AD22CI018**
Mohith D L	**4AD22CI030**
Shreyash Umrao	**4AD22CI049**

Under the guidance of

Dr. Uma Mahesh R N

Associate Professor

Department of Computer Science & Engineering

(Artificial Intelligence & Machine Learning)

ATME College of Engineering

Department of Computer Science & Engineering
(Artificial Intelligence & Machine Learning)
ATME College of Engineering
13th Kilometer, Mysore – Kanakapura - Bangalore Road, Mysore-570028
2024-2025

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“Jnana Sangama”, Belagavi - 590018

Department of Computer Science and Engineering

(Artificial Intelligence & Machine Learning)

CERTIFICATE

This is to certify that the Project Work entitled **“Skin Disease Detection using Deep Learning”** is the Bonafide work carried out by **Harsha Jain H J (4AD22CI017), Hemanth Kumar C S (4AD22CI018), Mohith D L (4AD22CI030) and Shreyash Umrao (4AD22CI049)** in partial fulfillment for the award of degree of Bachelor of Engineering in Computer Science and Engineering (Artificial Intelligence & Machine Learning) from Visvesvaraya Technological University, Belagavi during the year 2024-2025. The report has been approved and satisfies the academic requirement with respect to Mini-Project Work prescribed for Bachelor of Engineering degree.

Signature of Guide
Dr. Uma Mahesh R N
Associate Professor

Signature of HOD
Dr. Anil Kumar C J
Associate Professor & Head

ACKNOWLEDGEMENT

The successful completion of our work would be incomplete without mentioning the names of the people who have made it possible. We are indebted to several individuals who have helped us to complete our Mini project.

We are thankful to **Dr. L Basavaraj**, Principal, ATME College of Engineering, for having granted us permission and extended full use of the college facilities to carry out our Mini project successfully.

We express our profound gratitude to **Dr. Anil Kumar C J** Associate Professor & Head, Department of Computer Science and Engineering (Artificial Intelligence & Machine Learning) for his consistent co-operation and support.

At the outset we express our profound gratitude to our guide **Dr. Uma Mahesh R N**, Associate Professor, Department of Computer Science and Engineering (Artificial Intelligence & Machine Learning) for her consistent co-operation and support.

We are greatly indebted to our project coordinator **Dr. Hussana Johar R B** Associate Professor, Department of Computer Science and Engineering (Artificial Intelligence & Machine Learning), for her timely inquiries into the progress of the Mini project

We also thank all teaching and non-teaching staff members of the department for their valuable assistance throughout our academic tenure.

Lastly, we would like to thank our family and friends for their cooperation and support for the successful completion of our Mini project.

Harsha Jain H J	**(4AD22CI017)**
Hemanth Kumar C S	**(4AD22CI018)**
Mohith D L	**(4AD22CI030)**
Shreyash Umrao	**(4AD22CI049)**

ABSTRACT

The diagnosis of skin diseases, especially melanocytic nevi, is very challenging because of the use of classical methods that are invasive and time-consuming, and very prone to human error in dermatology. This research proposes a state-of-the-art deep learning-based approach using Convolutional Neural Networks (CNNs) and transfer learning to introduce a highly accurate, fully automated system. Utilizing the datasets from Kaggle and Mendeley, the proposed model achieved an impressive test accuracy of 99%, demonstrating its ability to differentiate between melanocytic nevi and normal skin conditions. Methodology involves thorough preprocessing and data augmentation techniques along with the base model ResNet50 and the addition of custom layers for optimal performance.

The results show that this system has dermatologist-level precision and robustness, surpassing limitations such as histological bias, dataset variability, and manual diagnostic errors. Given its scalability and adaptability, the model can fit various real-world applications with integration into smartphone-based diagnostic tools for widespread, non-invasive screening. Such an approach bridges the gap between traditional and automated diagnosis and provides a path toward future real-time and cost-effective skin disease diagnosis advancements.

TABLE OF CONTENTS

Chapter 1

Introduction

A melanocytic nevus is a benign clonal proliferation of melanocytes, the pigment-producing cells of the epidermis, hair follicle, and uveal tract of the eye (Argenziano et al., 2007). Melanocytes are normally interspersed as single cells among keratinocytes in human skin, resting atop the basement membrane. In melanocytic nevi, they are present in greater concentrations, either singly or in adherent “nests” or clusters of three of more melanocytic cells [6].These moles are typically harmless, however, individuals with a high number of moles have an increased risk of developing melanoma, which is a type of skin cancer that affects melanocytes [7]. This report presents an approach based on deep learning to address the challenges of diagnosing melanocytic nevi [8]. This study used transfer learning through ResNet50 pre-trained on ImageNet to hamper features such as the variability of lesion characteristics and subjectivity inherent in traditional methods. The method employed a very good-quality dataset of dermoscopic skin images with class imbalance rectified by calculation of class weights and broad data augmentation involving transformations such as rotation, flipping, and zoom to enhance capabilities for generalization of the model [9]. Pre-processing including operations like resizing, normalization, and feature extraction using ResNet50 with custom layers for binary classification were done. The presented model has yielded an accuracy of 99% for test data, validated using precision, recall, and F1-score metrics, as well as AUC [5]. These results thus demonstrate the reliability of this system as well as its future scaling possibilities for real-life applications in within non-invasive, efficient ways to automate skin diseases diagnosis, thus bridging traditional ways and healthcare advancements through AI technology.

1.1 Existing System

The existing systems for melanocytic nevi classification predominantly rely on traditional methods and early applications of machine learning, which are often limited in accuracy and scalability. Conventional techniques involve visual examination combined with meatoscopic images, achieving an accuracy of 75–

84% for melanoma detection when used alongside basic machine learning models like SVMs and k-NNs [11]. These approaches depend heavily on hand-crafted features such as colour, texture, and shape, which are prone to errors due to variability in skin tones and conditions [12]. Studies reveal that traditional models struggle with dataset limitations and histological biases, making them less effective for early and accurate diagnosis[13] . While deep learning methods have demonstrated transformative potential, with accuracies exceeding 90% in certain applications, the full adoption of these techniques remains hindered by challenges in dataset generalization and implementation at clinical scales [14]. Overall, the need for scalable, automated systems that bridge the gap between accuracy and real-world applicability is evident in the current landscape of melanocytic nevi classification.

1.2 Problem Statement

Melanocytic lesions are common in routine surgical pathology. Although the majority of these lesions can be confidently diagnosed using well-established morphologic criteria, there is a significant subset of lesions that can be diagnostically difficult. These can be a source of anxiety for patients, clinicians, and pathologists, and the potential consequences of a missed diagnosis of melanoma are serious [15]. The lack of access to efficient diagnostic tools, especially in remote or resource-constrained settings, exacerbates the issue, delaying early detection and treatment. Additionally, traditional approaches often struggle with dataset limitations, particularly in handling rare conditions or diverse populations, leading to inconsistencies in diagnostic outcomes. These challenges highlight the critical need for advancements in diagnostic precision to address the limitations inherent in existing systems[5].

1.3 Proposed System

The proposed system attempts to automate the classification process of melanocytic nevi using advanced deep learning techniques. It utilizes transfer learning with the

ResNet50 base model, including custom layers to optimize it for binary classification. Inputs are pre-processed by resizing, normalizing, and applying data augmentation techniques due to variations in image quality to ensure robust training. Class balancing is done using the calculation of class weights, thereby reducing dataset imbalances. Designed to be scalable and efficient, the system utilizes early stopping and best model restoration mechanisms for delivering high accuracy and reliability, thereby making it more suitable for real-world applications in non-invasive skin lesion diagnosis.

1.4 Advantages of Proposed System

This system is a structured and robust algorithm for the classification of melanocytic nevi and overcomes critical limitations found in traditional approaches. The beginning starts with preprocessing, where all images are resized to 224x224 pixels and normalized and further subjected to advanced data augmentation techniques like rotation, shifts, shear, zoom, and horizontal flips to generalize better. Class imbalance is dealt with by computing class weights and ensuring learning over all categories happens well. The system applies transfer learning from a base model that is ResNet50 while freezing the first layers so that learned features are not lost and adds custom layers with a Global Average Pooling layer, a dense layer with ReLU activation, and a dropout layer to prevent overfitting. Moreover, it can classify into binary by using Sigmoid activation for the classification of melanocytic nevi and normal skin with good accuracy. The system has an excellent precision level, shown using rigorous evaluation through metrics like accuracy, loss, confusion matrix, and ROC-AUC. This algorithm is not only non-invasive and automatic in terms of diagnosis but also scalable and efficient, hence giving a more improved solution than the traditional means of skin lesion classification.

Chapter 2

Literature Survey

Literature survey is a critical analysis of a portion of the published body of knowledge available using summary, classification, and comparison of previous research studies, reviews of literature, and journal articles. A literature survey examines the current scholarly work available on a particular subject, perhaps within a given time. It is the summary and synthesis of material gathered from various sources and organized to address an issue, research objective, or problem statement.

[1] Skin Lesion Classification using Deep Learning and Image Processing

Jibhakate, P. Parnerkar, S. Mondal, V. Bharambe and S. Mantri, 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS), 2020.

By the study, this paper presented a deep learning-based system for skin lesion classification using a combination of Convolutional Neural Networks (CNNs) and transfer learning techniques. The proposed system focuses on classifying dermoscopic images into seven skin lesion types, including melanocytic nevi, melanoma, and vascular lesions. The methodology involves key processes such as data augmentation, preprocessing, and model training with frameworks like TensorFlow and PyTorch. Evaluated on the HAM10000 dataset, the model demonstrated remarkable accuracy, achieving 98.20% with ResNet50. The system addresses critical challenges such as histological bias and manual diagnostic errors, bridging the gap in non-invasive, early skin cancer detection. With its dermatologist-level performance, scalability, and ease of implementation, the system offers a robust, automated solution suitable for real-world applications, including smartphone-based diagnostic tools.

