Skin Photo Analyzer: Comparative CNN Models Using Transfer Learning and Personalized Recommendations on Skin Type via OpenAI LLM Prompt and Generated CSV Data from ChatGPT

MD. ALI AHNAF

Research Methodology

Data Collection:

A custom dataset was thoroughly curated Roboflow (SkinClassification Image Dataset), featuring three distinct folders: Acne, Dry, and Oil. The dataset includes images of 3 skin types and conditions. A total of 692 images were thoroughly selected. The dataset had pre-processing applied to each image: auto-orientation of pixel data (with EXIF-orientation stripping) and resizing to 640x640 (Stretch).

Data Preparation:

The dataset was partitioned into training and testing sets. The training set has 80 percent of the data, and the test set has the rest (20 percent).

- **Loading Images:**

Images are loaded using the '**load_img'** function from Keras, and then converted to arrays using '**img_to_array'**. The images are loaded from the training directory (**TRAIN_DIR**), resized to the specified dimensions ('**resize x resize'**), and preprocessed using the **'preprocess_input'** function appropriate for the selected pre-trained model (MobileNetV2).

- **Label Encoding:**

The '**LabelEncoder'** is used to convert string labels/categorical data (skin types: acne, dry, oil) into numerical format, as required by the neural network for training. When it comes to unseen data (data that the model has not been exposed to during training), the same LabelEncoder instance should be used to ensure consistency. This is crucial because the mapping between original labels and numerical values should be the same for both the training and unseen data. Mainly, the LabelEncoder maintains the mapping between original class labels and numerical values, ensuring that the encoding is consistent across both training and unseen data.

- **Data Augmentation:**
    - ImageDataGenerator for Augmentation: The **'ImageDataGenerator'** is configured with various augmentation parameters to generate augmented images during training. These parameters include zooming, rotation, horizontal and vertical shifting, shear transformation, brightness adjustment, and horizontal flipping. These augmentations help increase the diversity of the training data and improve the model's generalization.
    - A custom function '**contrast_stretching'** is defined to apply contrast stretching during data augmentation. This function adjusts the contrast of the images based on the 2nd and 98th percentiles of pixel intensities.

- **Model Building:**
    - **Loading Pre-trained Base Model**: A pre-trained MobileNetV2 model is loaded with weights from 'imagenet'. The base model is configured to exclude the top classification layers and accepts input images with dimensions (resize, resize, 3).
    - **Custom Head for Classification:** A custom head for classification is added on top of the base model. Average pooling, flattening, dense (fully connected) layers, and dropout are applied. The final layer uses softmax activation for multi-class classification (for predicting skin types 'acne', 'dry', 'oil'). The base model and custom head are combined to create the final model.
    - **Freezing Base Model Layers:** The layers of the pre-trained MobileNetV2 model are frozen to retain the learned features. This prevents these layers from being updated during training.

Adam optimizer (**Adam**) is used to adjust the model's weights during training to minimize the loss. **"Categorical Crossentropy"** is used as the loss function. Chosen based on the number of skin type classes (more than two/multiclass). Initial learning rate (**INIT_LR**) is set to **1e-4** to determine the step size during optimization. 1e-4 is equivalent to 0.0001. The learning rate 1e-4 means the initial learning rate for the optimizer is set to 0.0001. Learning rate is a hyperparameter that determines the step size at each iteration during the optimization process. A smaller learning rate like 0.0001 is often chosen to ensure gradual and stable convergence during training.

Feature Extraction Models:

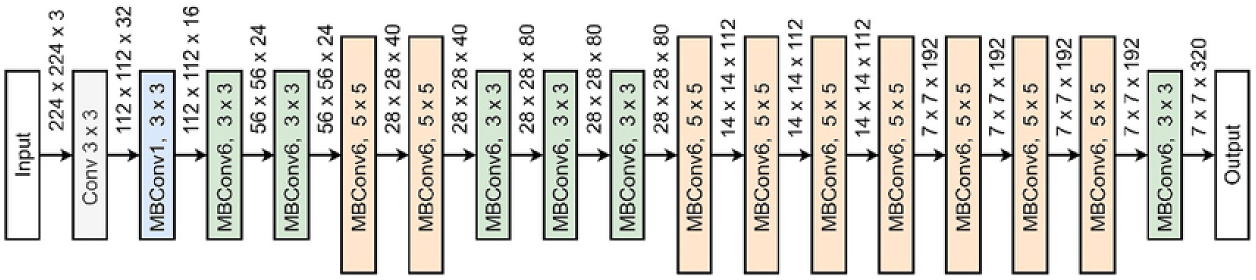


Figure 1: EfficientNetB0 Architecture

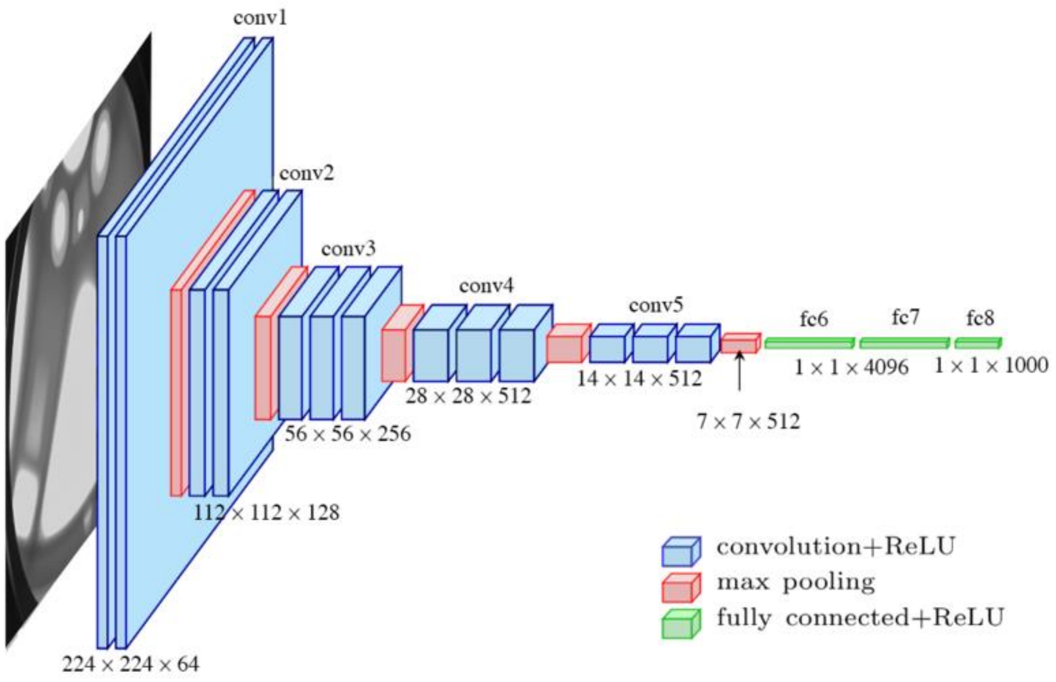
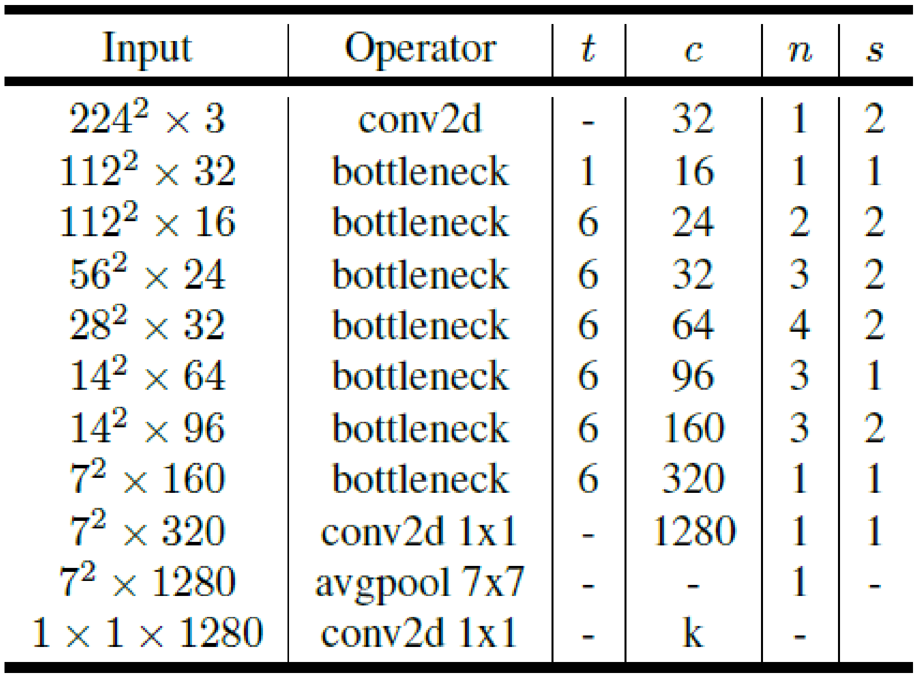
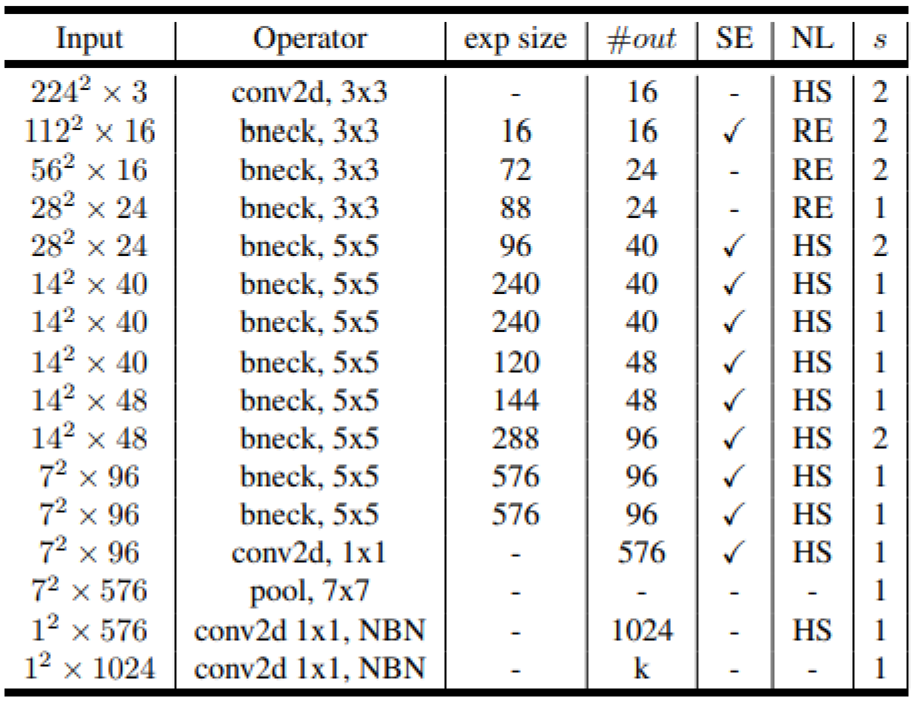


