PREMIER UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
A Project Final Report On

Deep learning based automated diagnosis of infectious skin lesions

Team Members:

Sharika Ali Suhi: ID 0222210005101103
Adity Barua: ID 0222410005101081
Asad Ullah Khan: ID 0222210005101104

Course Title:

Neural Network and Fuzzy Logic Laboratory
Course Code: CSE 452

Instructor:

MD Tamim Hossain
Lecturer
Department of Computer Science and Engineeering
Premier University, Chattogram

Date:

November 23, 2025

Deep learning based automated diagnosis of infectious skin lesions

Sharika Ali Suhi[1*], Adity Barua[2] and Asad Ullah Khan[3]

[*]Department of Computer Science and Engineering, Premier University, Chittagong, Bangladesh.

*Corresponding author(s). E-mail(s): sharikasuhi10@gmail.com;

Abstract

Infectious skin lesions are widespread across diverse populations of the world and can be caused by pathogenic microorganisms like viruses, bacteria, fungi, and parasites. These lesions should be detected at an early stage to treat the infected patients as well as to prevent the spread of the disease among the people who come in contact with infected persons. The goal of this project is to develop an automated deep learning-based system for classifying infectious skin lesions into 11 distinct categories using Convolutional Neural Networks (CNNs) and a transfer learning approach.Different studies are explored, and it is found that we can take pre-trained models and fine-tune them to solve various kinds of CNN-based classification problems. The MobileNetV2 model, which was trained on ImageNet, has previously provided outstanding results in such skin lesion detection. In the project, the MobileNetV2 model is taken as a base, and changes are being made to meet the classification goal. The model is trained on a custom dataset sourced from multiple public repositories, including Kaggle and Mendeley. As the dataset that is being combined is imbalanced, and images are of different resolutions, the data preprocessing encountered some challenges. The results of the different approaches are compared using standard performance metrics, including accuracy, precision, recall, and loss, with a focus on minimizing overfitting while maximizing generalization. Techniques like transfer learning and data augmentation can even help when there is not enough data to create better generalizations while training the deep learning-based models. The final model achieved a great accuracy on the validation set with minimal overfitting, showing promising potential for real-world deployment.

Keywords: Transfer learning, Convolutional Neural Network (CNN), Data augmentation

1 Introduction and Problem Statement

Skin disorders, particularly those caused by infectious microorganisms such as viruses, bacteria, fungi, and other pathogens, create a considerable challenge to healthcare systems worldwide. These ailments, which include prevalent infections such as chickenpox and scabies as well as more severe conditions such as monkeypox and cellulitis, can be challenging to diagnose due to the diverse range of symptoms and the similarities in dermatological conditions. A prompt and precise diagnosis is essential for effectively managing these infections, halting their transmission, and guaranteeing that patients obtain the suitable treatment.

Traditionally, the identification of skin lesions depends on the manual examination conducted by healthcare professionals, which typically includes visual evaluation, patient history, and physical assessment. Although dermatologists possess the training to differentiate among various skin conditions, this method is labor-intensive and susceptible to human error. Moreover, the need for dermatological expertise frequently exceeds the supply of qualified professionals, particularly in resource-constrained environments, resulting in postponed diagnoses and less-than-ideal treatment results.

In response to these challenges, the application of deep learning techniques has surfaced as a viable solution for the automation of skin lesion classification. With the progress made in machine learning, especially in Convolutional Neural Networks (CNNs), automated systems are now able to evaluate medical images with remarkable accuracy, thereby aiding healthcare professionals in achieving quicker and more dependable diagnoses. Deep learning models possess the capability to learn intricate patterns and features from extensive datasets of labeled images, enabling them to identify and categorize skin lesions that might be challenging to distinguish visually.

The Infectious Skin Lesion Classification project is dedicated to creating an automated system capable of categorizing skin lesions into 11 unique classifications. This initiative employs MobileNetV2, a compact deep learning model that has been pretrained on ImageNet, to classify skin lesions using image data. A significant challenge faced in this endeavor is the class imbalance present in the dataset, where some categories contain a considerably larger number of samples compared to others. This imbalance can lead to overfitting, causing the model to become biased towards the majority classes and perform inadequately on the minority classes. To mitigate this issue, strategies such as data augmentation, regularization, and fine-tuning are implemented to ensure that the model generalizes effectively across all categories.

Ultimately, the objective of this project is to create a robust and scalable system that can be utilized in resource-constrained environments, where trained dermatologists may be limited. By automating the classification of infectious skin lesions, this system aims to aid healthcare professionals in diagnosing conditions with greater efficiency and accuracy, resulting in earlier detection and enhanced treatment outcomes for patients.

2 Related Work

Recent research has made considerable advancements in the utilization of deep learning methodologies for the detection and classification of skin diseases, demonstrating the capability of automated systems to aid healthcare professionals in the diagnosis of skin lesions.

One significant study conducted by Deng and Zheng (2025) [1] presented HSCFNet, a lightweight Convolutional Neural Network (CNN) specifically developed for the classification of both infectious and non-infectious skin diseases. Their methodology emphasized the efficacy of compact, efficient models, particularly well-suited for mobile and edge computing environments.

Thieme et al. (2023) [2] formulated a deep learning algorithm explicitly designed to classify mpox lesions from medical imaging. Their model exhibited outstanding performance in differentiating between lesions induced by the mpox virus and other dermatological conditions. The application of automated lesion classification has the potential to significantly improve diagnostic processes by delivering rapid, consistent, and precise outcomes.

Another noteworthy contribution is from Ali et al. (2023) [3] who developed a web-based system for the detection of mpox skin lesions utilizing cutting-edge deep learning models. Their investigation highlighted the essential role of racial diversity within the training dataset, ensuring that the model operates effectively across varied populations. By incorporating data from multiple racial and ethnic backgrounds, their model contributed to reducing potential biases, thereby promoting fairer and more equitable predictions across diverse demographic groups.

Rimi et al. (2020) [4] investigated the application of CNNs for the detection of skin diseases, emphasizing their capacity to learn complex patterns and extract detailed features from images of skin lesions. Their study illustrated the effectiveness of deep learning within the medical domain, especially for the identification of skin diseases, by utilizing convolutional layers that autonomously learn hierarchical features from the images.

Furthermore, Burlina et al. (2019) [5] focused on the automated identification of erythema migrans and other overlapping skin lesions through the use of deep learning methods. Their research underscored the capabilities of CNNs in managing overlapping symptoms among skin diseases, a challenge that usually necessitates the expertise of dermatologists.

The Infectious Skin Lesion Classification project builds on these advancements by employing the MobileNetV2 architecture, which provides a lightweight and efficient approach for classifying skin lesions. Additionally, sophisticated techniques such as regularization, batch normalization, and data augmentation are utilized to address issues like class imbalance and overfitting, ultimately enhancing classification performance and generalization.

3 Dataset

3.1 Source

The data for this project was obtained from various repositories, such as Kaggle and Mendeley. These repositories offer a range of datasets for the classification of skin diseases and lesions, which include the following:

- **Mendeley Dataset:**
  - Multi-Class Viral Skin Lesion Dataset (MCVSLD) [6].
- **Kaggle Datasets:**
  - Mpox Skin Lesion Dataset Version 2.0 (MLSD v2.0) [7].
  - Skin diseases image dataset [8].
  - Dermnet [9].

