# Ulcervision — Smart Foot Ulcer Multi-Class Detection Using Decision Level Fusion


**Supervisor**

**Miss Sania Urooj**

**Muhammad Mushtaq**

**21K-3273**

**Abbas Mustafa**

**21K-3271**

**Saleh Shamoon**

**21K-3333**

**Final year project report submitted in partial fulfillment of the requirements for the degree of BS (Computer Science)**

**Department of Computer Science**

**NUCES, Karachi**

**May 2025**

**Abstract**

This project presents an automated system for classifying chronic foot wounds into six types using convolutional neural networks. We trained four pre-trained models VGG19, DenseNet201, MobileNetV2, and NASNetMobile on an augmented dataset of 14,640 images. To improve prediction accuracy, we implemented decision-level fusion techniques like hard voting, soft voting, maximum likelihood, and stacking. The stacking model using XG Boost achieved the highest accuracy of 89.47%, significantly outperforming individual models.

# Contents

# Chapter 1

# Introduction

## 1.1 Background

A wound is a disruption in the integrity of the skin or underlying tissues caused by physical, chemical, or biological factors. Wounds generally fall into two categories: acute and chronic. Acute wounds, such as surgical incisions or minor cuts, follow a predictable healing process and typically close within a few weeks [1]. In contrast, chronic wounds fail to progress through the normal healing phases in a timely manner, often remaining open for months or even years [1][2]. These wounds, including diabetic ulcers, pressure ulcers, venous ulcers, and surgical wounds, are particularly challenging to manage due to factors such as infection, poor circulation, and underlying health conditions like diabetes or vascular disease. Chronic wounds pose a significant clinical and economic burden, leading to increased hospitalizations, prolonged treatment durations, and a higher risk of complications like amputations [1][2][3]. These wounds affect 1% to 2% of the population in developed countries [2]. In the United States alone, acute wounds affect 11 million people and chronic wounds influence more than 6 million humans annually with an estimated Medicare burden of $28.1 billion to $96.8 billion [3]. Wound diagnosis typically involves clinical assessment through visual inspection, measurement of wound size and depth, evaluation of tissue types, and identification of signs of infection. Healthcare professionals may also utilize imaging techniques, bacterial cultures, and patient medical history to inform their diagnosis [1][4]. However, manual diagnosis can be subjective, time-consuming, and prone to inter-observer variability. Machine learning (ML) offers a promising solution by enabling automated and objective wound assessment. ML models, particularly those based on convolutional neural networks (CNNs), can analyze wound images to classify wound types, assess severity, and even predict healing outcomes. These systems enhance diagnostic accuracy, reduce clinician workload, and support real-time decision-making, thereby improving patient care and outcomes [5]. Previous research has explored various deep neural network architectures, including ensemble methods and feature fusion strategies, to enhance classification performance [2][3][6][4].

## 1.2 Problem Statements

Prior research in automated wound classification has largely concentrated on binary classification tasks—such as distinguishing between ulcerated and non-ulcerated wounds—or on a limited set of classes (typically 2 to 4), leaving comprehensive multi-class classification relatively underexplored. Machine learning (ML) and deep learning (DL) approaches, particularly convolutional neural networks (CNNs), have shown promise in automating this process by learning discriminative features directly from wound images, reducing reliance on subjective clinical judgment. Most studies to date have utilized well-established CNN

architectures like ResNet and EfficientNet; however, the potential of other state-of-the-art models—including VGG19, NasNet, MobileNet, and DenseNet—remains under-investigated, especially in the context of the AZH dataset. Beyond individual model performance, ML techniques also allow for enhanced diagnostic accuracy through decision-level fusion strategies (e.g., weighted fusion, maximum likelihood, hard voting) and stacking ensemble methods (e.g., XGBoost, Logistic Regression, Random Forest). Furthermore, feature-level fusion, which combines features extracted from multiple architectures, can yield richer and more robust representations for classifying various chronic wound types. These advanced ML/DL strategies not only improve predictive performance but also hold potential for supporting real-time, scalable wound assessment in clinical settings.

## 1.3 Research Objectives

The main objectives of this research are to:

- Investigate whether these models, which have not been previously fine-tuned on this dataset, can achieve improved accuracy compared to earlier approaches.
- Evaluate the performance of state-of-the-art pre-trained CNN models—VGG19, NasNet, MobileNet, and DenseNet—on the multi-class classification of chronic wound images from the AZH dataset.
- Enhance classification outcomes through decision-level fusion techniques (weighted fusion, max likelihood, hard voting) and stacking methods (using XGBoost, Logistic Regression, and Random Forest).
- Explore feature fusion strategies that combine the discriminative features extracted from different networks to further improve the robustness and accuracy of the classification.

## 1.4 Contribution

This study makes several key contributions:

- It provides a comprehensive evaluation of multiple state-of-the-art pre-trained CNN models that have not been previously applied to the AZH chronic wound dataset.
- It expands the classification task beyond previous studies by working with six wound classes: diabetic, surgical, venous, pressure, background, and non-classified wounds.
- It demonstrates the effectiveness of decision-level fusion and stacking strategies in boosting classification accuracy beyond that of individual networks, as supported by recent ensemble-based approaches [3][6].
- It introduces feature fusion techniques to integrate complementary features from VGG19, NasNet, MobileNet, and DenseNet, potentially yielding more robust and discriminative representations [7].

- The insights gained from this work offer a novel perspective on how modern deep learning methods can be leveraged to advance automated wound classification, ultimately aiding in the development of more accurate and efficient diagnostic tools.

## 1.5 Organization of Final Year Project

The remaining thesis is organized as follows: In Chapter 2 the existing literature related to chronic wound classification has been discussed. Chapter 3 will summarize the detailed proposed approach and functionality of the modes on the detection of six chronic wounds. In Chapter 4, a range of different experiments are being done, and results are generated. It briefly describes the experimental setup, and the entire research work conducted under the title. Chapter 5 gives the conclusion of this study. Discussion about the limitations and possible improvements in future are also stated.

---

# Chapter 2

# Literature Review

## 2.1. Chapter Overview

This chapter reviews existing work on chronic wound analysis using deep learning. It discusses the use of CNNs, fusion strategies, and segmentation techniques. The chapter evaluates various model architectures, such as DFU-SIAM, XAI-FusionNet, and U-Net derivatives, and compares their effectiveness. It also emphasizes the significance of deep feature fusion and ensemble methods for improving classification accuracy and positions the current research within this context.

## 2.2 Introduction

Chronic wounds such as diabetic foot ulcers (DFUs), venous ulcers, and pressure ulcers remain a significant problem in modern healthcare with their prolonged healing periods, risk of recurrence, and cost. In the United States alone, an estimated 6 million individuals have chronic wounds, with an additional healthcare cost per year in billions of dollars [15]. Wound evaluation based on current techniques relies on experience and visual inspection, which will be subjective and variable. Wound evaluation can be significantly improved by standardized image-based evaluation protocols, especially when integrated with current artificial intelligence systems [1].

## 2.3 Deep Learning in Wound Analysis

CNNs are now the standard architecture in designing wound classification models. Aldoulah et al. [2] presented a hybrid multi-class CNN model with good performance for different kinds of wounds. The model is derived from two EfficientNet-B4 models, differing in the layer freezing and the activation function applied. Specifically, we employed two models and concatenated them in an attempt to learn more informative features. On the AZH dataset with 6 classes, they achieved an accuracy of 83.19%.

Similarly, Rostami et al. [3] introduced an ensemble-based classifier that merges patch-wise and image-wise CNN predictions to enhance classification accuracy. The ensemble comprises two AlexNet classifiers with multi-class wound image classification. The first classifier operates at the image level, whereas the second classifier operates at the patch level with the sliding window method. The result of the two classifiers is merged, and the result is passed to an MLP for prediction of the class label. However, the patch processing and generation step in the patch-wise classifier makes the ensemble model complex, which requires extra time and computational cost. For the 3-class (venous-arterial, pressure, and diabetic) classification, average accuracies of 82.9% and 87.7% were achieved on the Medetec and AZH datasets, respectively. This approach [3] provides acceptable accuracy in

classifying venous and surgical wounds but has the lowest accuracy in classifying pressure and diabetic wounds.

These two papers lay the groundwork for our project by demonstrating the strength of multi-network architecture. Toofanee et al. [5] presented DFU-SIAM that successfully classified DFUs with a hybrid model that combined attention mechanisms with deep CNN layers. Biswas et al. [7] pushed the research forward with XAI-FusionNet with a balance between performance and model explainability using explainable AI methods, which is in line with the objective of our project to develop a reliable clinical tool.

## 2.4 Fusion Strategies in Practice

One of the prominent features of contemporary wound classification systems is feature and decision fusion. Biswas et al. [7] utilized multi-scale feature fusion to obtain varied spatial and contextual diabetic foot ulcer (DFU) features with an accuracy of 88.92%. Aldoulah et al. [2] also illustrated that feature fusion of varied CNNs improves classification accuracy, achieving a performance of 87.3% via a fused multiclass approach. In this work, we also explored decision-level fusion methods such as hard voting, weighted soft voting, maximum likelihood fusion, and stacking through meta-learning. Of these, stacking with a Random Forest meta-classifier exhibited the highest performance with a classification accuracy of 89.47%, surpassing individual CNNs and comparable with the literature. Similarly, Munadi et al. [6] illustrated decision fusion between RGB and thermal modalities for detecting DFU with an accuracy of 84.6%.

Our work extends the complementary model strengths framework by using four pre-trained convolutional neural networks—VGG19, DenseNet201, MobileNetV2, and NASNetMobile—individually as classifiers. We select each model for the unique architectural strengths and train each to generate class probabilities for multiclass wound classification into six clinically meaningful classes: diabetic, venous, pressure, surgical, natural (healthy), and background. Rather than fusing the feature embeddings, we apply a decision-level fusion strategy, where the predictions of the models are aggregated to generate final predictions. This ensemble strategy maintains heterogeneous decision patterns across models and enhances the robustness and classification accuracy. Although our work considers RGB images only, the publicly available dataset of 14,880 augmented images across these six classes provides a reasonable platform for testing decision fusion strategies for multiclass chronic wound diagnosis.

## 2.5 Segmentation and Wound Localization

Successful wound classification frequently relies on successful segmentation, since localized representation of wound areas isolates key features. In the 2024 work of Alabdulhafith et al. [8] published in Frontiers in Medicine, a hybrid model of ResNet34 and U-Net was proposed, with about 92% segmentation accuracy, and demonstrating residual learning in encoder-decoder architectures. Motivated by these layered representations, our FYP experimented with U-Net

derivatives in early-stage experiments to localize chronic wound areas prior to applying classification. Though we eventually shifted to a direct classification pipeline based on RGB images and deep feature fusion, the segmentation step guided our data preprocessing and augmentation approach. Oota et al. [11] proposed WSNet, trained on the WoundSeg dataset, using global and local patches for segmentation generalization, with IoU scores above 90%. Ohura et al. [12] compared segmentation models like LinkNet, SegNet, U-Net, and Unet_VGG16, establishing U-Net's better accuracy-computational cost ratio, which justified its use in our early-stage analysis. Scebba et al. [9] established the effectiveness of localization in classification through their 'detect-and-segment' pipeline, which supported that segmentation quality is strongly coupled with classification reliability. Though segmentation was not included in our final model architecture, these findings helped solidify the preprocessing rigor of our dataset and contributed to overall robustness of the classification framework.

Advanced classification networks like MobileNet, DenseNet, and NASNetMobile have been thoroughly studied. Pereira et al. [16] integrated MobileNet-UNet as a segmentation component with ML classifiers (SVM, RF, KNN) for outcome prediction for wounds. This is also similar to our decision to use MobileNet for light-weight feature extraction. Huang et al. [14] used Mask R-CNN for clinical wound data and achieved very high precision and recall scores, demonstrating that CNN-based object detection and segmentation networks are capable of performing well in actual clinical settings as well. Wang et al. [10] made an interesting contribution as a fully automated pipeline for healing progress analysis, highlighting that temporal characteristics of wound care are to be incorporated in future models as well.