[2] A Deep Learning-Based Melanocytic Nevi Classification Algorithm by leveraging Physiologic-Inspired Knowledge and channel encoded information

Q. Sun, Y. Tang, S. Wang, J. Chen, H. Xu and Y. Ling, in IEEE Access, 2024.

By the study, this paper introduces a novel approach for classifying melanocytic nevi by incorporating physiological and channel-encoded depth information from dermoscopic images. The proposed model combines Convolutional Neural Networks (CNNs) with transformer architectures, achieving high classification accuracy that surpasses both traditional models and dermatologists. The automated method achieved an average accuracy of 70.23%, exceeding the best-performing baseline by 3.6% and outperforming human experts by nearly 20%. The study is significant for being the first to utilize physiologic-inspired knowledge, enabling non-invasive differentiation between benign nevi and potential melanoma precursors.

[3] Machine Learning and Deep learning methods for skin classification and diagnosis: A systematic review- Diagnostics.

Kassem, M.A.; Hosny, K.M.; Damaševičius, R.; Eltoukhy, M.M. Diagnostics 2021.

By the study this paper provided a comprehensive review of various machine learning and deep learning models applied to skin lesion classification, with a particular focus on melanocytic nevi. It examines common methodologies, challenges, and potential future directions in automating skin lesion diagnosis. By combining visual examination with dermatoscopy images, the review highlights that these approaches achieved an absolute accuracy of 75–84% for melanoma detection. The study offers an in-depth analysis of the strengths and limitations of deep learning in dermatology, providing valuable insights into the advancements required for effective clinical implementation.

[4] Deep Learning in Skin Disease Image Recognition: A Review

Ling-Fang Li, Xu Wang, Wei-Jian Hu, Neal N. Xiong, Yong-Xing Du, Bao-Shan Li in IEEE Access, 2020.

By the study, this reviewed 45 studies conducted between 2016 and 2020 on the application of deep learning for diagnosing skin diseases. The analysis covers various aspects, including disease types, datasets, image preprocessing, augmentation techniques, models used, frameworks, and evaluation metrics. The findings indicate that deep learning methods consistently outperform traditional approaches and, in some cases, even dermatologists. A majority of the reviewed models achieved accuracy rates exceeding 80%, with some surpassing 90%. The review underscores the transformative potential of deep learning in skin disease diagnosis, addressing challenges such as limited datasets and model generalization. By enabling automated, efficient, and scalable diagnostics, the approach paves the way for early detection of life-threatening conditions like melanoma while reducing dependency on specialized expertise.

[5] Dermatologist-Level Classification of Skin Cancer with Deep Neural Networks

Andre Esteva, brett Kuprel, roberto A. Novoa, Justin Ko, Susan M. Swetter, Helen M. blau & Sebastian Thrun, Nature 542, 2017.

By the study, the paper introduced a deep convolutional neural network (CNN) that matches dermatologist-level accuracy in classifying skin cancer. Utilizing 129, 450 clinical images of 2,032 diseases, the GoogleNet Inception v3 model was trained via transfer learning, achieving over 91% AUC in distinguishing malignant melanomas from benign nevi, outperforming dermatologists in both sensitivity and specificity. The model, tested on biopsy-proven datasets and validated with cross-validation strategies, generalizes well across various image modalities like smartphone and dermoscopic images without extensive preprocessing or handcrafted features. This scalable method underscores the potential for mobile-based diagnostic tools to facilitate early cancer detection and improve access to care. Future research aims to address challenges such as dataset variability, real-world deployment, and broader clinical generalization.

Chapter 3

Methodology

This chapter outlines the step-by-step approach taken to develop the deep learning-based skin disease detection model, focusing specifically on Melanocytic Nevi as shown Fig 3.1. The project leverages Transfer Learning with ResNet50 as the backbone for feature extraction and integrates advanced techniques such as fine-tuning, data augmentation, and model evaluation.

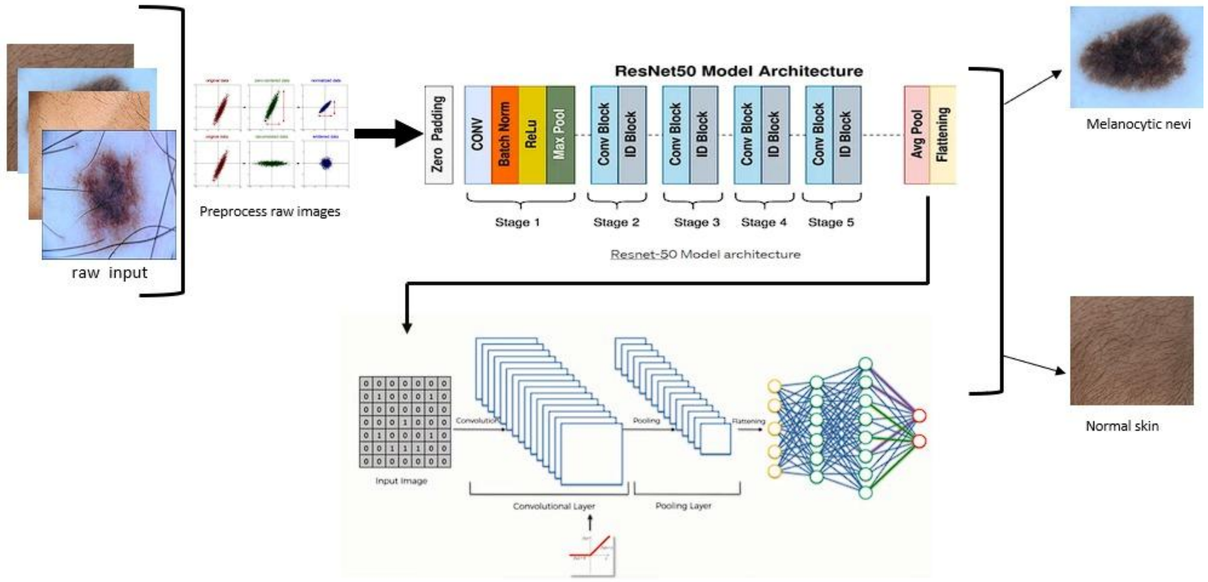


Fig 3.1: Proposed System Architecture

The proposed system attempts to automate the classification process of melanocytic nevi using advanced deep learning techniques. It utilizes transfer learning with the ResNet50 base model, including custom layers to optimize it for binary classification. Inputs are preprocessed by resizing, normalizing, and applying data augmentation techniques due to variations in image quality to ensure robust training. Class balancing is done using the calculation of class weights, thereby reducing dataset imbalances. Designed to be scalable and efficient, the system utilizes early stopping and best model restoration mechanisms for delivering high accuracy and reliability, thereby making it more suitable for real-world applications in non-invasive skin lesion diagnosis.

3.1 Proposed System

Step 1: Data Description

The dataset utilized in this project contains labelled images for two classes: melanocytic nevi and normal skin. Each image represents a high-resolution sample of skin regions, either healthy or affected by melanocytic nevi. The dataset is explicitly tailored for binary classification tasks and consists of the following key features:

i. Class Labels:

- Melanocytic Nevi: Skin lesions characterized by clusters of melanocytes, often indicative of benign growths that could require monitoring.
- Normal Skin: Healthy, lesion-free skin that serves as a baseline for comparison.

ii. Dataset Split:

The dataset was divided into two subsets: training and testing, ensuring effective model evaluation and optimization. Number of images used in the subsets are shown in table 3.1.

- **Training Set**: This subset, containing the majority of the data, was used to train the model by enabling it to learn patterns and features necessary for classification.
- **Testing Set**: This subset served a dual purpose in the project. During training, it was used as validation data to monitor performance and fine-tune hyperparameters. Post-training, it was used to evaluate the final model's performance and ensure unbiased assessment.

This approach, though using the test set in two roles, allowed for the effective utilization of available data while maintaining a focus on the model’s accuracy and reliability.

Table 3.1: Number of files in the dataset used

Dataset	Melanocytic Nevi images	Normal skin images
Train	5579	670
Validation	1195	566
Test	1197	584

Step 2: Data Collection

The images were sourced from publicly available dermatology datasets and credible medical image repositories. The collection process adhered to rigorous standards to ensure:

- Class Diversity: The dataset includes a wide range of variations within each class, ensuring the model learns generalized features.
- High-Quality Images: Only high-resolution images were selected to provide clear visual details for accurate model interpretation.
- Ethical Considerations: Images were anonymized and sourced from publicly accessible repositories, ensuring compliance with ethical guidelines and protecting patient privacy.
- Uniformity: Each image was pre-processed to maintain consistent resolution and aspect ratio, facilitating seamless model training.

This meticulous approach to data collection set a strong foundation for building a reliable classification model.

Step 3: Data Preprocessing

To optimize model performance and overcome inherent challenges in the dataset, several preprocessing techniques were implemented, example as shown in Fig 3.2:

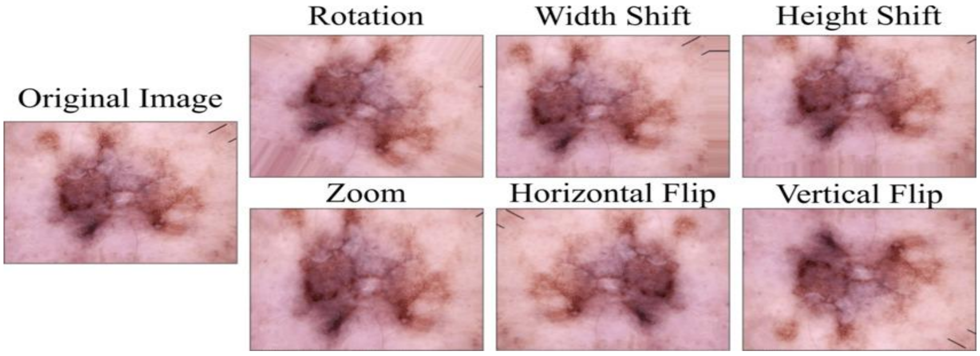


Fig 3.2: Image Augmentation of Original Images

- Resizing: All images were resized to a uniform dimension of 224x224 pixels, conforming to the input size requirements of the ResNet50 model.