The dataset consists of 11 unique categories of infectious skin lesions, which encompass a range of skin diseases induced by viruses, bacteria, and fungi. Infectious skin diseases' data are chosen from four repositories and then combined to create a more robust version of the skin lesion dataset. The allocation of images within each category is detailed as follows:

- Cellulitis, Impetigo, and other Bacterial Infections: 361 images
- Chickenpox: 1127 images
- Cowpox: 957 images
- Healthy: 1710 images
- Herpes, HPV, and other STDs: 507 images
- HFMD: 2415 images
- Measles: 825 images
- Monkeypox: 4260 images
- Scabies, Lyme Disease, and other Infestations: 539 images
- Tinea (Ringworm), Candidiasis, and other Fungal Infections: 3327 images
- Warts, Molluscum, and other Viral Infections: 2103 images

4 Exploratory Data Analysis (EDA)

4.1 Class Distribution

The dataset exhibited a notable class imbalance, with certain classes, such as Monkeypox and HFMD, having a significantly higher number of images, whereas others, like Herpes, HPV, and various STDs, had a smaller quantity of samples. Specifically:

- Monkeypox comprised 4260 images, rendering it the most prevalent class.
- Herpes, HPV, and other STDs were represented by only 507 images, marking them as the least represented class.

This imbalance has the potential to introduce bias into the model, which may lead to greater accuracy in classifying majority classes while facing challenges with the

underrepresented minority classes. A visualization is created to analyze the distribution of images across classes. Bar charts and pie charts were utilized to emphasize the imbalance, visually substantiating the necessity for strategies to rectify the class distribution problem.

Fig. 1 Class distribution of infectious skin lesion categories (illustrative placeholder).

4.2 Sample Image Inspection

The subsequent phase in the EDA process entailed examining sample images from every class. This phase was crucial for identifying quality concerns, such as low resolution or inconsistent labeling, and a mixture of RGB and grey-scale images, which could impact model performance. Illustrations of images from each class were presented to understand the variability and possible challenges in visual recognition.

5 Preprocessing

5.1 Resizing

Given that the MobileNetV2 model utilized in this project necessitates a fixed input size, all images were resized to 224×224 pixels. This guarantees consistency in the input dimensions, allowing the model to process them effectively without any dimension discrepancies.

5.2 Normalization

To simplify quicker convergence throughout the training process, the pixel values of every image were standardized to the interval [0, 1]. This was accomplished by dividing the pixel values by 255. Normalization serves to mitigate significant fluctuations in the data, which could potentially lead to instability during training and hinder the model's learning efficiency.

5.3 Data Augmentation

In light of the class imbalance and to improve the model's generalization abilities, various data augmentation techniques were implemented on the training dataset. These methods assist in generating variations of the training images, thereby increasing diversity within the dataset and enabling the model to acquire more robust features. The following augmentations were utilized:

- **Rotation**: Randomly rotating images to aid the model in learning rotational invariance.
- **Shifting**: Randomly shifting images in both horizontal and vertical directions to replicate translations of skin lesions.
- **Flipping**: Performing horizontal and vertical flips on images to represent different orientations of lesions.
- **Zooming**: Randomly zooming in and out to mimic varying distances of the lesions from the camera.

5.4 Class Balancing

To further tackle the issue of class imbalance, several additional measures were implemented:

- **Over-sampling**: To improve the representation of underrepresented classes like Herpes, HPV, and other STDs, images were augmented. For example, Cellulitis, Impetigo, and Chickenpox were oversampled using augmentation techniques such as random rotation, flipping, and noise addition, creating additional synthetic samples.

- **Down-sampling**: For classes that were overrepresented (like Monkeypox), the dataset was down-sampled to a mean value that approximated the average class count. Specifically, Monkeypox was reduced from 4260 images to approximately 1153 images.

By employing these augmentation and balancing strategies, the dataset achieved a more equitable distribution, ensuring that the model was exposed to a varied and representative collection of images for training.

6 Methodology

In this section, a comprehensive analysis of the methodology is presented that is employed for the model architecture, the selected hyperparameters, and the fine-tuning procedures executed to tackle the challenges associated with the "Infectious Skin Lesion Classification" task.

6.1 Model Architecture

The core of the model is MobileNetV2, which is a lightweight and efficient convolutional neural network (CNN) architecture pre-trained on ImageNet. MobileNetV2 was selected due to its use of depth-wise separable convolutions, which greatly decrease

computational complexity while preserving high performance. The architecture is composed of multiple residual blocks followed by depth-wise separable convolutions, rendering it appropriate for edge devices and mobile platforms.

Base Model – MobileNetV2:

- **Pre-trained Weights**: The model is initialized with weights from ImageNet, enabling the network to utilize features acquired from a vast and varied image dataset.
- **Base Layers**: The original fully connected layers of MobileNetV2 are eliminated to facilitate the integration of custom classification layers.
- **Fine-Tuning**: In the first approach, the first 20 layers and in the other two approaches, the first 30 layers of the model are kept trainable, and other layers are kept frozen to maintain the learned weights from ImageNet. This approach aids in mitigating overfitting during the early phases of training.

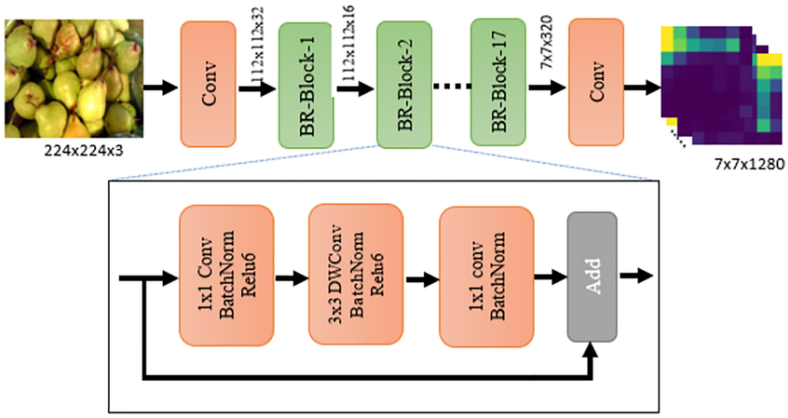

Fig. 2 MobileNetV2 architecture diagram (illustrative placeholder).

6.2 Three Approaches

6.2.1 Approach 1

- **GlobalAveragePooling2D**: Following the passage of feature maps through the MobileNetV2 base model, a Global Average Pooling layer is utilized. This layer condenses the spatial dimensions of the feature maps into a singular vector for each feature map. The benefit of GAP lies in its ability to significantly decrease the number of parameters in comparison to flattening, resulting in a more compact model size and a diminished risk of overfitting.

- **Dense Layer**: A dense layer is implemented after the pooling layer, which has one hidden layer that has 256 nodes and one output layer having 11 nodes to classify

11 classes, incorporating a ReLU activation function and a dropout rate of 0.4 to assist in mitigating overfitting.
- **Output Layer**: The concluding output layer is characterized by a softmax activation layer featuring 11 nodes, corresponding to each class within the classification task.