## 2.6 Clinical Context and Human Impact

It is stated by Sen et al. [13] that chronic wounds are biologically intricate and typically delayed in healing as a result of systemic disease states like diabetes, peripheral artery disease, and infection. Their review of the clinical aspect is focused on the necessity of smart tools that are capable of processing anatomical, textural, and contextual information which are effective in wound triaging and classification. Moreover, Sen's economic analysis [15] highlights the financial necessity: chronic wounds afflict more than 6 million people in the U.S. every year, with an estimated Medicare cost ranging from $28.1 to $96.8 billion. These figures point towards the necessity of AI-powered diagnostic tools which can speed up assessment without compromising accuracy. Our model helps address this need directly by automating six different wound classification types—diabetic ulcers, venous ulcers, pressure ulcers, surgical wounds, healthy (natural) tissue, and background which provide a larger diagnostic scope than previous efforts, which usually restricted classifications to binary or three-class ones [3, 7, 14]. Based on 14,880 augmented RGB images and fueled by decision fusion from VGG19, DenseNet201, MobileNetV2, and NASNetMobile. The performance justifies its suitability for deployment in clinical decision support systems, particularly in resource-constrained or high-volume environments where fast, automated wound classification can have a considerable impact on patient outcomes and decrease caregiver workload.

## 2.7 Positioning This Research

Our research is unique by employing a decision level fusion approach with four pre-trained CNNs i.e. VGG19, DenseNet201, MobileNetV2, and NASNetMobile, followed by a dense neural classifier to identify six classes of chronic wounds. Such an architecture supports the extraction and fusion of high-dimensional feature representations (with a total of 29,344 dimensions), providing a compact, expressive input to classification. In contrast to earlier models confined to binary or tri-class configurations, ours is trained on a highly varied, augmented dataset of 14,880 images across diabetic, venous, pressure, surgical, natural (healthy), and background wound classes. This pipeline enhances the foundation laid in segmentation [8, 11, 14], classification by feature fusion and ensemble learning [2 , 3 , 7], and explainability in clinical diagnosis [7, 17], and specifically tackles frequent issues of low-class granularity, dataset imbalance, and under-feature integration reported in research such as Wang et al. [10] and Ohura et al. [12]. Our methodology resulted in 89.47% accuracy of classification with stacked ensemble learning, confirming that it can function as a high-volume, real-time diagnostic modality. Our research is hence positioned to be compatible with outpatient, telehealth, or limited-resource clinical work settings, combining both accuracy and effectiveness in identifying and triaging chronic wounds.

---

# Chapter 3

# Methodology

## 3.1. Chapter Overview

This section describes the complete pipeline followed for the chronic wound classification using deep learning and feature fusion. The key steps include data acquisition, preprocessing, data augmentation, model selection, feature extraction, feature fusion, and evaluation.

## 3.2 Model Architectures

To extract meaningful and high-level features from chronic wound images, we utilized four well-established convolutional neural network (CNN) architectures that were pre-trained on the ImageNet dataset. Each architecture offers unique design principles and strengths, which contributed to robust and diverse feature representations. The architectures employed are VGG19, DenseNet201, MobileNetV2, and NASNetMobile.

1. VGG19

VGG19 is a deep CNN proposed by the Visual Geometry Group (VGG) at Oxford. It consists of 19 layers (16 convolutional and 3 fully connected) and is known for its simplicity and effectiveness.

- Architecture: The model uses small $3 \times 3$ convolutional filters applied with stride 1 and padding to preserve spatial resolution.
- Activation: ReLU (Rectified Linear Unit) activation is used after every convolutional layer to introduce non-linearity.
- Pooling: Max-pooling is used with a $2 \times 2$ window and stride 2 after every few convolutional layers to downsample feature maps.
- Adaptation: For our project, the fully connected (FC) layers and the classification head were removed. We extracted features from the last convolutional block (Conv5 4), resulting in a $7 \times 7 \times 512$ feature map, which was then flattened to a 1D vector of length 25,088.

2. DenseNet201

DenseNet201 is part of the Densely Connected Convolutional Networks family introduced by Huang et al. It addresses vanishing gradient issues and promotes feature reuse through dense connectivity.

- Dense Connectivity: Each layer receives the concatenated output of all preceding layers, which improves information flow and gradient propagation.
- Compression and Bottlenecks: Uses $1 \times 1$ convolutional bottleneck layers followed by $3 \times 3$ convolutions to reduce parameters.
- Transition Layers: These include batch normalization, ReLU activation, $1 \times 1$ convolutions, and average pooling to manage feature map sizes.
- Feature Extraction: We removed the classifier and applied Global Average Pooling (GAP) on the final convolutional output, resulting in a feature vector of length 1920.

3. MobileNetV2

MobileNetV2 is an efficient model optimized for mobile and embedded devices. It builds on MobileNetV1 by introducing inverted residuals and linear bottlenecks.

- Depthwise Separable Convolutions: Standard convolutions are replaced with depthwise and pointwise convolutions to reduce computation.
- Inverted Residuals: These structures expand the input dimensions first, apply depthwise con- volution, and then project back to a lower-dimensional space.
- Linear Bottlenecks: Prevent loss of information by using linear activations at the bottleneck layers.
- Feature Extraction: Features were extracted after removing the top FC layers. The GAP layer was applied to the final convolutional feature map, yielding a 1280-dimensional vector.

4. NASNetMobile

NASNetMobile (Neural Architecture Search Network for Mobile) is designed through automated architecture search to balance accuracy and efficiency.

- Cell-based Design: Uses normal and reduction cells designed by reinforcement learning. These cells are repeated to form the complete architecture.
- Factorized Convolutions: Similar to MobileNet, it uses separable convolutions to enhance computational efficiency.
- Scalability: Architectures can be scaled to meet mobile or server-based needs (NASNetMobile for mobile; NASNetLarge for servers).
- Feature Extraction: Final convolutional block output was extracted and passed through a GAP layer, resulting in a 1056-dimensional vector.

**Summary of Architectural Characteristics:**

| Model | Parameters | Feature Vector Size | Core Strength |
|---|---|---|---|
| VGG19 | 143.67 million | 25,088 | Deep architecture with uniform structure |
| DenseNet201 | 20.24 million | 1920 | Dense connections for efficient gradient flow |
| MobileNetV2 | 3.47 million | 1280 | Lightweight with inverted residuals |
| NASNetMobile | 5.3 million | 1056 | Automatically optimized for mobile |

Table 1: Summary of Model Architectures and Feature Extraction Outputs

## 3.3. Proposed Methodology

The proposed system is divided into five major stages:

- Data Acquisition: Images are obtained from a publicly available GitHub dataset.
- Preprocessing: Includes resizing, normalization, and data augmentation to enhance dataset diversity and balance.
- Feature Extraction: Deep features are extracted from VGG19, DenseNet201, NASNetMobile, and MobileNetV2 by removing their final classification heads.
- Feature Fusion: The extracted features are concatenated into a single feature vector for each image.
- Classification: A fully connected dense classifier with dropout and softmax is trained to classify the fused features.

### 3.3.1. Data Acquisition

The dataset used for this study is sourced from a publicly available chronic wound dataset on GitHub [**?**]. It contains foot images representing different categories of chronic wounds, including diabetic foot ulcers, pressure ulcers, venous ulcers, surgical wounds, as well as healthy foot images. Additionally, background images are included to represent non-foot objects, aiding in reducing false positives during classification. This dataset was carefully selected for its diversity in chronic wound types and its applicability in real-world clinical scenarios. The presence of multiple classes ensures that the model learns to distinguish between different wound types effectively. After data preprocessing, filtering, and cleaning, the dataset was divided into two subsets: the training set and the testing set.

### 3.3.2. Training Dataset

The training dataset comprises labeled images used for model training and optimization. It consists of 6 distinct classes:

- Diabetic Foot Ulcers (Diabetic): Diabetic ulcers are a severe complication of diabetes mellitus caused by poor circulation and neuropathy. Recognizing these ulcers early can prevent further complications, including infections and amputations. The training set contains 139 images.
- Pressure Ulcers (Pressure): These ulcers develop due to prolonged pressure on the skin, espe- cially in patients who are bedridden. Early detection is essential for effective treatment. There are 100 images of pressure ulcers in the training dataset.
- Venous Ulcers (Venous): Venous ulcers are caused by improper blood flow, often in the lower legs, leading to open sores. The dataset includes 185 images in this class.
- Surgical Wounds (Surgical): Surgical wounds may face complications like infections or de- layed healing. This class contains 122 images, helping the model recognize post-operative wound conditions.
- Healthy Foot Images (Natural): These are images of normal, healthy feet without any signs of chronic wounds. Serving as a baseline for the model, the dataset contains 75 images in this category.
- Background Images (Background): Background images include irrelevant objects such as floors, medical instruments, or environments with no visible feet. They assist in minimizing false positive classifications. There are 75 images in this class.

### 3.3.3. Testing Dataset

The testing dataset is used to evaluate the model's generalization ability. It also consists of 6 classes, with a balanced representation of wound types to ensure reliable performance analysis.

- Diabetic Foot Ulcers (Diabetic): 46 images
- Pressure Ulcers (Pressure): 34 images
- Venous Ulcers (Venous): 62 images
- Surgical Wounds (Surgical): 42 images
- Healthy Foot Images (Natural): 25 images
- Background Images (Background): 25 images

### 3.3.4 Total Dataset Composition

Combining the training and testing datasets, the final distribution across all classes is as follows:

- Diabetic Foot Ulcers (Diabetic): 185 images
- Pressure Ulcers (Pressure): 134 images
- Venous Ulcers (Venous): 247 images
- Surgical Wounds (Surgical): 164 images
- Healthy Foot Images (Natural): 100 images
- Background Images (Background): 100 images

This results in a total of 930 images used for both training and testing purposes. The dataset's balanced class distribution ensures the development of a robust and accurate model capable of detecting and classifying various wound types effectively.

The data was further augmented to enhance the generalization capability of the models. Data augmentation techniques such as random rotations, horizontal and vertical flips, brightness adjustments, and zoom operations were applied to generate additional images, making the dataset more diverse and resilient to overfitting. This expanded dataset significantly contributed to the improved accuracy and robustness of the model. Detailed augmentation results will be presented in the subsequent sections.

### 3.3.5. Data preprocessing and Augmentation

To ensure the effective training of the models and improve generalization capability, the dataset under- went extensive preprocessing and augmentation. These steps were essential for creating a more diverse set of images, reducing overfitting, and enhancing model performance.

### 3.3.6. Image preprocessing

The initial dataset images varied in size and quality. As the pre-trained models used in this study require fixed input dimensions, all images were resized to a standard resolution of 224 × 224 × 3 pixels. This resolution maintains the necessary visual details while being computationally efficient. Additionally, the images were normalized by scaling pixel values to the range of [0, 1] to accelerate convergence during training.

### 3.3.7 Data Augmentation

Data augmentation techniques were applied to artificially expand the dataset by generating new images with slight modifications. This not only increased the dataset size but also introduced variations in the data, which improved the model's ability to generalize to unseen images.

The following augmentation techniques were applied:

- Random Rotation: Images were randomly rotated within the range of $\pm 30^{\circ}$ to simulate different viewing angles of the wounds.
- Horizontal and Vertical Flipping: Both horizontal and vertical flips were applied to create mirror images, increasing data diversity.
- Random Zoom: A zoom range of up to 20% was applied to simulate closer or farther views of the wounds.
- Brightness Adjustment: Random brightness adjustments were applied to mimic different light- ing conditions in real-world scenarios.
- Image Normalization: Pixel values were normalized to a standard scale using mean subtraction and division by standard deviation. This ensured consistent input for the models.