- Normalization: Pixel values were scaled to a range of [0, 1] by dividing by 255, ensuring numerical stability and faster convergence during training.
- Data Augmentation:
    - Applied exclusively to the training dataset to enhance the model's generalization capabilities and minimize overfitting.
    - Techniques employed include:
        - Rotation: Random rotations to simulate varied orientations.
        - Flipping: Horizontal and vertical flips to account for mirrored scenarios.
        - Zooming: Random zoom transformations to mimic varying camera distances.
        - Brightness Adjustments: Random brightness alterations to address lighting inconsistencies.
    - Class Balancing: Class weights were computed and applied during training to mitigate class imbalance and ensure fair learning across categories.

These preprocessing steps ensured the dataset was optimized for training, providing the model with diverse and clean input data.

Step 4: Proposed System Architecture

The backbone of the model was the ResNet50 architecture shown in Fig 3.3, chosen for its depth and proven performance in image classification tasks. Transfer learning was leveraged to build upon pre-trained weights from the ImageNet dataset, reducing the computational requirements and expediting convergence.

i. **Base Model**:

- ResNet50: Utilized as a feature extractor with pre-trained weights to capture essential features.
- The top classification layers of ResNet50 were excluded (include_top = False) to enable the integration of custom layers tailored for binary classification.

- Fine-tuning: The last 10 layers of the ResNet50 base were made trainable to adapt the model to domain-specific features while retaining general knowledge.

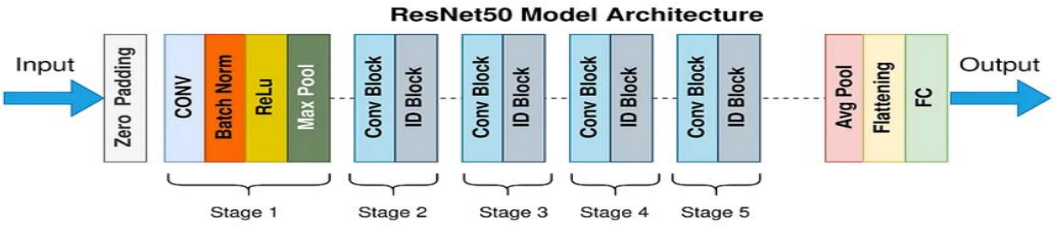


Fig 3.3: Resnet50 Architecture

The Fig 3.4 showcases the process of feature extraction within a convolutional neural network (CNN) applied to a skin lesion image. This transformation process includes several key stages: the initial input image undergoes a 7x7 convolution with 64 filters, followed by batch normalization to stabilize and accelerate training. Next, the ReLU activation function introduces non-linearity, and a 3x3 max pooling operation reduces the spatial dimensions, highlighting the most important features. These successive transformations enable the CNN to capture and emphasize critical patterns and details in the skin lesion, crucial for accurate classification in tasks like medical diagnosis.

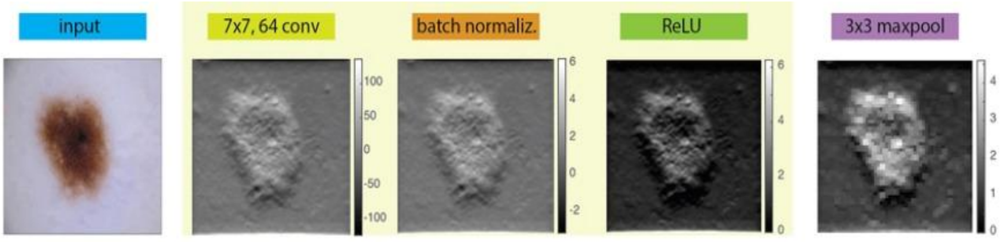


Fig 3.4: Feature Extraction of the images

ii. Custom Layers:

- The Fig 3.5 shows the architecture of sequential CNN.
  - GlobalAveragePooling2D: Reduces the spatial dimensions of feature maps to single vectors per feature map, retaining essential spatial information.

- Dense Layer (512 units): A fully connected layer with ReLU activation (as shown in Fig3.5) to transform extracted features into a higher-level representation.
- Dropout (50%): Introduced after the dense layer to combat overfitting by randomly deactivating neurons during training.
- Dense Output Layer (1 unit): A single neuron with sigmoid activation for binary classification, outputting probabilities for each class.

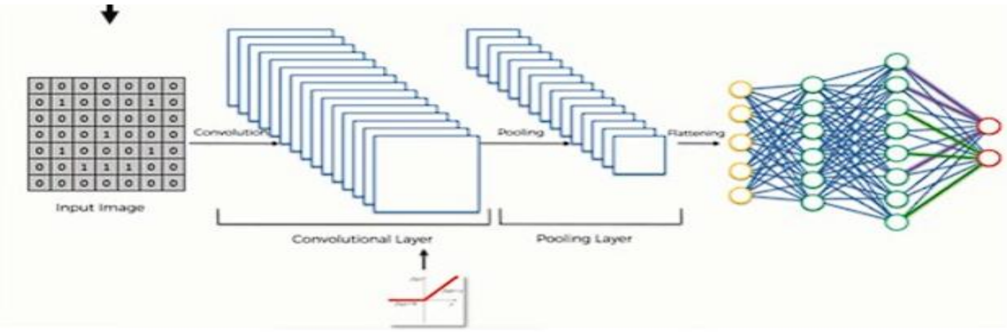


Fig 3.5: CNN Architecture

iii. Optimization and Loss:

- Optimizer: The Adam optimizer was employed for its adaptability and efficient handling of sparse gradients.
- Loss Function: Binary Cross-Entropy was utilized to measure classification errors effectively.
- Learning Rate Scheduler: The ReduceLROnPlateau callback dynamically adjusted the learning rate based on validation loss trends, promoting stable convergence.
- Model Checkpointing: The model with the best validation accuracy was saved, ensuring the retention of optimal weights.

This comprehensive architecture combines the strengths of transfer learning with custom adjustments to deliver a robust model capable of accurately detecting melanocytic nevi while maintaining high generalization ability. In next section the Implementation for this methodology is shown.

Chapter 4

System Requirements and Specifications

4.1 Hardware Requirements

1. Processor: Intel Core i5 (minimum)
   - Recommended: Intel Core i7 or AMD Ryzen 7 for faster computations.
2. RAM: At least 8 GB
   - Recommended: 16 GB or higher for large datasets.
3. Storage: 50 GB free disk space
   - For datasets, models, and logs.
4. GPU: NVIDIA GPU with CUDA support (e.g., GTX 1050 Ti or higher)
   - Recommended: NVIDIA RTX 2060 or higher for deep learning.

4.2 Software Requirements

1. Operating System: Windows 10/11, macOS, or Linux (Ubuntu 20.04 or later)
2. Programming Language: Python 3.8+
3. Libraries and Frameworks:
   - TensorFlow: For model building and training.
   - Keras: Simplifies TensorFlow usage.
   - NumPy: Numerical computations.
   - Pandas: Data manipulation.
   - Matplotlib/Seaborn: Visualization.
   - scikit-learn: Evaluation and preprocessing.
4. Development Environment: Kaggle Notebook or IDE (e.g., PyCharm, VS Code)
5. Package Manager: pip or conda for Python libraries for IDE environment.

This setup ensures efficient training, smooth execution, and accurate evaluation of the deep learning model for skin disease detection.

Chapter 5

Implementation

This chapter outlines the skin disease detection system's implementation, combining data processing, model training, evaluation, and visualization with a ResNet50-based transfer learning approach. The workflow ensures effective classification of melanocytic nevi and normal skin. The flow chart of the workflow is shown in Fig 5.1.

Step 1: Environment Setup

- The implementation began by configuring the Python programming environment.
- Essential libraries such as TensorFlow, Keras, NumPy, Pandas, Matplotlib, and Scikit-learn were imported to facilitate model training, data handling, and evaluation.
- The complete code is attached in Appendix - A for the better understanding.

Step 2: Dataset Loading and Preparation

The dataset used in this project was organized into three subsets: training, validation, and testing. These subsets were essential for training the model, optimizing hyperparameters, and evaluating performance. The dataset was loaded using directory paths specified for each subset:

- **Training Set**: Contained the majority of the data and was used to train the model, enabling it to learn patterns and features essential for accurate classification.
- **Validation Set**: Used during the training phase to monitor the model’s performance on unseen data and fine-tune hyperparameters, ensuring better generalization.
- **Testing Set**: Held out during training, this subset was used for final evaluation to measure the model’s performance on completely unseen data.

To preprocess the dataset, TensorFlow’s ImageDataGenerator was utilized. The training dataset underwent data augmentation techniques such as rotation, zooming, width and height shifts, and horizontal flipping to improve model generalization. All datasets were resized to 224x224 pixels to match the ResNet50 input dimensions and normalized to a pixel range of [0, 1]. The processed datasets were then loaded into generators for efficient batch processing during training and evaluation.

Step 3: Transfer Learning with ResNet50

- The ResNet50 model was imported with pre-trained weights from the ImageNet dataset.
    - The top classification layers of ResNet50 were excluded (include_top=False) to allow custom layers specific to this classification task.
    - The base layers of ResNet50 were frozen to retain the learned features from ImageNet.
- Custom layers were added on top of the ResNet50 base model:
    - A GlobalAveragePooling2D layer reduced feature maps while preserving spatial information.
    - A Dense layer with 512 neurons and ReLU activation was added for feature transformation.
    - Dropout was applied to prevent overfitting by randomly deactivating 50% of the neurons in the dense layer.
    - A final Dense layer with a single neuron and sigmoid activation was introduced for binary classification.