6.2.2 Approach 2

- **Flatten Layer**: Rather than utilizing GlobalAveragePooling2D, this method employs the Flatten layer to transform the 3D feature maps into a 1D vector. This flattening process preserves all spatial information; however, it results in an increase in the number of parameters within the model.

- **Dense Layers**: Following the flattened output, there is a dense layer that has four hidden layers having 1024 nodes, 512 nodes, 256 nodes, and 128 nodes, respectively. Hidden layers incorporate ReLU activation along with L2 regularization and He weight initialization. Batch normalization and a dropout rate of 0.4 are applied to reduce overfitting.

- **Output Layer**: For multi-class classification, a softmax activation layer is implemented, similar to Approach 1.

6.2.3 Approach 3

- **Flatten Layer**: This approach also employs the Flatten layer to transform the 3D feature maps into a 1D vector, like Approach 2. This flattening process preserves all spatial information to pass on to the dense layer to be trained with.

- **Dense Layers**: Following the flattened output, there is a dense layer that has three hidden layers having 1024 nodes, 512 nodes, and 256 nodes, respectively. Hidden layers incorporate ReLU activation function.

- **Output Layer**: The output layer contains eleven nodes for multi-class classification, and a softmax activation layer is implemented, similar to Approaches 1 and 2.

- **Regularization**: L1 and L2 regularization are incorporated into the dense layers to mitigate overfitting by imposing penalties on large weights. This method aids in stabilizing the model and preventing complexity that may result in overfitting.

- **Batch Normalization**: Implemented following each dense layer, batch normalization contributes to stabilizing the training process by minimizing internal covariate shift, enhancing convergence speed, and rendering the training more resilient.

- **Dropout**: This strategy includes dropout at various stages (with a dropout rate of 0.5) to more effectively address overfitting.

6.3 Hyperparameters

The model performance is greatly influenced by the adjustments of hyperparameters. The key hyperparameters chosen for the training of the model are the following:

- **Learning Rate**: The learning rate was initially set to 1e−4 for the first two methods and 1e−3 for the third method. This hyperparameter is vital; if it is too high, the model may overshoot the optimal solution, whereas a rate that is too low can lead to slow convergence, prolonging the training process. A learning rate scheduler was implemented to dynamically adjust the learning rate throughout the training phase.

- **Optimizer**: The Adam optimizer was selected for its adaptive learning rate characteristics. It employs moment-based updates, which are particularly advantageous for training deep learning models that require a substantial number of parameters to be trained.

- **Batch Size**: A batch size of 32 was chosen for training, as it provides a favorable balance between computation time and memory usage.

- **Dropout Rate**: Dropout serves as a regularization technique aimed at preventing overfitting. A dropout rate of 0.4 was applied following the dense layers in the first two approaches, while a rate of up to 0.5 was utilized in Approach 3 for more robust regularization.

6.4 Fine-Tuning Details

Fine-tuning the MobileNetV2 model involves training the model on a specific dataset after freezing the initial layers. In this project:

- **Freezing Initial Layers**: In CNN, initial layers of the model try to find the primitive features (e.g., edges), and later layers try to make complex features by combining the primitive features detected in initial layers. To take advantage of this, in the first approach, the first 33 layers, and in the other two approaches, the first 23 layers of the MobileNetV2 base model were kept frozen, as these layers contain general features like edges and textures that are transferable across different tasks. As MobileNetV2 was trained on millions of images from ImageNet, the initial trained layers' trained parameters can be used to detect the edges of our very own project. By freezing these layers, computational overhead can be minimized and the risk of overfitting can be mitigated during the initial phases of training.

- **Unfreezing Layers**: In the first approach, only the final 20 layers of the MobileNetV2 model were maintained as trainable, while the last 30 layers were kept trainable in the other two approaches. This strategy facilitates the learning of specific characteristics of skin lesions, as the deeper layers are responsible for capturing more advanced features. By unfreezing these deeper layers, the model

can learn more complex patterns relevant to the classification task.

- **Learning Rate Scheduler**: A tailored learning rate scheduler was implemented to decrease the learning rate by a factor of 0.5 if the validation loss failed to improve after a predetermined number of epochs. This approach facilitates smoother convergence of the model and prevents significant updates during the later phases of training, which may result in instability.

6.5 Comparative Analysis of the Three Approaches (Design)

Table 1 Comparative analysis of architectural and training features of the three approaches.

Feature	**Approach 1**	**Approach 2**	**Approach 3**
Layer for Reducing Dimensions	GlobalAveragePooling2D	Flatten	Flatten
Dropout	0.4	0.4	0.5
Regularization	None	L2 Regularization	L1 & L2 Regularization
Batch Normalization	Not used	After each dense layer	After each dense layer
Fine-tuning Layers	Last 20 layers trainable	Last 30 layers trainable	Last 20 layers trainable
Learning Rate Scheduler	Yes	Yes	Yes
Output Layer	Softmax	Softmax	Softmax
Advantages	Faster training, smaller model size	More spatial information is retained	Best regularization, robustness
Disadvantages	Risk of underfitting due to too few parameters	Larger model, risk of overfitting	Increased computational cost

These three methods illustrate various strategies for organizing the final layers of the MobileNetV2 model aimed at skin lesion classification. The first method (Global Average Pooling) strikes a balance between model size and performance, although it may lose some spatial information. The second method (Flattening) preserves spatial details but raises the complexity of the model, which could result in overfitting. The third method (Custom Dense Architecture with L1 and L2 Regularization and Batch Normalization) employs regularization techniques that assist in mitigating overfitting, yet it may lead to higher computational costs due to an increase in parameters.

7 Training Procedure

In this section, a comprehensive analysis is presented of the training methodologies employed in the three approaches utilized for the "Infectious Skin Lesion Classification" project. The training process encompasses information regarding the loss functions, optimizers, management of random seeds, environmental configurations, and hyperparameters applied to the models.

7.1 Loss Function

The loss function is essential in directing the model towards making optimal predictions throughout the training process. All three methods employed the same loss function for the classification task:

- **Categorical Cross-entropy**: Since this is a multi-class classification problem, categorical cross-entropy is utilized to assess the disparity between the predicted class probabilities and the actual labels, making it suitable for this kind of task.

7.2 Optimizers

The optimizer is responsible for managing the updates to the model's weights throughout the training process. In all three methodologies, the Adam optimizer was utilized, which adjusts the learning rate for each parameter during training. The primary hyperparameter settings across the methodologies are:

- **Adam Optimizer**: Adam was selected due to its widespread use and efficiency in training deep learning models. It employs adaptive moment estimation, making it particularly suitable for this project, especially when dealing with extensive parameter spaces.

- **Learning Rate**:
  - Approach 1: 1e−4 (the default for Adam)
  - Approach 2: 1e−4 (the default for Adam)
  - Approach 3: 1e−3 (a higher initial learning rate to promote quicker convergence)

The learning rate plays a vital role in regulating the convergence speed. An excessively high learning rate may cause the model to overshoot the optimal weights, whereas a rate that is too low can lead to slow learning, often creating a zig-zag pattern in the loss function.

7.3 Learning Rate Scheduling

All three approaches incorporated learning rate schedulers to decrease the learning rate at designated intervals when the model's performance stagnates:

- Approach 1: Employed the `ReduceLROnPlateau` callback to diminish the learning rate by a factor of 0.5 following 3 epochs without enhancement in validation loss.