### 3.3.8 Augmented Dataset Composition

After applying augmentation, the dataset size significantly increased. The final distribution of images in the augmented test and train datasets is presented below:

Test Dataset (Augmented) The augmented test dataset now includes:

- Augmented Background Images: 400 images
- Augmented Diabetic Foot Ulcer Images: 736 images
- Augmented Healthy Foot Images (Natural): 400 images
- Augmented Pressure Ulcer Images: 544 images
- Augmented Surgical Wound Images: 672 images
- Augmented Venous Ulcer Images: 992 images

Train Dataset (Augmented) The augmented train dataset expanded to:

- Augmented Background Images: 1200 images
- Augmented Diabetic Foot Ulcer Images: 2224 images
- Augmented Healthy Foot Images (Natural): 1200 images
- Augmented Pressure Ulcer Images: 1600 images
- Augmented Surgical Wound Images: 1952 images
- Augmented Venous Ulcer Images: 2960 images

### 3.3.9. Total Dataset Size after Augmentation

Combining both the augmented test and train datasets, the total number of images used in the study is:

- Background Images: 400 + 1200 = 1600 images
- Diabetic Foot Ulcer Images: 736 + 2224 = 2960 images
- Healthy Foot Images (Natural): 400 + 1200 = 1600 images
- Pressure Ulcer Images: 544 + 1600 = 2144 images
- Surgical Wound Images: 672 + 1952 = 2624 images
- Venous Ulcer Images: 992 + 2960 = 3952 images

This results in a total of 14,880 images after augmentation, significantly increasing the data available for model training and testing. The augmented dataset provided a rich variety of samples, representing diverse wound characteristics and enhancing the model's robustness in detecting chronic wounds. Further analysis of the impact of data augmentation on model performance will be presented in the results section.

## 3.4. Model Selection

For feature extraction, we employed four state-of-the-art pre-trained deep learning models, each offering distinct architectural advantages. These models were selected due to their superior performance on large-scale image classification tasks and their ability to generalize well across different datasets. The models used in this study are VGG19, DenseNet201, MobileNetV2, and NASNetMobile, each of which is described in detail below.

- **VGG19**: VGG19 (Visual Geometry Group 19-layer network) is a deep convolutional neural net- work (CNN) developed by Simonyan and Zisserman in 2014. It consists of 19 weighted layers with a uniform architecture of 3 × 3 convolutional kernels and ReLU activation functions. The model's key characteristics include: Architecture: 16 convolutional layers, 3 fully connected layers, and a softmax output layer. Depth: Increased depth allows for more complex feature representations. Feature Extraction: Uses small receptive fields (3×3 filters) with stride 1 and padding, ensuring fine-grained spatial feature extraction. Pooling Mechanism: Uses max-pooling (2×2 filters with stride 2) to reduce spatial dimensions progressively. Fully Connected Layers: The last three layers are fully connected, with 4096 neurons each, before the final classification layer. In our study, the fully connected layers were removed, and deep feature embeddings were extracted from the last convolutional block (before the flattening layer).

- **DenseNet201**: DenseNet201 (Densely Connected Convolutional Networks) was introduced by Huang et al. in 2017. Unlike traditional CNNs, where each layer has independent connections, DenseNet follows a dense connectivity pattern where each layer receives inputs from all previous layers and passes its own output to all subsequent layers. Key features include: Dense Connectivity: Each layer is connected to all preceding layers, promoting feature reuse and reducing the vanishing gradient problem. Efficient Parameter Usage: Uses fewer parameters compared to traditional deep networks, making it computationally efficient. Growth Rate: Controls how much each layer contributes to feature maps, improving gradient flow. Bottleneck Layers: 1×1 convolutional layers reduce dimensionality before applying expensive 3×3 convolutions. Pooling and Transition Layers: Transition layers with batch normalization and average pooling reduce dimensionality while preserving important spatial information. We extracted deep features from the final convolutional layer of DenseNet201, utilizing its feature-rich embeddings for chronic wound classification.
- **MobileNetV2**:MobileNetV2 is a lightweight CNN architecture optimized for mobile and edge computing applications. Introduced by Sandler et al. in 2018, it builds upon MobileNetV1 with inverted residual connections and linear bottlenecks, making it highly efficient. Key architectural enhancements include: Depthwise Separable Convolutions: Replaces standard convolutions with depthwise convolution followed by pointwise convolution, reducing computational cost. Inverted Residuals: Unlike traditional residual blocks, where input is expanded, MobileNetV2 first applies 1×1 convolutions to compress features, followed by depthwise convolution, and then 1×1 convolutions to restore dimensionality. Linear Bottleneck: Prevents information loss during feature transformations, improving representation capability. ReLU6 Activation: Used in intermediate layers for better quantization in low-power devices. Lightweight and Fast: Ideal for deployment on resource-constrained sys- tems. Deep features were extracted from the last convolutional layer before the classification head, capturing spatially rich representations with minimal computational overhead.
- **NASNetMobile**:NASNetMobile (Neural Architecture Search Network) was developed by Google Brain using reinforcement learning-based neural architecture search (NAS). It is optimized for mobile devices while maintaining high classification accuracy. Key innovations in NASNetMobile include: Automated Architecture Search: Designed through an evolutionary algorithm that optimizes performance and computational efficiency. Factorized Convolutions: Uses separable convolutions similar to MobileNet for efficient computation. Hierarchical Search Space: Uses a cell-based design, where optimized convolutional blocks are stacked to create a deep network. Reduction and Normal Cells: Two types of cells (normal cells

for feature extraction, reduction cells for downsampling) dynamically adjust network depth and complexity. Efficient Feature Learning: Extracts multi-scale features while maintaining computational efficiency. Deep feature embeddings were extracted from the final convolutional block of NASNetMobile and later fused with features from other models.

## 3.5 Feature Extraction

Feature extraction is a crucial step in leveraging the knowledge of pre-trained deep learning models for the chronic wound classification task. In this study, deep features were extracted from the final convolutional layers of four state-of-the-art models: VGG19, DenseNet201, MobileNetV2, and NASNetMobile. Each model was pre-trained on the large-scale ImageNet dataset, enabling them to capture robust and meaningful features from images.

To adapt these models for feature extraction, the fully connected (FC) layers were removed, and the output from the last convolutional block was used as a high-dimensional feature representation. These feature maps were then flattened into one-dimensional (1D) vectors, forming the final feature embeddings used for classification.

### 3.5.1. Feature Extraction Process

The process followed for feature extraction from each model is outlined below:

- Model Initialization: Each pre-trained model was loaded using weights learned from ImageNet. The models were initialized without their final classification layers (fully connected layers).
- Convolutional Feature Extraction: The final convolutional layers of the models act as high- level feature extractors. These layers capture hierarchical patterns such as edges, textures, and object shapes.
- Global Average Pooling (GAP): For some models, Global Average Pooling was applied to reduce the dimensionality of the feature maps. GAP computes the average value of each feature map, resulting in a compact feature vector.
- Flattening: The extracted feature maps were flattened into 1D vectors. This step ensures compatibility with traditional machine learning classifiers that accept vectorized input.
- Feature Embedding: The resulting feature embeddings represent rich information about the images, including texture, shape, and color patterns, which are essential for chronic wound classification.

### 3.5.2. Feature Extraction from Individual Models

Here is how feature extraction was specifically performed for each model:

VGG19

- Convolutional Layers: VGG19 consists of 16 convolutional layers followed by 3 fully connected layers. For feature extraction, only the convolutional layers were retained.
- Output Layer Selection: The output from the last convolutional block (Conv5 4) was used.
- Feature Dimensionality: The output feature map size was $7 \times 7 \times 512$, representing 512 feature channels.
- Flattening: The feature map was flattened into a 1D vector of size 25,088 ($7 \times 7 \times 512$).

DenseNet201

- Dense Connectivity**:** In DenseNet201, each layer receives inputs from all preceding layers, resulting in rich feature propagation.
- Output Layer Selection: The output was taken from the final convolutional layer (before the classification layer).
- Global Average Pooling (GAP): GAP was applied to the output feature map, reducing it to a 1D vector with 1920 feature values.
- Feature Dimensionality**:** The final vector had a size of 1920.

MobileNetV2

- Efficient Design: MobileNetV2 uses depthwise separable convolutions and inverted residuals to extract lightweight but effective features.
- Output Layer Selection: Features were extracted from the last convolutional block, just before the classification head.
- Global Average Pooling (GAP): Similar to DenseNet201, GAP was applied to reduce the spatial dimensions, producing a compact feature vector.
- Feature Dimensionality: The resulting vector had 1280 features.

NASNetMobile

- Neural Architecture Search (NAS): NASNetMobile was designed using an automated neural architecture search to optimize model performance for mobile applications.
- Output Layer Selection: Feature extraction was performed using the final convolutional block, which captures complex hierarchical patterns.

- Global Average Pooling (GAP): GAP was applied to generate a summarized representation of the feature maps.
- Feature Dimensionality: The final extracted feature vector contained 1056 features.

### 3.5.3. Summary of Extracted Feature Dimensions

The following table summarizes the feature dimensions extracted from each model:

| Model | Feature Map Size | Flattened Feature Vector Size |
|---|---|---|
| VGG19 | $7 \times 7 \times 512$ | 25,088 |
| DenseNet201 | $7 \times 7 \times 1920$ | 1920 |
| MobileNetV2 | $7 \times 7 \times 1280$ | 1280 |
| NASNetMobile | $7 \times 7 \times 1056$ | 1056 |

Table 2: Feature Dimensions Extracted from Each Model

## 3.6 Decision-Level Fusion Techniques

In this research, decision-level fusion techniques were employed to improve the performance of individual deep learning models for chronic wound classification. Decision-level fusion aggregates predictions from multiple independently trained classifiers. The ensemble strategies used in our work include Majority Voting, Weighted Soft Voting, Maximum Likelihood Fusion, and Stacking with Meta-Model learning. Each method is discussed in detail below.

### 3.6.1. Maximum Likelihood Fusion

This technique assumes that the outputs of each model can be interpreted as likelihoods of class membership. The final decision is made by maximizing the joint likelihood across all models. Formally, for class $c$ and models $M_1, M_2, \ldots, M_n$, the fused decision is:

$$c^* = \arg\max_{c} \prod_{i=1}^{n} P(c|x, M_i)$$

where $P(c|x, M_i)$ is the probability of class $c$ given input $x$ by model $M_i$. This probabilistic aggregation accounts for model confidence and balances uncertainty across outputs.

### 3.6.2. Weighted Soft Voting Fusion

In this approach, the softmax probability distributions from each model are combined using a weighted sum. Higher weights are assigned to models with better performance on validation data. The final predicted class is the one with the highest weighted average probability:

$$P_{fused}(c) \stackrel{n}{=} \sum_{i=1} w_i \cdot P(c|x, M_i) \quad \text{where} \sum w_i = 1$$

This allows stronger models to influence the final prediction more heavily. The weights were empiri- cally selected based on individual validation performance.

### 3.6.3. Hard Voting

Hard voting is the most basic ensemble method where each model votes for a predicted class, and the class receiving the majority of votes is selected as the final prediction:

$$c^* = \text{mode}\{M_1(x), M_2(x), ..., M_n(x)\}$$

This method assumes all models are equally important and does not consider prediction confidence.

It is simple yet effective for reducing overfitting in small datasets.

### 3.6.4. Meta-Learning via Stacking

Stacking is an advanced ensemble technique where predictions of base models are used as input features for a higher-level classifier (meta-model). The process consists of the following steps:

- The dataset is split into training and validation sets.
- Base models (VGG19, DenseNet201, MobileNetV2, NASNetMobile) are trained on the training set.
- Predictions (softmax probabilities or class labels) from each model on the validation set are collected to form a new dataset.
- A meta-model is trained using this new dataset to learn how to best combine the outputs of the base models.

Three types of meta-models were explored:

- Logistic Regression– A linear classifier suitable for learning from model predictions.
- Random Forest – An ensemble of decision trees that effectively captures nonlinear relationships in model outputs.
- XGBoost: The stacking approach with a Random Forest meta-learner yielded the highest classification accuracy in our study, demonstrating the power of model-level collaboration in decision making.

#### 3.6.5. Conclusion

The decision fusion strategy significantly enhanced classification performance, overcoming the limitations of individual models. The ensemble system provided robustness against misclassification and improved generalization on unseen chronic wound images. Among all methods, stacking with a Random Forest meta-learner demonstrated the best performance, making it the most effective ensemble strategy for this task.

## 3.7. Classification

The fused feature vectors were passed through a dense neural network for final classification. The classifier consisted of:

- Input layer with the fused feature vector
- Two hidden layers with 512 and 256 neurons using ReLU activation
- Dropout layers (rate = 0.5) to prevent overfitting
- Output layer with softmax activation for multi-class classification

## 3.8. Algorithm: Chronic Wound Classification System

Chronic Wound Detection and Classification using Decision Fusion

---

Chronic wound image dataset containing 6 classes: Background, Diabetic, Venous, Pressure, Surgical, and Natural. Class label prediction for each input image.