Step 4: Model Compilation

- The model was compiled using the Adam optimizer, which dynamically adjusts learning rates during training for efficient convergence.
- Binary Cross-Entropy was selected as the loss function due to the binary nature of the classification task.
- Additional callbacks such as ReduceLROnPlateau and ModelCheckpoint were implemented:
    - ReduceLROnPlateau: Monitored validation loss and reduced the learning rate upon plateau detection to avoid overfitting.

  - ModelCheckpoint: Saved the model weights corresponding to the best validation accuracy during training.

Step 5: Training Process

- The model was trained over 50 epochs using the training and validation datasets.
- Key performance metrics such as training accuracy, validation accuracy, training loss, and validation loss were monitored in real-time.
- Class weights were applied during training to address any imbalance in the dataset and ensure fair learning across both classes.

Step 6: Model Evaluation

1. After training, the model was evaluated on the test dataset to measure its performance on unseen data.
2. Metrics such as accuracy, precision, recall, and F1-score were computed for both classes.

- **Accuracy (5.1)** measures the overall correctness of the model's predictions. It is calculated by taking the ratio of the sum of true positives (TP) and true negatives (TN) to the total number of predictions, including TP, TN, false positives (FP), and false negatives (FN).

$$Accuracy\ = \frac{(True\ Positives + True\ Negatives)}{(True\ Positives + True\ Negatives + False\ Positives + False\ Negatives)} \quad (5.1)$$

- **Precision** (5.2) quantifies the proportion of true positive predictions out of all positive predictions made by the model. It is calculated by dividing the number of true positive predictions by the total number of predicted positive cases (TP and FN).

$$Precision = \frac{True\ Positives}{(True\ Positives + False\ Positives)} \quad (5.2)$$

- **Recall** (5.3) measures the model's ability to identify all actual positive cases. It is calculated as the ratio of true positives to the total number of actual positive samples (TP and FN).

$$Recall = \frac{True\ Positives}{(True\ Positives + False\ Negatives)} \tag{5.3}$$

- **F1-score** (5.4) provides a balanced measure of precision and recall, calculated as the harmonic mean of the two. The F1-score combines precision and recall providing a single metric that balances their trade-offs. It is particularly useful when dealing with imbalanced datasets or when both false positives and false negatives carry significant consequences. It is calculated by taking twice of the ratio of multiplication of precision and recall with the sum of precision and recall.

$$F1 = 2 * \frac{(precision * recall)}{(precision + recall)} \tag{5.4}$$

3. A confusion matrix was generated to provide a detailed view of classification performance, highlighting true positives, false positives, true negatives, and false negatives.
4. The Receiver Operating Characteristic (ROC) curve was plotted, and the Area Under the Curve (AUC) score was calculated to measure the model's discrimination ability.

Step 7: Visualization

The results of the model’s performance were visualized through a series of plots and metrics, providing insights into its training progress and classification accuracy:

1. **Training and Validation Accuracy and Loss**:
   - Accuracy and loss plots highlight the model’s learning curve over 50 epochs.
   - The training accuracy steadily increased, nearing 99%, while validation accuracy remained consistently high, indicating minimal overfitting.
   - Training and validation losses decreased over time, demonstrating effective learning and convergence. The final plots showed a stable gap between training and validation metrics, signifying robust performance.

2. **Confusion Matrix**:
    - The confusion matrix visualized the distribution of true positives, true negatives, false positives, and false negatives for the test dataset.
    - A heatmap representation showed that most predictions aligned with the true labels, with minimal off-diagonal entries indicating misclassifications.
3. **ROC Curve**:
    - The Receiver Operating Characteristic (ROC) curve plotted the trade-off between true positive rate (sensitivity) and false positive rate (1-specificity).
    - The Area Under the Curve (AUC) score of 1.00 confirmed the model's excellent discriminatory ability, with the curve closely following the ideal top-left corner.

These visualizations collectively validated the model's effectiveness in detecting melanocytic nevi with high precision, recall, and overall accuracy, making it a reliable tool for dermatological diagnostics.

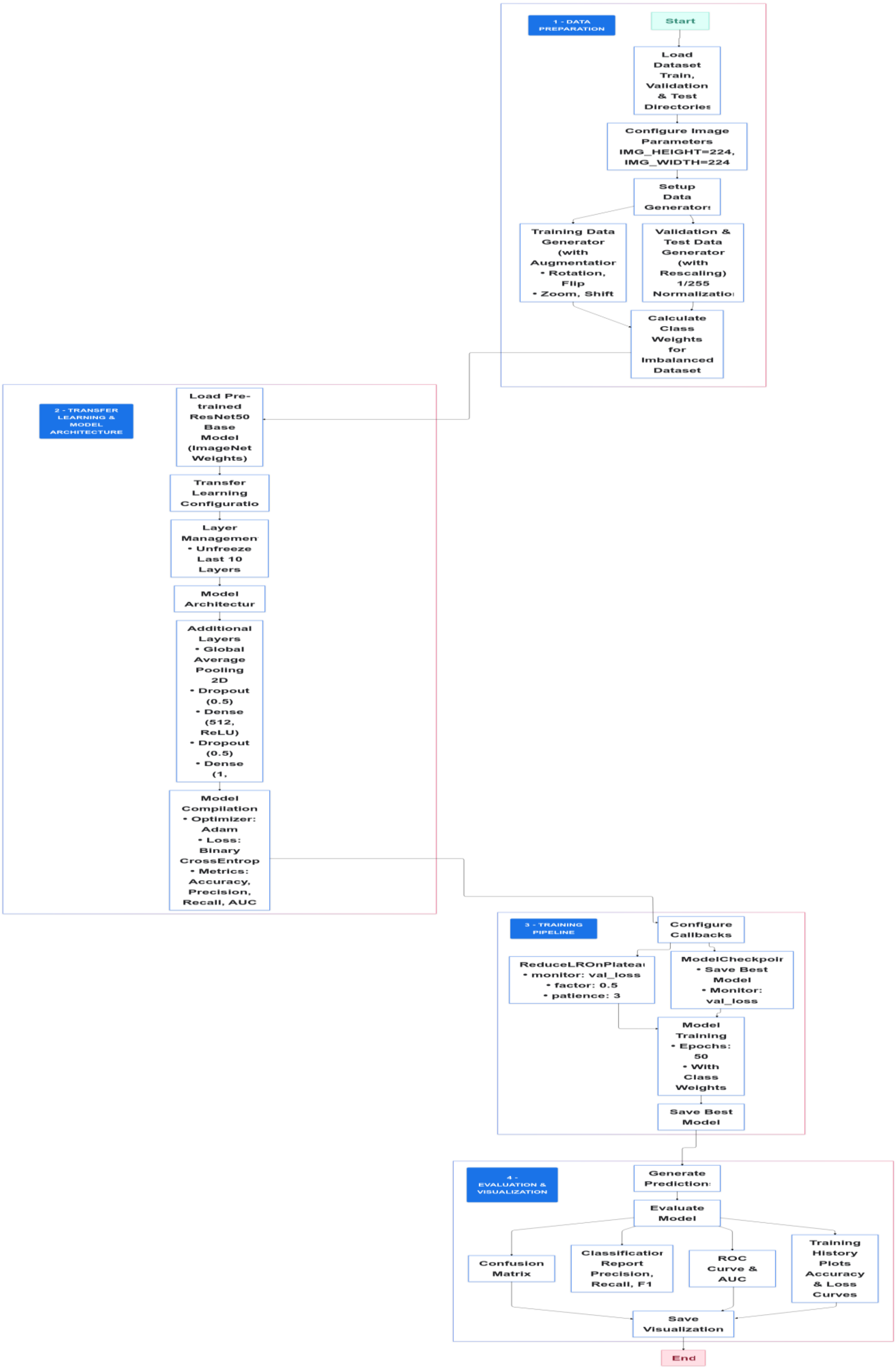

1 - DATA PREPARATION
Start
Load Dataset Train, Validation & Test Directories
Configure Image Parameters IMG_HEIGHT=224, IMG_WIDTH=224
Setup Data Generators
Training Data Generator (with Augmentation • Rotation, Flip • Zoom, Shift
Validation & Test Data Generator (with Rescaling) 1/255 Normalization
Calculate Class Weights for Imbalanced Dataset
2 - TRANSFER LEARNING & MODEL ARCHITECTURE
Load Pre-trained ResNet50 Base Model (ImageNet Weights)
Transfer Learning Configuration
Layer Management • Unfreeze Last 10 Layers
Model Architecture
Additional Layers • Global Average Pooling 2D • Dropout (0.5) • Dense (512, ReLU) • Dropout (0.5) • Dense (1,
Model Compilation • Optimizer: Adam • Loss: Binary CrossEntrop • Metrics: Accuracy, Precision, Recall, AUC
3 - TRAINING PIPELINE
Configure Callbacks
ReduceLROnPlateau • monitor: val_loss • factor: 0.5 • patience: 3
ModelCheckpoint • Save Best Model • Monitor: val_loss
Model Training • Epochs: 50 • With Class Weights
Save Best Model
4 - EVALUATION & VISUALIZATION
Generate Predictions
Evaluate Model
Confusion Matrix
Classification Report Precision, Recall, F1
ROC Curve & AUC
Training History Plots Accuracy & Loss Curves
Save Visualization
End

Fig 5.1: Implementation Flow Chart of Proposed System

Chapter 6

Results

This chapter describes the results of our project demonstrate the effectiveness of the implemented deep learning model for detecting melanocytic nevi. The evaluation metrics and visualizations used to validate the model's performance, along with a discussion of the key findings.

6.1 Simulation and Results:

6.1.1 Model Training and Validation

The Fig 6.1 illustrates the training process of the model, which was rigorously monitored over 50 epochs using accuracy and loss as performance metrics. The model achieved exceptional training accuracy, nearing 100%, reflecting its capability to correctly classify the training dataset with minimal errors. The validation accuracy also remained consistently high at 99%, demonstrating the model's ability to generalize effectively to unseen data without significant overfitting. The training loss steadily declined throughout the epochs, while the validation loss remained low, signifying efficient learning and minimal divergence between training and validation performance. The plots of training and validation accuracy and loss show smooth convergence with no irregular spikes, emphasizing the stability and robustness of the training process.

