COMPARATIVE ANALYSIS OF DEEP LEARNING ARCHITECTURES FOR SKIN DISEASE CLASSIFICATION

A Comprehensive Study on Transfer Learning, Hybrid Models, Attention Mechanisms, and Multi-Scale Feature Extraction

---

AUTHOR

Ömer Faruk KURTULUŞ

December 25, 2025

TABLE OF CONTENTS

1. Introduction, Significance and Objective of the Project

1.1. Significance of the Project

Skin diseases are among the most prevalent health problems worldwide. Particularly, conditions such as Acne, Vitiligo, Hyperpigmentation, Nail Psoriasis, and Stevens-Johnson Syndrome (SJS-TEN) can severely diminish patients' quality of life when not diagnosed early, and in some cases (such as SJS-TEN) can pose life-threatening risks. In regions with limited access to dermatology specialists or in busy clinical settings, visual diagnosis of these diseases can be time-consuming and subjective. AI-powered image processing techniques have the potential to provide dermatologists with a low-cost and rapid preliminary diagnostic mechanism to assist in the decision-making process.

1.2. Objective of the Project

The primary objective of this project is to develop models that classify 5 different skin diseases with high accuracy using deep learning techniques. Within the scope of the project, in addition to Transfer Learning models with proven success in the literature, novel Hybrid Models that combine the strengths of different **architectures have been** designed. The goal is not only to achieve high accuracy but also to minimize the risk of missed diagnoses by reducing false negative rates (high recall). The ultimate aim is for the best-performing model to achieve a level of reliability suitable for use in clinical decision support systems.

2. Dataset Description

In this study, the "Skin Disease Classification Dataset" published on the Mendeley Data platform was used. The original dataset is a comprehensive collection gathered for dermatological research.

- **Data Source: Mendeley** Data (Skin Disease Classification Dataset)
- **Categories: The** dataset contains 5 main classes:
    1. **Acne - 410 Images**
    2. **Vitiligo - 721 Images**
    3. **Hyperpigmentation - 250 Images**
    4. **Nail-Psoriasis - 900 Images**
    5. **SJS-TEN (Stevens-Johnson Syndrome) - 1,130 Images**

Data Cleaning and Preparation: Although the original dataset contained a total of 9,548 images, pre-processed artificial images (with superpixel, saturation, and hue filter applied) were removed from the dataset to enhance the reliability of this study. Only raw images were used to ensure the

models were trained with real-world data. As a result of this cleaning process, the study was **conducted on a total of "3,411" h**igh-quality dermatological images.

This distribution indicates a numerical class imbalance in the dataset (e.g., the SJS-TEN class has 1,130 images while Hyperpigmentation has only 250). To prevent this imbalance from adversely affecting model performance, Data Augmentation techniques were applied during training and class weighting strategies were taken into consideration.

3. EDA (Exploratory Data Analysis) Results

In this section, the structural properties of the dataset, class distributions, image dimensions, and color statistics were examined in detail. The analyses performed played a critical role in determining model training strategies.

3.1. Class Distribution and Data Imbalance Analysis

When the distribution of the 5 different disease classes in the dataset was examined, a significant numerical imbalance *was identified.* According to the analysis results:

- **Class with Most Data: SJS-TEN** (1,130 images)
- **Class with Least Data:** Hyperpigmentation (250 images)
- **Imbalance Ratio:** 4.52x

This situation poses the risk of the model making biased decisions toward the majority class (SJS-TEN). To minimize this risk, **Class Weighting was applied *during training and* a**ugmentation coefficients for minority classes were ***dynamically*** adjusted.

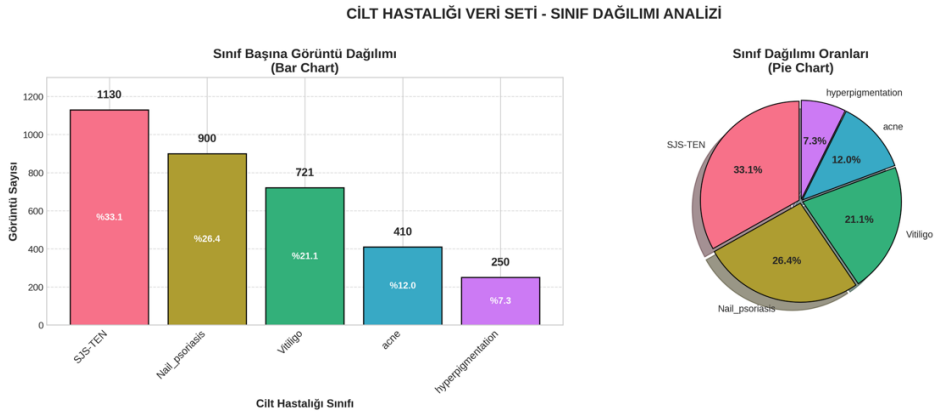

Figure 3.1: *Distribution graph of disease classes in the dataset.*

3.2. Image Dimensions and Statistical Analysis

All 3,411 images in the dataset were analyzed in terms of pixel dimensions (using multi-threading). The obtained statistical data are as follows:

- **Width: Average 279.66** pixels (Min: 194, Max: 494)
- **Height: Average 209.86** pixels (Min: 122, Max: 422)
- **Aspect Ratio: Average** 1.42

The analysis revealed that the images were not of a standard size and contained 10.03% outliers. Since the input layers of deep learning models (EfficientNet, DenseNet, etc.) require fixed-size matrices, all images were resized to **224x224 pixels prior** to training using bilinear/bicubic *interpolation methods that* preserve image quality.

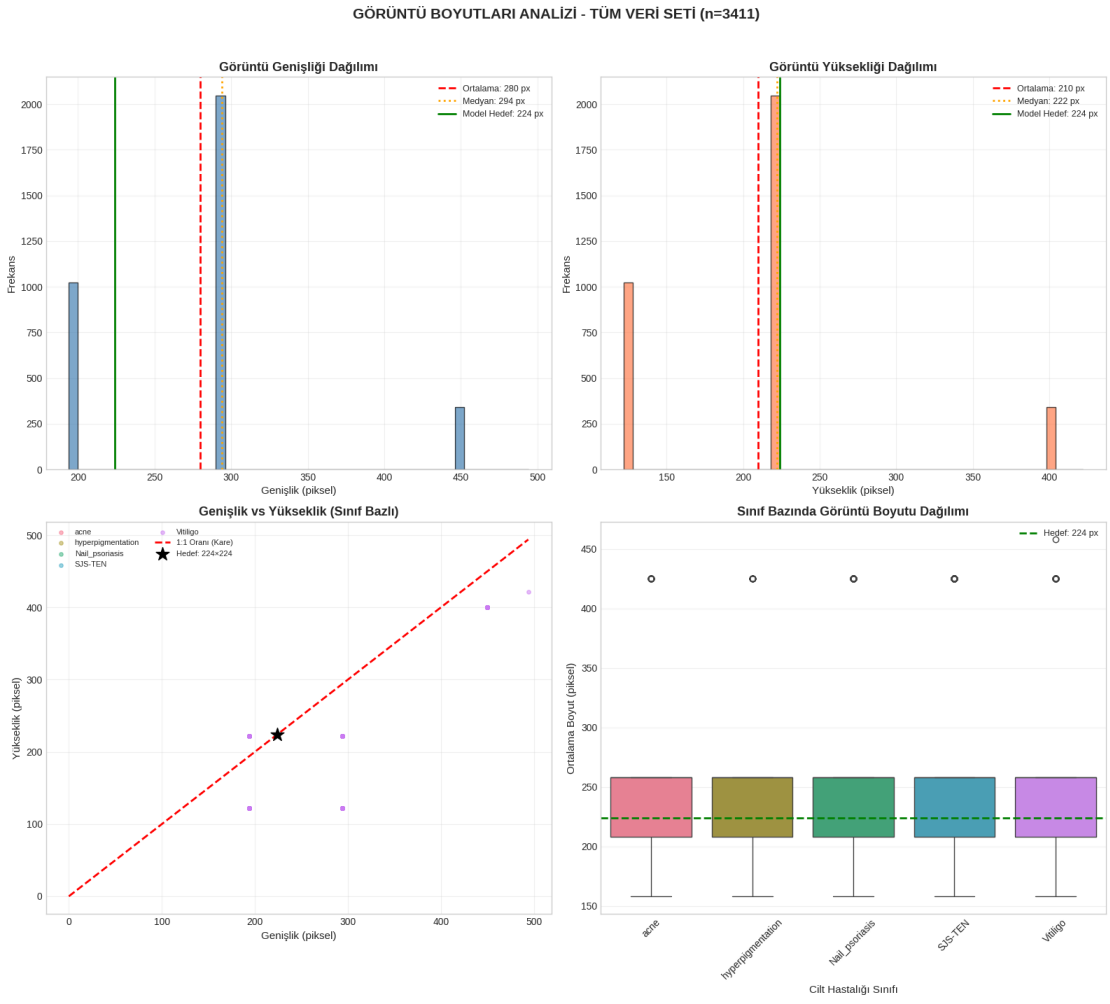

***Figure 3.2: Width** and height distribution of images in the dataset.*

3.3. RGB Color Space and Channel Analysis

Color changes (redness, pigment loss, etc.) are the most important distinguishing features in the diagnosis of skin diseases. According to the RGB (Red-Green-Blue) channel analysis:

- **The SJS-TEN** class has the highest Red **channel value** with an average of 168.97. This is clinically consistent with the disease being characterized by intense inflammation and redness.
- **The Nail-Psoriasis class has** the lowest average **brightness** value at 109.39.
- **The color contrast** differences **between the** Hyperpigmentation and Vitiligo classes will serve as an important feature for the model to distinguish between these diseases.

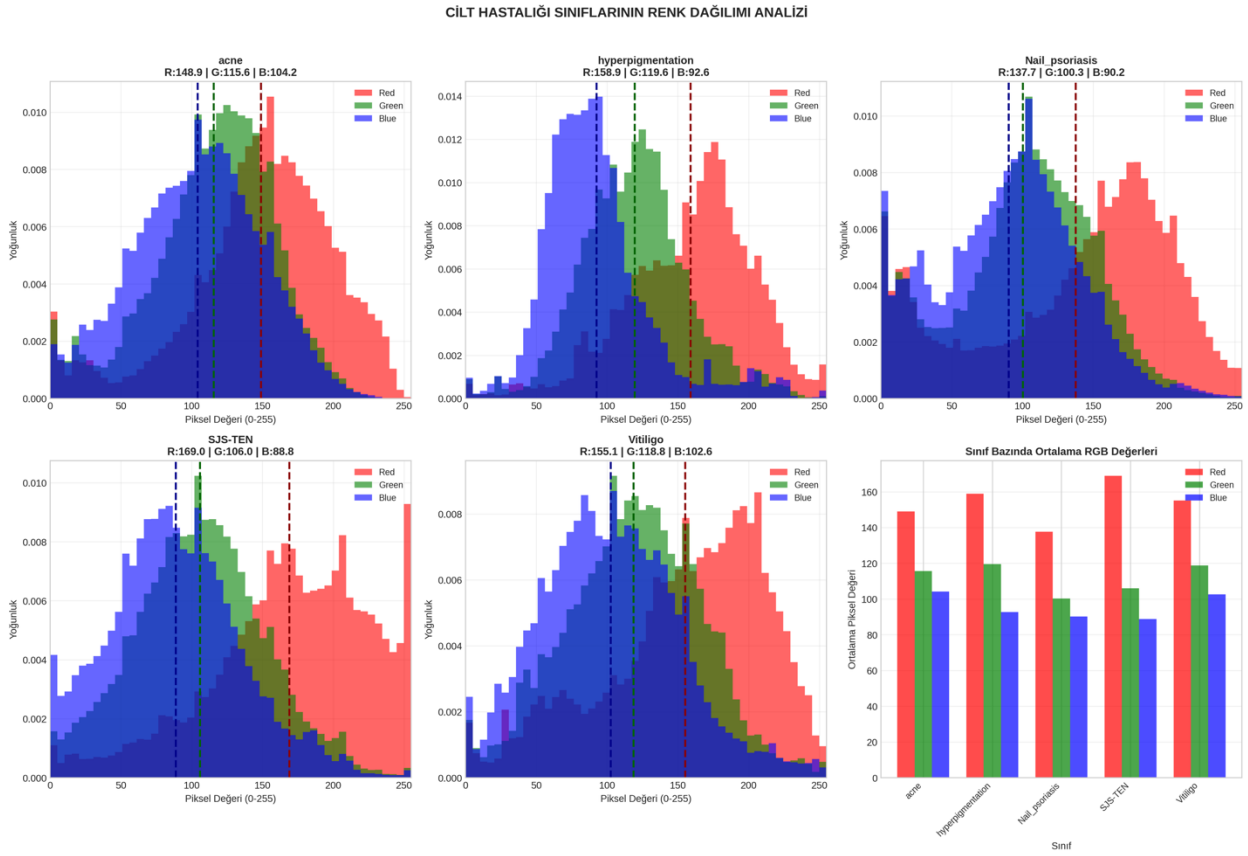

Figure 3.3: *Average intensity distribution of RGB color channels by class.*

3.4. Medical Characteristics of Classes (Qualitative Analysis)

To understand what the model classifies, the visual characteristics of the diseases in the dataset were examined:

1. **Acne: Lesions** caused by clogged hair follicles.
2. **Hyperpigmentation: Dark** spots caused by increased melanin.
3. **Vitiligo: White** patches caused by melanocyte loss (critical for edge detection by the model).
4. **Nail Psoriasis: Deterioration** and pitting in nail structure.
5. **SJS-TEN: Severe** skin rashes and extensive surface wounds (the most complex class challenging the model's texture analysis).

Figure 3.4: *Sample images randomly selected from each disease class.*

4. DATA PREPARATION AND TRAINING STRATEGY

Considering the data imbalance and structural features identified during the exploratory data analysis phase, a training strategy was designed to enhance the model's generalization capability.

This section details data splitting, imbalance management, data augmentation techniques, and training optimization processes.

4.1. Dataset Splitting (Stratified Splitting)

To objectively evaluate the model's performance, the dataset was **divided into three** parts: **Training (70%),** Validation **(15%), and Test** (15%). During the splitting process, Stratified Sampling was used to maintain the distribution across **classes.**

Through this method, the class proportions in each dataset were ensured to match exactly with the original dataset. Validation tests showed that the maximum deviation between the original distribution and the split sets was only **0.11%, meaning** the dataset characteristics were preserved across all partitions.

The numerical distribution of the created datasets is as follows:

- **Training Set: 2,387 Images** (for model learning)
- **Validation Set: 512 Images (for** hyperparameter tuning during training)
- **Test Set: 512** Images (for final model performance)

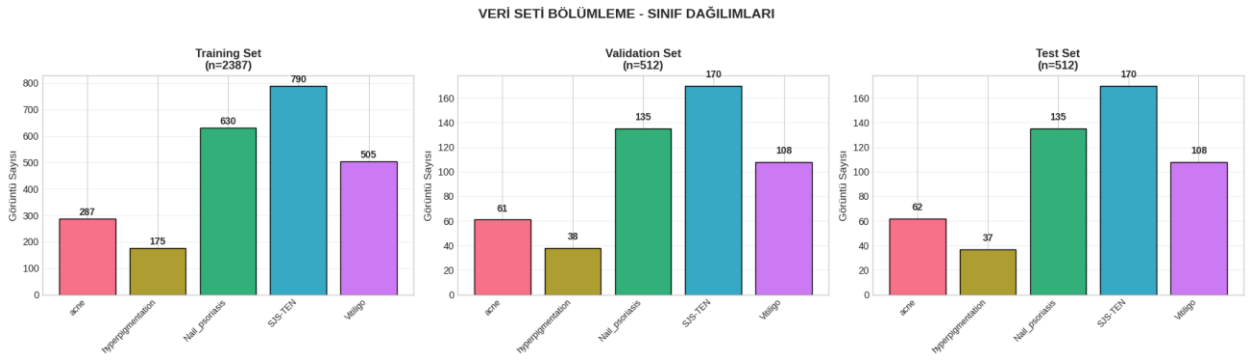

Figure 4.1: *Distribution graph of the dataset into Training, Validation, and Test sets.*

4.2. Imbalanced Data Management and Class Weighting

To compensate for the 4.51x data **imbalance** identified during the EDA phase (SJS-TEN class having excessive samples while Hyperpigmentation class having **few), Class Weighting was applied. In this** method, higher loss penalties were assigned to classes with fewer samples and lower penalties to classes with more samples, preventing the model from neglecting minority classes.

The calculated weight coefficients are as follows:

- **Hyperpigmentation (Least data): 2.7280** (High Priority)
- **Acne:** 1.6634
- **Vitiligo:** 0.9453
- **Nail-Psoriasis:** 0.7578
- **SJS-TEN (Most data): 0.6043** (Low Priority)

During training, the model used these coefficients to optimize the loss function, thereby improving the diagnostic accuracy for numerically underrepresented diseases.

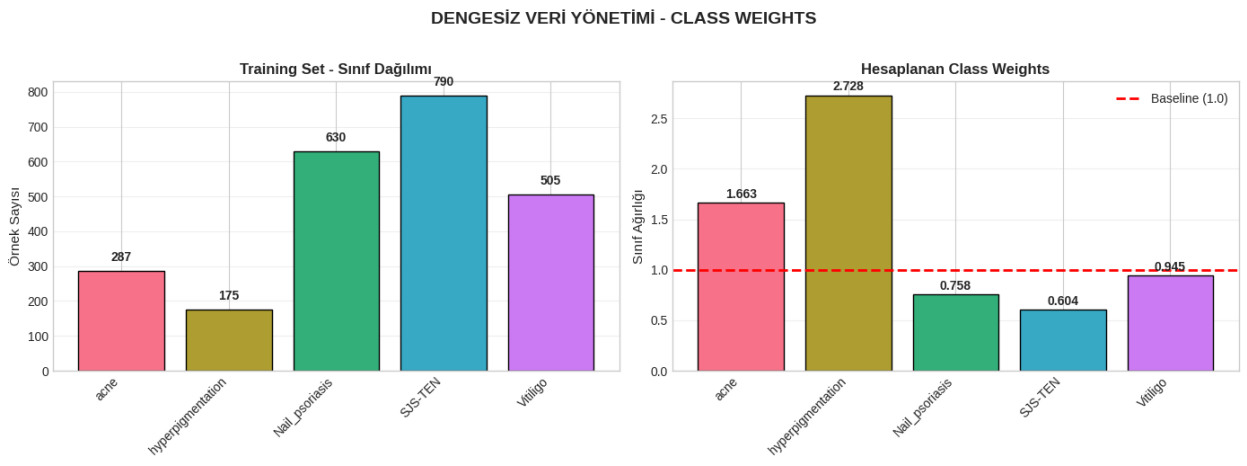

***Figure 4.2: Weight** coefficients calculated inversely proportional to class frequencies.*

4.3. Data Augmentation and Data Pipeline

Data Augmentation was applied to the training set to prevent overfitting in deep learning models and to increase the model's robustness against **different variations. The following** operations were applied in the high-performance data pipeline created using the TensorFlow tf.data API:

1. **Random Rotation: ±10** degrees
2. **Random Zoom:** ±10%
3. **Random Flip: Horizontal** mirroring of images

These operations were **applied only to the** Training Set; the Validation and Test sets were kept in their original (raw) state to ensure the reliability of results.

Important Note (Adaptive Approach): Furthermore, the applied data augmentation strategies were not kept constant for all models. Since each model's architectural depth and feature extraction capacity differ, augmentation types **and parameter values were customized on a per-model basis. To achieve the highest** accuracy rate, iterative experiments were conducted on the parameters during training; rotation angles or zoom ratios were dynamically updated according to model performance to target the optimal configuration.

4.4. Data Leakage Control

One of the most critical steps in terms of academic and technical reliability, ***"Data Leakage"*** control, was performed. The file lists of the training, validation, and test sets were compared, and it was confirmed that the intersection of the sets **was 0** files.

This result proves that the model never saw the test data during training and that the achieved performance results are entirely realistic.

4.5. Training Optimization and Callback Mechanisms

Dynamic Callback functions were defined to optimize the **model's** training process:

1. **EarlyStopping: If the model's** validation accuracy (val_accuracy) does not improve for 15 epochs, training is automatically stopped to prevent overfitting and wasted time.
2. **ModelCheckpoint: Only the** model weights with the "best validation accuracy" are saved throughout training.
3. **ReduceLROnPlateau: When** the model struggles to learn (if loss does not decrease for 5 epochs), the learning rate is halved (factor=0.5) to enable finer learning.

5. OVERALL EXPERIMENTAL RESULTS AND PERFORMANCE COMPARISON / TABLES

Within the scope of the project, a total of 10 different deep learning / transfer learning architectures were trained and tested, ranging from baseline models to complex hybrid structures. All models were evaluated on the same training, validation, and test sets; results were compared across Accuracy, Precision, Recall, F1-Score, Kappa Coefficient, and ROC-AUC metrics.

The data presented in the tables below show the final performance of the models on the test set.

Important Note: The tables showing the results of all models are presented below; individual model details will be discussed separately in Section "6."

5.1. Performance Indicators and Color Codes

To enable clearer interpretation of the analysis tables, results have been categorized and color-coded:

Green (Best Performance): Represents the model with the highest performance on the test set (Eff**icientNetV2-S -** 99.61%).