- Approach 2: Utilized both `ReduceLROnPlateau` and a bespoke learning rate scheduler that reduces the learning rate by a factor of 0.1 after the completion of epoch 10.
- Approach 3: Applied `ReduceLROnPlateau` to decrease the learning rate by 0.5 after 5 epochs of no progress in validation loss. Furthermore, a custom scheduler halved the learning rate after epoch 50.

Learning rate schedulers facilitate faster convergence by adaptively modifying the learning rate in response to the model's performance, thereby preventing both overfitting and slow learning.

7.4 Regularization Techniques

To prevent overfitting, the following regularization methods were implemented across the three approaches:

Dropout

- Approach 1: A dropout rate of 0.4 was applied following the dense layers to randomly deactivate neurons during the training process, thereby mitigating overfitting.
- Approach 2: A dropout rate of 0.4 was utilized after each dense layer.
- Approach 3: A higher dropout rate of 0.5 was employed to enhance regularization, applied after the dense layers.

L2 Regularization

This was applied in Approach 2 and Approach 3, incorporating a penalty (`l2(0.01)`) to prevent large weights by integrating a regularization term into the loss function.

L1–L2 Regularization

This was utilized in Approach 3 (`l1_l2(l1=0.01, l2=0.01)`), which combines both L1 and L2 penalties to further regularize the model.

These methods ensure that the model does not overfit the training data and performs better on unseen data.

7.5 Model Checkpoint and Early Stopping

To avoid overfitting and guarantee optimal training of the model:

- **Model Checkpoint**: This feature saves the model exclusively when there is an improvement in the validation loss, thereby ensuring that the most effective model is preserved.
- **Early Stopping**: This mechanism observes the validation loss, and if there is no improvement over a predetermined number of epochs (5 for Approaches 1 and 2, and 10 for Approach 3), the training is halted prematurely to conserve time and prevent overfitting.

7.6 Training Environment

The training was carried out utilizing the Kaggle platform alongside NVIDIA GPUs to enhance the efficiency of the training process. Kaggle offers a well-optimized setting for executing deep learning models on extensive datasets.

- **Batch Size**: A batch size of 32 was utilized in all methodologies. This size achieves a balance between memory consumption and model efficacy, facilitating efficient gradient updates without overloading the GPU memory.

- **Epochs**: Each methodology trained the model for a maximum of 100 epochs; however, early stopping was implemented to halt training once the model's performance stopped improving.

7.7 Randomization and Data Augmentation

- **Random Seed**: A constant random seed (42) was utilized to guarantee the reproducibility of the experiments and their outcomes.

- **Data Augmentation**:
  - Approach 1: Implemented basic augmentation techniques such as rotation, shifting, flipping, and zooming.
  - Approach 2: Employed similar augmentation methods with a few additional strategies like `RandomResizedCrop` to enhance generalization.
  - Approach 3: Adopted more aggressive augmentation techniques (for instance, `RandomBrightnessContrast`, `HueSaturationValue`, and `Cutout`) to ensure that the model acquires more robust features from the dataset.

8 Comparative Analysis of the Three Approaches (Training Setup)

Table 2 Comparative analysis of training configurations for all three approaches.

Aspect	**Approach 1**	**Approach 2**	**Approach 3**
Base Model	MobileNetV2 (ImageNet weights)	MobileNetV2 (ImageNet weights)	MobileNetV2 (ImageNet weights)
Preprocessing	Image resizing (224×224), data augmentation	Image resizing (224×224), data augmentation	Image resizing (224×224), advanced augmentation
Regularization	Dropout (0.4)	Dropout (0.4), L2 Regularization (0.01)	Dropout (0.5), L1–L2 Regularization (0.01, 0.01)
Model Hidden & Output Layers	Hidden layer: 1 (256 units, ReLU) Output: 11 units (softmax)	Hidden layers: 4 (1024–512–256–128 units, ReLU) Output: 11 units (softmax)	Hidden layers: 3 (1024–512–256 units, ReLU) Output: 11 units (softmax)
Optimizer	Adam (learning rate 1e−4)	Adam (learning rate 1e−4)	Adam (learning rate 1e−3)
Learning Rate Scheduler	ReduceLROnPlateau (factor 0.5)	ReduceLROnPlateau (factor 0.5), custom scheduler	ReduceLROnPlateau (factor 0.5), custom scheduler
Fine-Tuning	Train last 20 layers	Train last 30 layers	Train last 30 layers
Epochs	100	100	100
Batch Size	32	32	32
Callbacks	EarlyStopping, ModelCheckpoint, ReduceLROnPlateau	EarlyStopping, ModelCheckpoint, ReduceLROnPlateau	EarlyStopping, ModelCheckpoint, ReduceLROnPlateau
Evaluation Metrics	Accuracy, Precision, Recall	Accuracy, Precision, Recall	Accuracy, Precision, Recall

This comprehensive analysis offers a clear insight into the training methodologies and strategies utilized in each approach, illustrating how hyperparameters, regularization methods, and model optimization techniques have been adjusted to enhance performance.

9 Result

In this section, we outline the outcomes of the three methods applied to the "Infectious Skin Lesion Classification" task. The evaluation of performance is conducted using various essential metrics, including accuracy, loss, precision, recall, confusion matrices, and additional significant visualizations, such as Precision–Recall curves and AUC–ROC curves.

9.1 Approach 1

9.1.1 Classification Report

Table 3 Classification report for Approach 1.

Class	Precision	Recall	F1-Score
Cellulitis Impetigo and other Bacterial Infections	0.38	0.17	0.23
Chickenpox	0.92	0.85	0.89
Cowpox	0.95	0.96	0.95
HFMD	0.97	0.97	0.97
Healthy	0.95	0.97	0.96
Herpes, HPV, and other STDs	0.53	0.28	0.36
Measles	0.89	0.90	0.90
Monkeypox	0.94	0.94	0.94
Scabies Lyme Disease and other Infestations	0.71	0.40	0.51
Tinea (Ringworm), Candidiasis, and other Fungal Infections	0.81	0.91	0.86
Warts Molluscum and other Viral Infections	0.70	0.81	0.75

9.1.2 Training vs. Validation Accuracy

The Training vs. Validation Accuracy plot demonstrates the progression of the model's accuracy throughout the epochs. An effective model is expected to exhibit a consistent rise in both training and validation accuracy, with minimal disparity between the two. If the validation accuracy plateaus or declines while the training accuracy persists in its upward trend, this could suggest the occurrence of overfitting. Here we can see the model is a bit overfitting.

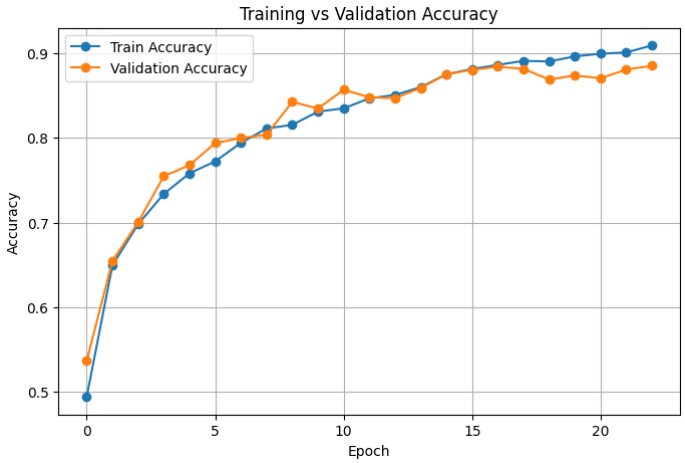

Fig. 3 Training vs. Validation Accuracy curve for Approach 1.