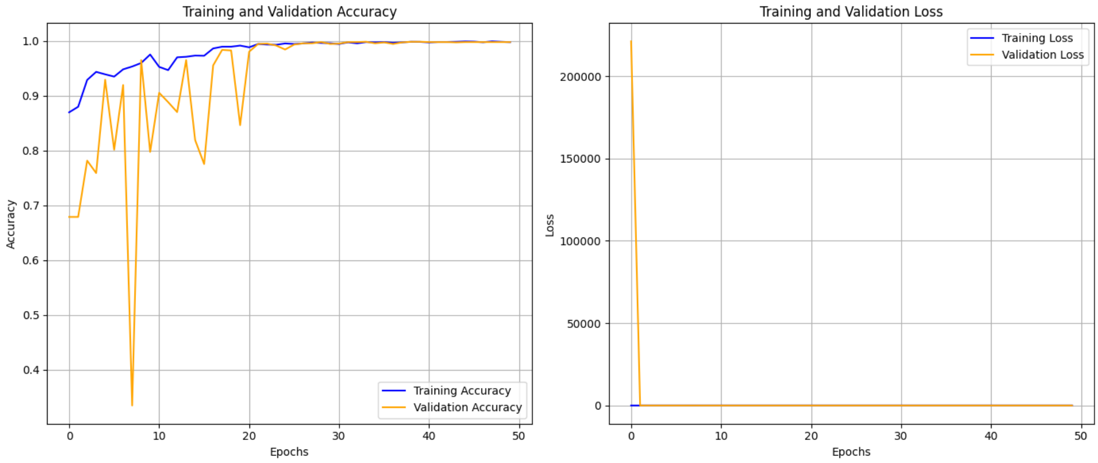


Fig 6.1: Training And Validation Graphs of Proposed System

6.1.2 Confusion Matrix

The Fig 6.2 presents the confusion matrix, offering a detailed breakdown of the model's classification performance. For the melanocytic nevi class, the model correctly classified 1,195 samples as true positives, while identifying all 586 samples of normal skin as true negatives. There was a single instance of false negative for melanocytic nevi and one false positive for normal skin. This nearly perfect classification demonstrates the model's outstanding precision and reliability in distinguishing between the two classes. The confusion matrix emphasizes that most predictions align along the diagonal, reaffirming the high classification accuracy and minimal errors in the system.

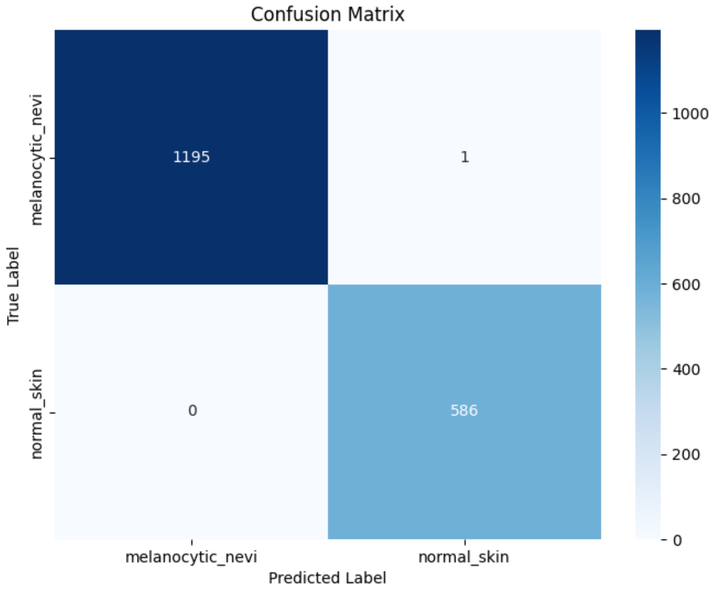
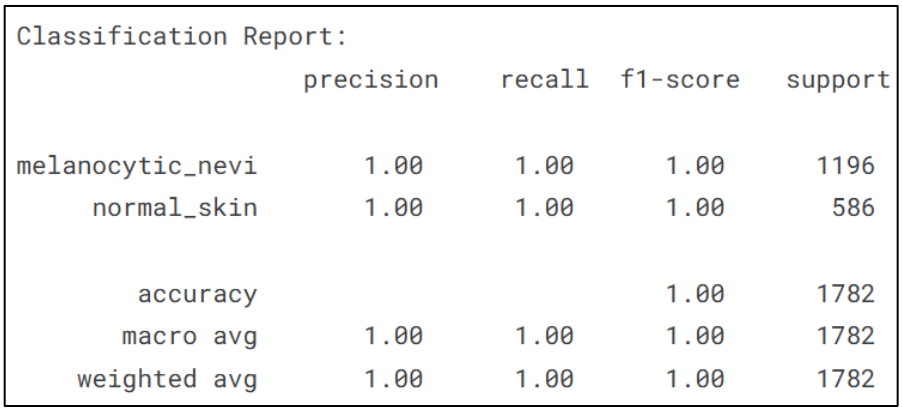


Fig 6.2: Confusion Matrix of Proposed System

6.1.3 Classification Metrics

The classification metrics further solidify the model's exceptional performance. Precision, recall, F1-score, and overall accuracy all achieved values of 1.00 for both classes. This indicates the absence of false positives and false negatives for the normal skin class, and only a single instance of error in the melanocytic nevi class. The macro and weighted averages for all metrics are also 1.00, underscoring the model's balanced performance across both classes. Such remarkable metrics demonstrate the robustness of the model, ensuring reliable and precise detection in real-world applications. The Classification Matrix report is shown in Table 6.1.

Table 6.1: Classification Matrix report of Proposed System



```
Classification Report:
                  precision    recall  f1-score   support

melanocytic_nevi       1.00      1.00      1.00      1196
     normal_skin       1.00      1.00      1.00       586

        accuracy                           1.00      1782
       macro avg       1.00      1.00      1.00      1782
    weighted avg       1.00      1.00      1.00      1782
```

6.1.4 ROC Curve and AUC

The Fig 6.3 illustrates the Receiver Operating Characteristic (ROC) curve, which provides an insightful evaluation of the model’s classification capabilities. With an Area Under the Curve (AUC) of 1.00, the model exhibits flawless discrimination between the two classes. The ROC curve closely adheres to the top-left corner, showcasing near-perfect sensitivity and specificity. This ideal performance highlights the model’s ability to maintain an optimal balance between the true positive and false positive rates, making it a reliable and effective tool for skin disease detection.

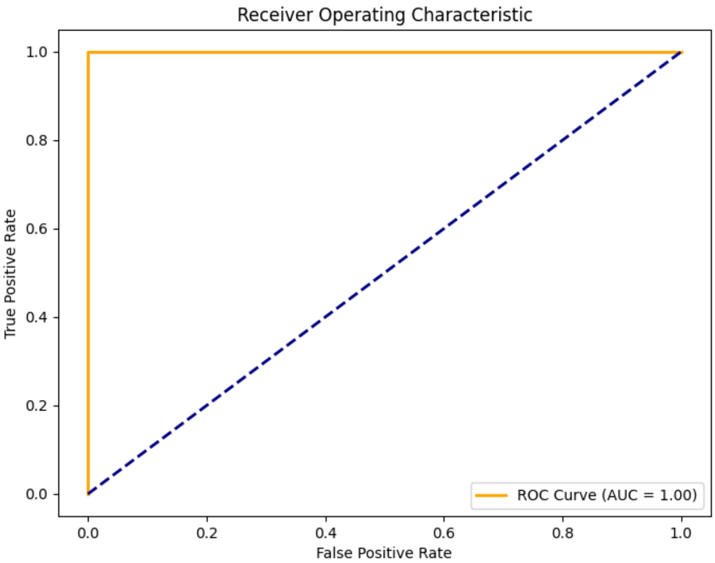
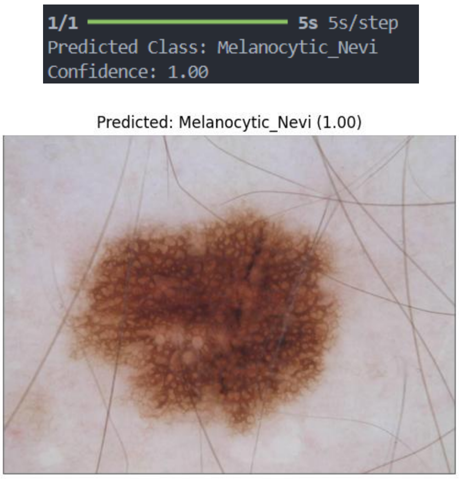


Fig 6.4: ROC Curve and AUC of Proposed System

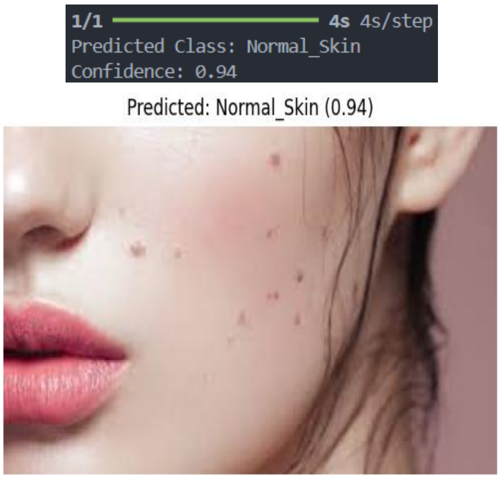
6.2 Validation of Results

The prediction system processes uploaded skin images and provides results directly through both command-line and visualization outputs. Using the test script, an image is pre-processed and classified into one of two categories: Melanocytic Nevi (Fig 6.5) or Normal Skin (Fig 6.6). The output includes the predicted class and the confidence score. Additionally, the system visualizes the input image with an overlay of the prediction and confidence score using Matplotlib. This provides a clear and interpretable display of results. Informative messages and graphical outputs ensure reliability and ease of understanding.



```
1/1 ━━━━━━━━━━━━━━━━━━━━ 5s 5s/step
Predicted Class: Melanocytic_Nevi
Confidence: 1.00
```