Yellow (Original Architecture): Represents the original architecture developed from ***scratch by Ömer Faruk*** Kurtuluş without using any pre-built networks from the **literature** (SalvationNet*).

Red (Baseline Model): Represents the simple CNN model trained without transfer learning (Custom CNN - **32.23%).**

Important Note: The asterisk (*) **next to the** SalvationNet* name in the tables indicates that this model did not use any pre-trained weights and is an original design developed by the researcher.

5.2. Comparative Analysis of Models (Table 1)

Table 1 presents the basic performance metrics of all developed models and the average time spent per epoch (training cycle) in seconds.

The most notable finding here is that while transfer learning-based models (EfficientNet, DenseNet, ConvNeXt, etc.) achieved over 98% accuracy, the simple CNN model trained from scratch (Baseline) remained at 32%. This demonstrates the necessity of transfer learning for dermatological images. Additionally, **the DenseNet121 + InceptionV3** hybrid model achieving 99.41% accuracy in just 5.65 seconds is a critical finding in terms of speed/performance balance.

Table 1: Model Comparison - Performance Metrics

Method	Precision	Recall	F1-Score	Accuracy	Avg time for an epoch (s)
Custom CNN (Baseline)	0.2	0.32	0.22	0.3223	10.77
EfficientNetV2-S	1.0	1.0	1.0	0.9961	8.17
ConvNeXt-Tiny	0.99	0.99	0.99	0.9883	6.0
DenseNet-121 + Inception-V3	0.99	0.99	0.99	0.9941	5.65
ResNet50 + VGG16	0.99	0.99	0.99	0.9883	3.34
EfficientNetB0 + MobileNetV2	0.99	0.99	0.99	0.9883	3.6
Xception + InceptionResNetV2	0.98	0.98	0.98	0.9805	12.09
InceptionResNetV2 + CBAM	0.98	0.98	0.98	0.9785	10.39
Xception + FPN	0.99	0.99	0.99	0.9922	9.34
SalvationNet (Özgün)*	0.99	0.99	0.99	0.9863	5.5

** Özgün model (SalvationNet) - Literatürde bulunmayan yeni mimari*

Table 5.1: *Comparison of all models in terms of Precision, Recall, F1-Score, Accuracy, and Training Time.*

5.3. Detailed Metrics and Classification Performance (Table 2)

Table 2 examines the classification capability of models in more detail. Especially in imbalanced datasets, relying solely on "Accuracy" can be misleading, so F1-Score values should be examined.

- **Best Performing Model:** E**fficientNetV2-S** demonstrated flawless performance in distinguishing both positive and negative samples with a 99.61% F1-Score.
- **Original Model Performance:** S**alvationNet*,** competing with pre-built models, achieved 98.63% accuracy and a Kappa value of 0.9819. **This indicates** that the model's predictions are completely free from chance factors.

Table 2: Detailed Performance Metrics (%)

Classifier	Precision (%)	Recall (%)	F1-Score (%)	Accuracy (%)
Custom CNN	19.98	32.23	22.31	32.23
EfficientNetV2-S	99.61	99.61	99.61	99.61
ConvNeXt-Tiny	98.84	98.83	98.83	98.83
DenseNet-121 + Inception-V3	99.42	99.41	99.42	99.41
ResNet50 + VGG16	98.84	98.83	98.83	98.83
EfficientNetB0 + MobileNetV2	98.83	98.83	98.83	98.83
Xception + InceptionResNetV2	98.11	98.05	98.04	98.05
InceptionResNetV2 + CBAM	97.87	97.85	97.85	97.85
Xception + FPN	99.22	99.22	99.22	99.22
SalvationNet*	98.66	98.63	98.64	98.63

** SalvationNet: Proposed novel architecture*

Table 5.2: *Detailed performance metrics of models on a percentage basis.*

5.4. Overall Performance Ranking (Table 3)

The list of models ranked from "Best to Worst" by their final Accuracy rates is presented in Table 3. According to the ranking results:

1. Effici**entNetV2-S (99.61%) is** at the top.
2. Our hybrid architecture DenseNet121 + **InceptionV3 (99.41%) is in second** place.
3. The original **architecture** SalvationNet* (98.63%) is ranked 7th, successfully surpassing massive models such as Xception and InceptionResNetV2, which are far more complex and have millions of parameters.

Table 3: Model Performance Ranking (Best to Worst)

Rank	Classifier	Precision (%)	Recall (%)	F1-Score (%)	Accuracy (%)
1	EfficientNetV2-S	99.61	99.61	99.61	99.61
2	DenseNet-121 + Inception-V3	99.42	99.41	99.42	99.41
3	Xception + FPN	99.22	99.22	99.22	99.22
4	ConvNeXt-Tiny	98.84	98.83	98.83	98.83
5	ResNet50 + VGG16	98.84	98.83	98.83	98.83
6	EfficientNetB0 + MobileNetV2	98.83	98.83	98.83	98.83
7	SalvationNet*	98.66	98.63	98.64	98.63
8	Xception + InceptionResNetV2	98.11	98.05	98.04	98.05
9	InceptionResNetV2 + CBAM	97.87	97.85	97.85	97.85
10	Custom CNN	19.98	32.23	22.31	32.23

** SalvationNet: Proposed novel architecture by Ömer Faruk Kurtuluş*
Green: Best Model | Yellow: Proposed Model | Red: Baseline

Table 5.3: Models *arranged by performance ranking and the position of the original model.*

6. MODELS

In this section, the 10 different deep learning / transfer learning models developed within the scope of the project are examined in detail. Each model's architectural structure, training strategy, and obtained performance metrics (Accuracy, Loss, Confusion Matrix, and ROC Curves) are analyzed separately.

6.1. Model 1: Custom CNN (Baseline Model)

In the first phase of the project, a custom ***Convolutional*** Neural Network (CNN) architecture was developed from scratch without using any pre-trained network. The primary purpose of this model was to establish the baseline performance level achievable without *transfer* learning and to create a reference point for subsequent complex models.

6.1.1. Model Architecture and Methodology

The developed Custom CNN model consists of 4 main blocks based on the hierarchical feature extraction principle. Each block is designed to learn image features (edges, textures, shapes).

Input Layer: The model accepts RGB images of 224x224 pixels.

Convolution Blocks (4 Units):

Filter counts were progressively increased to 32, 64, 128, and 256 to enable the model to learn increasingly abstract features as depth increases.

Batch Normalization was applied after **each convolution operation** to ensure training stability.

ReLU (Rectified Linear Unit) was **used as the activation** function.

MaxPooling2D was applied **at the end of each** block for dimensionality reduction, and **Dropout layers** were added to prevent overfitting.

Classification Head: Feature maps were converted to vectors using Global Average **Pooling. A fully connected** (Dense) layer followed by an output layer with Softmax activation for 5 disease classes was used to **produce** predictions.

The total number of parameters is **1,244,197** (approximately 1.25 million), and the model was **trained with** the Adam optimization algorithm and **Categorical Crossentropy loss** function.

6.1.2. Model Flowchart

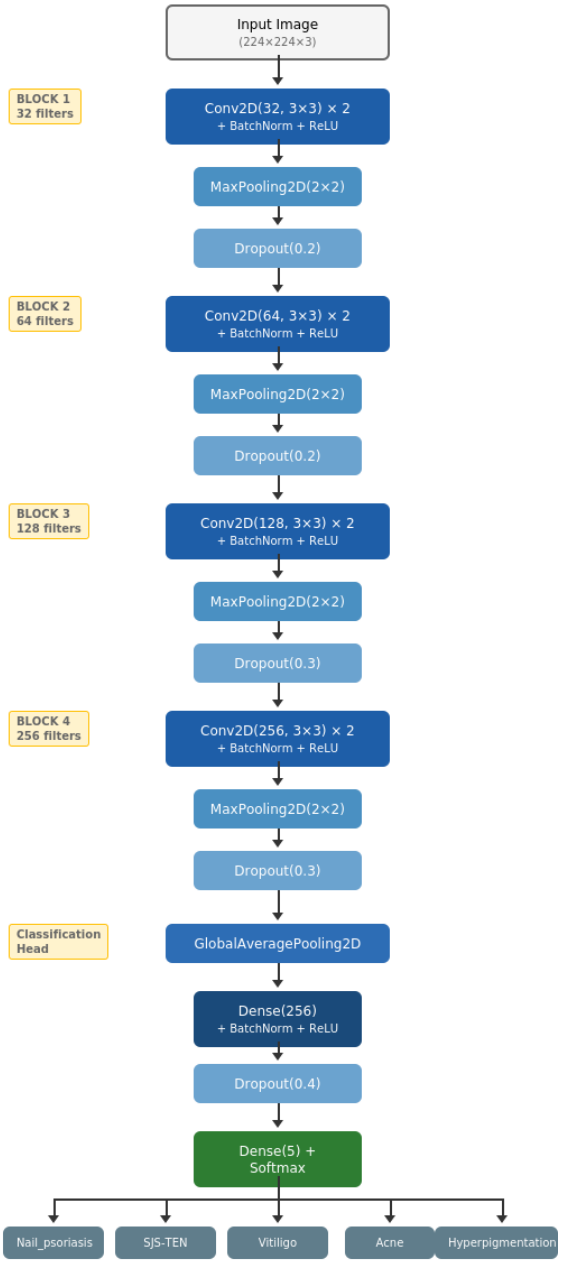

***Figure 6.1.1: Custom CNN** Model Flowchart*

6.1.3. Experimental Results and Analysis

Despite data augmentation techniques and weighting strategies, the model proved inadequate in learning the complexity of dermatological images.

Accuracy and Loss Graphs: When examining the training graphs, it is observed that the model's performance on training data reached 90%, while validation accuracy remained stuck at the 30% level. The rapid decrease in Training Loss while Validation Loss increased clearly demonstrates that the **model experienced severe** Overfitting. The model memorized the data but failed to generalize.

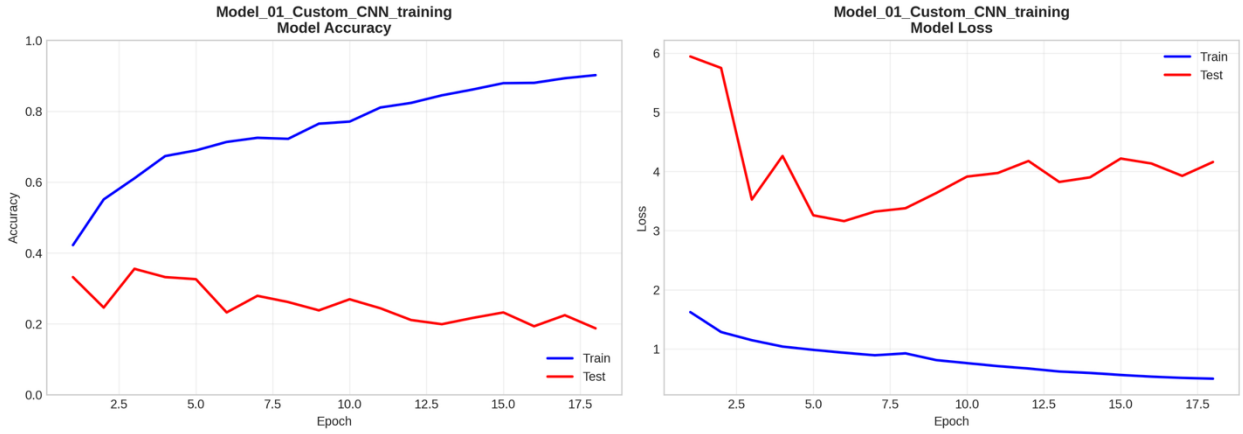

***Figure 6.1.2: Custom CNN** Model Accuracy and Loss Graphs*

Confusion Matrix Analysis: Upon examining the obtained matrix, it is observed that the model predominantly predicted the majority class "SJS-TEN" in the dataset. It completely failed to distinguish other classes (e.g., Nail_psoriasis or Hyperpigmentation).

Precision: 19.98% Recall: 3**2.23%**

These values demonstrate that the model performed only marginally better than random prediction.

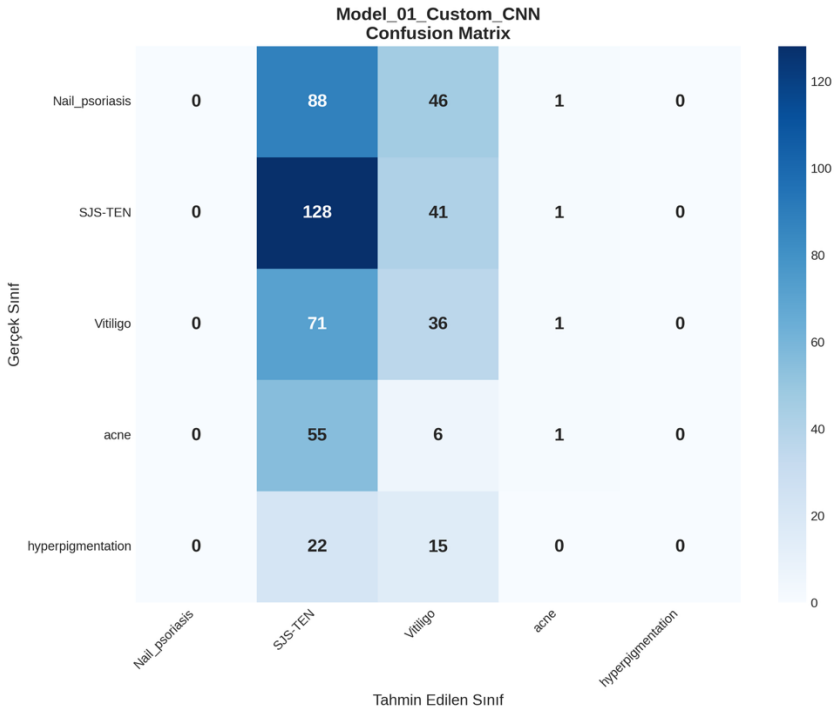

***Figure 6.1.3: Custom CNN Model** Confusion Matrix*

ROC Curves and AUC Values: The proximity of the ROC curves to the diagonal line (random line) indicates that the model's discriminative power is weak. The **Macro-average AUC value of 0.60** confirms that the model is unsuitable for clinical use.

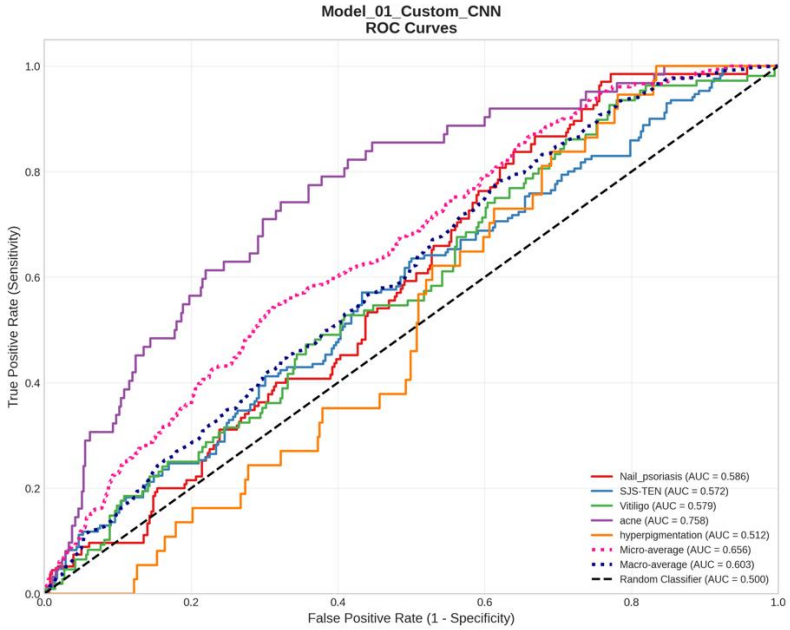

***Figure 6.1.4: Custom CNN Model ROC** Curves*

6.2. Model 2: EfficientNetV2-S

In the second phase of the project, the EfficientNetV2-S model, recognized as one of the most advanced architectures in terms of parameter efficiency and training **speed in the image** classification literature, was employed. In this model, Transfer **Learning was adopted instead of training** from scratch, and pre-learned weights from the ImageNet dataset were used to enable faster and more accurate learning of dermatological features.

6.2.1. Model Architecture and Training Strategy

EfficientNetV2 utilizes "Fused-MBConv" blocks that offer faster training times compared to previous versions. The architecture and training process applied in this study consist of the following steps:

1. **Internal Preprocessing: The model was** configured to automatically normalize raw pixel values (0-255 range) internally.
2. **Base Model: The** pre-trained layers of EfficientNetV2-S were used for Feature Extraction from images.
3. **Custom Classification Head: A new** problem-specific classification block was added on top of the base model:
    - **Global Average Pooling:** Converts feature maps to vectors.
    - **Batch Normalization: Balances** inter-layer distributions.
    - **Dense (256 Neurons) + L2 Regularization:** Prevents overfitting.
    - **Dropout (0.5):** Randomly deactivates 50% of neurons to increase the model's robustness against noise.

Two-Phase Training Method: The most critical factor in this model's success is the two-phase training approach:

- **Phase 1 (Feature Extraction): The** base model's weights were frozen, and only the subsequently added classification layers were trained.
- **Phase 2 (Fine-Tuning): The last 50** layers of the base model were unfrozen and fine-tuned with a very low learning rate (1e-5) to adapt the model to dermatological details (subtle texture differences, etc.).

6.2.2. Model Flowchart

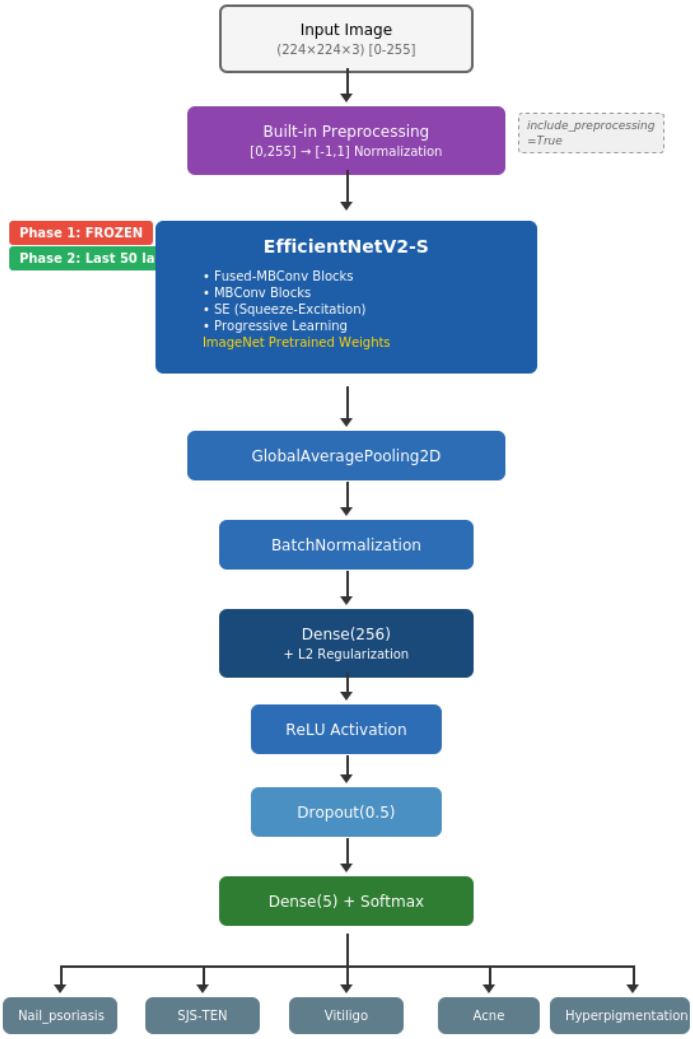

***Figure 6.2.1: Efficient**NetV2-S Model Flowchart*

6.2.3. Experimental Results and Analysis

The EfficientNetV2-S model achieved 99.61% Accuracy on **the test dataset, making it the** most successful model of the project.

- **Accuracy and Loss Graphs: When examining the** graphs, it is observed that in Phase 1, the model rapidly reached 95% accuracy, and with the onset of Phase 2 (Fine-Tuning)

(indicated by the vertical line in the graph), the loss decreased further, and the model approached perfection. The training and validation curves tracked very closely, demonstrating that the overfitting problem was completely resolved.