Input: Raw image dataset with class labels.

Output: Predicted wound class for each test image.

Step 1: Data Preprocessing:

Resize all input images to 224 × 224 × 3 to match model input requirements. Apply data augmentation techniques:

- Random rotations

- Horizontal and vertical flips
- Random zoom
- Brightness and gamma adjustments
- Gaussian blur and noise addition
- Organize dataset into training and testing folders by class.

Step 2: Load Pre-trained Models

- Load VGG19, DenseNet201, MobileNetV2, and NASNetMobile with ImageNet weights.
- Remove the fully connected (top) layers.
- Freeze base model layers to prevent overfitting on small dataset.

Step 3: Transfer learning:

- Custom classification layer is added in the last for the 6 classes
- Images in the dataset are fed into each of the 4 pre-trained models.
- Train model with training data over multiple epochs (e.g., 20-30) and using categorical cross-entropy loss and Adam optimizer ($lr = 0.0001$)..
- Apply callbacks like EarlyStopping and ReduceLROnPlateau to prevent overfitting

  Fine tuning:

  Some pre-trained layers are unfrozen and allowed to change with a low learning rate for better results

  Finally, results are evaluated and the models are saved in .keras format.

- VGG19: output shape $7 \times 7 \times 512 \rightarrow$ flattened to 25,088
- DenseNet201: GAP output $\rightarrow$ 1920 features
- MobileNetV2: GAP output $\rightarrow$ 1280 features
- NASNetMobile: GAP output $\rightarrow$ 1056 features

Step 4: Decision level Fusion

- Load the trained models
- Individually run them on the test dataset
- Combine the output probabilities
- $R_{fusion} = [R_{VGG19}, R_{MobileNetV2}, R_{NASNetMobile}]$
- Use that fused result to perform Decision level fusion:
  - Weighted fusion
  - Max likelihood

  - Hard voting
  - Training a Meta-classifier (XGBoost, RandomForest, Logistic Regression)

Step 5: Model Evaluation

- Use the proposed approaches to test the models on all 6 classes (D,P,V,S,N,BG)
- Evaluate trained model on the test set.
- Compute metrics:
  - Accuracy
  - Precision
  - Recall
  - F1-Score
  - Confusion Matrix
- Compare individual model performances vs. fused model.
- Return: Predicted class labels for test images with evaluation metrics.

---

### 3.9. System Workflow

The following flowchart (Figure 1) outlines the step-by-step workflow of our chronic wound classification system using decision level fusion and deep learning.

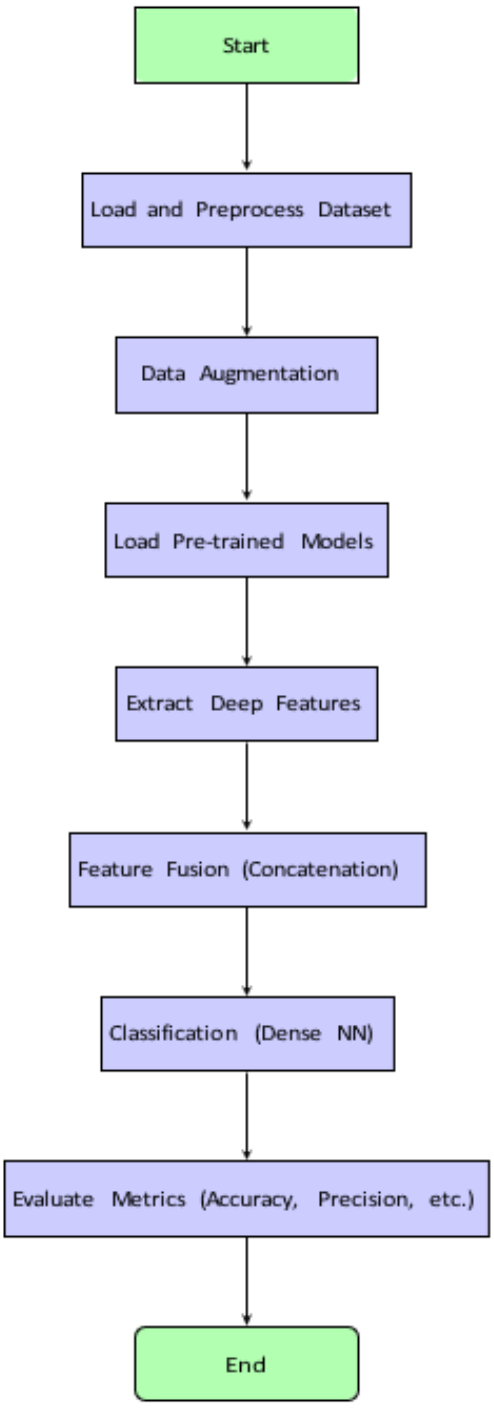


Figure 1: Workflow of Chronic Wound Classification System

## 3.10. Decision level fusion Architecture

Figure 2 illustrates the architecture used for feature fusion. Features are extracted from the final convolutional layers of VGG19, DenseNet201, MobileNetV2, and NASNetMobile. These are concatenated and passed through a dense neural network for final classification.

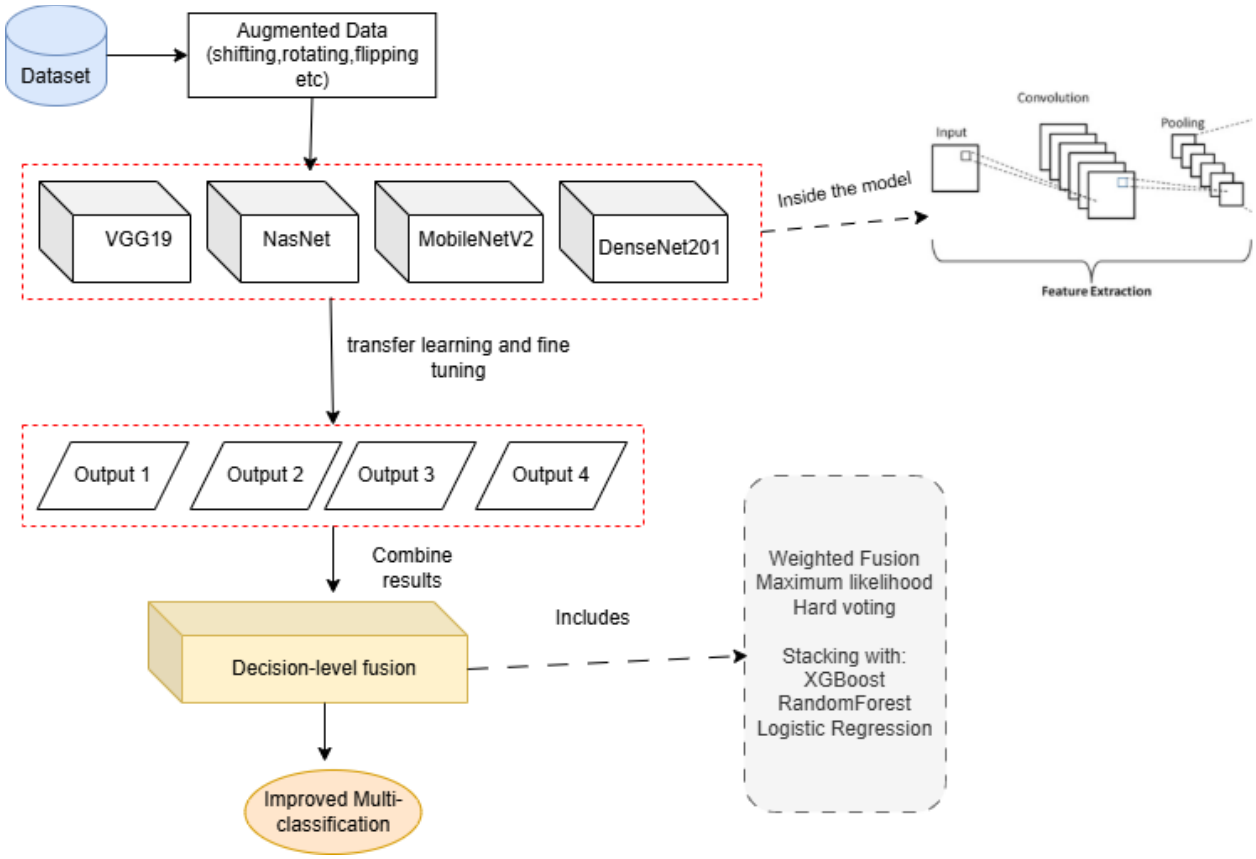


Figure 2: Architecture of the decision level Fusion-based Classification System

## 3.11. Summary of Chapter

This research demonstrated the effectiveness of Decision fusion in chronic wound classification. By combining the strengths of multiple pre-trained models, we achieved a significant increase in classification accuracy. Future work may involve extending this method to other medical image classification tasks.

# Chapter 4

# RESULT AND EVALUATION

## 4.1. Overview of chapter

*This chapter presents the experimental results performed by different models and our dataset. These experiments are being generated and evaluated based on proposed approach which has been discussed in chapter 3. Section 4.2 will briefly put a light on the experimental setup. The complete results and discussion will be carried out in section 4.3 and section 4.4 respectively.*

## 4.2. Experimental Setup

Experimental studies were conducted on chronic wound images obtained from the AZH dataset, which includes six wound classes: diabetic, pressure, venous, surgical, and others. The dataset was split into two parts: 80% of the data was used for training, while 20% was used for testing. To achieve better results, existing data was augmented before training and results from multiple were combined using decision level fusion. Training parameters were selected carefully for all the models used in this study. Several performance parameters were also used to assess the models described in sub-section 4.2.8.

### 4.2.1. System Configuration

In this research work, the experiments were performed on Intel® core ™ i9-10850K at CPU speed 3.6 GHz with 64 GB RAM including an NVIDIA 3060 GPU with 12 GB memory. Operating System was Windows 10 Pro and system type is 64-bit operating system. The summary of experimental configuration is shown in Table 4.1.

**Table 4.1: System Configuration**

| | |
|---|---|
| OS | Windows |
| CPU | Intel® core ™ i9-10850K @ 3.6 GHz, 10 cores, 20 logical processors |
| GPU | NVIDIA GeForce RTX -3060 |
| IDE | VS code |
| Keras | 3.9.2 |
| Python | 3.12.7 |

### 4.2.2. Model Parameter Optimization

Activation Functions

Activation functions play a crucial role in deep neural networks by introducing non-linearity and helping the model learn complex patterns. In this study:

- ReLU (Rectified Linear Unit) activation was used for all hidden layers due to its simplicity, computational efficiency, and ability to mitigate the vanishing gradient problem.
  The ReLU function is defined as:

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

- Softmax activation was applied in the output layer to perform multi-class classification. Softmax normalizes the output values into a probability distribution across the six wound classes.
  The Softmax function is defined as:

$$e^y / sum(e^y)$$

Cost Function

The model was trained using Categorical Cross-Entropy as the loss function, which is well-suited for multi-class classification tasks. It measures the dissimilarity between the true label distribution and the predicted distribution.
The Categorical Cross-Entropy loss is given by:

$$\text{Loss} = \sum p(x\ i{=}1\ i) \log (qi)$$

where p(xi) is the true probability distribution and q(xi) is the predicted distribution over C classes.

Optimizer

The optimization algorithm plays a crucial role in minimizing the loss and improving model performance. In this study:

- The Adam optimizer was initially used with a learning rate of 0.001 during the training phase for its adaptive learning rate capabilities and faster convergence.
- After initial training, fine-tuning was performed by unfreezing the top layers of the base model and using a lower learning rate of 1e-5 to prevent large weight updates that could destabilize the pre-trained features.

Additionally, a ReduceLROnPlateau callback was implemented to automatically reduce the learning rate by a factor of 0.5 if the validation loss did not improve for three consecutive epochs, with a minimum learning rate of 1e-6.

Regularization and Training Strategies

To further enhance generalization and prevent overfitting:

- Batch Normalization layers were added after dense layers to stabilize and accelerate training.
- Dropout with a rate of 0.7 was applied after dense layers to randomly deactivate neurons during training.
- L2 regularization (with a penalty term of 0.01) was introduced in dense layers to penalize large weights and improve generalization.
- Mixed precision training (using mixed_float16 policy) was employed to speed up training and reduce memory consumption without compromising accuracy.