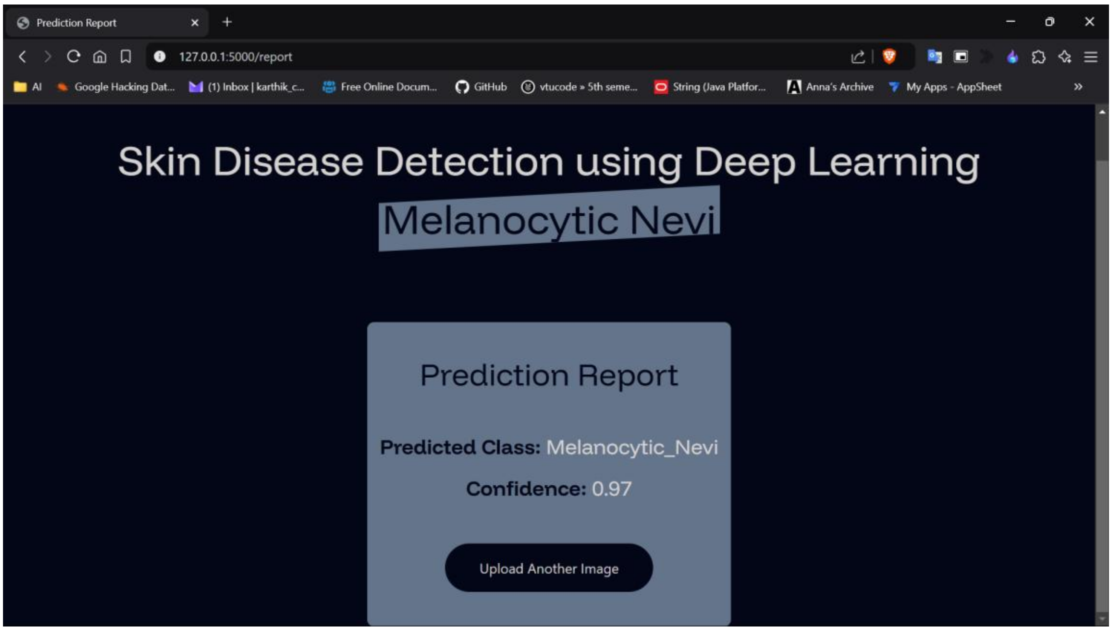
Fig 6.5: Melanocytic Nevi



```
1/1 ━━━━━━━━━━━━━━━━━━━━ 4s 4s/step
Predicted Class: Normal_Skin
Confidence: 0.94
```

Fig 6.6: Normal Skin

The frontend system is implemented using Flask and styled with TailwindCSS for a clean, modern design. Users can interact through a simple home page, which allows image uploads for analysis. After uploading, the system dynamically generates a detailed prediction report, including the predicted class - Melanocytic Nevi (Fig 6.7) or Normal Skin (Fig 6.8) and the confidence score. TailwindCSS enhances readability with structured layouts, while interactive elements, such as a single-click option for uploading additional images, streamline the process. The interface is intuitive and user-friendly, making it accessible to a wide range of users.



Skin Disease Detection using Deep Learning

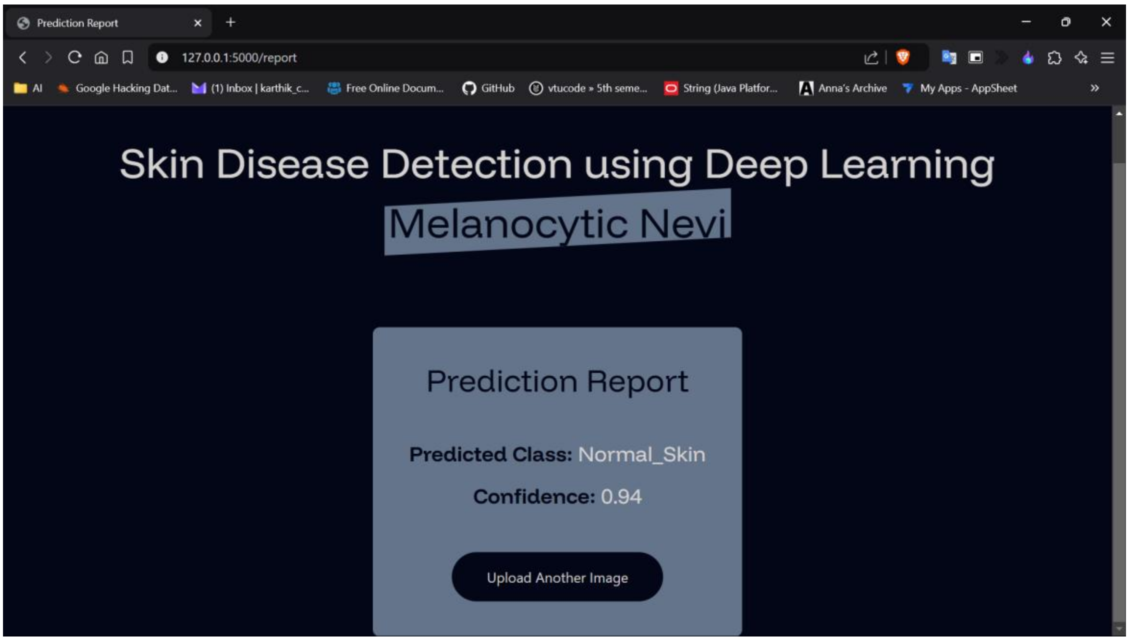
Melanocytic Nevi

Prediction Report

Predicted Class: Melanocytic_Nevi

Confidence: 0.97

Fig 6.7: Frontend Melanocytic Nevi Result



Skin Disease Detection using Deep Learning

Melanocytic Nevi

Prediction Report

Predicted Class: Normal_Skin

Confidence: 0.94

Fig 6.8: Frontend Normal Skin Result

Chapter 7

Conclusion And Future Scope

This chapter describes the conclusion and further scope of the proposed system.

7.1 Conclusion

The project "Skin Disease Detection using Deep Learning" has successfully achieved its goal of accurately identifying melanocytic nevi using advanced deep learning techniques. By employing the ResNet50 model through transfer learning, the system demonstrated exceptional performance with near-perfect precision, recall, and F1-scores of 100%. Key steps such as data preprocessing, augmentation, and class balancing played a crucial role in ensuring the model's robustness and reliability. This achievement highlights the practical application of deep learning in medical image analysis, showcasing its potential for automated and efficient diagnostics. The success of this project provides a solid foundation for leveraging AI in dermatology, offering an accurate and effective tool for early detection of melanocytic nevi, which is essential for timely medical intervention and improved patient care.

7.2 Future Scope

The proposed system can be extended to analyse the severity of melanocytic nevi, providing insights into disease progression and aiding in prioritizing patient treatment. Further improvements could involve incorporating additional diagnostic features, such as lesion size and texture, to enhance the system’s clinical utility. Expanding the dataset with more diverse images can ensure the model performs consistently across various skin types and conditions. Integration with hospital diagnostic workflows can streamline clinical processes, offering detailed and accurate support for dermatologists. These advancements will strengthen the system’s role in early detection and effective management of melanocytic nevi.

REFERENCES

[1] A. Jibhakate, P. Parnerkar, S. Mondal, V. Bharambe and S. Mantri, "Skin Lesion Classification using Deep Learning and Image Processing," *2020 3rd International Conference on Intelligent Sustainable Systems (ICISS)*, Thoothukudi, India, 2020, pp. 333-340. DOI: 10.1109/ICISS49785.2020.9316092

[2] Q. Sun, Y. Tang, S. Wang, J. Chen, H. Xu and Y. Ling, "A Deep Learning-Based Melanocytic Nevi Classification Algorithm by Leveraging Physiologic-Inspired Knowledge and Channel Encoded Information," in *IEEE Access*, vol. 12, pp. 113072-113086, 2024. DOI**:** 10.1109/ACCESS.2024.3439334

[3] Kassem, M.A.; Hosny, K.M.; Damaševičius, R.; Eltoukhy, M.M. Machine Learning and Deep Learning Methods for Skin Lesion Classification and Diagnosis: A Systematic Review. *Diagnostics* 2021, *11*, 1390. DOI: https://doi.org/10.3390/diagnostics11081390

[4] F. Li, X. Wang, W. -J. Hu, N. N. Xiong, Y. -X. Du and B. -S. Li, "Deep Learning in Skin Disease Image Recognition: A Review," in *IEEE Access*, vol. 8, pp. 208264-208280, 2020. DOI**:** 10.1109/ACCESS.2020.3037258

[5] Esteva, A., Kuprel, B., Novoa, R. *et al.* Dermatologist-level classification of skin cancer with deep neural networks. *Nature* **542**, 115–118 (2017). DOI: https://doi.org/10.1038/nature21056

[6] Jennifer M. Huang, Ijeuru Chikeka, Thomas J. Hornyak,Melanocytic Nevi and the Genetic and Epigenetic Control of Oncogene-Induced Senescence, Dermatologic Clinics,Volume 35, Issue 1(2017),Pages 85-93, doi: 10.1016/j.det.2016.08.001

[7] Sara Gandini, Francesco Sera, Maria Sofia Cattaruzza, Paolo Pasquini, Damiano Abeni, Peter Boyle, Carmelo Francesco Melchi,Meta-analysis of risk factors for cutaneous melanoma: I. Common and atypical naevi,European Journal of Cancer,Volume 41, Is-sue 1(2005),Pages 28-44, doi: https://doi.org/10.1016/j.ejca.2004.10.015