9.1.3 Training vs. Validation Loss

The Training vs. Validation Loss curve illustrates the progression of the model's loss, if it is increasing or decreasing. It is preferable to observe a decline in loss for both the training and validation datasets. Should the validation loss rise while the training loss falls, it may indicate the presence of overfitting, necessitating the consideration of measures such as early stopping or regularization. Here we can again see the overfitting tendency.

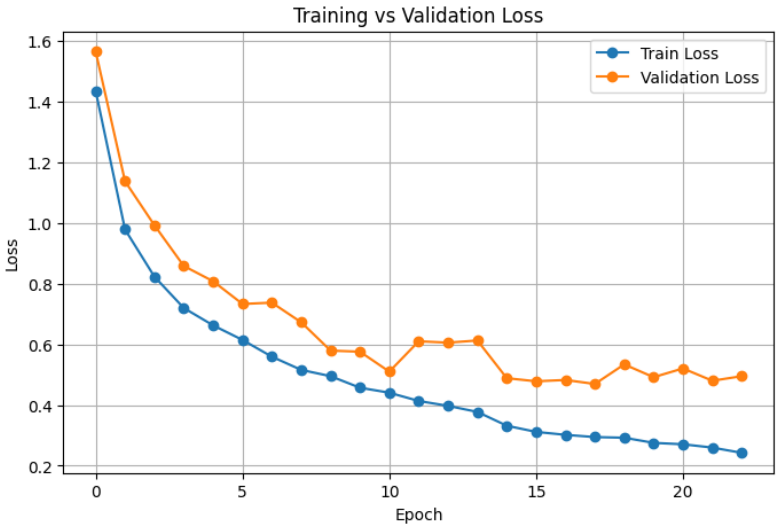

Fig. 4 Training vs. Validation Loss curve for Approach 1.

9.1.4 Confusion Matrix

The Confusion Matrix illustrates the distribution between predicted and actual classes. It provides a clear insight into which classes are misclassified and the nature of errors committed by the model. The diagonal values signify correct predictions, while the off-diagonal values denote misclassifications.

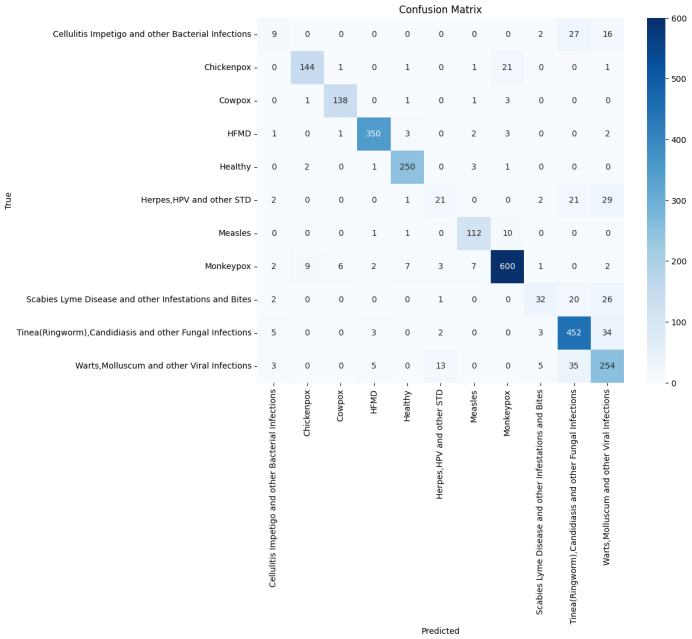

Fig. 5 Confusion Matrix for Approach 1.

9.1.5 Precision–Recall Curve

The Precision–Recall Curve is especially beneficial when working with imbalanced datasets. It illustrates the balance between precision (the proportion of relevant instances within the retrieved instances) and recall (the proportion of relevant instances that have been retrieved compared to the total number of relevant instances).

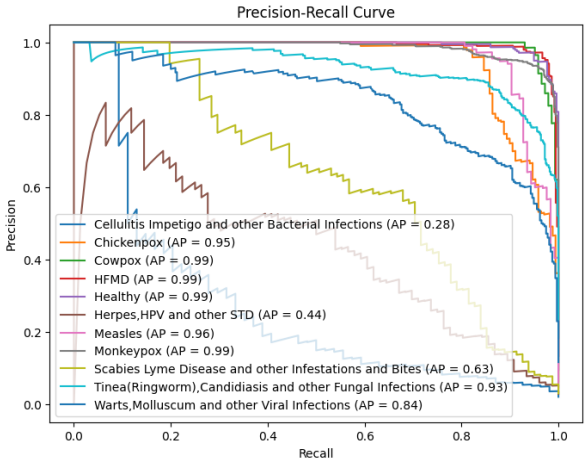

Fig. 6 Precision–Recall Curve for Approach 1

9.1.6 AUC–ROC Curve

The AUC–ROC Curve illustrates the relationship between the True Positive Rate (TPR) and the False Positive Rate (FPR) across different classification thresholds. A greater AUC (Area Under the Curve) signifies a more effective model. A model that achieves a higher area under the curve (AUC) demonstrates superior capability in balancing precision and recall. The nearer the AUC value approaches 1, the more proficient the model is at differentiating between positive and negative classes. A value closer to 0.5 means the model performs no better than random chance.

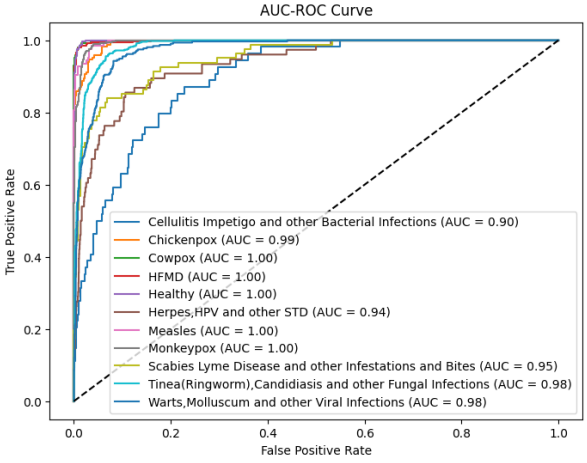

Fig. 7 AUC–ROC Curve for Approach 1.

9.2 Approach 2

9.2.1 Classification Report

Table 4 Classification report for Approach 2.