Early stopping based on validation loss with a patience of 5 epochs and model checkpointing were also utilized to save the best model weights during training

### 4.2.3. Training the pre-trained VGG19 model

The training and testing datasets were loaded using TensorFlow's ImageDataGenerator, with all images resized to 224 × 224 pixels and normalized to [0, 1] range.
The model was built using TensorFlow Keras, based on a pretrained VGG19 model with added custom dense, batch normalization, and dropout layers.

Categorical Cross-Entropy was used as the loss function, and Adam was selected as the optimizer with an initial learning rate of 0.001.
Training was performed with a batch size of 32 for 20 epochs, followed by fine-tuning (unfreezing top layers) with a reduced learning rate of 1e-5 for 10 additional epochs.
The model was trained using model.fit() with callbacks including EarlyStopping, ModelCheckpoint, and ReduceLROnPlateau for optimal performance and early convergence.

### 4.2.4. Training the pre-trained DenseNet201 Model

The training and validation datasets were loaded using ImageDataGenerator with normalization (rescaling pixel values to [0,1]). Images were resized to 224 × 224 pixels to match the DenseNet201 input requirements.

A model based on DenseNet201 pretrained on ImageNet was built by adding a GlobalAveragePooling2D layer, a Dropout layer (0.5), and a final Dense softmax layer for classification across six classes.
The base DenseNet201 layers were frozen to retain pretrained features.

The model was compiled using the Adam optimizer with a learning rate of 0.0001, and categorical cross-entropy was used as the loss function.
Training was performed for 10 epochs using model.fit(), and the best model was saved after training completion.

### 4.2.5. Training the pre-trained MobileNetV2 Model

The training and testing datasets were loaded using ImageDataGenerator, with normalization (rescaling pixel values to [0,1]) and resizing images to 224 × 224 pixels.

A model was built using a MobileNetV2 pretrained on ImageNet, where a GlobalAveragePooling2D layer, Batch Normalization, Dropout (0.5), and Dense layers with L2 regularization were added for classification across six classes.
The base model layers were frozen during initial training to retain learned features.

The model was compiled using the Adam optimizer with a learning rate of 0.001 and categorical cross-entropy as the loss function.
Training was performed for 20 epochs with model.fit(), utilizing EarlyStopping, ModelCheckpoint, and ReduceLROnPlateau callbacks to monitor validation loss, save the best model, and dynamically adjust the learning rate.

### 4.2.6. Training the pre-trained NASNetMobile Model

Images from the training and testing datasets were resized to 224 × 224 pixels and normalized using ImageDataGenerator.

The model was constructed using a NASNetMobile base pretrained on ImageNet, with added layers including GlobalAveragePooling2D, a Dense layer with L2 regularization, Dropout (0.5), and a final softmax output layer for six-class classification. The base model layers were frozen to retain pretrained features.

The model was compiled using the Adam optimizer with a learning rate of 0.0001, and categorical cross-entropy was used as the loss function.
Training was performed for 10 epochs using model.fit(), with callbacks including EarlyStopping, ModelCheckpoint, and ReduceLROnPlateau to monitor validation loss, prevent overfitting, and adjust learning rate dynamically.

### 4.2.7. Decision Level Fusion

To further enhance classification performance, decision-level fusion techniques were applied by combining the outputs of individual models (VGG19, DenseNet201, MobileNetV2, and NasNetMobile).

The following fusion strategies were employed:

- Weighted Soft Voting:

    - The output class probabilities from the individual models were combined using weighted averaging, based on their individual validation accuracies.
    - Weights used: 0.5 for VGG19, and 0.25 each for MobileNetV2 and NasNetMobile.
- Maximum Likelihood Fusion:
    - For each sample, the maximum confidence score across models was selected for each class, and the class with the highest maximum probability was assigned.
- Hard Voting:
    - Predicted class labels from each model were collected, and the majority vote determined the final class for each test sample.
- Stacking Ensemble:
    - The concatenated output probabilities from the base models were used as feature vectors.
    - A train-test split (70% training, 30% validation) was used for training the meta-classifiers.
    - Three meta-classifiers were explored:
        - XGBoost Classifier with max_depth=3, n_estimators=100, and learning_rate=0.1.
        - Random Forest Classifier with n_estimators=100 and max_depth=10.
        - Logistic Regression with multi_class='multinomial' and max_iter=1000.
- All fusion experiments used a batch size of 32 and images resized to 224×224 pixels with rescaling (1/255 normalization) during testing.

These ensemble strategies aimed to leverage the complementary strengths of individual CNNs to achieve improved accuracy, generalization, and robustness for chronic wound classification.

### 4.2.8. Evaluation Metrics

The performance of the individual CNN models (VGG19, DenseNet201, MobileNetV2, and NasNetMobile) and the ensemble fusion methods (weighted soft voting, hard voting, maximum likelihood fusion, and stacking) was evaluated using standard classification metrics: Accuracy, Precision, Recall (Sensitivity), Specificity, and F1-score. These metrics were chosen to provide a comprehensive evaluation of the models across all six chronic wound classes.

- Accuracy (Acc):
  Accuracy represents the proportion of correctly classified samples among the total number of test instances. In a multi-class classification problem, it is calculated as:

$$Acc = \frac{1}{N}\sum_{c=1}^{N}\frac{TPc + TNc}{TPc + TNc + FPc + FNc}$$

- Precision:
  Precision measures the proportion of correctly predicted positive samples out of all predicted positives. It is defined as:

$$PRE = \frac{TP}{TP+FP}$$

- Recall (Sensitivity):
  Recall indicates how many actual positive cases were correctly identified. It is calculated as:

$$REC = \frac{TP}{TP + FN}$$

- F1-score:
  The F1-score provides a harmonic mean of Precision and Recall, offering a balanced evaluation metric especially useful in cases of class imbalance:

$$F1 - score = 2 * \frac{PRE * REC}{PRE + REC}$$

These metrics were computed using the sklearn.metrics module for all individual models and fusion strategies, with the results reported in Section 4.3.

---

## 4.3. Results

The results below were evaluated using the augmented dataset with 6 classes. The training set with around 1500 images per class and testing set with around 500 images per class. In sub-sections, results of the experiments will be thoroughly presented in the form of tables. confusion matrix, ROC curve.

### 4.3.1. VGG19 model

Table 4.2 describes how the model achieved an overall accuracy of 80%, demonstrating strong performance across most classes. It performed particularly well on BG, N, and D with F1-scores above 0.95, indicating high precision and recall. Moderate performance was observed for S and V, while P showed relatively lower accuracy, suggesting potential confusion with other classes or data imbalance. Overall, the macro and weighted averages of 0.80–0.82 confirm the model's

balanced and consistent classification ability across all six wound types. The confusion matrix in Fig 4.1 and ROC curve in Fig 4.2 gives class specific details.

**Table 4.2: Performance values of Vgg19**

| Class | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| BG | 0.99 | 0.98 | 0.98 | 475 |
| D | 0.93 | 0.64 | 0.76 | 414 |
| N | 0.95 | 0.99 | 0.97 | 475 |
| P | 0.54 | 0.59 | 0.56 | 442 |
| S | 0.78 | 0.74 | 0.76 | 420 |
| V | 0.69 | 0.84 | 0.76 | 434 |
| Accuracy | | | | |
| Macro Avg | – | – | 0.80 | 2660 |
| Weighted Avg | 0.81 | 0.80 | 0.80 | 2660 |

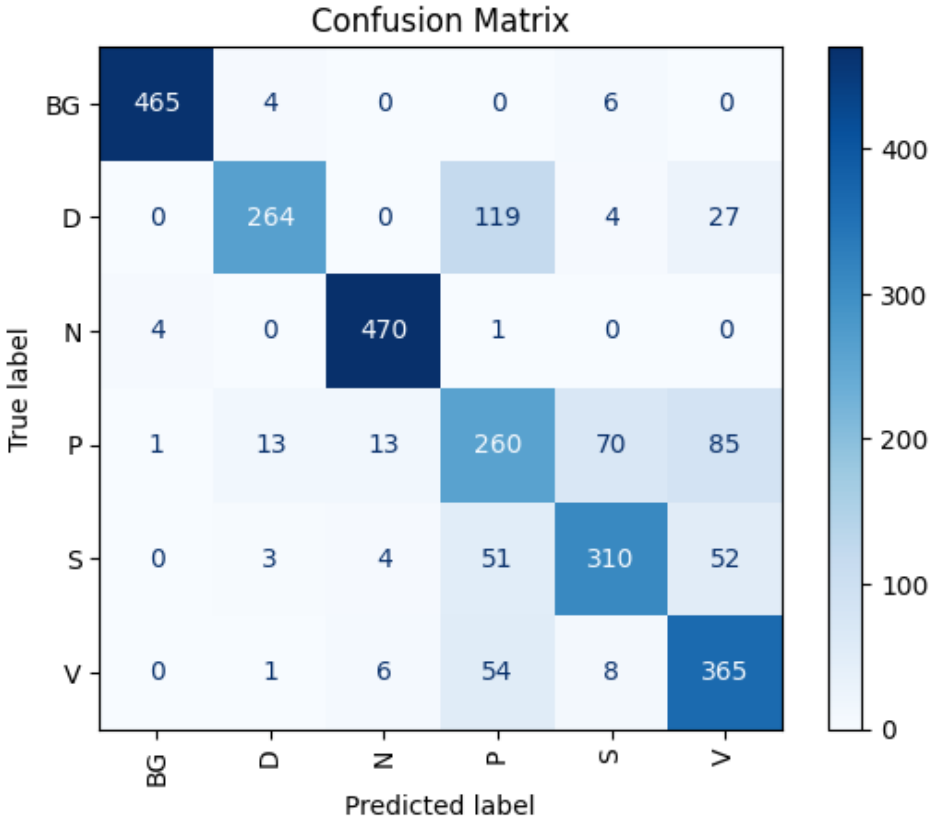


**Figure 4.1: Confusion matrix of VGG19**

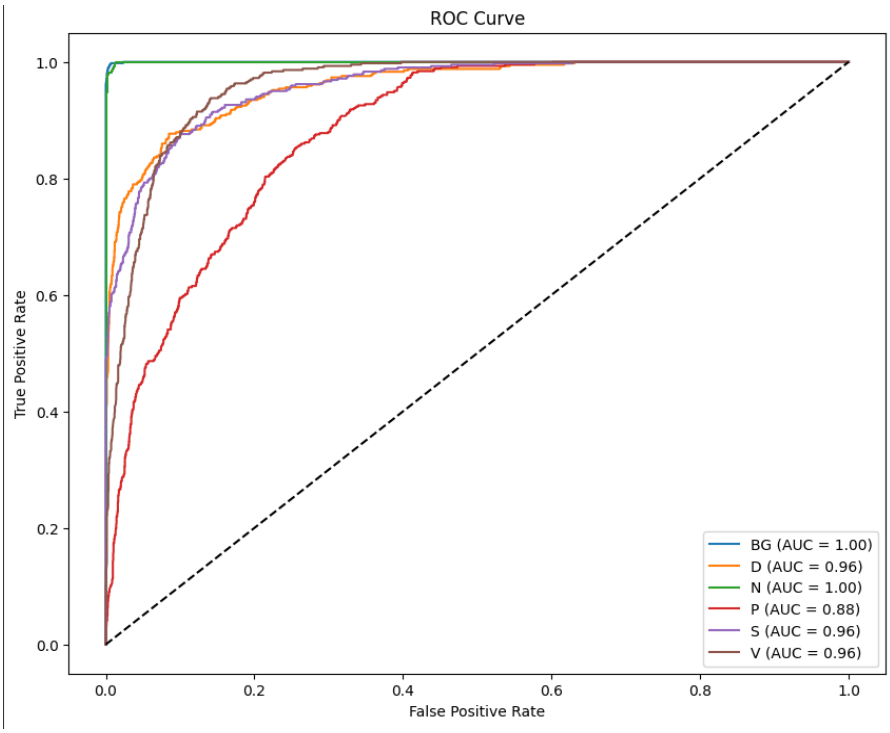


**Figure 4.2: ROC curve of VGG19**

### 4.3.2. DenseNet201

Table 4.3 describes how the model achieved an overall accuracy of 72%, showing relatively strong performance for classes BG, N, and V, with F1-scores above 0.70. The model performed exceptionally well on class N (F1-score: 0.88) and BG (F1-score: 0.94). However, performance on classes P and S was lower (F1-score: 0.50 each), suggesting challenges in predicting these classes accurately. Overall, macro and weighted averages near 0.70 indicate fair but improvable performance across all classes. The confusion matrix in Fig 4.3 and ROC curve in Fig 4.4 gives class specific details.