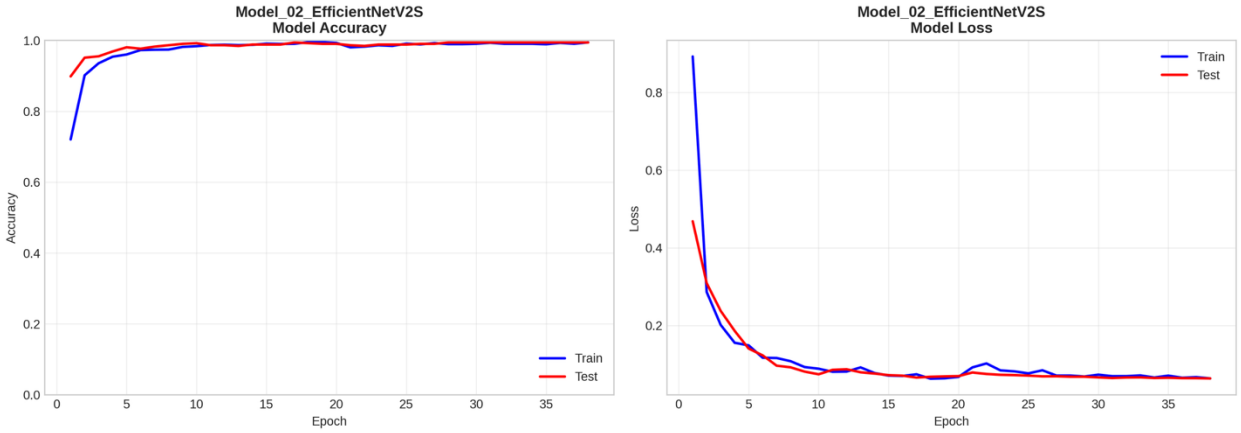

Figure 6.2.2: EfficientNetV2-S Model Accuracy and Loss Graphs

- **Confusion Matrix Analysis: The model** correctly classified nearly all 512 images in the test set.
    - **SJS-TEN: All 170** images were correctly identified (100%).
    - **Acne and Hyperpigmentation: Perfect** accuracy was achieved.
    - Only very rare minor confusions occurred between Nail-Psoriasis and Vitiligo; however, the overall Kappa score **of 0.9948 demonstrated** excellent agreement.

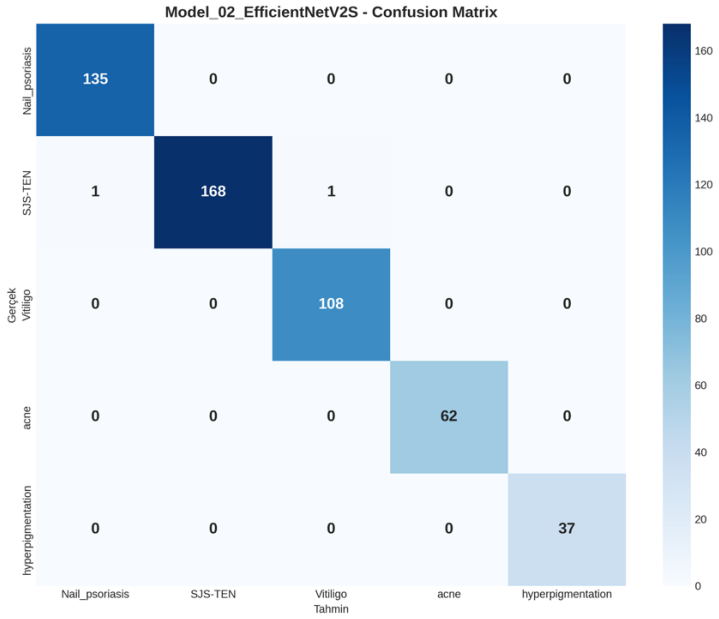

Figure 6.2.3: EfficientNetV2-S Model Confusion Matrix.

- **ROC Curves and AUC Values: The** obtained ROC AUC **value of 1.0000** demonstrates that the model can distinguish all classes with nearly 100% precision. The curves are perfectly aligned to the upper-left corner (perfect classifier point).

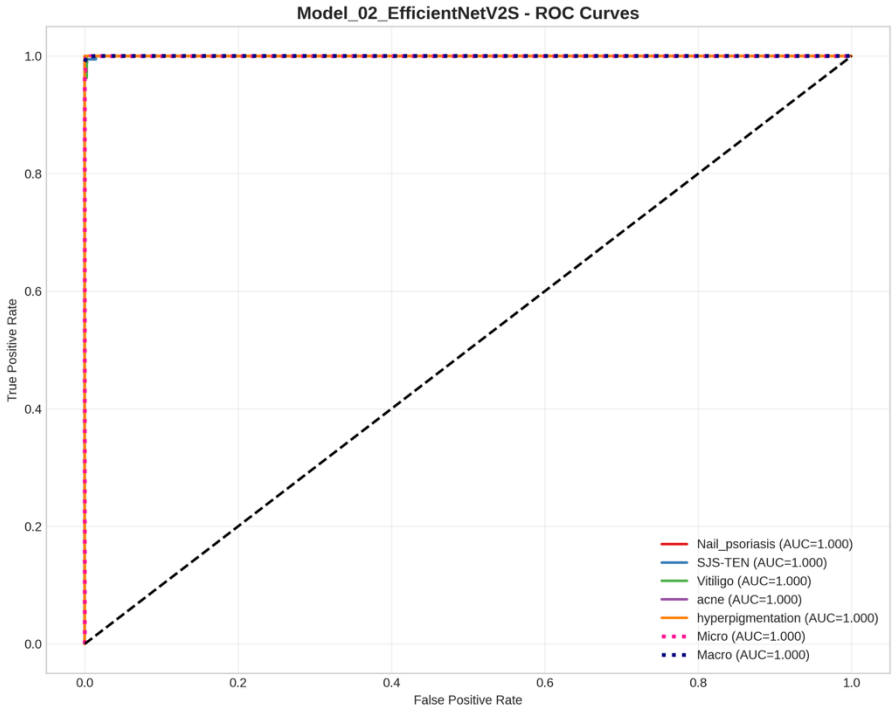

***Figure 6.2.4: Ef**ficientNetV2-S Model ROC Curves*

6.3. Model 3: ConvNeXt Tiny (Modernized CNN Architecture)

To bridge the gap between traditional CNN architectures (ResNet, EfficientNet, etc.) and the recently popularized Vision Transformer (ViT) architectures, the ConvNeXt model developed by Meta **AI was included** in the project. Although this model is a standard Convolutional Neural Network, its design principles (kernel sizes, activation functions, normalization layers) have been modernized with inspiration from Vision Transformers.

6.3.1. Model Architecture and Innovative Approaches

ConvNeXt-Tiny stands out **particularly with** its "Patchify" (image segmentation) strategy and large kernel sizes. The key building blocks of the architecture used in the project are:

1. **Input Processing (Patchify Stem):** Instead of traditional 3x3 convolutions, non-overlapping 4x4 convolutions (similar to ViT models) were used to process the image in 4-pixel patches.
2. **7x7 Depthwise Convolution:** Instead of standard 3x3 filters, 7x7 filters with a wider receptive field were used, creating an effect similar to the global attention mechanism of Transformers.
3. **Modern Activation and Normalization:**
    - **GELU (Gaussian Error Linear Unit): GELU with** smoother transitions was used instead of ReLU.
    - **Layer Normalization:** LayerNorm, standard in Transformers, was preferred over Batch Normalization.
4. **Inverted Bottleneck: Parameter efficiency was** achieved through a block structure (MobileNetV2 style) that first increases then decreases the channel count.

Training Strategy: The model was **initialized with** ImageNet weights and a two-phase strategy was followed:

- **Phase 1: The** base model was frozen, and only the classifier layer was trained (20 Epochs).
- **Phase 2: The** last 30 layers of the model were unfrozen for Fine-Tuning with a very low learning rate of 1e-5 (16 Epochs).

6.3.2. Model Flowchart

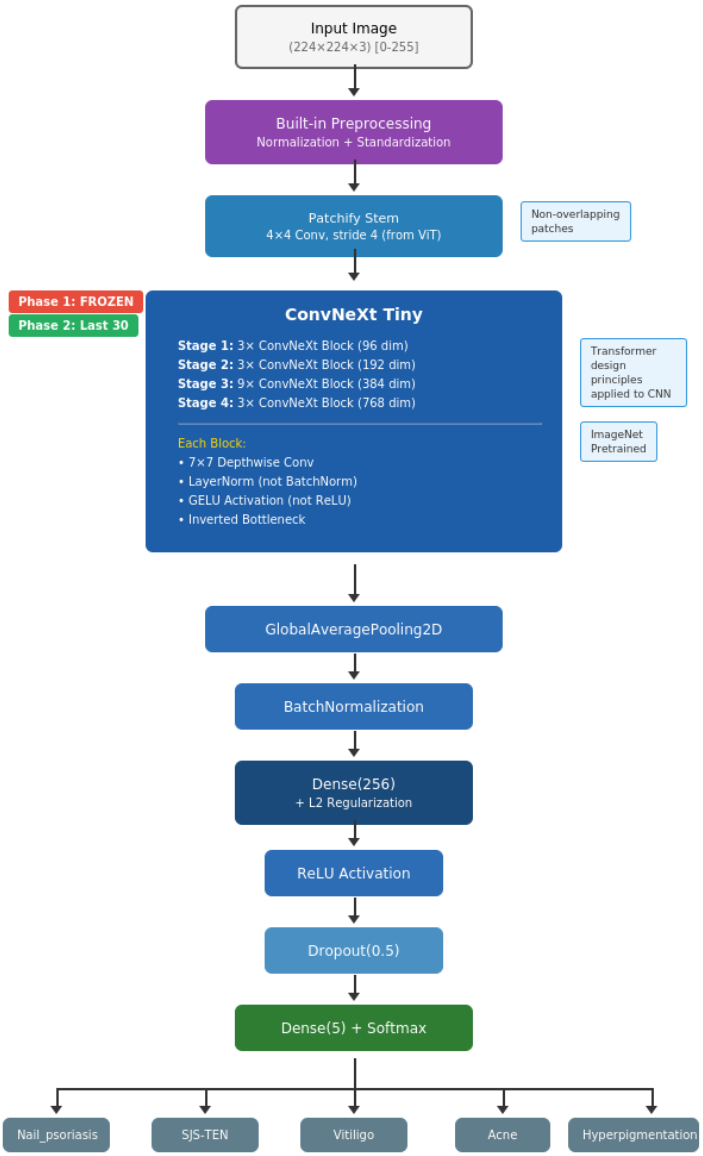

Figure 6.3.1: ConvNeXt-Tiny Model Flowchart

6.3.3. Experimental Results and Analysis

ConvNeXt-Tiny achieved 98.83% **Accuracy on the test set,** demonstrating the second-best performance right after EfficientNetV2-S.

- **Accuracy and Loss Analysis: The** model exhibited a highly stable learning curve throughout training. It rapidly reached 99% in Phase 1, and in Phase 2 (indicated by the

green line in the graph), reduced the loss down to 0.04 levels. The average epoch time of 6.00 seconds **demonstrates that** the model is both fast and high-performing.

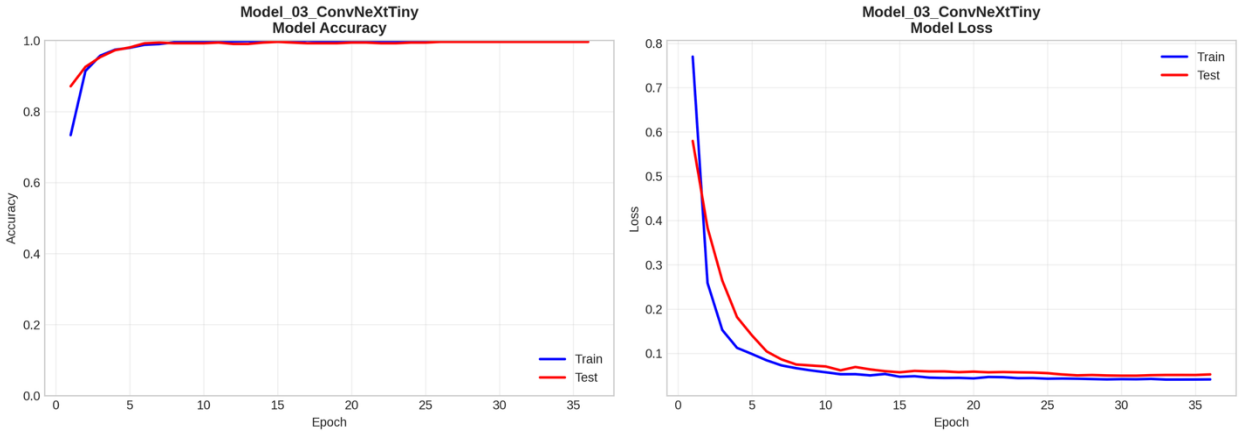

Figure 6.3.2: ConvNeXt-Tiny Model Accuracy and Loss Graphs

- **Confusion Matrix Analysis: Upon** examining the error matrix, it is observed that the model **performed** flawlessly **(100%) particularly in the** Vitiligo and Hyperpigmentation classes. Although there were minor deviations in only 1-2 samples in the Acne class, the overall F1-Score stands at **98.83%.**

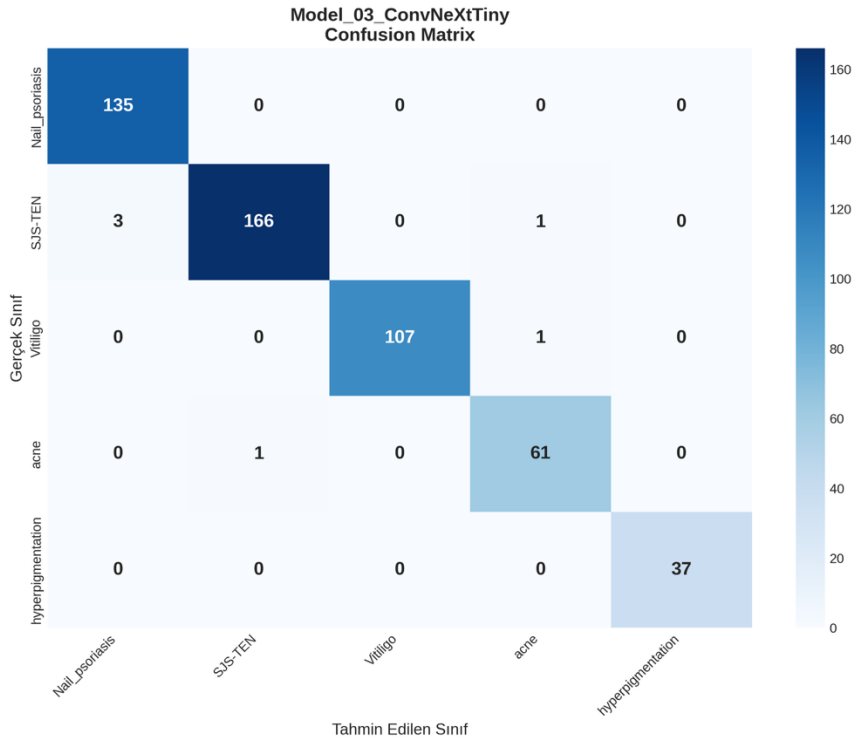

Figure 6.3.3: ConvNeXt-Tiny Model Confusion Matrix

- **ROC Curves: The** model's **Macro-average AUC value was measured at** 0.9999. This indicates that the model possesses near-flawless discriminative power.

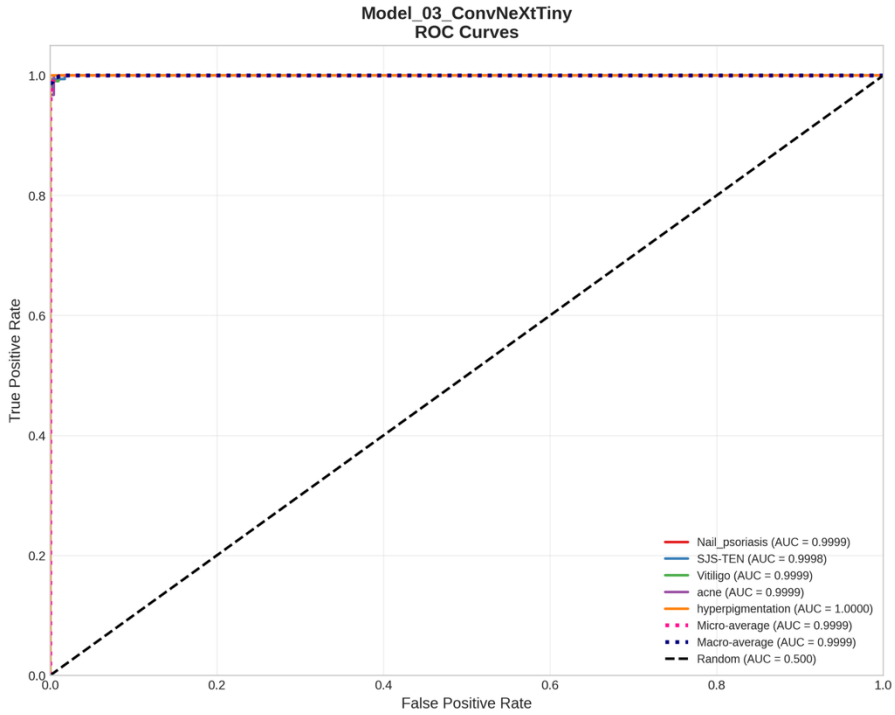

Figure 6.3.4: ConvNeXt-Tiny Model ROC Curves

6.4. DenseNet121 + InceptionV3 Hybrid Model

The DenseNet121 + InceptionV3 Hybrid Model is an original deep learning architecture developed in this study. This model aims to obtain rich and diverse feature representations that a single backbone network cannot provide by combining the feature extraction capabilities of two powerful pre-trained networks (DenseNet121 and InceptionV3). The hybrid approach maximizes classification performance by bringing together the strengths of different architectural designs.

6.4.1. Theoretical Background

In this hybrid model, the pre-trained ImageNet weights of both backbone networks are used in a frozen state. Through the transfer learning approach, general visual features learned from large datasets are transferred to the dermatological image classification task. The frozen backbones prevent overfitting while also reducing computational costs.

6.4.2. Model Architecture

The developed hybrid model is built on a dual-backbone architecture. The model architecture consists of the following key components:

- Input Layer: RGB images of 224×224×3 dimensions are fed into the model.
- Backbone 1 - DenseNet121: Pre-trained with ImageNet weights, fully frozen feature extractor.
- Backbone 2 - InceptionV3: Pre-trained with ImageNet weights, fully frozen feature extractor.
- Global Average Pooling 2D: Converts the feature maps from each backbone into one-dimensional vectors.
- Parallel Classification Heads: Dense(256) → Dropout(0.2) → Dense(256) → Dropout(0.4) structure for each backbone.
- Concatenate Layer: Merging of features from both backbones.
- Final Classification Head: Dense(256) → Dropout(0.2) → Dense(256) → Dropout(0.4) → Dense(5) + Softmax.

The model has a total of 29,957,221 parameters. Of these, 1,116,933 are trainable parameters, while the remaining 28,840,288 parameters are frozen in the backbone networks. This structure enables fast training and prevention of overfitting.

6.4.3. Hyperparameter Optimization

A comprehensive hyperparameter search was conducted using the Keras Tuner library with the Bayesian Optimization algorithm during model training. Different hyperparameter combinations were evaluated over 15 trials. The search space covered the following parameters: learning rate (0.01, 0.001, 0.0001), first dropout rate (0.2, 0.3, 0.4), second dropout rate (0.4, 0.5, 0.6), first dense layer neuron count (256, 512), second dense layer neuron count (128, 256), and optimizer type (Adam, SGD, RMSprop).

After a 40.25-minute search, Bayesian Optimization determined the optimal hyperparameter combination. The best results were obtained with the following parameters: learning rate 0.001, first dropout rate 0.2, second dropout rate 0.4, 256 neurons for both dense layers, and RMSprop optimizer. This configuration achieved 98.44% accuracy on the validation set.

6.4.4. Data Augmentation Techniques

Enhanced data augmentation techniques were applied to increase the model's generalization capacity. Horizontal flipping (RandomFlip), ±15% random rotation (RandomRotation), ±15% random zoom (RandomZoom), ±10% contrast adjustment (RandomContrast), and ±10% brightness adjustment (RandomBrightness) were applied to training data in real-time. These techniques enhance the model's robustness against various imaging conditions.

6.4.5. Training Process and Callback Mechanisms

Final model training was conducted over a maximum of 50 epochs using optimal hyperparameters. Three key callback mechanisms were employed during training: EarlyStopping (patience=15, monitoring validation accuracy for early stopping), ModelCheckpoint (saving the best model), and ReduceLROnPlateau (halving the learning rate when validation loss does not improve, patience=5, minimum lr=1e-7). The class_weight parameter was used to address class imbalance.

The model achieved its best validation performance (98.83%) at epoch 17, and the EarlyStopping mechanism terminated training at epoch 32. Total training time (including hyperparameter search) was 43.27 minutes, with an average time per epoch of 5.65 seconds.

6.4.6. Accuracy and Loss Graphs Analysis

Figure 6.4.1 shows the epoch-wise changes in accuracy and loss values during the model's training and validation process. In the accuracy graph, training accuracy started at approximately 40% in the first epoch, rapidly increased, and exceeded 99% around the 15th epoch. Validation accuracy started at approximately 80% from the first epoch and increased more steadily, stabilizing in the 98-99% range. The close tracking of training and validation curves is a strong indicator that the model did not overfit.