Class	Precision	Recall	F1-Score
Cellulitis Impetigo and other Bacterial Infections	0.40	0.11	0.17
Chickenpox	0.88	0.85	0.86
Cowpox	0.99	0.92	0.95
HFMD	0.94	0.96	0.95
Healthy	0.93	0.96	0.95
Herpes, HPV, and other STDs	0.55	0.24	0.33
Measles	0.93	0.87	0.90
Monkeypox	0.92	0.93	0.92
Scabies Lyme Disease and other Infestations	0.47	0.35	0.40
Tinea (Ringworm), Candidiasis, and other Fungal Infections	0.80	0.91	0.85
Warts Molluscum and other Viral Infections	0.70	0.78	0.74

9.2.2 Training vs. Validation Accuracy

Here, in the Training Accuracy vs. Validation Accuracy curve, we can see that till 33 epochs, training and validation accuracy both increased, but validation accuracy seems to surpass training accuracy, which means the model is underfitting.

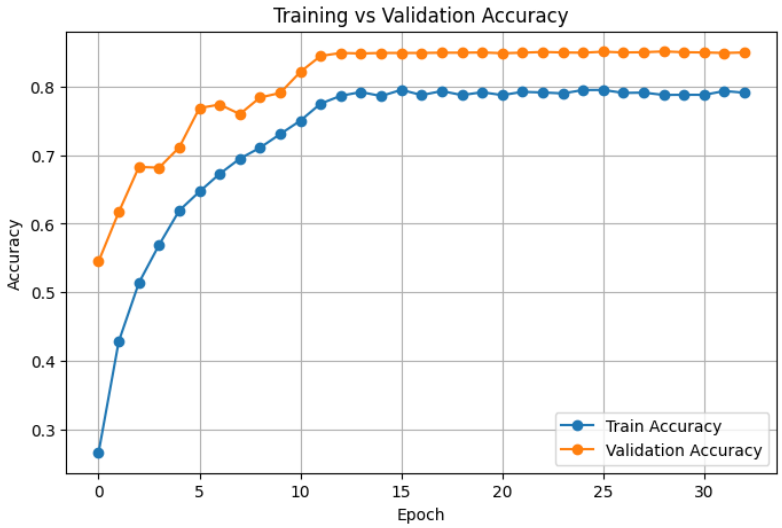

Fig. 8 Training vs. Validation Accuracy curve for Approach 2.

9.2.3 Training vs. Validation Loss

Here, in the Training vs. Validation Loss curve, loss seems to be gradually decreasing, and training and validation loss are nearly the same, approaching zero. We can come to the conclusion that parameters should be tuned to increase accuracy and to mitigate under-fitting.

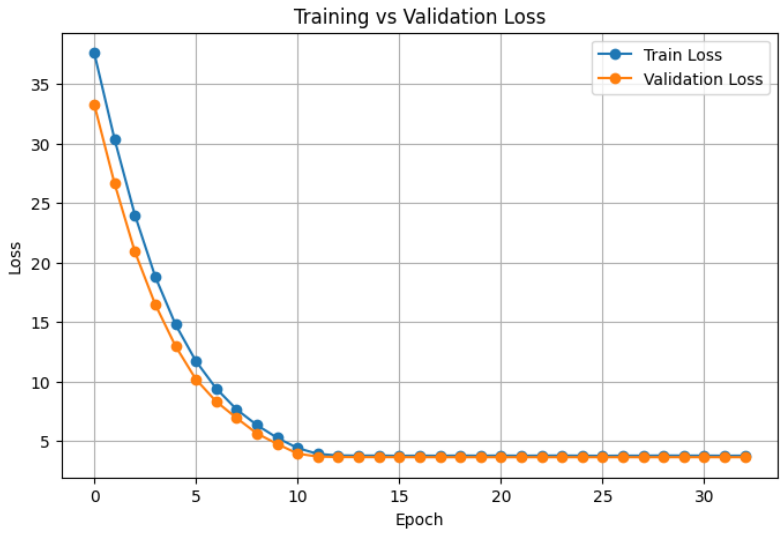

Fig. 9 Training vs. Validation Loss curve for Approach 2.

9.2.4 Confusion Matrix

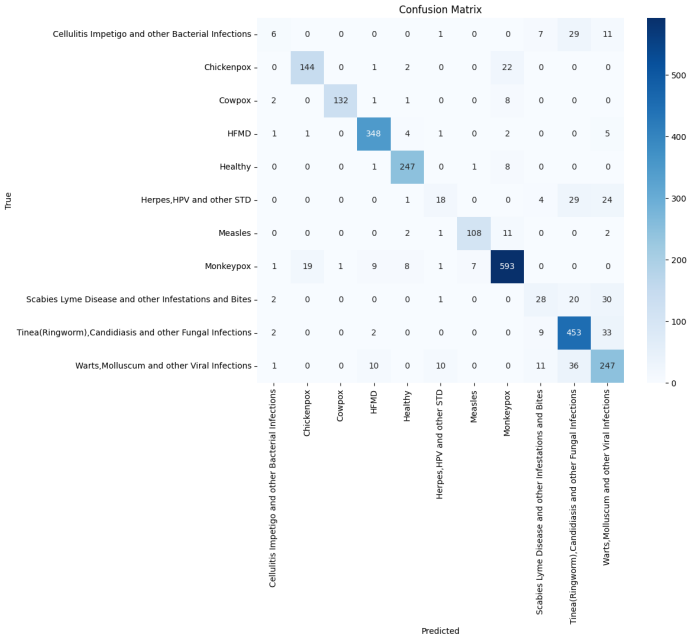

Fig. 10 Confusion Matrix for Approach 2.

The diagonal values signify correct predictions, while the off-diagonal values denote misclassifications. We can see misclassified points did not much improve than that was in Approach 1.

9.2.5 Precision–Recall Curve

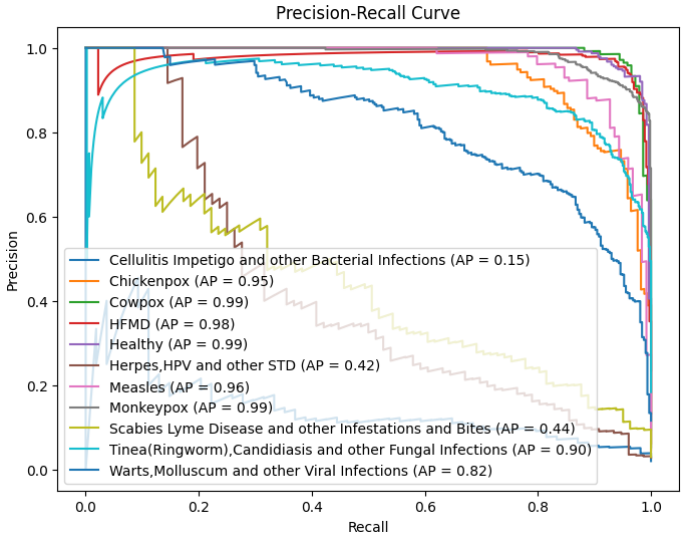

Fig. 11 Precision–Recall Curve for Approach 2.

It seems from the Precision–Recall curve, desirable improvement is still not there in Approach 2, which means there is still scope for improvement.

9.2.6 AUC–ROC Curve

The following AUC–ROC curve shows that some classes have more misclassified points that need to be addressed.