**Table 4.3: Performance values of DenseNet201**

| Class | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| BG | 0.89 | 0.99 | 0.94 | 475 |
| D | 0.72 | 0.63 | 0.67 | 414 |
| N | 0.79 | 1.00 | 0.88 | 475 |
| P | 0.53 | 0.47 | 0.50 | 442 |
| S | 0.56 | 0.45 | 0.50 | 420 |
| V | 0.71 | 0.74 | 0.72 | 434 |
| Accuracy | – | – | 0.72 | 2660 |
| Macro Avg | 0.70 | 0.71 | 0.70 | 2660 |
| Weighted Avg | 0.71 | 0.72 | 0.71 | 2660 |

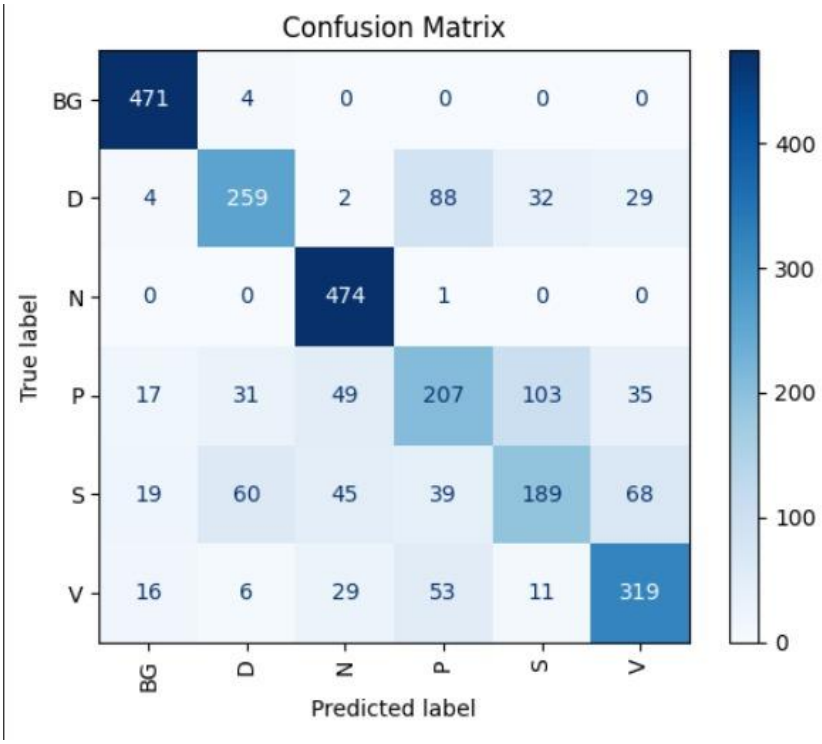


**Figure 4.3: Confusion matrix of DenseNet201**

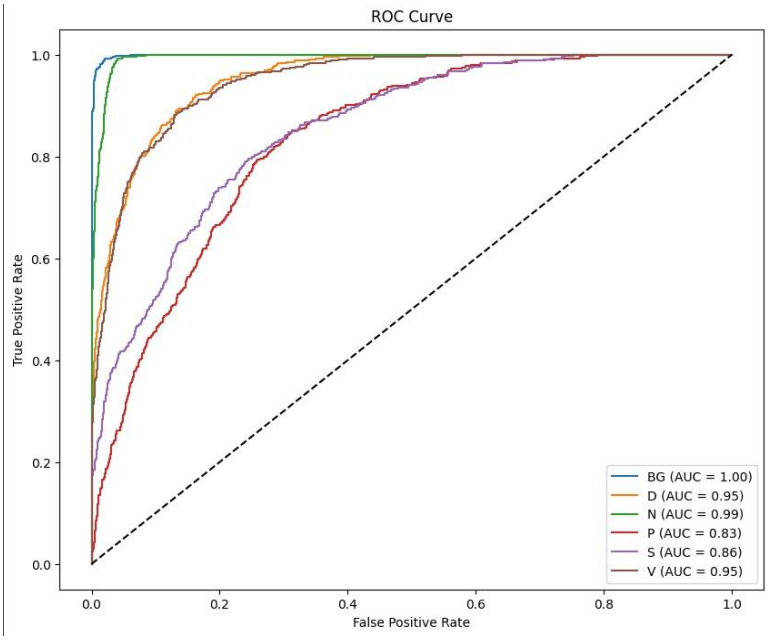


**Figure 4.4: ROC curve of DenseNet201**

### 4.3.3. MobileNetV2

Table 4.4 describes how the model achieved an overall accuracy of 72%, showing strong performance for BG and N classes with F1-scores above 0.90, reflecting reliable classification. Moderate performance was observed for D, S, and V, each scoring between 0.61–0.68. The P class remained the most challenging to classify, with a lower F1-score of 0.47. The macro and weighted averages of 0.71–0.73 suggest a balanced and stable performance across the six chronic wound categories. The confusion matrix in Fig 4.5 and ROC curve in Fig 4.6 gives class specific details.

**Table 4.4: Performance values of MobileNetV2**

| Class | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| BG | 0.90 | 0.98 | 0.94 | 475 |
| D | 0.77 | 0.61 | 0.68 | 414 |
| N | 0.88 | 0.92 | 0.90 | 475 |
| P | 0.48 | 0.46 | 0.47 | 442 |
| S | 0.73 | 0.52 | 0.61 | 420 |
| V | 0.58 | 0.81 | 0.68 | 434 |
| Accuracy | – | – | 0.72 | 2660 |
| Macro Avg | 0.73 | 0.72 | 0.71 | 2660 |
| Weighted Avg | 0.73 | 0.72 | 0.72 | 2660 |

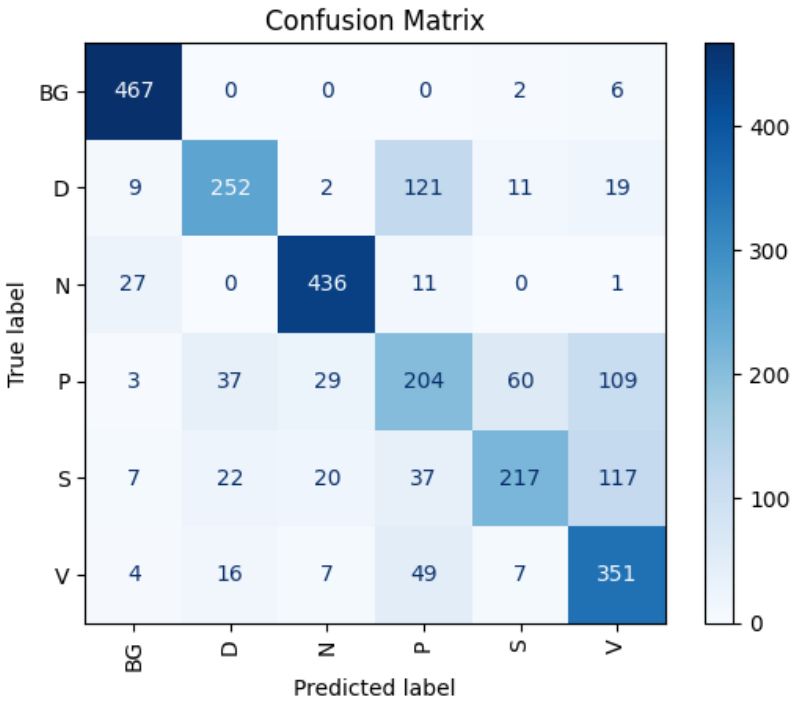

**Figure 4.5: Confusion matrix of DenseNet201**

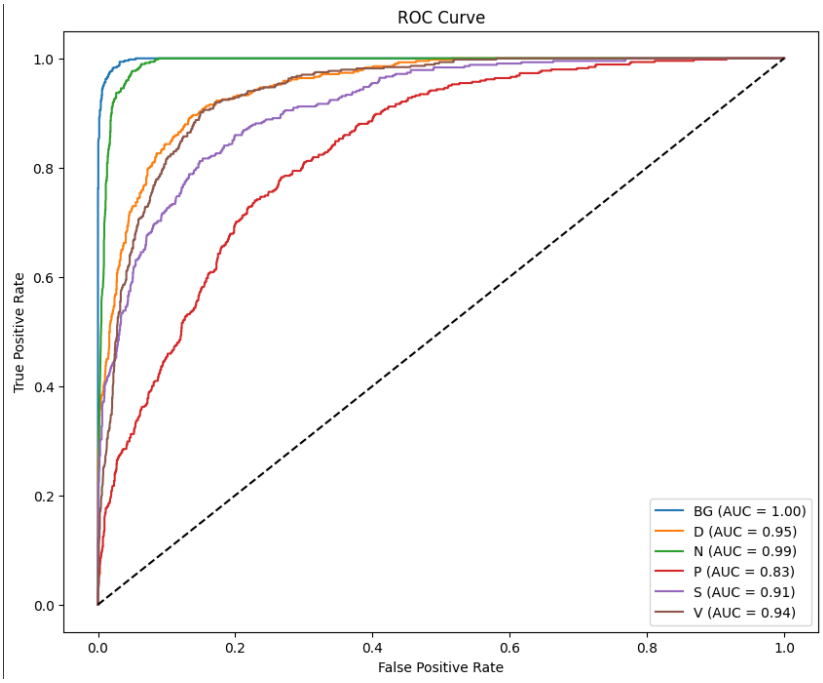


**Figure 4.6: Confusion matrix of DenseNet201**

### 4.3.4. NasNet

Table 4.5 describes how the model achieved an overall accuracy of 74%, with a balanced macro and weighted F1-score of 0.74, indicating consistent performance across all six wound classes. It performed exceptionally well on class BG (F1-score: 0.96) and N (0.91), showing high precision and recall. Moderate classification performance was seen for classes D, S, and V, with F1-scores ranging from 0.62 to 0.73. The model struggled most with class P, which had the lowest F1-score of 0.53, suggesting it may require additional training data or improved feature extraction for better recognition. Overall, the model demonstrates reliable generalization with some room for improvement in minority or complex classes. The confusion matrix in Fig 4.7 and ROC curve in Fig 4.8 gives class specific details.