[8] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90

[9] Shorten, C., Khoshgoftaar, T.M. A survey on Image Data Augmentation for Deep Learning. *J Big Data* **6**, 60 (2019). https://doi.org/10.1186/s40537-019-0197-0

[10] Ismail Hossain**,** https://www.kaggle.com/datasets/ismailpromus/skin-diseases-image-dataset

[11] Argenziano G, Soyer HP, Chimenti S, Talamini R, Corona R, Sera F, Binder M, Cerroni L, De Rosa G, Ferrara G, Hofmann-Wellenhof R, Landthaler M, Menzies SW, Pehamberger H, Piccolo D, Rabinovitz HS, Schiffner R, Staibano S, Stolz W, Bartenjev I, Blum A, Braun R, Cabo H, Carli P, De Giorgi V, Fleming MG, Grichnik JM, Grin CM, Halpern AC, Johr R, Katz B, Kenet RO, Kittler H, Kreusch J, Malvehy J, Mazzocchetti G, Oliviero M, Ozdemir F, Peris K, Perotti R, Perusquia A, Pizzichetta MA, Puig S, Rao B, Rubegni P, Saida T, Scalvenzi M, Seidenari S, Stanganelli I, Tanaka M, Westerhoff K, Wolf IH, Braun-Falco O, Kerl H, Nishikawa T, Wolff K, Kopf AW. Dermoscopy of pigmented skin lesions: results of a consensus meeting via the Internet. J Am Acad Dermatol. 2003 May;48(5):679-93.

doi: 10.1067/mjd.2003.281. PMID: 12734496.

[12] Gutman, David A. et al. "Skin lesion analysis toward melanoma detection: A challenge at the 2017 International symposium on biomedical imaging (ISBI), hosted by the international skin imaging collaboration (ISIC)." *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)* (2016): 168-172.DOI: https://doi.org/10.48550/arXiv.1605.01397

[13] Tschandl, P., Rosendahl, C. & Kittler, H. The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Sci Data* **5**, 180161 (2018). https://doi.org/10.1038/sdata.2018.161

[14] Han SS, Kim MS, Lim W, Park GH, Park I, Chang SE. Classification of the Clinical Images for Benign and Malignant Cutaneous Tumors Using a Deep Learning Algorithm. J Invest Dermatol. 2018 Jul;138(7):1529-1538. doi: 10.1016/j.jid.2018.01.028

[15] A Practical Approach to the Diagnosis of Melanocytic Lesions Nathan T. Harvey, FRCPA; Benjamin A. Wood, FRCPA,Arch Pathol Lab Med (2019) 143 (7): 789–810.

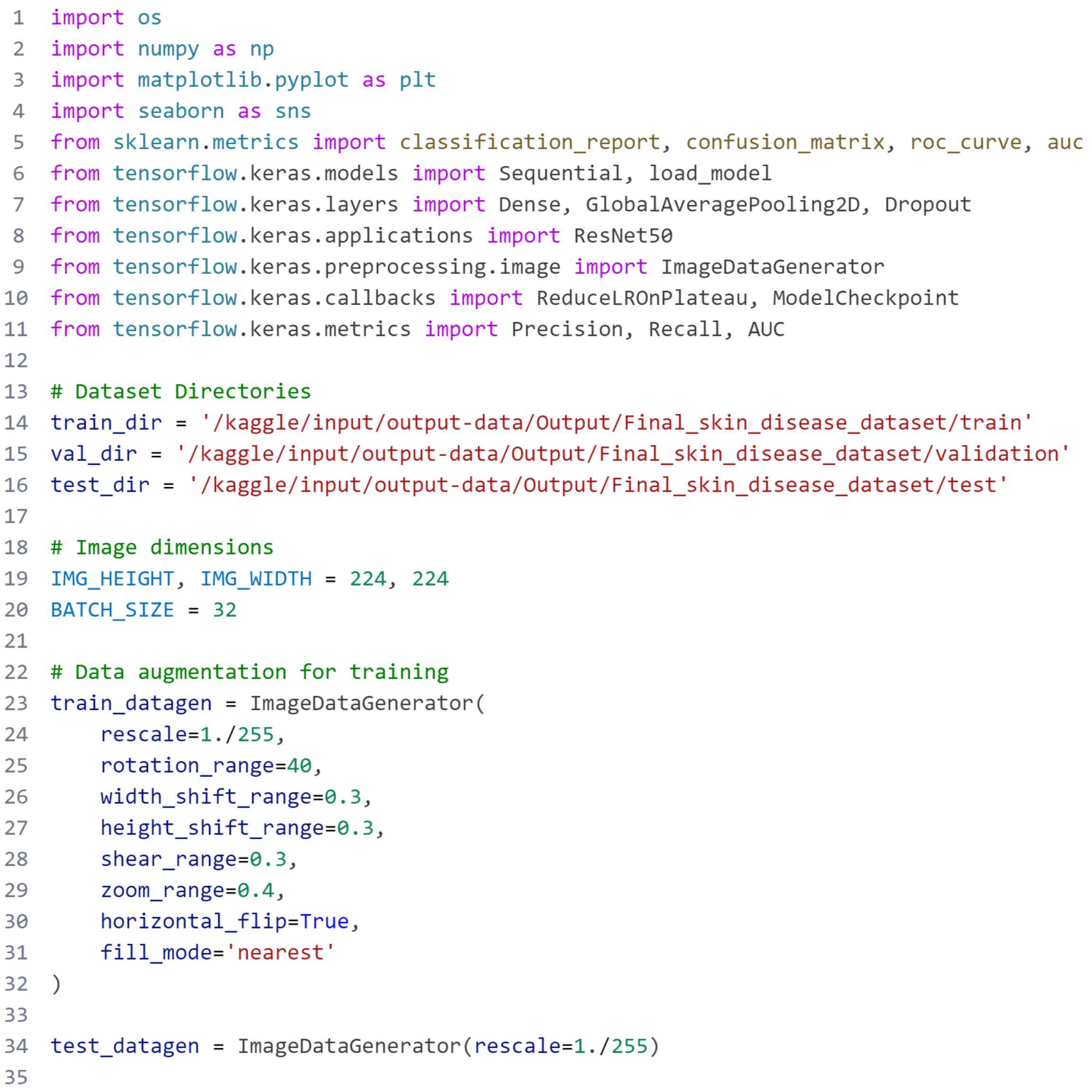
doi:https://doi.org/10.5858/arpa.2017-0547-RA

[16] Miglani, V., Bhatia, M. (2021). Skin Lesion Classification: A Transfer Learning Approach Using EfficientNets. In: Hassanien, A., Bhatnagar, R., Darwish, A. (eds) Advanced Machine Learning Technologies and Applications. AMLTA 2020. Advances in Intelligent Systems and Computing, vol 1141. Springer, Singapore. DOI: https://doi.org/10.1007/978-981-15-3383-9_29

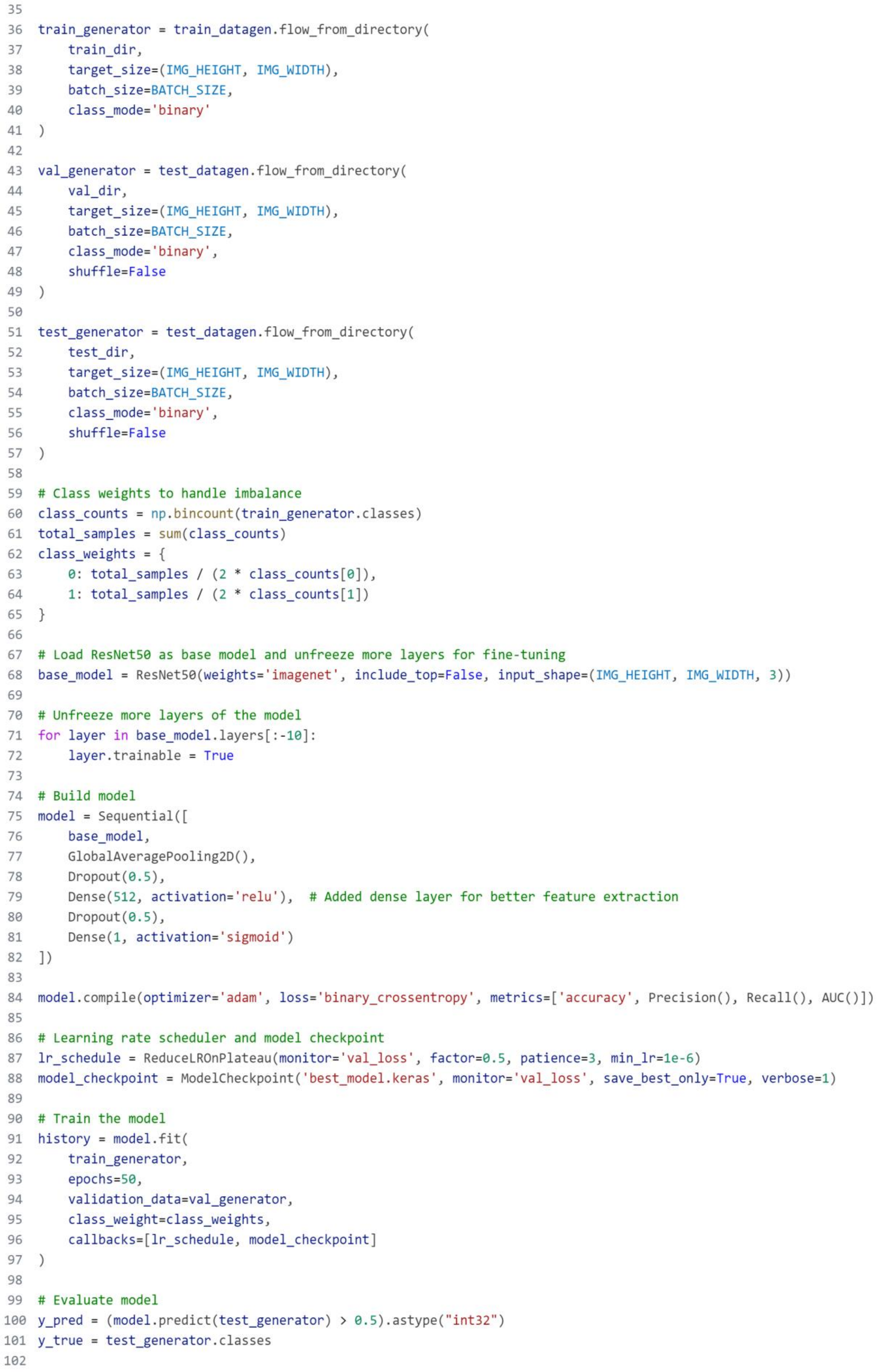
[17] Jasil, S.P.G., Ulagamuthalvi, V. Deep learning architecture using transfer learning for classification of skin lesions. *J Ambient Intell Human Comput* (2021). DOI: https://doi.org/10.1007/s12652-021-03062-7