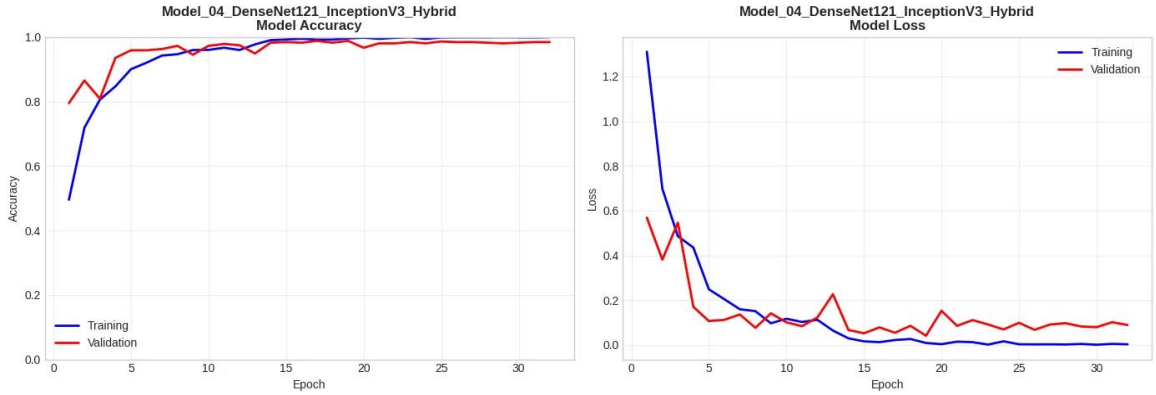

Figure 6.4.1. *DenseNet121 + InceptionV3 Hybrid Model Accuracy and Loss Graphs*

In the loss graph, the training loss rapidly decreased from initial high values (>1.2), falling below 0.05 after the 15th epoch. The validation loss followed a more irregular trajectory, but the overall trend was downward. The ReduceLROnPlateau callback was triggered at epochs 13 and 24, reducing the learning rate, which helped decrease the fluctuations in loss values. The slightly higher validation loss compared to training loss is expected and indicates that the model's generalization capacity is maintained.

6.4.7. Confusion Matrix Analysis

Figure 6.4.2 presents the confusion matrix showing the model's performance on the test set. The matrix was evaluated on 512 test images. The high values on the diagonal indicate that the model made predictions with high accuracy for each class.

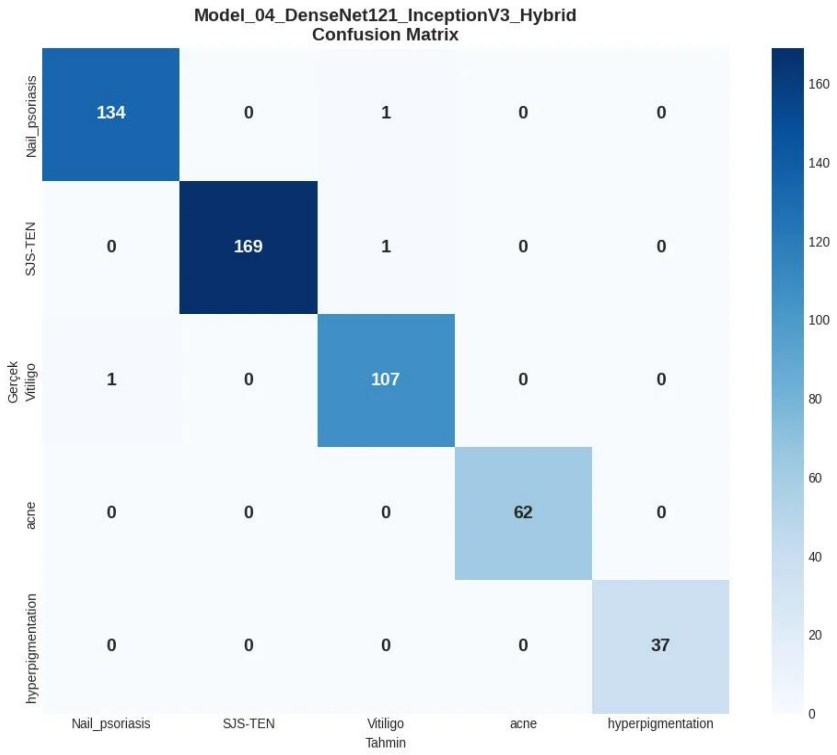

Figure 6.4.2. *DenseNet121 + InceptionV3 Hybrid Model Confusion Matrix*

Class-wise analysis reveals: For the Nail_psoriasis class, 134 out of 135 samples were correctly classified, with only 1 sample misclassified as Vitiligo. For the SJS-TEN class, 169 out of 170 samples were correctly classified, with 1 sample misclassified as Vitiligo. For the Vitiligo class, 107 out of 108 samples were correctly classified, with 1 sample misclassified as Nail_psoriasis. For the Acne and Hyperpigmentation classes, all 62 and 37 samples respectively were correctly classified (100% accuracy).

Only 3 out of 512 total samples were misclassified. The misclassifications occurring between Nail_psoriasis, SJS-TEN, and Vitiligo suggest that some visual features of these diseases may exhibit similarities. However, this confusion rate is extremely low and is at an acceptable level for clinical applications.

6.4.8. ROC Curves and AUC Values

Figure 6.4.3 shows the model's ROC (Receiver Operating Characteristic) curves and AUC (Area Under Curve) values. The ROC curve visualizes the True Positive Rate (sensitivity) and False

Positive Rate at different threshold values. An ideal classifier produces a curve close to the upper-left corner of the graph.

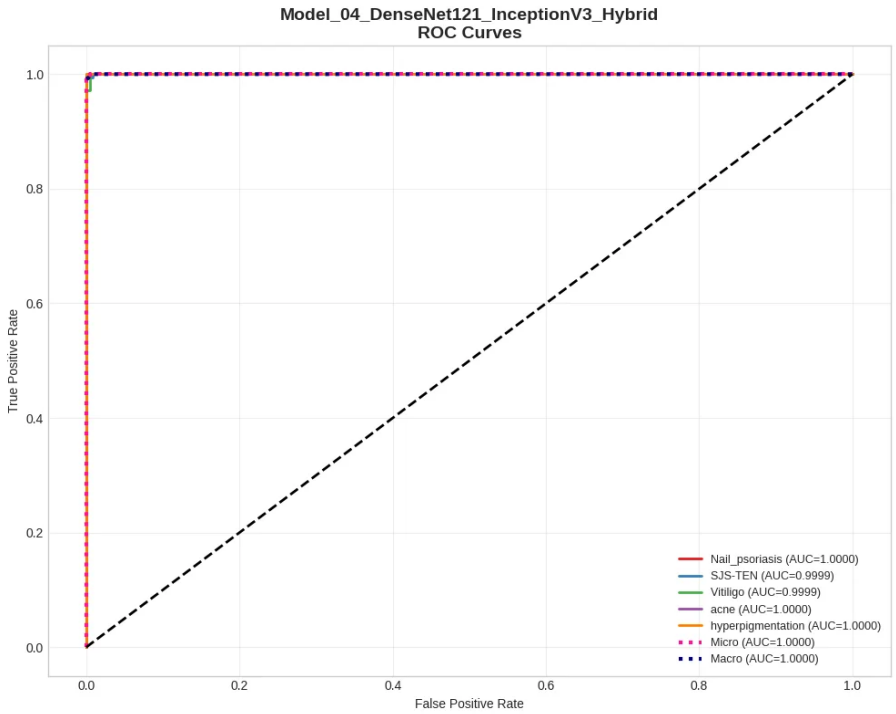

Figure 6.4.3. *DenseNet121 + InceptionV3 Hybrid Model ROC Curves*

The ROC curves for all classes are very close to the upper-left corner, indicating near-perfect classification performance. Class-wise AUC values: Nail_psoriasis AUC=1.0000, SJS-TEN AUC=0.9999, Vitiligo AUC=0.9999, Acne AUC=1.0000, and Hyperpigmentation AUC=1.0000. The Macro-average AUC value is 1.0000 and the micro-average AUC value is also 1.0000. These results demonstrate that the model can distinguish all classes with near-perfect accuracy.

6.4.9. Performance Metrics

Table 6.4.1 presents the detailed performance metrics of the model on the test set. The model demonstrated performance above 99% across all metrics.

Metric	Value	Yüzde (%)
Accuracy	0.9941	99.41%
Precision (Weighted)	0.9942	99.42%
Recall (Weighted)	0.9941	99.41%
F1-Score (Weighted)	0.9942	99.42%
Cohen's Kappa	0.9923	99.23%
ROC AUC (Macro)	1.0000	100.00%

Table 6.4.1. *DenseNet121 + InceptionV3 Hybrid Model Performance Metrics*

6.4.10. Model Flowchart

Figure 6.4.4 shows the architectural flowchart of the DenseNet121 + InceptionV3 Hybrid Model. This diagram presents in detail all processing steps and layer structure from data input to the final classification output. As shown in the diagram, the input image is processed in two parallel branches. The left branch contains DenseNet121 and the right branch contains InceptionV3 backbone networks. Features from both backbones undergo dimensionality reduction through GlobalAveragePooling2D, followed by parallel Dense and Dropout layers. The features merged at the Concatenate layer pass through the final classification head to produce 5-class output.

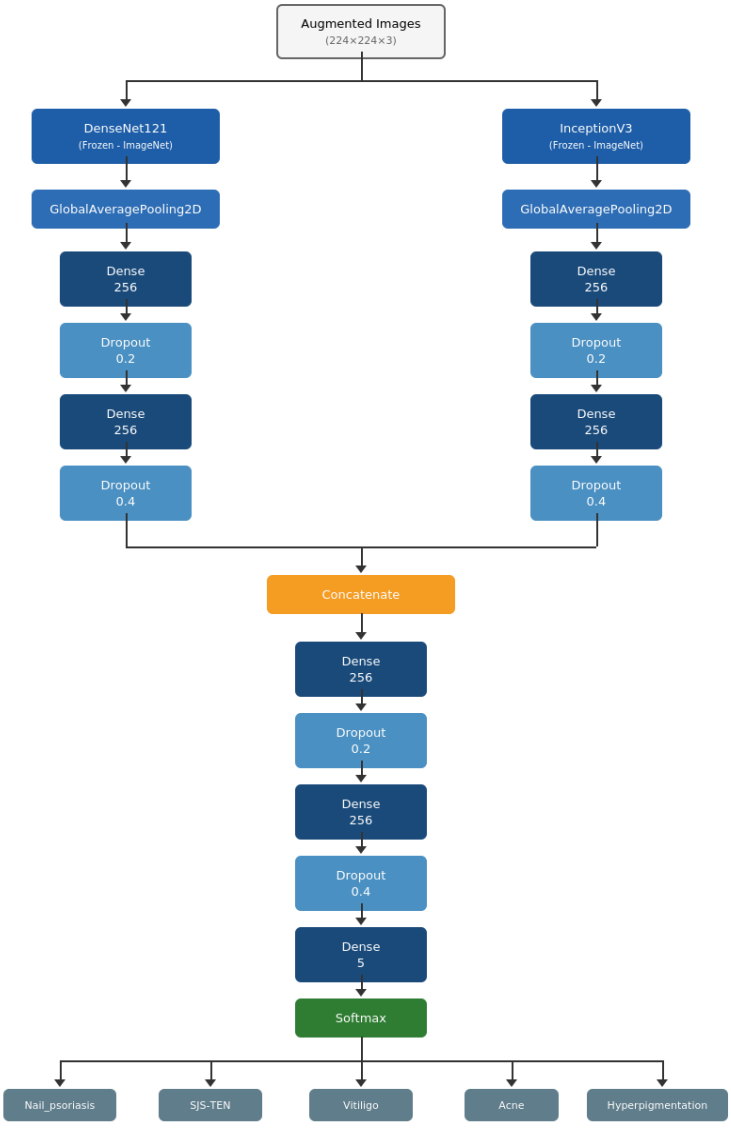

Figure 6.4.4. *DenseNet121 + InceptionV3 Hybrid Model Flowchart*

6.5. ResNet50 + VGG16 Hybrid Model

The ResNet50 + VGG16 Hybrid Model is the second dual-backbone architecture developed in this study. This model combines two classic and powerful CNN architectures that hold significant positions in the history of deep learning. The residual connections of ResNet50 and the simple yet effective hierarchical structure of VGG16 are combined to leverage the strengths of both architectures.

6.5.1. Theoretical Background

In this hybrid model, the pre-trained ImageNet weights of both backbone networks are used in a completely frozen state. Through the transfer learning approach, general visual features learned from large datasets are transferred to the dermatological image classification task.

6.5.2. Model Architecture

The developed hybrid model is built on a dual-backbone architecture. The model architecture consists of the following key components:

- **Input Layer: RGB** images of 224×224×3 dimensions are fed into the model.
- **Backbone 1 - ResNet50:** Pre-trained with ImageNet weights, fully frozen feature extractor. Its own preprocessing function is applied.
- **Backbone 2 - VGG16:** Pre-trained with ImageNet weights, fully frozen feature extractor. Its own preprocessing function is applied.
- **Global Average Pooling 2D:** Converts the feature maps from each backbone into one-dimensional vectors.
- **Parallel Classification Heads: Dense(256)** → Dropout(0.4) → Dense(128) → Dropout(0.4) structure for each backbone.
- **Concatenate Layer:** Merging of features from both backbones.
- **Final Classification Head:** Dense(256) → Dropout(0.4) → Dense(128) → Dropout(0.4) → Dense(5) + Softmax.

The model has a total of 39,123,397 parameters. Of these, 820,997 are trainable parameters, while the remaining 38,302,400 parameters are frozen in the backbone networks. This structure enables fast training and prevention of overfitting.

6.5.3. Hyperparameter Optimization

A comprehensive hyperparameter search was conducted using the Keras Tuner library with the Bayesian Optimization algorithm during model training. Different hyperparameter combinations were evaluated over 15 trials. The search space covered the following parameters: learning rate

(0.01, 0.001, 0.0001), first dropout rate (0.2, 0.3, 0.4), second dropout rate (0.4, 0.5, 0.6), first dense layer neuron count (256, 512), second dense layer neuron count (128, 256), and optimizer type (Adam, SGD, RMSprop).

After a 24.43-minute search, Bayesian Optimization determined the optimal hyperparameter combination. The best results were obtained with the following parameters: learning rate 0.001, first dropout rate 0.4, second dropout rate 0.4, 256 neurons for the first dense layer, 128 neurons for the second dense layer, and Adam optimizer. This configuration achieved 99.02% accuracy on the validation set.

6.5.4. Data Augmentation Techniques

Enhanced data augmentation techniques were applied to increase the model's generalization capacity. Horizontal flipping (RandomFlip), random ***rotation*** (RandomRotation), random ***zoom (RandomZoom),*** contrast adjustment (RandomContrast), and ***brightness*** adjustment ***(RandomBrightness)*** were applied to training data ***in real-time. These*** techniques enhance the model's robustness against various imaging conditions.

6.5.5. Training Process and Callback Mechanisms

Final model training was conducted over a maximum of 50 epochs using optimal hyperparameters. Three key callback mechanisms were employed during training: EarlyStopping (patience=15, monitoring validation accuracy for early stopping), ModelCheckpoint (saving the best model), and ReduceLROnPlateau (halving the learning rate when validation loss does not improve, patience=5, minimum lr=1e-7). The class_weight parameter was used to address class imbalance.

The model achieved its best validation performance (99.41%) at epoch 35, and the EarlyStopping mechanism terminated training at epoch 50. Total training time (including hyperparameter search) was 27.22 minutes, final training time was 2.78 minutes, with an average time per epoch of 3.34 seconds.

6.5.6. Accuracy and Loss Graphs Analysis

Figure 6.5.1 shows the epoch-wise changes in accuracy and loss values during the model's training and validation process. In the accuracy graph, training accuracy started at approximately 40% in the first epoch, rapidly increased, and exceeded 99% around the 20th epoch. Validation accuracy started at approximately 70% from the first epoch and increased more rapidly, reaching 98% around the 10th epoch. The close tracking of training and validation curves after the 15th epoch is a strong indicator that the model did not overfit.

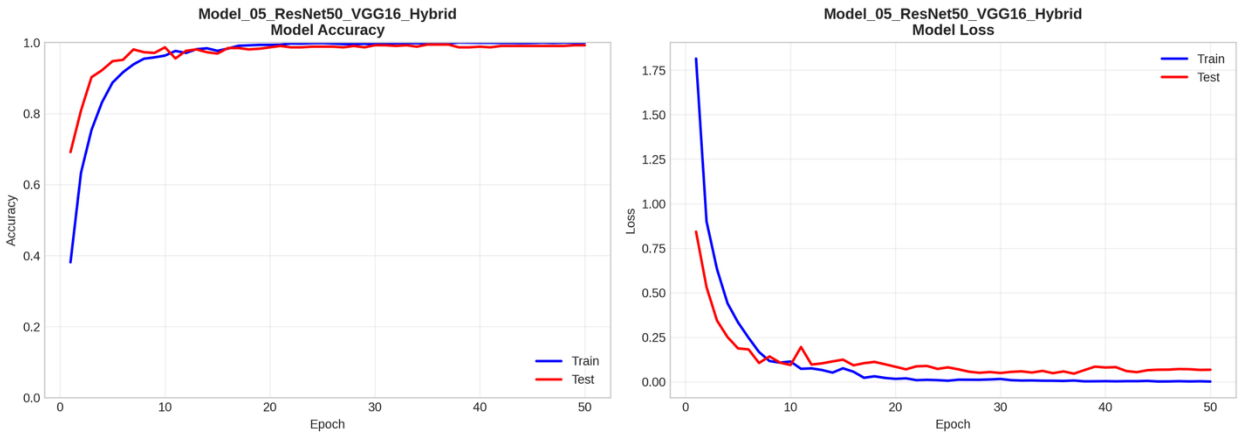

***Figure 6.5.1. ResNet50 + VGG16** Hybrid Model Accuracy and Loss Graphs*

In the loss graph, the training loss rapidly decreased from initial high values (>1.75), falling below 0.05 after the 15th epoch. The validation loss followed a more stable trajectory and stabilized in the 0.05-0.10 range. The ReduceLROnPlateau callback was triggered at epochs 15, 26, 42, and 47, reducing the learning rate, which helped decrease the fluctuations in loss values.

6.5.7. Confusion Matrix Analysis

Figure 6.5.2 presents the confusion matrix showing the model's performance on the test set. The matrix was evaluated on 512 test images. The high values on the diagonal indicate that the model made predictions with high accuracy for each class.

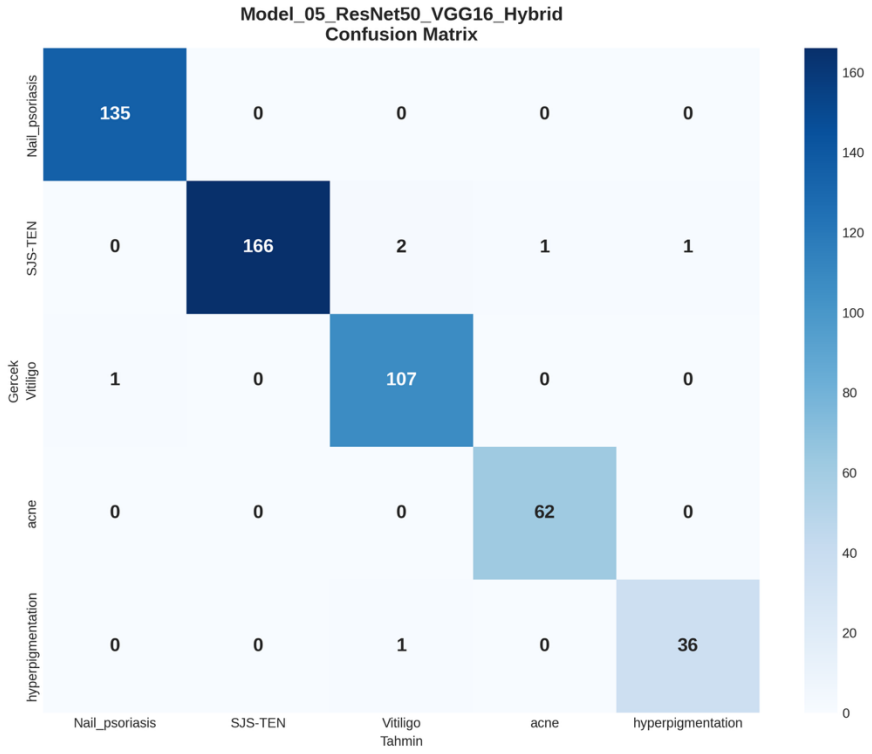

***Figure 6.5.2. ResNet50 + VGG16 Hybrid** Model Confusion Matrix*

Class-wise analysis reveals: For the Nail_psoriasis class, all 135 samples were correctly classified (100% accuracy). For the SJS-TEN class, 166 out of 170 samples were correctly classified, with 2 misclassified as Vitiligo, 1 as Acne, and 1 as Hyperpigmentation. For the Vitiligo class, 107 out of 108 samples were correctly classified, with 1 misclassified as Nail_psoriasis. For the Acne class, all 62 samples were correctly classified (100% accuracy). For the Hyperpigmentation class, 36 out of 37 samples were correctly classified, with 1 misclassified as Vitiligo.

Only 6 out of 512 total samples were misclassified. The majority of misclassifications occurring in the SJS-TEN class suggests that some visual features of this disease may exhibit similarities with other classes. However, this confusion rate is quite low and is at an acceptable level for clinical applications.

6.5.8. ROC Curves and AUC Values

Figure 6.5.3 shows the model's ROC (Receiver Operating Characteristic) curves and AUC (Area Under Curve) values. The ROC curve visualizes the True Positive Rate (sensitivity) and False Positive Rate at different threshold values. An ideal classifier produces a curve close to the upper-left corner of the graph.