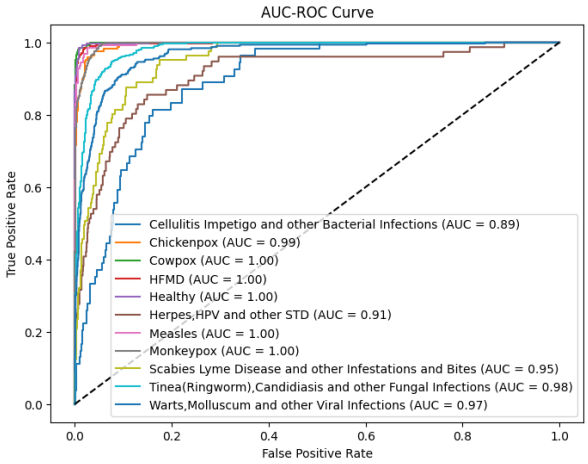

Fig. 12 AUC–ROC Curve for Approach 2.

9.3 Approach 3

9.3.1 Classification Report

Table 5 Classification report for Approach 3.

Class	Precision	Recall	F1-Score
Cellulitis Impetigo and other Bacterial Infections	0.75	0.22	0.34
Chickenpox	0.91	0.85	0.87
Cowpox	0.97	0.95	0.96
HFMD	0.96	0.96	0.96
Healthy	0.91	0.99	0.95
Herpes, HPV, and other STDs	0.59	0.34	0.43
Measles	0.93	0.86	0.90
Monkeypox	0.94	0.94	0.94
Scabies Lyme Disease and other Infestations	0.59	0.63	0.61
Tinea (Ringworm), Candidiasis, and other Fungal Infections	0.88	0.87	0.88
Warts Molluscum and other Viral Infections	0.69	0.84	0.76

9.3.2 Training vs. Validation Accuracy

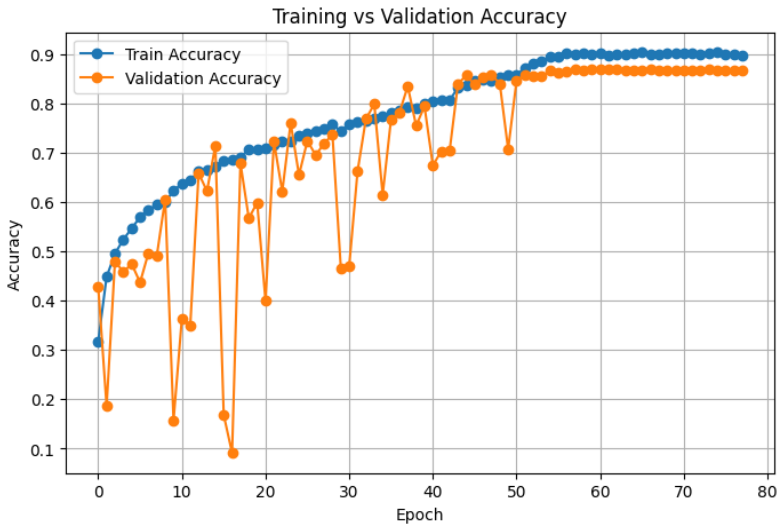

Fig. 13 Training vs. Validation Accuracy curve for Approach 3.

From Training Accuracy vs. Validation Accuracy, model training was unstable initially. That might happen for the learning rate scheduler. But after around 50 epochs, it became stable and showed a considerable performance.

9.3.3 Training vs. Validation Loss

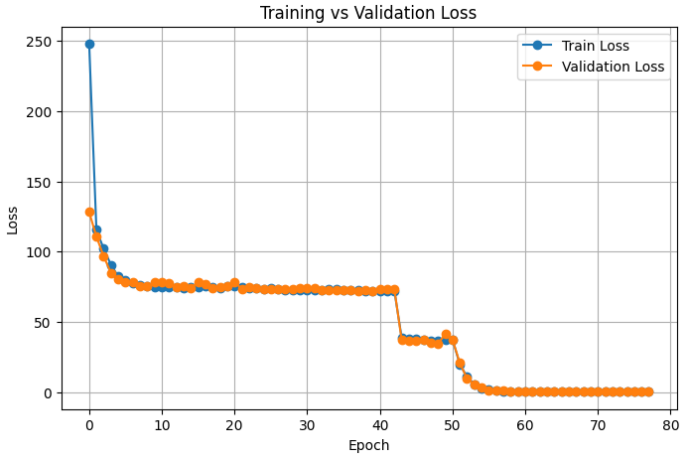

Fig. 14 Training vs. Validation Loss curve for Approach 3.

The Training Loss vs. Validation Loss curve showed through the epochs, loss kept on decreasing, and after nearly 55 epochs, it approached zero.

9.3.4 Confusion Matrix

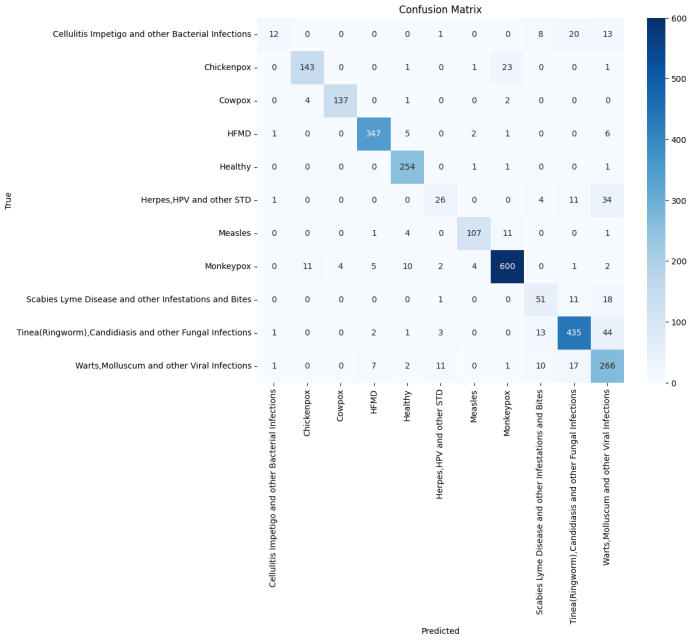

Fig. 15 Confusion Matrix for Approach 3.

The confusion matrix plot shows that this time the model's performance improved in decreasing misclassified points compared to the other two approaches.

9.3.5 Precision–Recall Curve

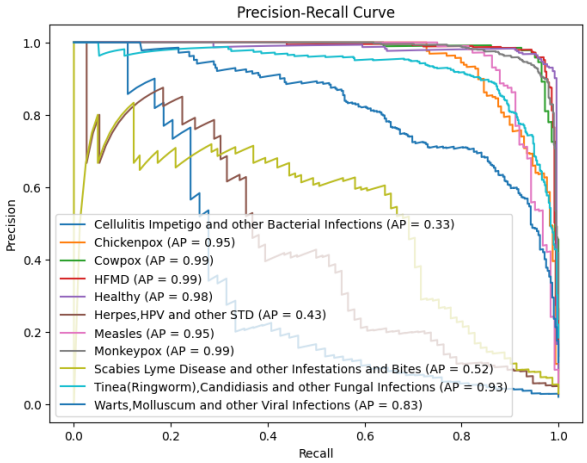

Fig. 16 Precision–Recall Curve for Approach 3.

The Precision–Recall curve also shows improved performance in identifying true-positive and true-negative points.