**Table 4.5: Performance values of NasNet**

| Class | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| BG | 0.96 | 0.96 | 0.96 | 475 |
| D | 0.69 | 0.66 | 0.67 | 414 |
| N | 0.88 | 0.93 | 0.91 | 475 |
| P | 0.56 | 0.50 | 0.53 | 442 |
| S | 0.59 | 0.66 | 0.62 | 420 |
| V | 0.73 | 0.72 | 0.73 | 434 |
| Accuracy | – | – | 0.74 | 2660 |
| Macro Avg | 0.74 | 0.74 | 0.74 | 2660 |
| Weighted Avg | 0.74 | 0.74 | 0.74 | 2660 |

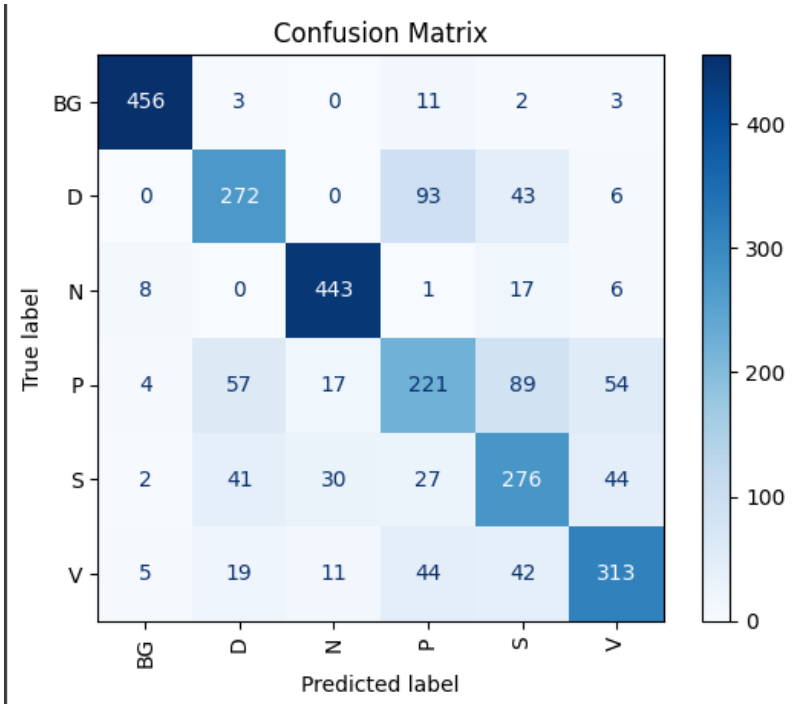


**Figure 4.7: Confusion matrix of NasNet**

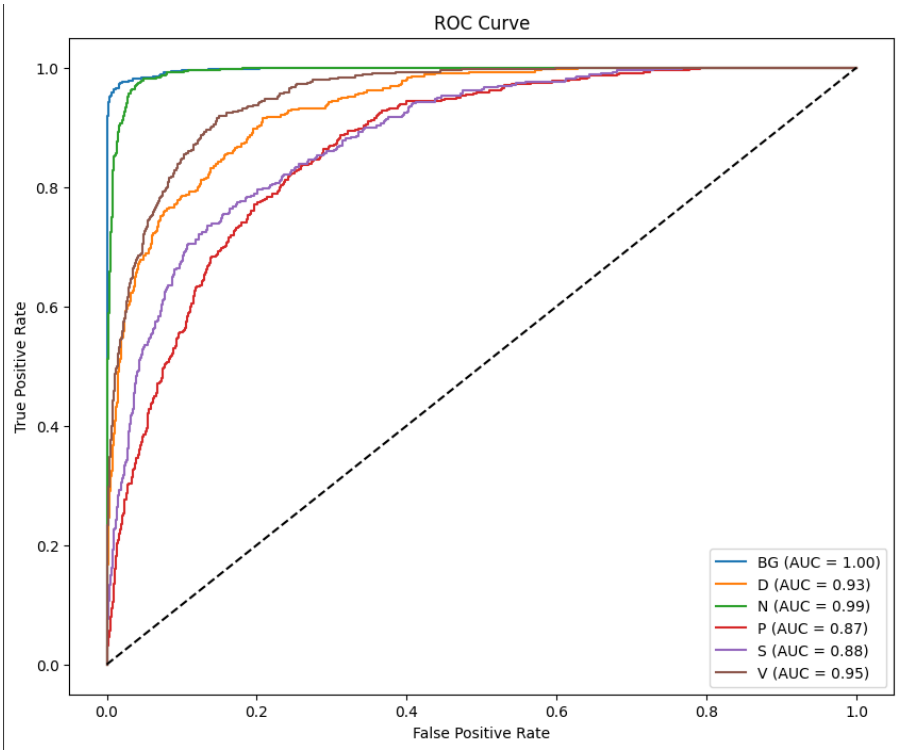


**Figure 4.8: Confusion matrix of NasNet**

### 4.3.5. Decision level Fusion Results

Following are the results by using the different proposed fusion method using VGG19, MobileNet and NasNet.

**Table 4.6: Performance values of Weighted fusion**

| Class | Precision | recall | f1-score | support |
|---|---|---|---|---|
| BG | 0.99 | 1.00 | 0.99 | 475 |
| D | 0.93 | 0.75 | 0.83 | 414 |
| N | 0.93 | 0.99 | 0.96 | 475 |
| P | 0.63 | 0.58 | 0.60 | 442 |
| S | 0.81 | 0.75 | 0.78 | 420 |
| V | 0.70 | 0.88 | 0.78 | 434 |
| accuracy | - | - | 0.83 | 2660 |
| macro avg | 0.83 | 0.82 | 0.82 | 2660 |
| weighted avg | 0.83 | 0.83 | 0.83 | 2660 |

**Table 4.7: Performance values of maximum likelihood**

| Class | precision | recall | f1-score | support |
|---|---|---|---|---|
| BG | 0.98 | 1.00 | 0.99 | 475 |
| D | 0.86 | 0.76 | 0.81 | 414 |
| N | 0.91 | 0.99 | 0.95 | 475 |
| P | 0.64 | 0.48 | 0.55 | 442 |
| S | 0.72 | 0.72 | 0.72 | 420 |
| V | 0.70 | 0.88 | 0.78 | 434 |
| accuracy | - | - | 0.81 | 2660 |
| macro avg | 0.80 | 0.80 | 0.80 | 2660 |
| weighted avg | 0.81 | 0.81 | 0.80 | 2660 |

**Table 4.8: Performance values of hard voting**

| Class | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| BG | 0.96 | 1.00 | 0.98 | 475 |
| D | 0.77 | 0.80 | 0.79 | 414 |
| N | 0.88 | 0.99 | 0.93 | 475 |
| P | 0.65 | 0.48 | 0.55 | 442 |
| S | 0.78 | 0.64 | 0.70 | 420 |
| V | 0.70 | 0.85 | 0.77 | 434 |
| Accuracy | | | 0.80 | 2660 |
| Macro Avg | 0.79 | 0.79 | 0.79 | 2660 |
| Weighted Avg | 0.79 | 0.80 | 0.79 | 2660 |

**Table 4.9: Performance values using XGBoost**

| Class | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| BG | 0.99 | 1.00 | 1.00 | 142 |
| D | 0.91 | 0.81 | 0.86 | 124 |
| N | 0.99 | 1.00 | 1.00 | 143 |
| P | 0.72 | 0.81 | 0.76 | 133 |
| S | 0.95 | 0.84 | 0.89 | 126 |
| V | 0.83 | 0.88 | 0.85 | 130 |
| Accuracy | | | 0.89 | 798 |
| Macro Avg | 0.90 | 0.89 | 0.89 | 798 |
| Weighted Avg | 0.90 | 0.89 | 0.90 | 798 |

**Table 4.10: Performance values using Logistic Regression**

| Class | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| BG | 0.99 | 1.00 | 1.00 | 142 |
| D | 0.87 | 0.80 | 0.83 | 124 |
| N | 0.96 | 1.00 | 0.98 | 143 |
| P | 0.62 | 0.64 | 0.63 | 133 |
| S | 0.86 | 0.79 | 0.83 | 126 |
| V | 0.78 | 0.82 | 0.80 | 130 |
| Accuracy | | | 0.85 | 798 |
| Macro Avg | 0.85 | 0.84 | 0.84 | 798 |
| Weighted Avg | 0.85 | 0.85 | 0.85 | 798 |

**Table 4.11: Performance values of Random forest**

| Class | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| BG | 0.99 | 1.00 | 1.00 | 142 |
| D | 0.95 | 0.81 | 0.81 | 124 |
| N | 0.99 | 1.00 | 1.00 | 143 |
| P | 0.69 | 0.78 | 0.78 | 133 |
| S | 0.94 | 0.83 | 0.83 | 126 |
| V | 0.80 | 0.88 | 0.88 | 130 |
| Accuracy | – | – | 0.89 | 798 |
| Macro Avg | 0.89 | 0.88 | 0.89 | 798 |
| Weighted Avg | 0.90 | 0.89 | 0.89 | 798 |

Fig compares and shows that among all methods, the XGBoost stacking ensemble achieved the best performance with the highest overall accuracy (89.47%) and macro F1-score (0.89), consistently outperforming others across all classes. It was followed closely by Random Forest (88.72%) and Logistic Regression (84.71%), both showing strong performance on difficult classes like P, S, and V. While Weighted Soft Voting performed well (accuracy: 82.96%), it was less effective than stacking methods in minority classes like P (F1: 0.60). Max Likelihood and Hard Voting showed lower performance overall, especially in classes P and S, suggesting that simple fusion techniques may not fully capture the diversity across model outputs.

**Table 4.12: Comparison across all fusion techniques**

| Method | Accuracy | Macro F1 | BG F1 | D F1 | N F1 | P F1 | S F1 | V F1 |
|---|---|---|---|---|---|---|---|---|
| **Weighted Soft Voting** | 0.829699 | 0.823664 | 0.994764 | 0.829987 | 0.959267 | 0.601415 | 0.775713 | 0.780836 |
| **MaxLikelihood** | 0.810526 | 0.799399 | 0.990615 | 0.807198 | 0.946519 | 0.549223 | 0.720764 | 0.782077 |
| **Hard Voting** | 0.800000 | 0.786901 | 0.980392 | 0.786730 | 0.931615 | 0.553966 | 0.702350 | 0.766355 |
| **XGBoost** | 0.894737 | 0.892889 | 0.996491 | 0.859574 | 0.996516 | 0.763251 | 0.890756 | 0.850746 |
| **LogisticReg** | 0.847118 | 0.843356 | 0.996491 | 0.831933 | 0.979452 | 0.627306 | 0.826446 | 0.798507 |
| **Random Forest** | 0.887218 | 0.885969 | 0.996491 | 0.873362 | 0.996516 | 0.732394 | 0.877637 | 0.839416 |