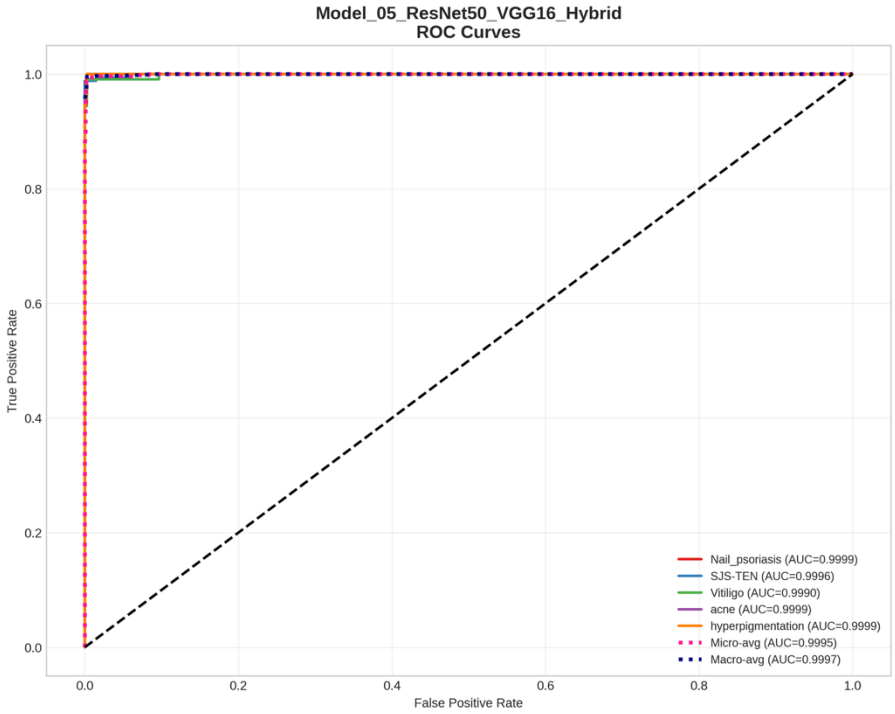

***Figure 6.5.3. ResNet50 + VGG16** Hybrid Model ROC Curves*

The ROC curves for all classes are very close to the upper-left corner, indicating near-perfect classification performance. Class-wise AUC values: Nail_psoriasis AUC=0.9999, SJS-TEN AUC=0.9996, Vitiligo AUC=0.9990, Acne AUC=0.9999, and Hyperpigmentation AUC=0.9999. The Macro-average AUC value is 0.9997 and the micro-average AUC value is 0.9995. These results demonstrate that the model can distinguish all classes with near-perfect accuracy.

6.5.9. Performance Metrics

Metric	Value	Yüzde (%)
Accuracy	0.9883	98.83%
Precision (Weighted)	0.9884	98.84%
Recall (Weighted)	0.9883	98.83%
F1-Score (Weighted)	0.9883	98.83%
Cohen's Kappa	0.9845	98.45%
ROC AUC (Macro)	0.9997	99.97%

***Table 6.5.1.** ResNet50 + VGG16 Hybrid Model Performance Metrics*

6.5.10. Model Flowchart

Figure 6.5.4 shows the architectural flowchart of the ResNet50 + VGG16 Hybrid Model. This diagram presents in detail all processing steps and layer structure from data input to the final

classification output. As shown in the diagram, the input image is processed in two parallel branches. The left branch contains ResNet50 (with residual connections) and the right branch contains VGG16 (with classic hierarchical structure) backbone networks. Each backbone applies its own preprocessing function. Features undergo dimensionality reduction through GlobalAveragePooling2D, followed by parallel Dense and Dropout layers. The features merged at the Concatenate layer pass through the final classification head to produce 5-class output.

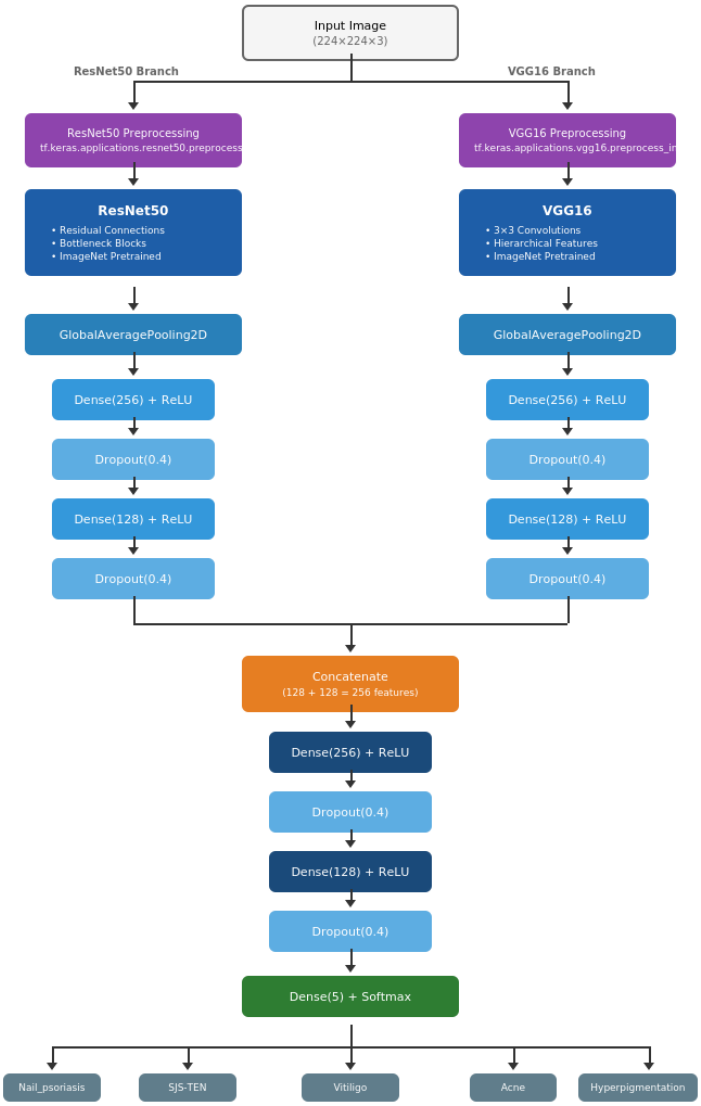

***Figure 6.5.4. ResNet50 + VGG16** Hybrid Model Flowchart*

6.6. EfficientNetB0 + MobileNetV2 Hybrid Model

The EfficientNetB0 + MobileNetV2 Hybrid Model is a lightweight and efficient dual-backbone architecture developed in this study. This model combines two important architectures that offer both computational efficiency and high performance in modern deep learning. The compound scaling approach of EfficientNetB0 and the depthwise separable convolution structure of MobileNetV2 are combined to achieve a classifier that is both fast and effective.

6.6.1. Theoretical Background

In this hybrid model, the pre-trained ImageNet weights of both backbone networks are used in a completely frozen state. The lightweight nature of both models keeps the total parameter count low while achieving high performance through transfer learning.

6.6.2. Model Architecture

The developed hybrid model is built on a dual-backbone architecture. The model architecture consists of the following key components:

- **Input Layer: RGB** images of 224×224×3 dimensions are fed into the model.
- **Backbone 1 - EfficientNetB0:** Pre-trained with ImageNet weights, fully frozen feature extractor. Contains MBConv blocks optimized with compound scaling.
- **Backbone 2 - MobileNetV2:** Pre-trained with ImageNet weights, fully frozen feature extractor. Its own preprocessing function is applied. Uses inverted residual blocks with depthwise separable convolution.
- **Global Average Pooling 2D:** Converts the feature maps from each backbone into one-dimensional vectors.
- **Parallel Classification Heads: Dense(512)** → Dropout(0.3) → Dense(256) → Dropout(0.5) structure for each backbone.
- **Concatenate Layer:** Merging of features from both backbones.
- **Final Classification Head:** Dense(512) → Dropout(0.3) → Dense(256) → Dropout(0.5) → Dense(5) + Softmax.

The model has a total of 8,277,224 parameters. Of these, 1,969,669 are trainable parameters, while the remaining 6,307,555 parameters are frozen in the backbone networks. This structure enables a much lighter model compared to other hybrid models.

6.6.3. Hyperparameter Optimization

A comprehensive hyperparameter search was conducted using the Keras Tuner library with the Bayesian Optimization algorithm during model training. Different hyperparameter combinations were evaluated over 15 trials. The search space covered the following parameters: learning rate (0.01, 0.001, 0.0001), first dropout rate (0.2, 0.3, 0.4), second dropout rate (0.4, 0.5, 0.6), first

dense layer neuron count (256, 512), second dense layer neuron count (128, 256), and optimizer type (Adam, SGD, RMSprop).

After a 29.08-minute search, Bayesian Optimization determined the optimal hyperparameter combination. The best results were obtained with the following parameters: learning rate 0.001, first dropout rate 0.3, second dropout rate 0.5, 512 neurons for the first dense layer, 256 neurons for the second dense layer, and RMSprop optimizer. This configuration achieved 99.41% accuracy on the validation set.

6.6.4. Data Augmentation Techniques

Enhanced data augmentation techniques were applied to increase the model's generalization capacity. Horizontal flipping (RandomFlip), random rotation (RandomRotation), random zoom (RandomZoom), contrast adjustment (RandomContrast), and brightness adjustment (RandomBrightness) were applied to training data in real-time. These techniques enhance the model's robustness against various imaging conditions.

6.6.5. Training Process and Callback Mechanisms

Final model training was conducted over a maximum of 50 epochs using optimal hyperparameters. Three key callback mechanisms were employed during training: EarlyStopping (patience=15, monitoring validation accuracy for early stopping), ModelCheckpoint (saving the best model), and ReduceLROnPlateau (halving the learning rate when validation loss does not improve, patience=5, minimum lr=1e-7). The class_weight parameter was used to address class imbalance.

The model achieved its best validation performance (99.61%) at epoch 19, and the EarlyStopping mechanism terminated training at epoch 34. Total training time (including hyperparameter search) was 31.12 minutes, final training time was 2.04 minutes, with an average time per epoch of 3.60 seconds.

6.6.6. Accuracy and Loss Graphs Analysis

Figure 6.6.1 shows the epoch-wise changes in accuracy and loss values during the model's training and validation process. In the accuracy graph, training accuracy started at approximately 55% in the first epoch, rapidly increased, and exceeded 98% around the 10th epoch. Validation accuracy started at 85% from the first epoch, clearly demonstrating the effect of transfer learning. Both curves tracked very closely in the 98-99% range after the 15th epoch, which is a strong indicator that the model did not overfit.

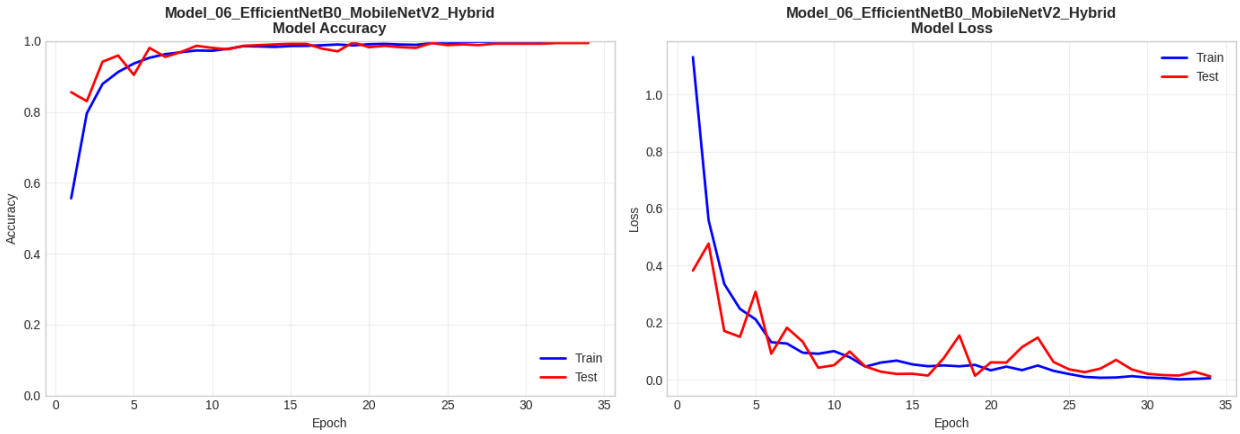

Figure 6.6.1. Eff*icientNetB0 + MobileNetV2 Hybrid Model Accuracy and Loss Graphs*

In the loss graph, the training loss rapidly decreased from initial high values (>1.0), falling below 0.1 after the 10th epoch. The validation loss showed some fluctuations, with sudden increases particularly at epochs 17-18 and 22-24. The ReduceLROnPlateau callback was triggered at epochs 24 and 29, reducing the learning rate, which helped decrease the fluctuations in loss values.

6.6.7. Confusion Matrix Analysis

Figure 6.6.2 presents the confusion matrix showing the model's performance on the test set. The matrix was evaluated on 512 test images. The high values on the diagonal indicate that the model made predictions with high accuracy for each class.

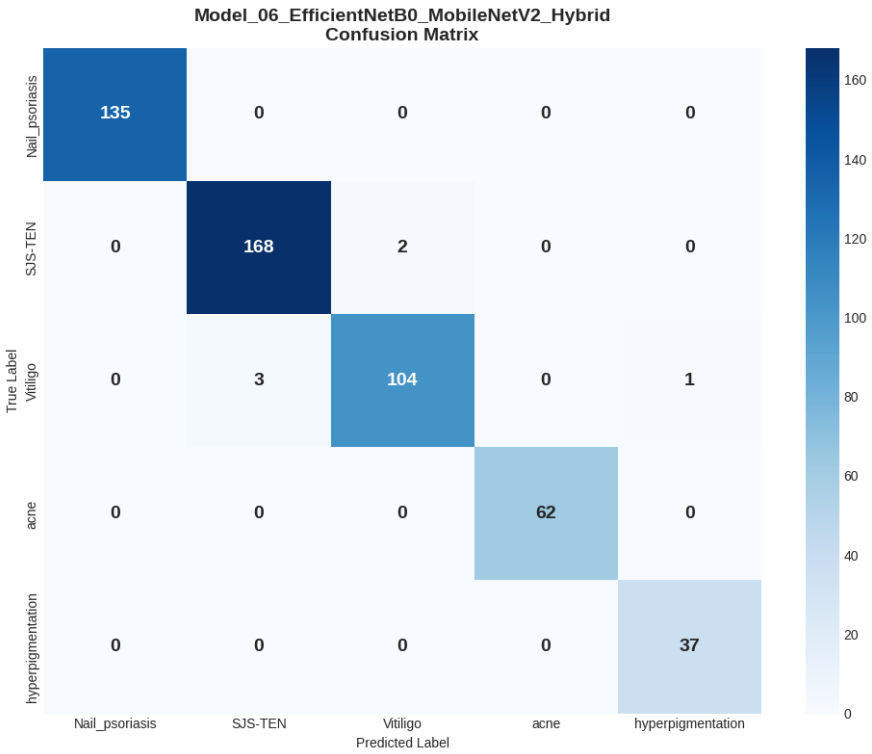

***Figure 6.6.2. Effi**cientNetB0 + MobileNetV2 Hybrid Model Confusion Matrix*

Class-wise analysis reveals: For the Nail_psoriasis class, all 135 samples were correctly classified (100% accuracy). For the SJS-TEN class, 168 out of 170 samples were correctly classified, with 2 misclassified as Vitiligo. For the Vitiligo class, 104 out of 108 samples were correctly classified, with 3 misclassified as SJS-TEN and 1 as Hyperpigmentation. For the Acne class, all 62 samples were correctly classified (100% accuracy). For the Hyperpigmentation class, all 37 samples were correctly classified (100% accuracy).

Only 6 out of 512 total samples were misclassified. The majority of misclassifications occurring between SJS-TEN and Vitiligo suggests that some visual features of these two diseases may exhibit similarities. The 100% accuracy achieved in Nail_psoriasis, Acne, and Hyperpigmentation classes is noteworthy.

6.6.8. ROC Curves and AUC Values

Figure 6.6.3 shows the model's ROC (Receiver Operating Characteristic) curves and AUC (Area Under Curve) values. The ROC curve visualizes the True Positive Rate (sensitivity) and False Positive Rate at different threshold values. An ideal classifier produces a curve close to the upper-left corner of the graph.

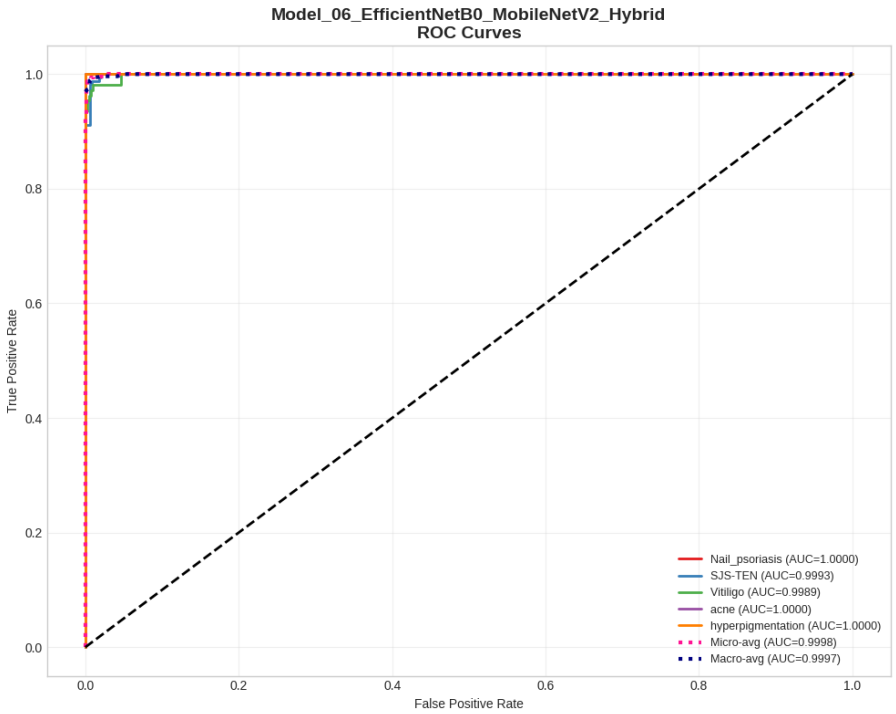

Figure 6.6.3. EfficientNetB0 + MobileNetV2 Hybrid Model ROC Curves

The ROC curves for all classes are very close to the upper-left corner, indicating near-perfect classification performance. Class-wise AUC values: Nail_psoriasis AUC=1.0000, SJS-TEN AUC=0.9993, Vitiligo AUC=0.9989, Acne AUC=1.0000, and Hyperpigmentation AUC=1.0000. The Macro-average AUC value is 0.9997 and the micro-average AUC value is 0.9998. These results demonstrate that the model can distinguish all classes with near-perfect accuracy.

6.6.9. Performance Metrics

Table 6.6.1. EfficientNetB0 + MobileNetV2 Hybrid Model Performance Metrics

Metric	Value	Percentage (%)
Accuracy	0.9883	98.83%
Precision (Weighted)	0.9883	98.83%
Recall (Weighted)	0.9883	98.83%
F1-Score (Weighted)	0.9883	98.83%
Cohen's Kappa	0.9845	98.45%
ROC AUC (Macro)	0.9997	99.97%

Table 6.6.1. EfficientNetB0 + MobileNetV2 Hybrid Model Performance Metrics

6.6.10. Model Flowchart

Figure 6.6.4 shows the architectural flowchart of the EfficientNetB0 + MobileNetV2 Hybrid Model. This diagram presents in detail all processing steps and layer structure from data input to the final classification output. As shown in the diagram, the input image is processed in two parallel branches. The left branch contains EfficientNetB0 (optimized with compound scaling) and the right branch contains MobileNetV2 (using depthwise separable convolution) backbone networks. The MobileNetV2 branch applies its own preprocessing function. Features undergo dimensionality reduction through GlobalAveragePooling2D, followed by parallel Dense and Dropout layers. The features merged at the Concatenate layer pass through the final classification head to produce 5-class output.

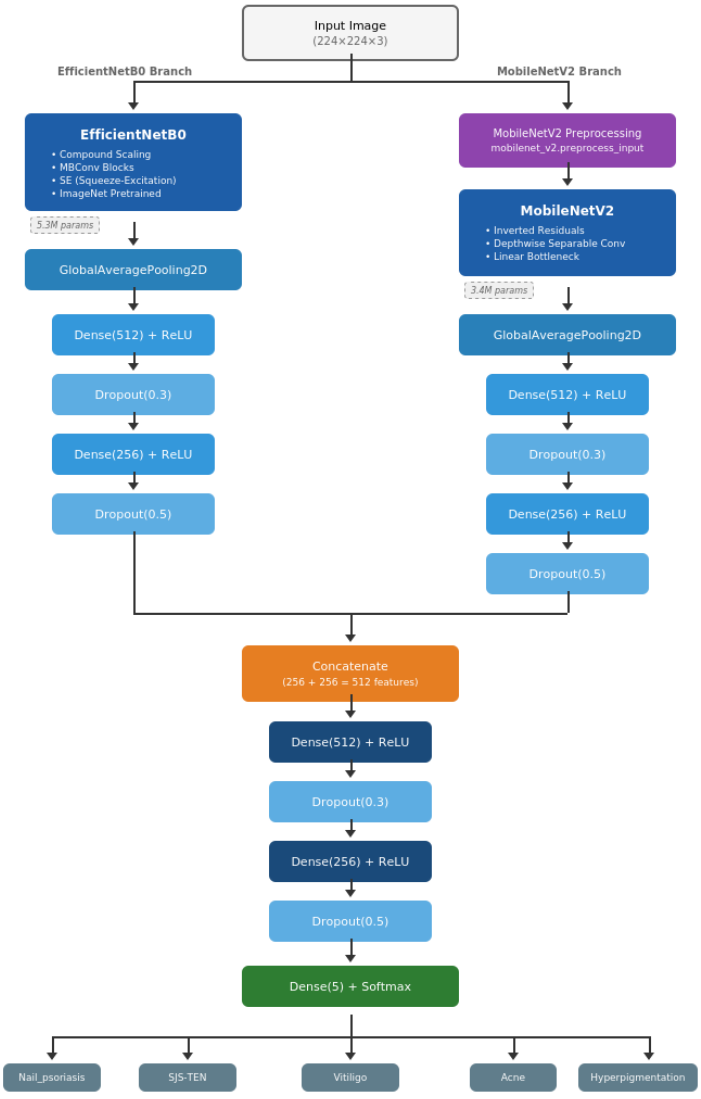

***Figure 6.6.4. Efficient**NetB0 + MobileNetV2 Hybrid Model Flowchart*

6.7. Xception + InceptionResNetV2 (Hybrid Model)

The Xception + InceptionResNetV2 Hybrid Model is the most comprehensive and parameter-wise largest dual-backbone architecture developed in this study. This model combines two powerful architectures with significant innovations in modern deep learning. The depthwise separable convolution efficiency of Xception and the power of InceptionResNetV2's inception modules and residual connections are combined to achieve rich feature extraction.