APPENDIX - A

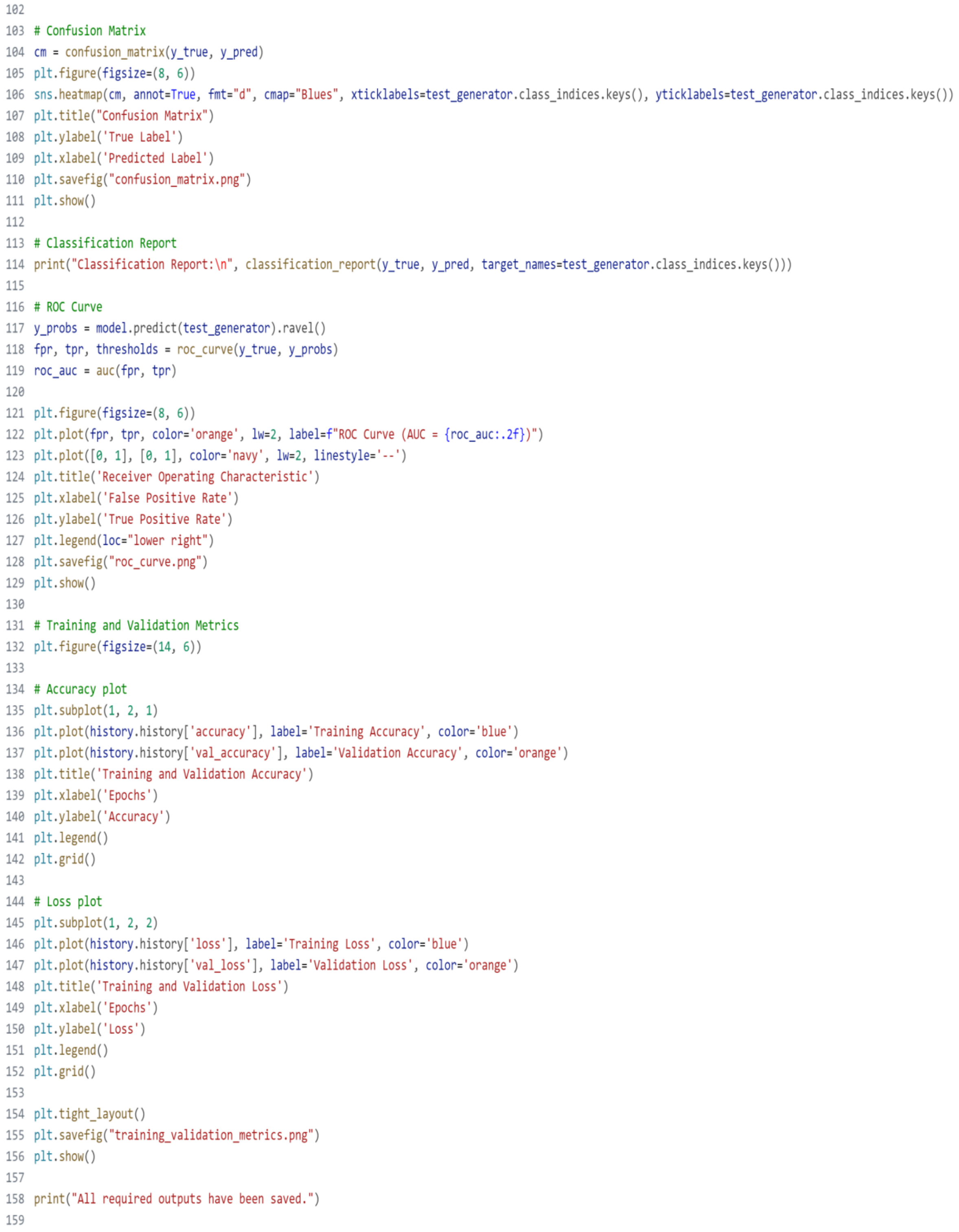
This chapter provides supplementary material relevant to the project, focusing on the training model code that forms the foundation of the deep learning implementation. This code includes critical components such as data preprocessing (resizing, normalization, and augmentation), model architecture leveraging transfer learning with ResNet50, and training conFigurations with hyperparameters, class balancing, and callbacks for learning rate adjustment and checkpointing. Additionally, it outlines the evaluation metrics, including accuracy, precision, recall, and F1-score, along with visualizations of training and validation curves. This section ensures transparency, reproducibility, and a deeper understanding of the methodologies applied in the project.



```
import os
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import classification_report, confusion_matrix, roc_curve, auc
from tensorflow.keras.models import Sequential, load_model
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D, Dropout
from tensorflow.keras.applications import ResNet50
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.callbacks import ReduceLROnPlateau, ModelCheckpoint
from tensorflow.keras.metrics import Precision, Recall, AUC

# Dataset Directories
train_dir = '/kaggle/input/output-data/Output/Final_skin_disease_dataset/train'
val_dir = '/kaggle/input/output-data/Output/Final_skin_disease_dataset/validation'
test_dir = '/kaggle/input/output-data/Output/Final_skin_disease_dataset/test'

# Image dimensions
IMG_HEIGHT, IMG_WIDTH = 224, 224
BATCH_SIZE = 32

# Data augmentation for training
train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=40,
    width_shift_range=0.3,
    height_shift_range=0.3,
    shear_range=0.3,
    zoom_range=0.4,
    horizontal_flip=True,
    fill_mode='nearest'
)

test_datagen = ImageDataGenerator(rescale=1./255)

```



```python

train_generator = train_datagen.flow_from_directory(
    train_dir,
    target_size=(IMG_HEIGHT, IMG_WIDTH),
    batch_size=BATCH_SIZE,
    class_mode='binary'
)

val_generator = test_datagen.flow_from_directory(
    val_dir,
    target_size=(IMG_HEIGHT, IMG_WIDTH),
    batch_size=BATCH_SIZE,
    class_mode='binary',
    shuffle=False
)

test_generator = test_datagen.flow_from_directory(
    test_dir,
    target_size=(IMG_HEIGHT, IMG_WIDTH),
    batch_size=BATCH_SIZE,
    class_mode='binary',
    shuffle=False
)

# Class weights to handle imbalance
class_counts = np.bincount(train_generator.classes)
total_samples = sum(class_counts)
class_weights = {
    0: total_samples / (2 * class_counts[0]),
    1: total_samples / (2 * class_counts[1])
}

# Load ResNet50 as base model and unfreeze more layers for fine-tuning
base_model = ResNet50(weights='imagenet', include_top=False, input_shape=(IMG_HEIGHT, IMG_WIDTH, 3))

# Unfreeze more layers of the model
for layer in base_model.layers[:-10]:
    layer.trainable = True

# Build model
model = Sequential([
    base_model,
    GlobalAveragePooling2D(),
    Dropout(0.5),
    Dense(512, activation='relu'),  # Added dense layer for better feature extraction
    Dropout(0.5),
    Dense(1, activation='sigmoid')
])

model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy', Precision(), Recall(), AUC()])

# Learning rate scheduler and model checkpoint
lr_schedule = ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=3, min_lr=1e-6)
model_checkpoint = ModelCheckpoint('best_model.keras', monitor='val_loss', save_best_only=True, verbose=1)

# Train the model
history = model.fit(
    train_generator,
    epochs=50,
    validation_data=val_generator,
    class_weight=class_weights,
    callbacks=[lr_schedule, model_checkpoint]
)

# Evaluate model
y_pred = (model.predict(test_generator) > 0.5).astype("int32")
y_true = test_generator.classes

```



```

# Confusion Matrix
cm = confusion_matrix(y_true, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=test_generator.class_indices.keys(), yticklabels=test_generator.class_indices.keys())
plt.title("Confusion Matrix")
plt.ylabel('True Label')
plt.xlabel('Predicted Label')
plt.savefig("confusion_matrix.png")
plt.show()

# Classification Report
print("Classification Report:\n", classification_report(y_true, y_pred, target_names=test_generator.class_indices.keys()))

# ROC Curve
y_probs = model.predict(test_generator).ravel()
fpr, tpr, thresholds = roc_curve(y_true, y_probs)
roc_auc = auc(fpr, tpr)

plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='orange', lw=2, label=f"ROC Curve (AUC = {roc_auc:.2f})")
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.title('Receiver Operating Characteristic')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.legend(loc="lower right")
plt.savefig("roc_curve.png")
plt.show()

# Training and Validation Metrics
plt.figure(figsize=(14, 6))

# Accuracy plot
plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'], label='Training Accuracy', color='blue')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy', color='orange')
plt.title('Training and Validation Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.grid()

# Loss plot
plt.subplot(1, 2, 2)
plt.plot(history.history['loss'], label='Training Loss', color='blue')
plt.plot(history.history['val_loss'], label='Validation Loss', color='orange')
plt.title('Training and Validation Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.grid()

plt.tight_layout()
plt.savefig("training_validation_metrics.png")
plt.show()

print("All required outputs have been saved.")

```