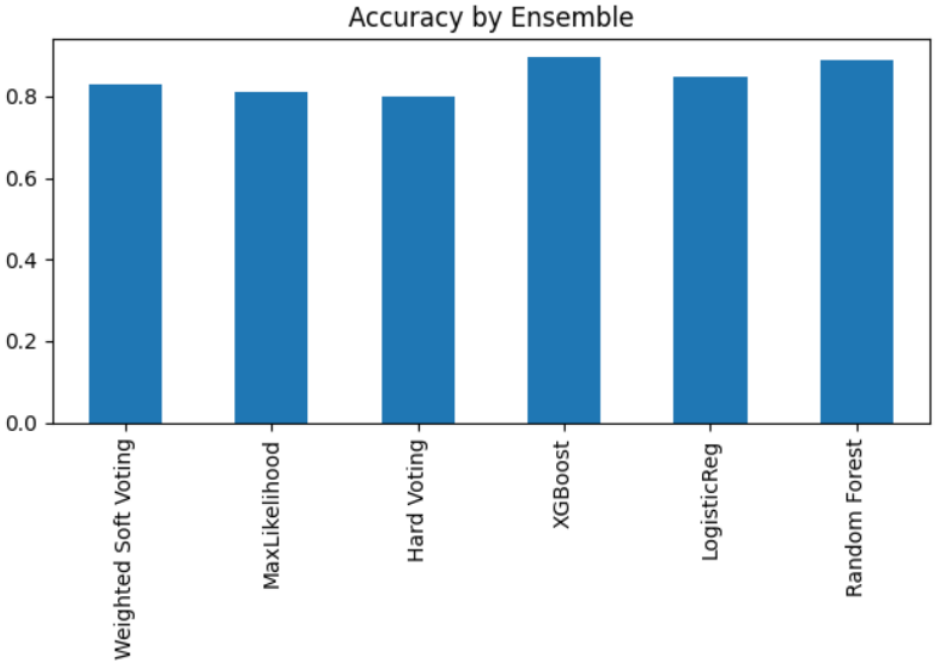


**Figure 4.9: Accuracy across all fusion techniques**

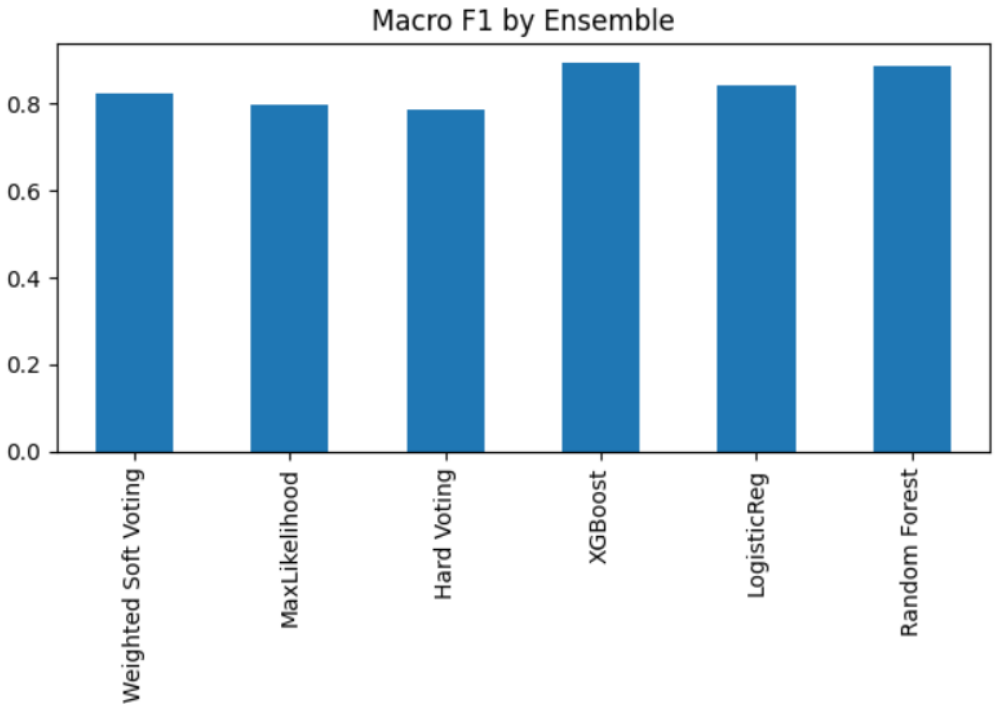


**Figure 4.10: Macro F1 across all fusion techniques**

## 4.4. Discussion

Table 4.13 summaries the performance evaluation of wounds classification using convolution neural network. Several studies have explored the use of deep learning models for chronic wound and diabetic foot ulcer (DFU) classification, with varying success across classification types, datasets, and model architectures. Zaid et al. [2] addressed a relatively complex six-class chronic wound classification task using the AZH dataset and achieved 83% accuracy with EfficientNetB4. This performance is notable given the increased difficulty of multi-class classification, whereas inter-class similarity and data imbalance often hinder model accuracy. In comparison, Behrouz et al. [3] applied an ensemble strategy combining patch-wise and image-wise classification pipelines using AlexNet and a multilayer perceptron, achieving only 68.7% accuracy for a four-class problem on the same dataset. This lower performance may stem from reliance on a relatively shallow network like AlexNet and a more limited fusion strategy, highlighting the importance of both depth and integration design in ensemble methods.

In binary classification, the model's task is inherently simpler, higher accuracies are typically observed. Shuvo et al. [7] focused on DFU detection and applied a feature fusion method using DenseNet201, VGG19, and NASNetMobile. They achieved a respectable 83% accuracy, reflecting strong performance in visual image classification despite variability in lesion presentation. Khairul et al. [6], on the other hand, achieved 100% accuracy using thermal images and decision-level fusion of ShuffleNet and MobileNet. The remarkable performance can be attributed to the

complementary nature of thermal imaging, which captures temperature-based physiological differences. Furthermore, their decision fusion strategy capitalizes on agreement between classifiers to suppress misclassifications, suggesting that modality selection and decision integration are critical factors in DFU detection accuracy.

**Table 4.13: Comparison of the proposed techniques with state-of-the-art studies on wounds classification**

| Authors | Disease | No. of classes | Data Set | Models | Accuracy |
|---|---|---|---|---|---|
| [2] | Chronic wounds | 6(D,V,S,P,N,BG) | AZH | EfficientNetB4 | 83% |
| [3] | Chronic wounds | 4 | AZH | AlexNet | 68.7% |
| [7] | Foot ulcer | Binary | DFU | **Fusion:**<br>Densenet201<br>VGG19<br>NASNetMobile | 83% |
| [6] | Diabetic foot | Binary | Thermogram dataset | **Fusion:**<br>ShuffleNet<br>MobileNet | 100% |
| Our proposed approach | Chronic wounds | 6(D,V,S,P,N,BG) | Augmented AZH | **Individual Results:**<br>VGG19<br>NASNet<br>MobileNetV2<br>DenseNet201<br><br>**Decision-level Fusion:**<br>Weighted<br>Max likelihood<br>Hard voting<br>XGBoost<br>Random Forest<br>Logistic Regression | <br><br>80%<br>74%<br>72%<br>73%<br><br><br><br>82.96%<br>81.05%<br>80%<br>89.47%<br>88.72%<br>84.71% |

Overall, these findings highlight that while binary classification of DFUs may achieve good performance using optimized pipelines and specialized modalities, multiclass chronic wound classification remains a greater challenge due to the visual similarities between wound types and class imbalance issues. Our proposed approach, applied within the more challenging multiclass domain, demonstrates competitive performance and highlights the effectiveness of ensemble architectures and decision-level fusion in addressing such complexities. Moreover, our comparative analysis suggests that researchers should consider not only architectural enhancements, but also factors such as domain-specific imaging (e.g., thermal vs. RGB), dataset quality, and the design of decision fusion strategies to further improve wound classification outcomes.

## 4.5. Summary of Chapter

In this chapter, our proposed approach has been evaluated on AZH dataset. Firstly, the chapter defines the training parameters and evaluation metrics. Also, technologies and the libraries used in this experimental study are stated. The results of all the models and fusion techniques have been described in detail. In the end, a brief comparison has been made based on classification results. The next chapter will conclude the present study and highlight some future directions.

# Chapter 5

# Conclusion and Future Work

## 5.1. Conclusion

This research presents a robust decision-level ensemble learning approach for chronic wound classification using deep learning. Chronic wounds, including diabetic, venous, pressure, and surgical ulcers, pose a significant burden on healthcare systems and require timely, accurate identification for effective treatment. Our study focused on building a reliable AI system capable of classifying wound types into six categories: Background, Diabetic, Pressure, Venous, Surgical, and Natural (healthy skin).

In the initial phase (FYP-I), we fine-tuned and evaluated four pre-trained convolutional neural network architectures—VGG19, DenseNet201, MobileNetV2, and NASNetMobile—on an augmented dataset of over 14,800 images. The following results were obtained:

- VGG19: Achieved an accuracy of 80%
- DenseNet201: Achieved an accuracy of 72%
- MobileNetV2: Achieved an accuracy of 72%
- NASNetMobile: Achieved an accuracy of 74%

Each model exhibited distinct strengths in feature representation due to architectural differences. However, standalone models were limited in capturing complex inter-class variations across wound types.

In FYP-II, we implemented a Decision-Level Fusion strategy. Rather than combining features, we fused the final predictions (softmax outputs) from each of the four trained models using a majority voting scheme. This method improved the system's robustness and reduced the likelihood of model-specific biases or misclassifications.

The decision level approach achieved improved classification accuracies with XGBoost reaching around 89.4%, surpassing the individual models. Evaluation metrics such as precision, recall, and F1-score further demonstrated the reliability and generalizability of the fused model.

This project provides a scalable solution for automated wound classification and can assist healthcare professionals in clinical decision-making, especially in resource- constrained settings.

## 5.2 Future Work

While the current system achieves high accuracy, several directions exist for future enhancements:

- Thermal Image Integration: We plan to extend the dataset by including thermal imaging data to capture physiological cues like inflammation and heat distribution—useful for early wound detection.
- Explainable AI (XAI): The incorporation of explainability techniques such as Grad-CAM or SHAP will help clinicians understand the reasoning behind the model's predictions and foster trust in AI systems.
- Real-Time Deployment: We aim to develop a lightweight version of the model suitable for deployment on mobile devices or cloud-based web applications for on- the-go wound analysis in clinics.
- Larger and Diverse Dataset: Collection of wound images from diverse populations and clinical environments will help the model generalize better and reduce biases.
- Severity Grading and Segmentation: Beyond classification, future iterations may include wound segmentation and severity scoring to support prognosis and treatment planning.
- Research Dissemination: A comprehensive research paper based on this study will be submitted to peer-reviewed journals and conferences in the field of biomedical AI and medical imaging.

Our proposed system sets the groundwork for a clinically usable chronic wound classifier and demonstrates the potential of decision fusion in medical image classification. With continued improvements, this work can be transitioned into a practical diagnostic aid in wound care management.

## References


[1] Nagle, S. M., Stevens, K. A., & Wilbraham, S. C. (2025). Wound assessment. *StatPearls*. ref

[2] Aldoulah, Z. A., Malik, H., & Molyet, R. (2023). A novel fused multi-class deep learning approach for chronic wounds classification. Applied Sciences, 13(21), 11630. ref

[3] Rostami, B., Anisuzzaman, D. M., Wang, C., et al. (2021). Multiclass wound image classification using an ensemble deep CNN-based classifier. Computers in Biology and Medicine, 137, 104536. ref

[4] Zhang, R., Tian, D., Xu, D., Qian, W., & Yao, Y. (2022). A survey of wound image analysis using deep learning: Classification, detection, and segmentation. IEEE Access, 10, 79502–79515. ref

[5] Toofanee, M. S. A., et al. (2023). DFU-SIAM: A novel diabetic foot ulcer classification with deep learning. IEEE Access, 11, 98315–98332. ref

[6] Munadi, K., Saddami, K., Oktiana, M., et al. (2022). A deep learning method for early detection of diabetic foot using decision fusion and thermal images. Applied Sciences, 12(15), 7524. ref

[7] Biswas, S., Mostafiz, R., Uddin, M. S., & Paul, B. K. (2024). XAI-FusionNet: Diabetic foot ulcer detection based on multi-scale feature fusion with explainable artificial intelligence. Heliyon. ref

[8] Alabdulhafith, M., Ba Mahel, A. S., Samee, N. A., Mahmoud, N. F., Talaat, R., Muthanna, M. S. A., & Nassef, T. M. (2024). Automated wound care by employing a reliable U-Net architecture combined with ResNet feature encoders for monitoring chronic wounds. *Frontiers in Medicine, 11*. ref

[9] Scebba, G., Zhang, J., Catanzaro, S., Mihai, C., Distler, O., Berli, M., & Karlen, W. (2022). Detect-and-segment: A deep learning approach to automate wound image segmentation. *Informatics in Medicine Unlocked, 29*, 100884. ref

[10] Wang, C., Anisuzzaman, D. M., Williamson, V., et al. (2020). Fully automatic wound segmentation with deep convolutional neural networks. *Scientific Reports, 10*, 21897. ref

[11] Oota, S. R., Rowtula, V., Mohammed, S., Liu, M., & Gupta, M. (2023). WSNet: Towards an effective method for wound image segmentation. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 3234–3243. ref

[12] Ohura, N., Mitsuno, R., Sakisaka, M., Terabe, Y., Morishige, Y., Uchiyama, A., Okoshi, T., Shinji, I., & Takushima, A. (2019). Convolutional neural networks for wound detection: The role of artificial intelligence in wound care. *Journal of Wound Care, 28*(Suppl 10), S13–S24. ref

[13] Sharma, Y., Ghatak, S., Sen, C.K. *et al.* Emerging technologies in regenerative medicine: The future of wound care and therapy. *J Mol Med* **102**, 1425–1450 (2024). ref

[14] Huang, ST., Chu, YC., Liu, LR. *et al.* Deep Learning-Based Clinical Wound Image Analysis Using a Mask R-CNN Architecture. *J. Med. Biol. Eng.* **43**, 417–426 (2023). ref

[15] Sen, C. K. (2023). Human wound and its burden: Updated 2022 compendium of estimates. *Advances in Wound Care, 12*(12), 657–670. ref

[16] Pereira, C., Guede-Fernández, F., Vigário, R., Coelho, P., Fragata, J., & Londral, A. (2023). Image analysis system for early detection of cardiothoracic surgery wound alterations based on artificial intelligence models. *Applied Sciences, 13*(4), 2120. ref

[17] Buschi, D., Curti, N., Cola, V., Carlini, G., Sala, C., Dall'Olio, D., Castellani, G., Pizzi, E., Del Magno, S., Foglia, A., Giunti, M., Pisoni, L., & Giampieri, E. (2023). Automated wound image segmentation: Transfer learning from human to pet via active semi-supervised learning. *Animals (Basel), 13*(6), 956. ref

[18] University of Wisconsin-Milwaukee Big Data Analytics and Visualization Lab. (2024). *Wound classification using images and locations* [Data set]. GitHub. ref

[19] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*. ref

[20] Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4700-4708). ref

[21] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4510-4520). ref

[22] Zoph, B., Vasudevan, V., Shlens, J., & Le, Q. V. (2018). Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 8697-8710). ref