6.7.1. Theoretical Background

In this hybrid model, the pre-trained ImageNet weights of both backbone networks are used in a completely frozen state. Since both models require a 299×299 pixel input size, the dataset was resized to this resolution.

6.7.2. Model Architecture

The developed hybrid model is built on a dual-backbone architecture. The model architecture consists of the following key components:

- **Input Layer: RGB images** of 299×299×3 dimensions are fed into the model. This is the native resolution for both backbones.
- **Backbone 1 - Xception:** Pre-trained with ImageNet weights, fully frozen feature extractor. Its own preprocessing function is applied. Contains depthwise separable convolution blocks.
- **Backbone 2 - InceptionResNetV2:** Pre-trained with ImageNet weights, fully frozen feature extractor. Its own preprocessing function is applied. Contains inception modules and residual connections.
- **Global Average Pooling 2D:** Converts the feature maps from each backbone into one-dimensional vectors.
- **Parallel Classification Heads: Dense(512)** → Dropout(0.3) → Dense(256) → Dropout(0.5) structure for each backbone.
- **Concatenate Layer:** Merging of features from both backbones.
- **Final Classification Head:** Dense(512) → Dropout(0.3) → Dense(256) → Dropout(0.5) → Dense(5) + Softmax.

The model has a total of 77,692,173 parameters. Of these, 2,493,957 are trainable parameters, while the remaining 75,198,216 parameters are frozen in the backbone networks. This structure constitutes the largest hybrid model in the study and offers rich feature representations.

6.7.3. Hyperparameter Optimization

A comprehensive hyperparameter search was conducted using the Keras Tuner library with the Bayesian Optimization algorithm during model training. Different hyperparameter combinations were evaluated over 15 trials. The search space covered the following parameters: learning rate (0.01, 0.001, 0.0001), first dropout rate (0.2, 0.3, 0.4), second dropout rate (0.4, 0.5, 0.6), first dense layer neuron count (256, 512), second dense layer neuron count (128, 256), and optimizer type (Adam, SGD, RMSprop).

After a 77.55-minute search, Bayesian Optimization determined the optimal hyperparameter combination. The best results were obtained with the following parameters: learning rate 0.001, first dropout rate 0.3, second dropout rate 0.5, 512 neurons for the first dense layer, 256 neurons for the second dense layer, and Adam optimizer. This configuration achieved 97.46% accuracy on the validation set.

6.7.4. Data Augmentation Techniques

Enhanced data augmentation techniques were applied to increase the model's generalization capacity. Horizontal flipping (RandomFlip), random rotation (RandomRotation), random zoom (RandomZoom), contrast adjustment (RandomContrast), and brightness adjustment (RandomBrightness) were applied to training data in real-time. In this model, images were resized to 299×299 pixels.

6.7.5. Training Process and Callback Mechanisms

Final model training was conducted over a maximum of 50 epochs using optimal hyperparameters. Three key callback mechanisms were employed during training: EarlyStopping (patience=15, monitoring validation accuracy for early stopping), ModelCheckpoint (saving the best model), and ReduceLROnPlateau (halving the learning rate when validation loss does not improve, patience=5, minimum lr=1e-7). The class_weight parameter was used to address class imbalance.

The model achieved its best validation performance (99.02%) at epoch 27, and the EarlyStopping mechanism terminated training at epoch 42. Total training time (including hyperparameter search) was 86.01 minutes, final training time was 8.46 minutes, with an average time per epoch of 12.09 seconds. This duration is attributed to the model's large parameter count and 299×299 input size.

6.7.6. Accuracy and Loss Graphs Analysis

Figure 6.7.1 shows the epoch-wise changes in accuracy and loss values during the model's training and validation process. In the accuracy graph, training accuracy started at approximately 45% in the first epoch, gradually increased, and exceeded 98% around the 25th epoch. Validation accuracy started at approximately 70% from the first epoch and increased more rapidly, reaching 95% around the 10th epoch. Both curves tracked closely in the 98-99% range after the 25th epoch, indicating that the model did not overfit.

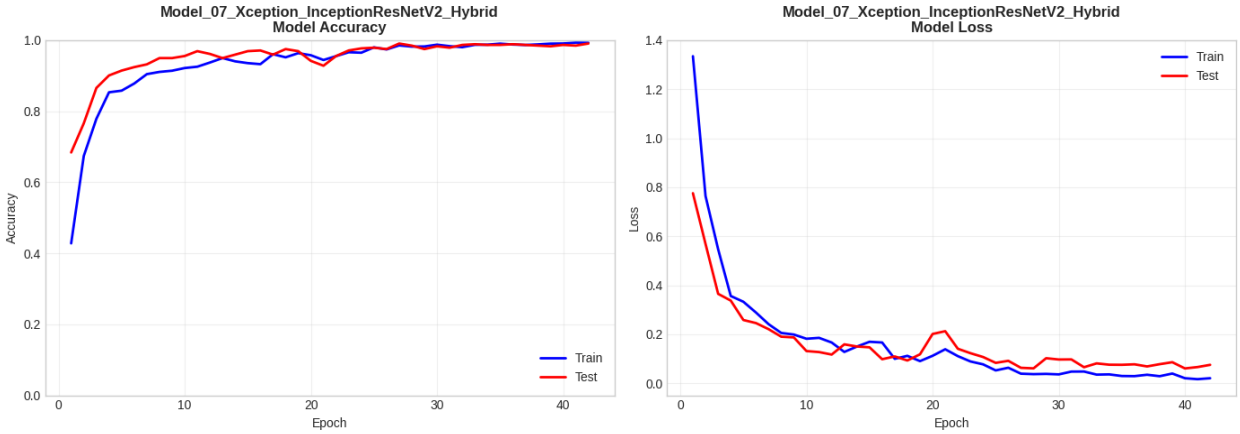

Figure 6.7.1. Xception + InceptionResNetV2 Hybrid Model Accuracy and Loss Graphs

In the loss graph, the training loss gradually decreased from initial high values (>1.3), falling below 0.05 after the 30th epoch. The validation loss showed some fluctuations, with sudden increases particularly at epochs 20-22. The ReduceLROnPlateau callback was triggered at epochs 23, 33, and 38, reducing the learning rate, which helped decrease the fluctuations in loss values.

6.7.7. Confusion Matrix Analysis

Figure 6.7.2 presents the confusion matrix showing the model's performance on the test set. The matrix was evaluated on 512 test images. The high values on the diagonal indicate that the model made predictions with high accuracy for each class.

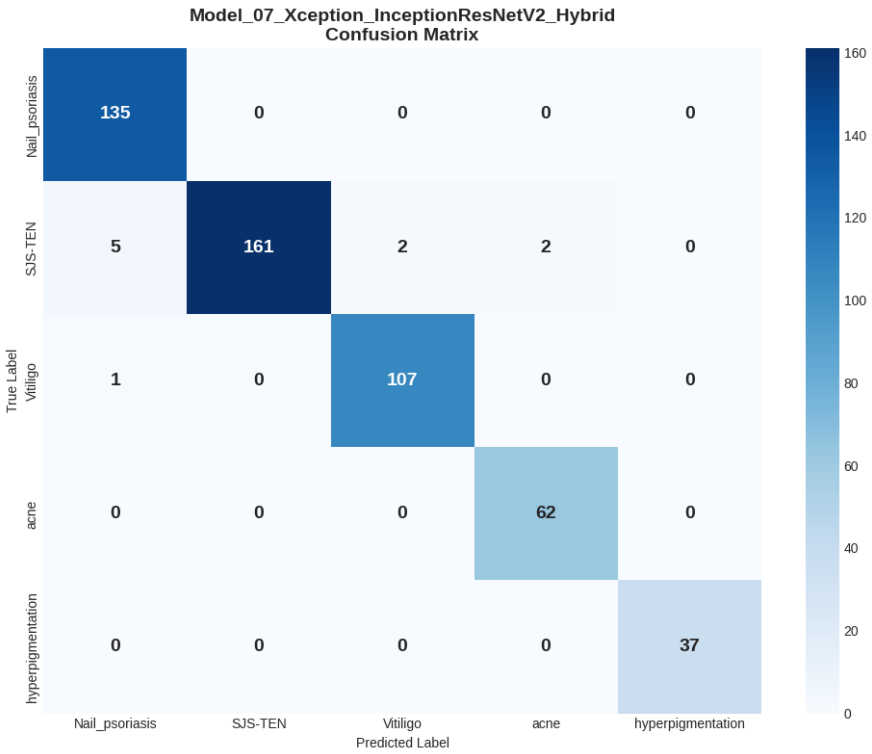

Figure 6.7.2. Xception + InceptionResNetV2 Hybrid Model Confusion Matrix

Class-wise analysis reveals: For the Nail_psoriasis class, all 135 samples were correctly classified (100% accuracy). For the SJS-TEN class, 161 out of 170 samples were correctly classified, with 5 misclassified as Nail_psoriasis, 2 as Vitiligo, and 2 as Acne. For the Vitiligo class, 107 out of 108 samples were correctly classified, with 1 misclassified as Nail_psoriasis. For the Acne class, all 62 samples were correctly classified (100% accuracy). For the Hyperpigmentation class, all 37 samples were correctly classified (100% accuracy).

A total of 10 out of 512 samples were misclassified. The majority of misclassifications occurring in the SJS-TEN class suggests that some visual features of this disease may exhibit similarities with other classes, particularly Nail_psoriasis. The 100% accuracy achieved in Nail_psoriasis, Acne, and Hyperpigmentation classes is noteworthy.

6.7.8. ROC Curves and AUC Values

Figure 6.7.3 shows the model's ROC (Receiver Operating Characteristic) curves and AUC (Area Under Curve) values. The ROC curve visualizes the True Positive Rate (sensitivity) and False Positive Rate at different threshold values. An ideal classifier produces a curve close to the upper-left corner of the graph.

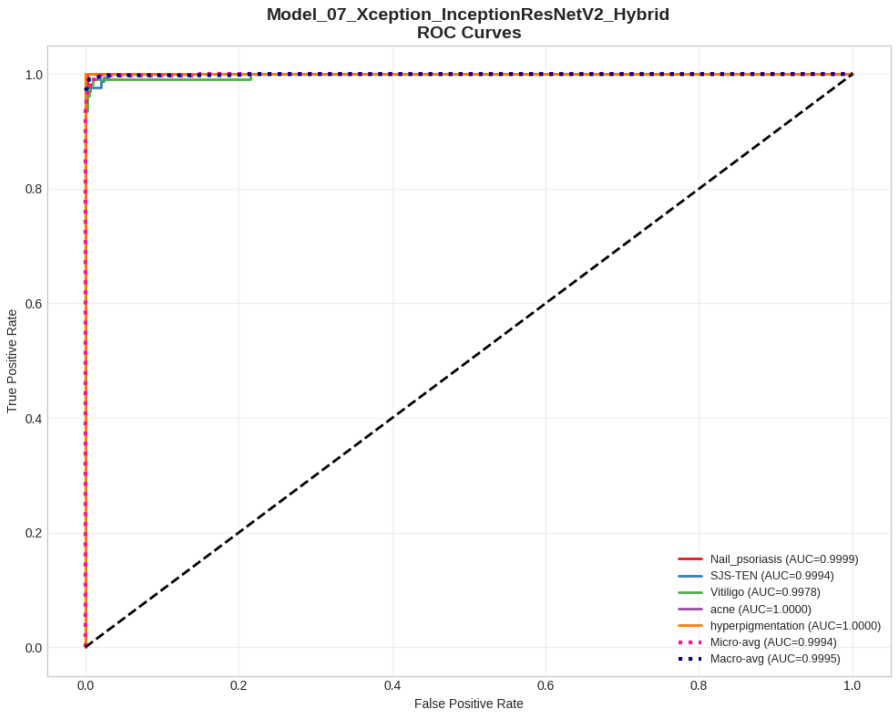

Figure 6.7.3. Xception + InceptionResNetV2 Hybrid Model ROC Curves

The ROC curves for all classes are very close to the upper-left corner, indicating near-perfect classification performance. Class-wise AUC values: Nail_psoriasis AUC=0.9999, SJS-TEN AUC=0.9994, Vitiligo AUC=0.9978, Acne AUC=1.0000, and Hyperpigmentation AUC=1.0000. The Macro-average AUC value is 0.9995 and the micro-average AUC value is 0.9994. These results demonstrate that the model can distinguish all classes with near-perfect accuracy.

6.7.9. Performance Metrics

Table 6.7.1. Xception + InceptionResNetV2 Hybrid Model Performance Metrics

Metric	Value	Yüzde (%)
Accuracy	0.9805	98.05%
Precision (Weighted)	0.9811	98.11%
Recall (Weighted)	0.9805	98.05%
F1-Score (Weighted)	0.9804	98.04%
Cohen's Kappa	0.9742	97.42%
ROC AUC (Macro)	0.9995	99.95%

Table 6.7.1. Xception + InceptionResNetV2 Hybrid Model Performance Metrics

6.7.10. Model Flowchart

Figure 6.7.4 shows the architectural flowchart of the Xception + InceptionResNetV2 Hybrid Model. This diagram presents in detail all processing steps and layer structure from data input to the final classification output. As shown in the diagram, the 299×299×3 input image is processed in two parallel branches. The left branch contains Xception (using depthwise separable convolution) and the right branch contains InceptionResNetV2 (containing inception modules and residual connections) backbone networks. Each backbone applies its own preprocessing function. Features undergo dimensionality reduction through GlobalAveragePooling2D, followed by parallel Dense and Dropout layers. The features merged at the Concatenate layer pass through the final classification head to produce 5-class output.

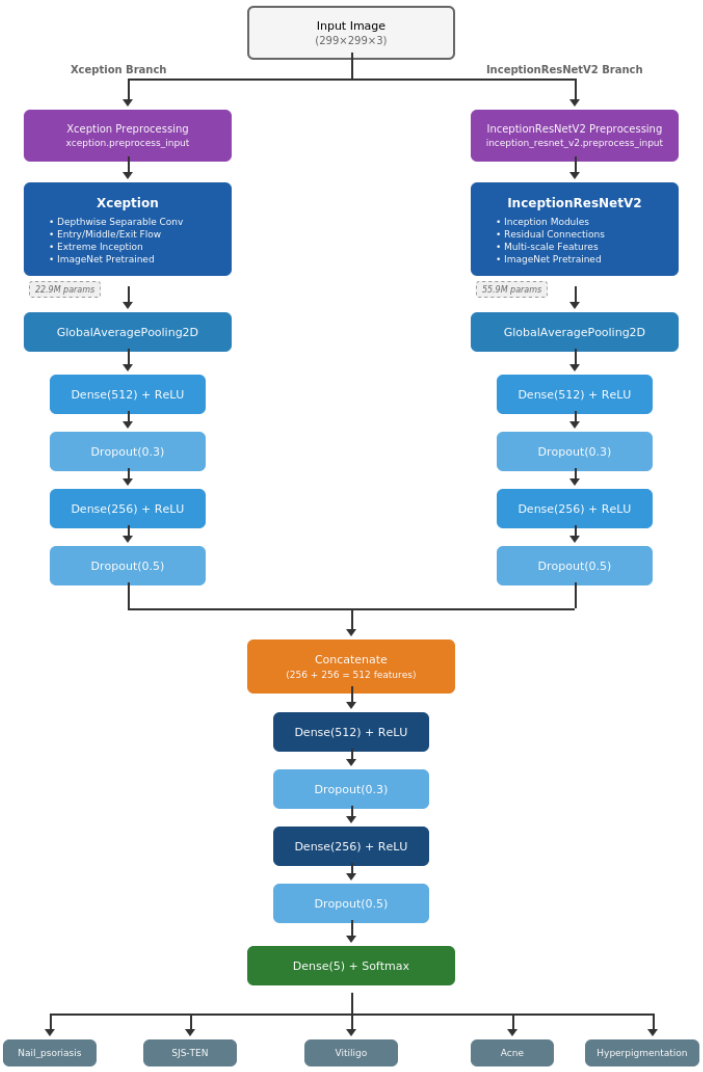

Figure 6.7.4. Xception + InceptionResNetV2 Hybrid Model Flowchart

6.8. InceptionResNetV2 + CBAM (Attention Mechanism)

The InceptionResNetV2 + CBAM Model is the first attention mechanism-based model developed in this study. This model combines a powerful backbone network, InceptionResNetV2, with the CBAM (Convolutional Block Attention Module) attention mechanism. CBAM aims to improve classification performance by enabling the model to focus on important regions and feature channels of the image.

6.8.1. Theoretical Background

In this model, the pre-trained ImageNet weights of the InceptionResNetV2 backbone network are used in a completely frozen state. The CBAM module is applied on the feature maps to learn attention weights.

6.8.2. Model Architecture

The developed attention mechanism-based model consists of the following key components:

- **Input Layer: RGB** images of 299×299×3 dimensions are fed into the model.
- **Preprocessing:** InceptionResNetV2's own preprocessing function is applied.
- **Backbone - InceptionResNetV2:** Pre-trained with ImageNet weights, fully frozen feature extractor. Contains inception modules and residual connections.
- **CBAM Attention** Module:
    - **Channel Attention: Global** Average/Max Pooling → Shared MLP (512/ratio → 512) → Sigmoid
    - **Spatial Attention: Channel** Average/Max → Concatenate → Conv2D(7×7) → Sigmoid
    - CBAM Ratio: 16
- **Global Average Pooling 2D: Converts** attention-applied feature maps into a one-dimensional vector.
- **Classification Head:** Dense(512) → Dropout(0.4) → Dense(256) → Dropout(0.5) → Dense(5) + Softmax.

The model has a total of 55,552,936 parameters. Of these, 1,216,200 are trainable parameters, while the remaining 54,336,736 parameters are frozen in the backbone network. The CBAM module and classification head constitute the trainable parameters.

6.8.3. Hyperparameter Optimization

A comprehensive hyperparameter search was conducted using the Keras Tuner library with the Bayesian Optimization algorithm during model training. Different hyperparameter combinations

were evaluated over 15 trials. The search space covered the following parameters: learning rate (0.01, 0.001, 0.0001), first dropout rate (0.2, 0.3, 0.4), second dropout rate (0.4, 0.5, 0.6), first dense layer neuron count (256, 512), second dense layer neuron count (128, 256), optimizer type (Adam, SGD, RMSprop), and CBAM ratio (8, 16).

After a 67.49-minute search, Bayesian Optimization determined the optimal hyperparameter combination. The best results were obtained with the following parameters: learning rate 0.001, first dropout rate 0.4, second dropout rate 0.5, 512 neurons for the first dense layer, 256 neurons for the second dense layer, RMSprop optimizer, and CBAM ratio 16. This configuration achieved 97.07% accuracy on the validation set.

6.8.4. Data Augmentation Techniques

In this model, reduced augmentation techniques were applied for more stable training. Horizontal flipping (RandomFlip), random rotation (RandomRotation, 0.10), random zoom (RandomZoom, 0.10), contrast adjustment (RandomContrast, 0.05), and brightness adjustment (RandomBrightness, 0.05) were applied to training data. These values were reduced from the 0.15 and 0.10 rates in previous models to enable more stable learning of the attention mechanism.

6.8.5. Training Process and Callback Mechanisms

Final model training was conducted over a maximum of 50 epochs using optimal hyperparameters. Three key callback mechanisms were employed during training: EarlyStopping (patience=15, monitoring validation accuracy for early stopping), ModelCheckpoint (saving the best model), and ReduceLROnPlateau (halving the learning rate when validation loss does not improve, patience=5, minimum lr=1e-7). The class_weight parameter was used to address class imbalance.

The model achieved its best validation performance (98.63%) at epoch 36, and training continued for all 50 epochs. Total training time (including hyperparameter search) was 76.15 minutes, final training time was 8.66 minutes, with an average time per epoch of 10.39 seconds.