9.3.6 AUC–ROC Curve

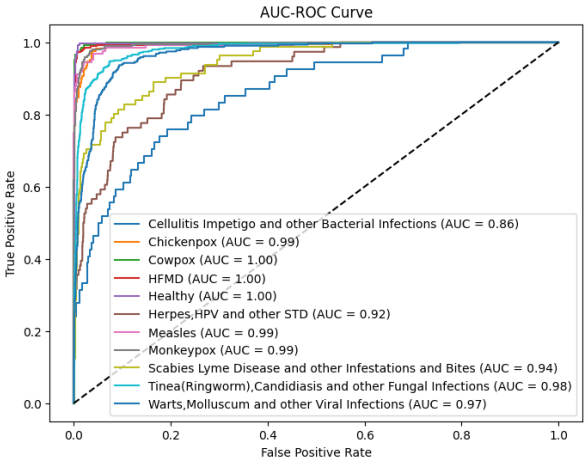

Fig. 17 AUC–ROC Curve for Approach 3.

Most of the classes' misclassified points decreased compared to the other two approaches, resulting in a higher area under the curve (AUC), denoting better performance.

9.4 Comparative Analysis of Test Results for All Three Approaches

Table 6 Comparative analysis of test results for all three approaches.

Metric	Approach 1	Approach 2	Approach 3
Test Accuracy	86.84%	85.44%	87.43%
Test Loss	0.4858	3.6173	0.6256
Test Precision	88.18%	87.70%	88.39%
Test Recall	86.14%	83.38%	86.73%
Micro Avg. Precision	0.79	0.77	0.83
Weighted Avg. Precision	0.86	0.84	0.87
Micro Avg. Recall	0.74	0.72	0.77
Weighted Avg. Recall	0.87	0.85	0.87
Micro Avg. F1 Score	0.76	0.73	0.78
Weighted Avg. F1 Score	0.86	0.84	0.87

The Results section provided a comparative analysis of the three methodologies utilizing essential classification metrics. The AUC–ROC and Precision–Recall curves offered valuable insights into the balance between precision and recall, with the AUC–ROC curve delivering a more profound comprehension of the model's capacity to differentiate between classes. The confusion matrix was instrumental in recognizing misclassifications, which could subsequently inform enhancements to the model.

10 Discussion

10.1 Limitations

1. **Class Imbalance**: Even with the application of data augmentation methods, the model may still encounter difficulties with classes that are underrepresented. For example, classes that contain a limited number of images (such as Herpes and HPV) tend to be more challenging to classify with precision.

2. **Dataset Variability**: The dataset's quality was inconsistent, with certain images exhibiting lower resolution or having labeling discrepancies. Such inconsistencies could adversely affect the accuracy of the model.

3. **Model Complexity**: Although MobileNetV2 is efficient, it may not be the most suitable architecture for all skin lesion classification tasks. For more intricate lesions, utilizing deeper networks like ResNet or DenseNet might yield superior performance.

10.2 Ethical Considerations

1. **Data Bias**: The dataset utilized may lack sufficient diversity to accurately represent all skin tones and ethnicities. This inadequacy can result in biased predictions from the model, thereby impacting its fairness across various demographic groups.

2. **Privacy and Consent**: Images of skin lesions constitute personal and sensitive medical information. It is imperative to ensure that proper informed consent is acquired from patients whose images are utilized. Furthermore, compliance with data privacy regulations is essential to safeguard individuals' rights.

3. **Clinical Application**: Although the model demonstrates encouraging outcomes, it should not be regarded as a substitute for professional diagnosis. The model ought to function as a decision support tool, with medical professionals responsible for making the final diagnosis.

11 Conclusion and Future Work

The Infectious Skin Lesion Classification project illustrates the capabilities of deep learning in the automatic classification of skin lesions from medical images. By fine-tuning a MobileNetV2 model using various methods, this research investigated how to tackle prevalent issues in skin lesion classification, including class imbalance and overfitting. The custom dense architecture, which incorporates L2 regularization, batch normalization, and dropout, achieved the optimal balance between training accuracy and generalization, resulting in minimal overfitting.

Although the model produced encouraging outcomes, there remains potential for enhancement:

1. **More Diverse Data**: Future endeavors could involve datasets that encompass a wider range of skin tones and lesion types, which may enhance the model's robustness across different demographics.

2. **Advanced Architectures**: Investigating alternative models, such as DenseNet or ResNet, could lead to improved accuracy, especially for more challenging classes.

3. **Real-Time Deployment**: Refining the model for use in clinical environments, including mobile and edge devices, would facilitate the integration of the model into practical diagnostic applications.

This research establishes a foundation for the development of more sophisticated tools in dermatology, assisting healthcare professionals in diagnosing infectious skin lesions more swiftly and accurately.

Data Availability

The datasets used in this study are publicly available from:

- Mendeley: https://data.mendeley.com/datasets/dfztdtfsxz/1
- Kaggle (Mpox Skin Lesion Dataset v2.0): https://www.kaggle.com/datasets/joydippaul/mpox-skin-lesion-dataset-version-20-msld-v20
- Kaggle (Skin diseases image dataset): https://www.kaggle.com/datasets/ismailpromus/skin-diseases-image-dataset
- Kaggle (Dermnet): https://www.kaggle.com/datasets/shubhamgoel27/dermnet

Code Availability

Github link:
https://github.com/sharika1103/Deep-learning-based-automated-diagnosis-of-infectious-skin-lesions

References

[1] Deng, X., & Zheng, Y. (2025). HSCFNet: Lightweight Convolutional Neural Network for the Classification of Infectious and Non-Infectious Skin Diseases. *International Journal of Imaging Systems and Technology*, 35. https://doi.org/10.1002/ima.70052.

[2] Thieme, A., Zheng, Y., et al. (2023). A deep-learning algorithm to classify skin lesions from mpox virus infection. *Nature Medicine*, 29, 738–747. https://doi.org/10.1038/s41591-023-02225-7.

[3] Ali, S., Ahmed, M., et al. (2023). A Web-based Mpox Skin Lesion Detection System Using State-of-the-art Deep Learning Models Considering Racial Diversity. *Biomedical Signal Processing and Control*, 98, 106742. https://doi.org/10.48550/arxiv.2306.14169.

[4] Rimi, T., Sultana, N., & Foysal, M. (2020). Derm-NN: Skin Diseases Detection Using Convolutional Neural Network. In *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)* (pp. 1205–1209). https://doi.org/10.1109/iciccs48265.2020.9120925.

[5] Burlina, P., Joshi, N., et al. (2019). Automated detection of erythema migrans and other confounding skin lesions via deep learning. *Computers in Biology and Medicine*, 105, 151–156. https://doi.org/10.1016/j.compbiomed.2018.12.007.

[6] Multi-Class Viral Skin Lesion Dataset (MCVSLD). Mendeley Data. https://data.mendeley.com/datasets/dfztdtfsxz/1.

[7] Mpox Skin Lesion Dataset Version 2.0 (MSLD v2.0). Kaggle. https://www.kaggle.com/datasets/joydippaul/mpox-skin-lesion-dataset-version-20-msld-v20.

[8] Skin diseases image dataset. Kaggle. https://www.kaggle.com/datasets/ismailpromus/skin-diseases-image-dataset.

[9] Dermnet dataset. Kaggle. https://www.kaggle.com/datasets/shubhamgoel27/dermnet.