6.8.6. Accuracy and Loss Graphs Analysis

Figure 6.8.1 shows the epoch-wise changes in accuracy and loss values during the model's training and validation process. In the accuracy graph, training accuracy started at approximately 55% in the first epoch, gradually increased, and exceeded 97% around the 20th epoch. Validation accuracy started at approximately 77% from the first epoch and exceeded 97% after the learning rate was reduced at the 17th epoch. Both curves tracked closely in the 97-99% range after the 20th epoch, indicating that the model did not overfit.

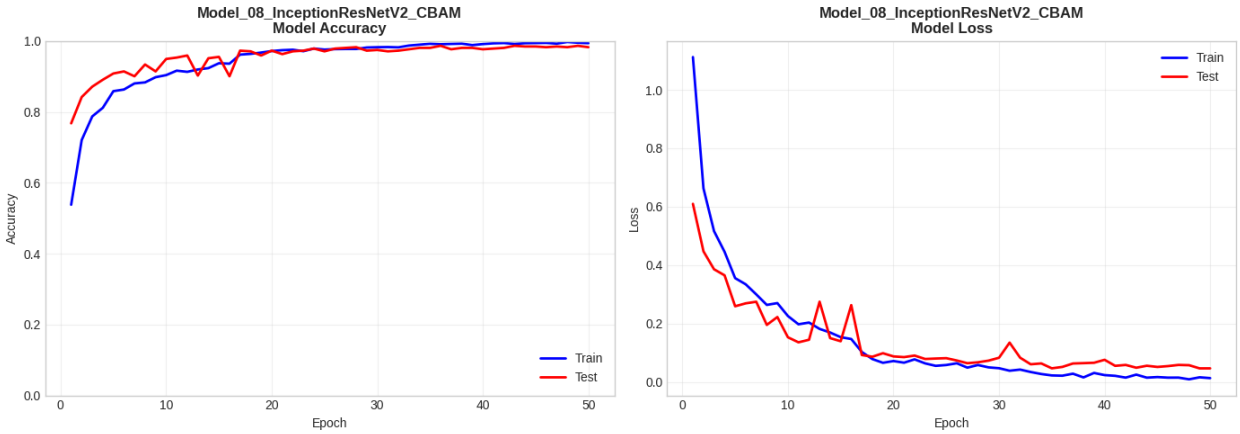

Figure 6.8.1. InceptionResNetV2 + CBAM Model Accuracy and Loss Graphs

In the loss graph, the training loss gradually decreased from initial high values (>1.0), falling below 0.05 after the 20th epoch. The validation loss showed some fluctuations, with sudden increases particularly at epochs 13 and 16. The ReduceLROnPlateau callback was triggered at epochs 16, 32, 40, and 45, reducing the learning rate, which helped decrease the fluctuations in loss values.

6.8.7. Confusion Matrix Analysis

Figure 6.8.2 presents the confusion matrix showing the model's performance on the test set. The matrix was evaluated on 512 test images. The high values on the diagonal indicate that the model made predictions with high accuracy for each class.

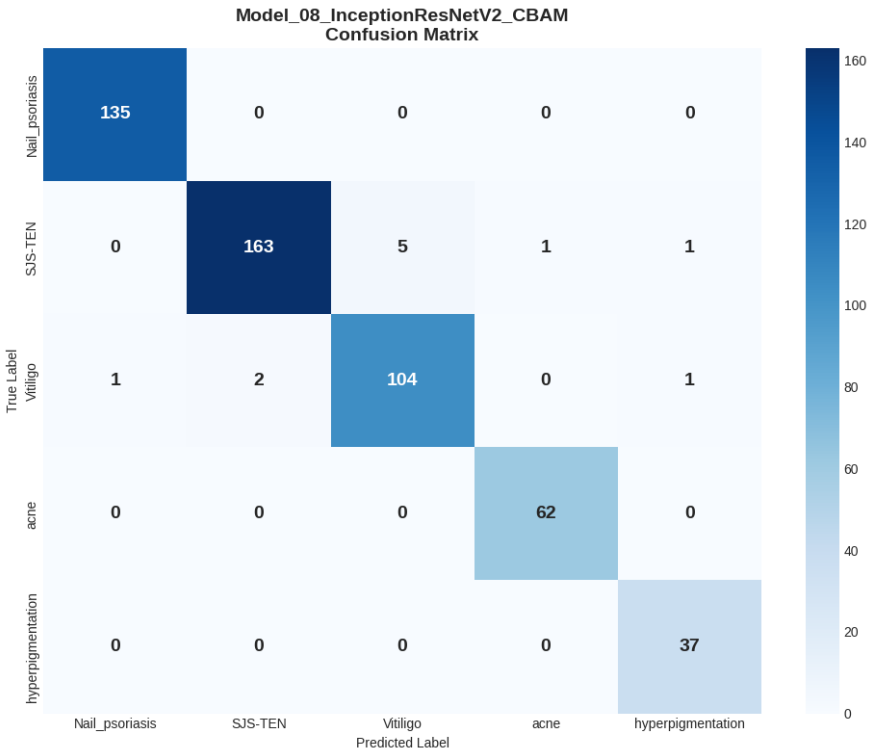

Figure 6.8.2. InceptionResNetV2 + CBAM Model Confusion Matrix

Class-wise analysis reveals: For the Nail_psoriasis class, all 135 samples were correctly classified (100% accuracy). For the SJS-TEN class, 163 out of 170 samples were correctly classified, with 5 misclassified as Vitiligo, 1 as Acne, and 1 as Hyperpigmentation. For the Vitiligo class, 104 out of 108 samples were correctly classified, with 1 misclassified as Nail_psoriasis, 2 as SJS-TEN, and 1 as Hyperpigmentation. For the Acne class, all 62 samples were correctly classified (100% accuracy). For the Hyperpigmentation class, all 37 samples were correctly classified (100% accuracy).

A total of 11 out of 512 samples were misclassified. The majority of misclassifications occurring between SJS-TEN and Vitiligo suggests that some visual features of these two diseases may exhibit similarities. The 100% accuracy achieved in Nail_psoriasis, Acne, and Hyperpigmentation classes is noteworthy.

6.8.8. ROC Curves and AUC Values

Figure 6.8.3 shows the model's ROC (Receiver Operating Characteristic) curves and AUC (Area Under Curve) values. The ROC curve visualizes the True Positive Rate (sensitivity) and False Positive Rate at different threshold values. An ideal classifier produces a curve close to the upper-left corner of the graph.

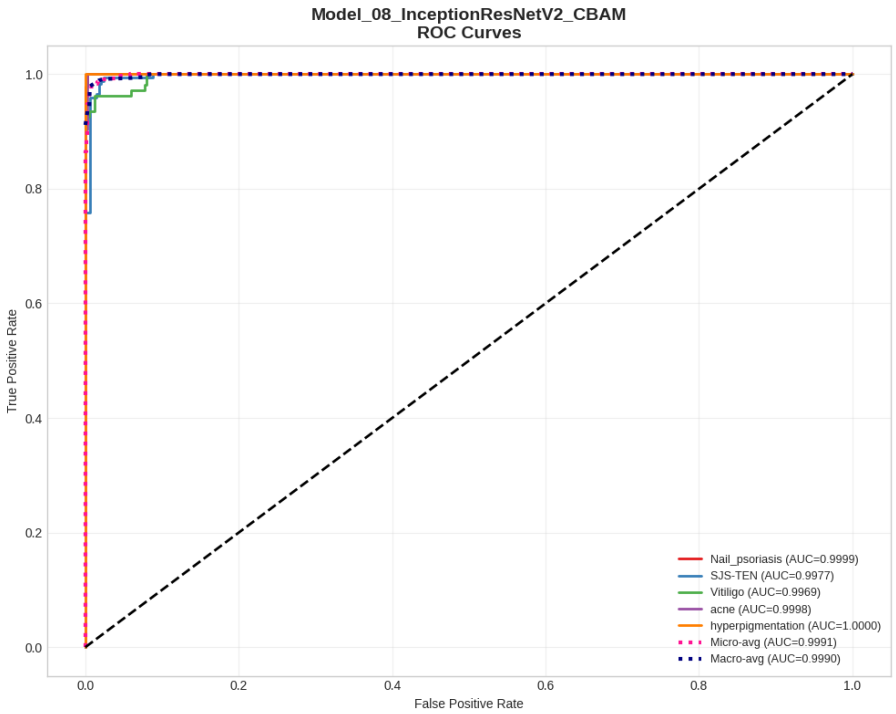

Figure 6.8.3. InceptionResNetV2 + CBAM Model ROC Curves

The ROC curves for all classes are very close to the upper-left corner, indicating near-perfect classification performance. Class-wise AUC values: Nail_psoriasis AUC=0.9999, SJS-TEN AUC=0.9977, Vitiligo AUC=0.9969, Acne AUC=0.9998, and Hyperpigmentation AUC=1.0000. The Macro-average AUC value is 0.9990 and the micro-average AUC value is 0.9991. These results demonstrate that the CBAM attention mechanism improved the model's capacity to distinguish between classes.

6.8.9. Performance Metrics

Metric	Value	Percentage (%)
Accuracy	0.9785	97.85%
Precision (Weighted)	0.9787	97.87%
Recall (Weighted)	0.9785	97.85%
F1-Score (Weighted)	0.9785	97.85%
Cohen's Kappa	0.9716	97.16%
ROC AUC (Macro)	0.9990	99.90%

Table 6.8.1. InceptionResNetV2 + CBAM Model Performance Metrics

6.8.10. Model Flowchart

Figure 6.8.4 shows the architectural flowchart of the InceptionResNetV2 + CBAM Model. This diagram presents in detail all processing steps and layer structure from data input to the final classification output. As shown in the diagram, the 299×299×3 input image first passes through InceptionResNetV2 preprocessing, then feature extraction is performed by the frozen InceptionResNetV2 backbone. The extracted features enter the CBAM attention module, where Channel Attention and Spatial Attention are applied sequentially. The attention-weighted features undergo dimensionality reduction through GlobalAveragePooling2D and pass through the classification head to produce 5-class output.

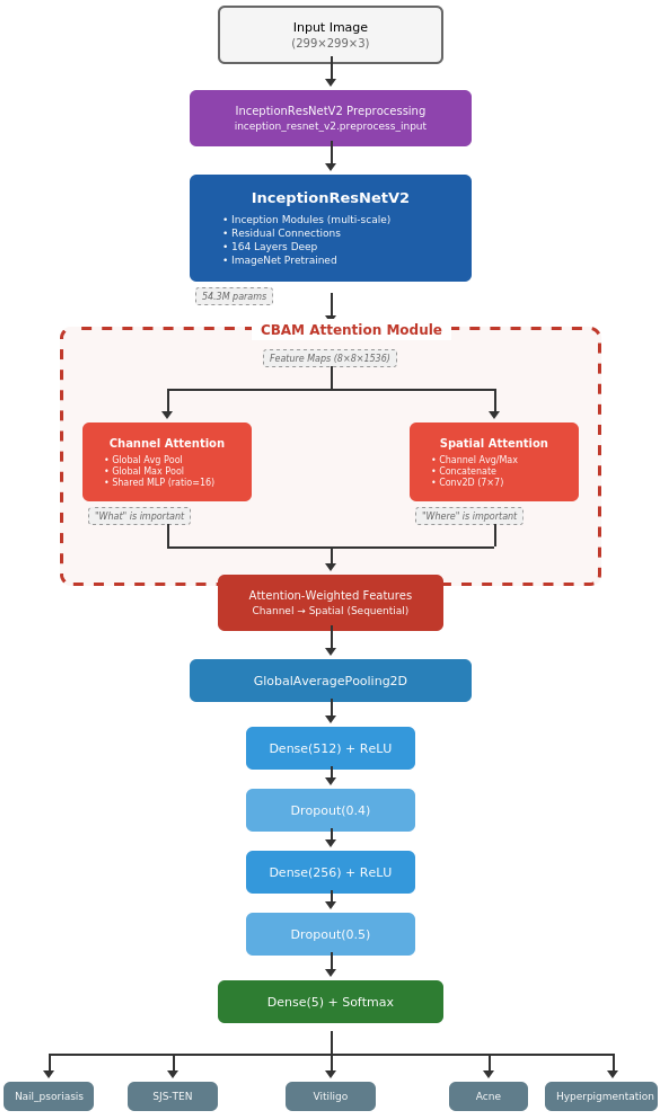

Figure 6.8.4. InceptionResNetV2 + CBAM Model Flowchart

6.9. Xception + FPN (Multi-Scale Feature Extraction)

The Xception + FPN Model is a multi-scale feature extraction-based model developed in this study. This model combines a powerful backbone network, Xception, with the FPN (Feature Pyramid Network) architecture. FPN enables effective detection of both small and large skin lesions by combining features at different scales.

6.9.1. Theoretical Background

In this model, the pre-trained ImageNet weights of the Xception backbone network are used in a completely frozen state. The FPN module extracts feature maps at three different levels (C3, C4, C5) from Xception to construct P3, P4, P5 pyramid levels.

6.9.2. Model Architecture

The developed multi-scale classification model consists of the following key components:

- **Input Layer: RGB** images of 299×299×3 dimensions are fed into the model.
- **Preprocessing:** Xception's own preprocessing function is applied.
- **Backbone - Xception:** Pre-trained with ImageNet weights, fully frozen feature extractor. Feature maps are extracted from three different levels:
    - C3: From the block4_sepconv2_bn layer
    - C4: From the block13_sepconv2_bn layer
    - C5: From the block14_sepconv2_act layer
- **FPN** Module:
    - **Lateral Connections: 1×1** Conv2D(256) for dimension matching at each level
    - **Top-Down Pathway: Bilinear** interpolation upsampling and Add operation
    - **Smooth Layers: 3×3** Conv2D(256) for smoothing at each level
    - P5, P4, P3 pyramid levels are constructed
- **Multi-Scale** Head:
    - GlobalAveragePooling2D from each pyramid level
    - Concatenation for merging (256×3 = 768 features)
- **Classification Head:** Dense(256) → Dropout(0.4) → Dense(128) → Dropout(0.4) → Dense(5) + Softmax.

The model has a total of 23,835,693 parameters. Of these, 2,974,213 are trainable parameters, while the remaining 20,861,480 parameters are frozen in the backbone network.

6.9.3. Hyperparameter Configuration

In this model, fixed hyperparameters were used instead of the Keras Tuner hyperparameter search used in previous models. The model configuration was created with the following parameters: learning rate 0.001, dropout rate 0.4, 256 neurons for the first dense layer, 128 neurons for the second dense layer, and Adam optimizer. 256 filters were used in FPN lateral connections and smooth layers.

6.9.4. Data Augmentation Techniques

Standard data augmentation techniques were applied to increase the model's generalization capacity. Horizontal flipping (RandomFlip), random rotation (RandomRotation), random zoom (RandomZoom), contrast adjustment (RandomContrast), and brightness adjustment (RandomBrightness) were applied to training data. Images were resized to 299×299 pixels.

6.9.5. Training Process and Callback Mechanisms

Final model training was conducted over a maximum of 40 epochs. Three key callback mechanisms were employed during training: EarlyStopping (patience=10, monitoring validation accuracy for early stopping), ModelCheckpoint (saving the best model), and ReduceLROnPlateau (halving the learning rate when validation loss does not improve, patience=3). The class_weight parameter was used to address class imbalance. Additionally, the jit_compile=False setting was used to prevent GPU compatibility issues.

The model achieved its best validation performance (98.83%) at epoch 22, and the EarlyStopping mechanism terminated training at epoch 32. Total training time was 4.98 minutes, with an average time per epoch of approximately 9.3 seconds. This fast training time is attributed to the efficient structure of the FPN architecture.

6.9.6. Accuracy and Loss Graphs Analysis

Figure 6.9.1 shows the epoch-wise changes in accuracy and loss values during the model's training and validation process. In the accuracy graph, training accuracy started at approximately 55% in the first epoch, rapidly increased, and exceeded 97% around the 15th epoch. Validation accuracy started at 85% from the first epoch, clearly demonstrating the effect of transfer learning and FPN's multi-scale feature extraction. Both curves tracked very closely in the 97-99% range after the 15th epoch, which is a strong indicator that the model did not overfit.

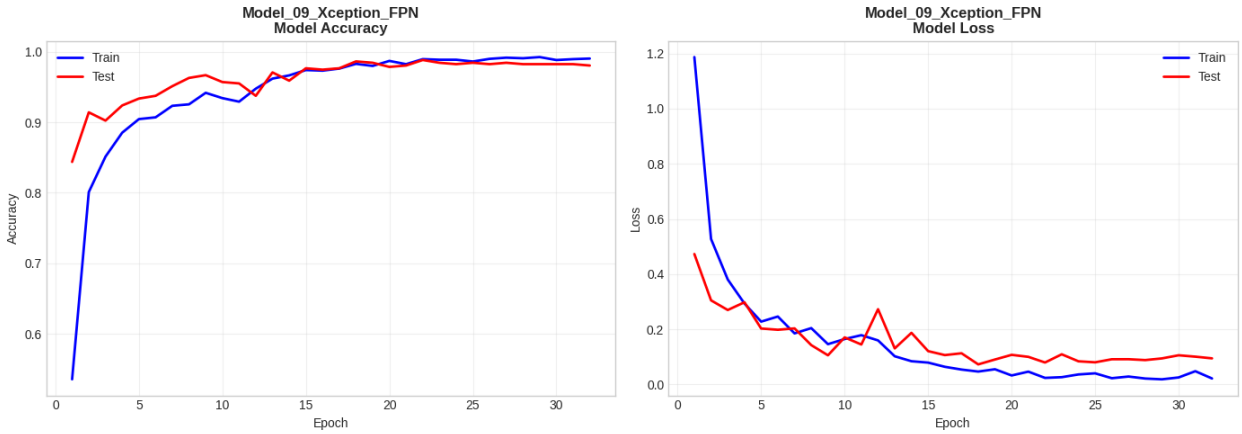

Figure 6.9.1. Xception + FPN Model *Accuracy and Loss Graphs*

In the loss graph, the training loss rapidly decreased from initial high values (>1.1), falling below 0.05 after the 15th epoch. The validation loss showed some fluctuations, with increases particularly at epochs 10-12. The ReduceLROnPlateau callback was triggered at epochs 12, 15, 21, 24, 27, and 30, reducing the learning rate, which allowed the model to perform finer tuning.

6.9.7. Confusion Matrix Analysis

Figure 6.9.2 presents the confusion matrix showing the model's performance on the test set. The matrix was evaluated on 512 test images. The high values on the diagonal indicate that the model made predictions with high accuracy for each class.

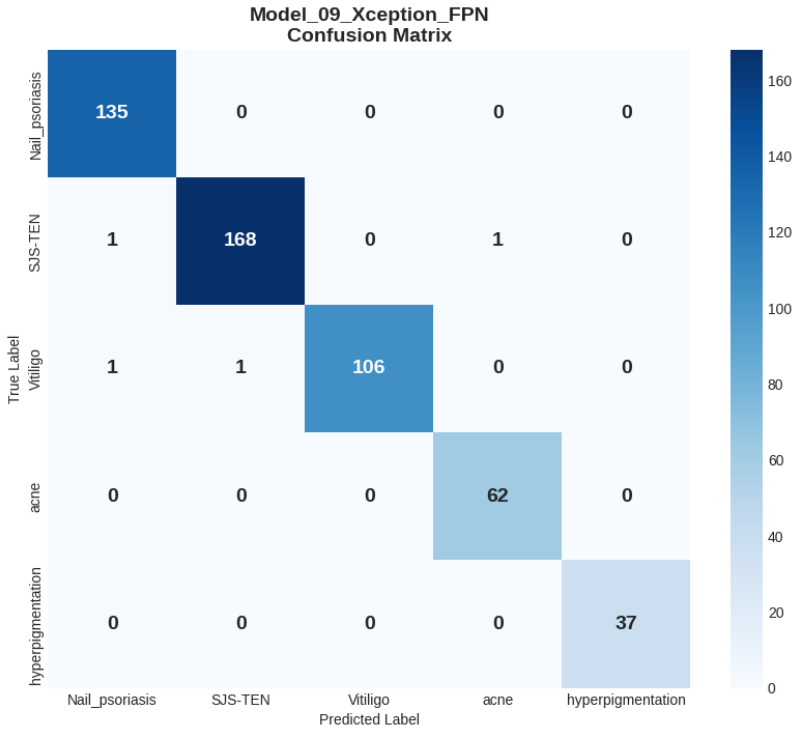

Figure 6.9.2. Xception + FPN Model *Confusion Matrix*

Class-wise analysis reveals: For the Nail_psoriasis class, all 135 samples were correctly classified (100% accuracy). For the SJS-TEN class, 168 out of 170 samples were correctly classified, with 1 misclassified as Nail_psoriasis and 1 as Acne. For the Vitiligo class, 106 out of 108 samples were correctly classified, with 1 misclassified as Nail_psoriasis and 1 as SJS-TEN. For the Acne class, all 62 samples were correctly classified (100% accuracy). For the Hyperpigmentation class, all 37 samples were correctly classified (100% accuracy).

Only 4 out of 512 total samples were misclassified. This is one of the lowest error counts in the study. FPN's multi-scale feature extraction contributed to better recognition of lesions of different sizes. The 100% accuracy achieved in Nail_psoriasis, Acne, and Hyperpigmentation classes is noteworthy.

6.9.8. ROC Curves and AUC Values

Figure 6.9.3 shows the model's ROC (Receiver Operating Characteristic) curves and AUC (Area Under Curve) values. The ROC curve visualizes the True Positive Rate (sensitivity) and False Positive Rate at different threshold values. An ideal classifier produces a curve close to the upper-left corner of the graph.

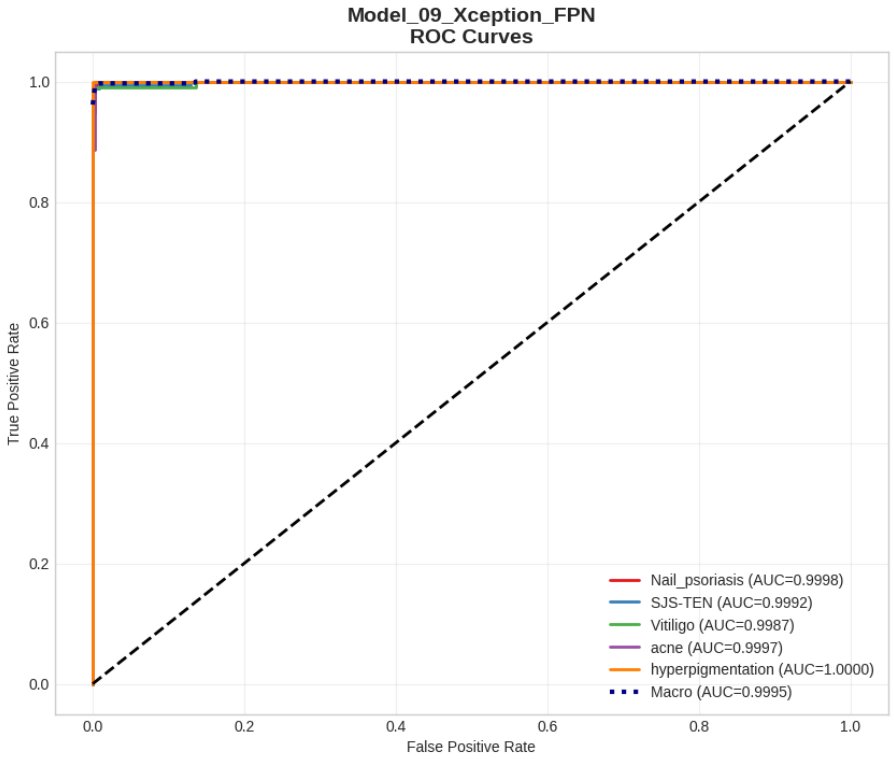

***Figure 6.9.3. Xception + FPN Model** ROC Curves*

The ROC curves for all classes are very close to the upper-left corner, indicating near-perfect classification performance. Class-wise AUC values: Nail_psoriasis AUC=0.9998, SJS-TEN AUC=0.9992, Vitiligo AUC=0.9987, Acne AUC=0.9997, and Hyperpigmentation AUC=1.0000. The Macro-average AUC value is 0.9995. These results demonstrate that FPN architecture's multi-scale feature extraction made a significant contribution to classification performance.

6.9.9. Performance Metrics

Metric	Value	Percentage (%)
Accuracy	0.9922	99.22%
F1-Score (Weighted)	0.9922	99.22%
ROC AUC (Macro)	0.9995	99.95%

***Table 6.9.1. Xception + FPN Model** Performance Metrics*

6.9.10. Model Flowchart

Figure 6.9.4 shows the architectural flowchart of the Xception + FPN Model. This diagram presents in detail all processing steps and layer structure from data input to the final classification output. As shown in the diagram, the 299×299×3 input image first passes through Xception preprocessing, then feature maps are extracted at three different levels (C3, C4, C5) from the frozen Xception backbone. In the FPN module, dimension matching is performed through lateral connections, high-level features are transferred to lower levels through the top-down pathway, and pyramid levels (P3, P4, P5) are constructed through smooth layers. Features are extracted from each pyramid level via GlobalAveragePooling2D, merged through Concatenation, and passed through the classification head to produce 5-class output.

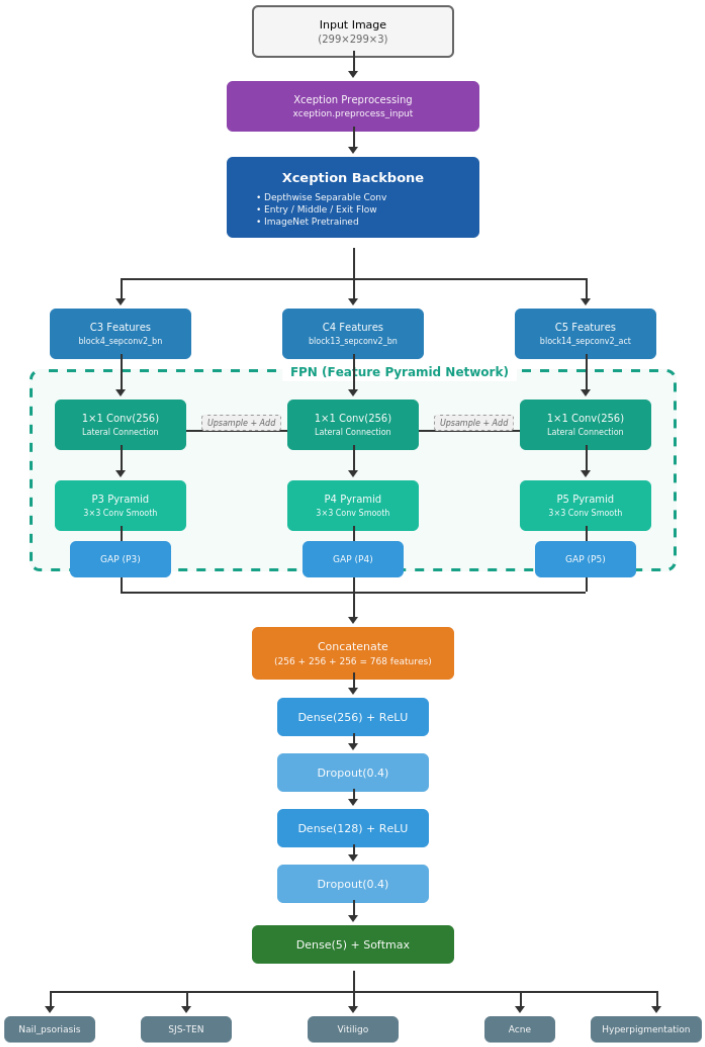

***Figure 6.9.4. Xception + FPN Model** Flowchart*

6.10. SalvationNet: Tri-Domain Adaptive Fusion Network

Tri-Domain Adaptive Fusion Network for Skin Disease Classification – Author: Ömer Faruk KURTULUŞ

SalvationNet is a novel deep learning architecture entirely designed and developed by Ömer Faruk Kurtuluş within the scope of this study. This model was built from scratch (*from scratch*) without utilizing any pre-existing network or pre-trained weights available in the literature. The fundamental philosophy of SalvationNet is based on a tri-domain parallel processing approach that emulates the dermatological examination process.

6.10.1. Theoretical Background and Dermatological Motivation

In dermatological examination, specialist physicians follow a systematic approach when evaluating skin lesions. This approach consists of three fundamental components:

- **Color Analysis:** Conditions such as Vitiligo and Hyperpigmentation are primarily characterized by color changes.
- **Texture Analysis:** Conditions such as Acne and Nail Psoriasis exhibit distinctive texture changes.
- **Morphological Analysis:** Severe conditions such as Stevens-Johnson Syndrome present characteristic morphological features.

6.10.2. Model Architecture

The SalvationNet architecture is designed in a modular and hierarchical structure. The model accepts RGB images of 224×224×3 dimensions as input:

- **Stem Block:** Conv(32,3×3,s=2) → Conv(32) → Conv(64) → MaxPool. Output: 56×56×64
- **Color Domain Branch (CDB):** Color analysis with 1×1 + 3×3 + 5×5 Conv. Target: Vitiligo, Hyperpigmentation
- **Texture Domain Branch (TDB):** Texture analysis with SepConv 3×3 + 5×5 + 7×7. Target: Acne, Nail Psoriasis
- **Morphology Domain Branch (MDB):** Structural analysis with Edge + Dilated + AvgPool. Target: SJS-TEN
- **Cross-Domain Attention (CDA):** Color→Texture, Texture→Morph, Color→Morph attention mechanisms
- **Adaptive Domain Fusion Module (ADFM):** Dynamic weighting via GAP → Concat → Dense(128) → Dense(3, Softmax)
- **Hierarchical Feature Refinement (HFR):** Dense(512)+Dropout(0.4) → Dense(192)+Dropout(0.3)

6.10.3. Hyperparameter Optimization

15 trials were conducted using Keras Tuner Bayesian Optimization. Optimal parameters: Base Filters=80, Learning Rate=0.0005, Dropout1=0.4, Dropout2=0.3, Dense1=512, Dense2=192, Optimizer=RMSprop. Total search duration was 32.33 minutes.

6.10.4. Data Augmentation Techniques

Moderate data augmentation for from-scratch training: RandomFlip (horizontal), RandomRotation (0.10), RandomZoom (0.10), RandomContrast (0.05), RandomBrightness (0.05).

6.10.5. Training Process

Final training utilized a maximum of 80 epochs, EarlyStopping (patience=20), ModelCheckpoint, and ReduceLROnPlateau (patience=7, factor=0.5). The model reached its best performance at epoch 57 and was early-stopped at epoch 77. Total duration: 39.38 minutes, per epoch: 5.50 seconds.

6.10.6. Accuracy and Loss Curves Analysis

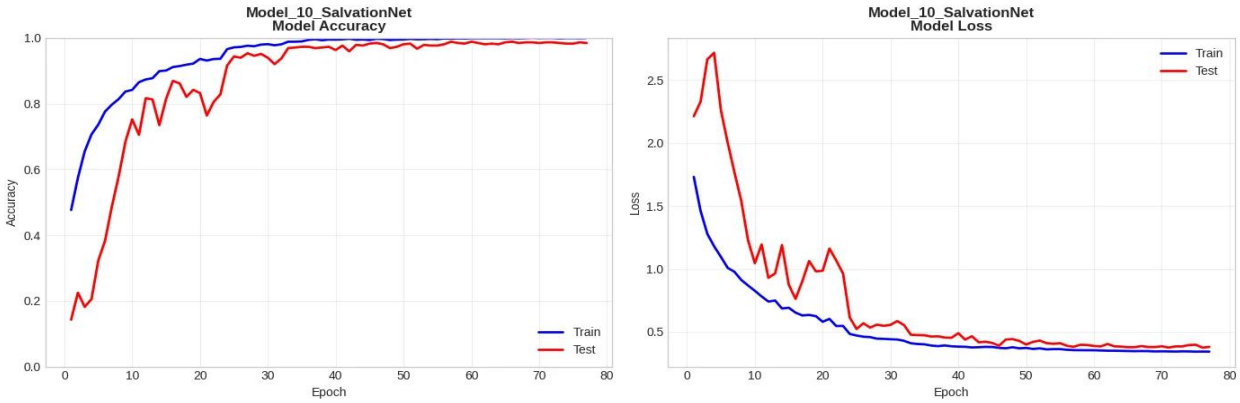

Figure 6.10.1. *SalvationNet Model Accuracy and Loss Curves*

Figure 6.10.1 displays the accuracy and loss values during the model's training and validation process. Training accuracy started at 45% and progressively rose above 99%. Validation accuracy began at 14% in the first epoch and reached 98.83% at epoch 57. The convergence of training and validation curves indicates that the model generalizes well.

6.10.7. Confusion Matrix Analysis

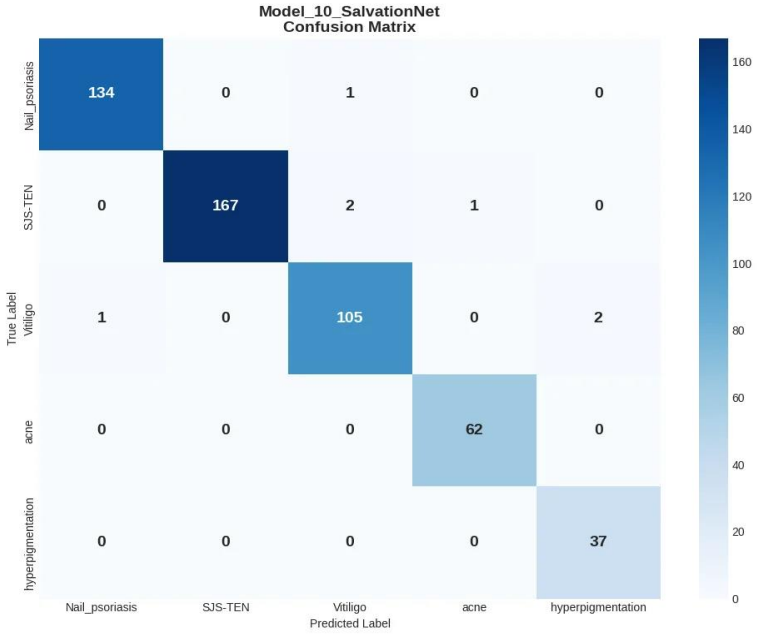

Figure 6.10.2. *SalvationNet Model Confusion Matrix*

Class-level analysis:

- **Nail_psoriasis:** 134/135 correct (99.26%), 1 Vitiligo error
- **SJS-TEN:** 167/170 correct (98.24%), 2 Vitiligo + 1 Acne errors
- **Vitiligo:** 105/108 correct (97.22%), 1 Nail_psoriasis + 2 Hyperpigmentation errors
- **Acne:** 62/62 correct (100%)
- **Hyperpigmentation:** 37/37 correct (100%)

Only 7 errors out of 512 total samples. The 100% accuracy in Acne and Hyperpigmentation classes validates the effectiveness of the tri-domain feature extraction strategy.

6.10.8. ROC Curves and AUC Values

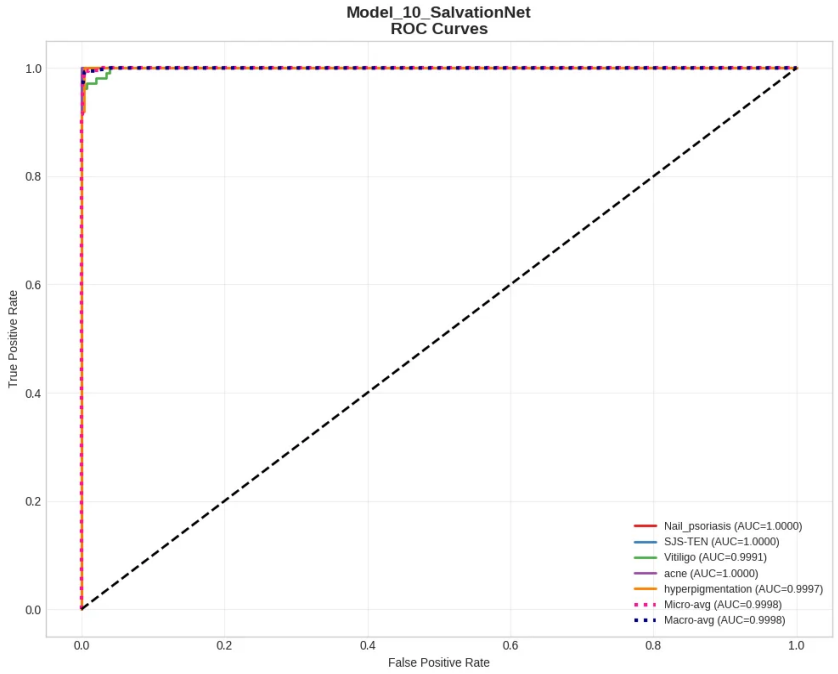

Figure 6.10.3. *SalvationNet Model ROC Curves*

Class-level AUC values:

- Nail_psoriasis: AUC = 1.0000
- SJS-TEN: AUC = 1.0000
- Vitiligo: AUC = 0.9991
- Acne: AUC = 1.0000
- Hyperpigmentation: AUC = 0.9997
- **Macro-average AUC: 0.9998**
- **Micro-average AUC: 0.9998**

6.10.9. Performance Metrics

Metric	Value	Percentage (%)
Accuracy	0.9863	98.63%
Precision (Weighted)	0.9866	98.66%
Recall (Weighted)	0.9863	98.63%
F1-Score (Weighted)	0.9864	98.64%
Cohen's Kappa	0.9819	98.19%
ROC AUC (Macro)	0.9998	99.98%

Table 6.10.1. *SalvationNet Model Performance Metrics*

6.10.10. Model Flowchart

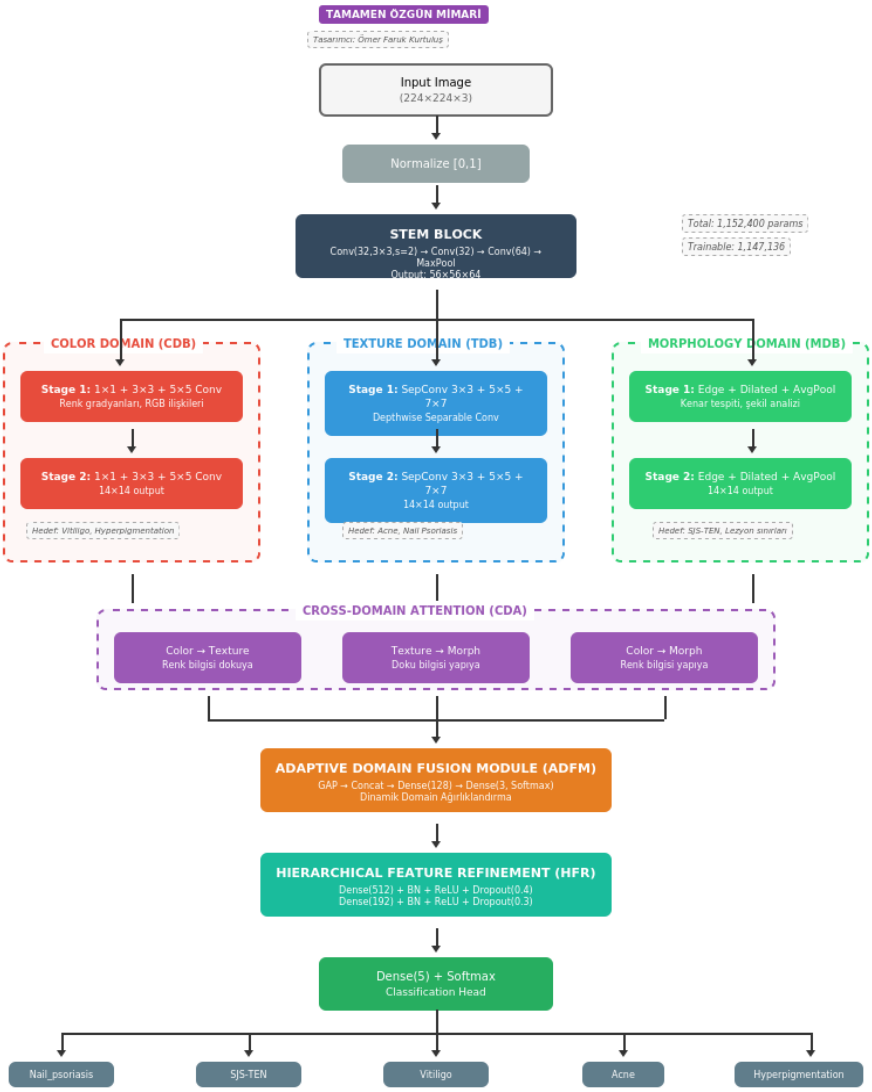

Figure 6.10.4. *SalvationNet Model Flowchart*

Figure 6.10.4 presents the architectural flowchart of the SalvationNet model. The 224×224×3 input passes through the Stem Block and is then split into three parallel branches (Color, Texture, Morphology). Cross-Domain Attention enables inter-branch information exchange, the Adaptive Domain Fusion Module performs dynamic weighting, and finally a 5-class output is produced.

6.10.11. Conclusion and Evaluation

SalvationNet is a novel deep learning architecture entirely designed and developed by Ömer Faruk Kurtuluş. The model was trained from scratch without utilizing any pre-existing network or pre-trained weights, achieving near-perfect results with 98.63% accuracy, 0.9864 F1-score, and 0.9998 macro AUC.

Original Contributions:

- **Tri-Domain Parallel Processing (TDPP):** Tri-domain parallel feature extraction that emulates the dermatological examination process.
- **Cross-Domain Attention (CDA):** A novel attention mechanism enabling information exchange between branches.
- **Adaptive Domain Fusion Module (ADFM):** Dynamic domain weighting for each input sample.
- **Hierarchical Feature Refinement (HFR):** Progressive feature refinement and multi-level regularization.

The model has an extremely compact architecture with only 1,152,400 parameters (compared to InceptionResNetV2: 55.9M, Xception: 22.9M). The 100% accuracy in Acne and Hyperpigmentation classes validates the effectiveness of the tri-domain feature extraction strategy. The total training time of 39.38 minutes demonstrates remarkable efficiency.

6.10.12. Model Parameter Summary

Table 6.10.2. SalvationNet Model Configuration

Property	Value
Model Name	SalvationNet
Model Type	Fully Original (Custom)
Designer	Ömer Faruk Kurtuluş
Pre-trained	NO (Trained from Scratch)
Input Size	224×224×3
Total Parameters	1,152,400
Trainable Parameters	1,147,136
Optimizer	RMSprop (LR=0.0005)
Total Training Time	39.38 minutes
Number of Epochs	77 (best at 57)

Table 6.10.2. *SalvationNet Model Parameter Summary*