Figure 2: VGG16 Architecture



Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	

Figure 3: MobilenetV2 Architecture



Input	Operator	exp size	$\#out$	SE	NL	s
$224^2 \times 3$	conv2d, 3x3	-	16	-	HS	2
$112^2 \times 16$	bneck, 3x3	16	16	✓	RE	2
$56^2 \times 16$	bneck, 3x3	72	24	-	RE	2
$28^2 \times 24$	bneck, 3x3	88	24	-	RE	1
$28^2 \times 24$	bneck, 5x5	96	40	✓	HS	2
$14^2 \times 40$	bneck, 5x5	240	40	✓	HS	1
$14^2 \times 40$	bneck, 5x5	240	40	✓	HS	1
$14^2 \times 40$	bneck, 5x5	120	48	✓	HS	1
$14^2 \times 48$	bneck, 5x5	144	48	✓	HS	1
$14^2 \times 48$	bneck, 5x5	288	96	✓	HS	2
$7^2 \times 96$	bneck, 5x5	576	96	✓	HS	1
$7^2 \times 96$	bneck, 5x5	576	96	✓	HS	1
$7^2 \times 96$	conv2d, 1x1	-	576	✓	HS	1
$7^2 \times 576$	pool, 7x7	-	-	-	-	1
$1^2 \times 576$	conv2d 1x1, NBN	-	1024	-	HS	1
$1^2 \times 1024$	conv2d 1x1, NBN	-	k	-	-	1

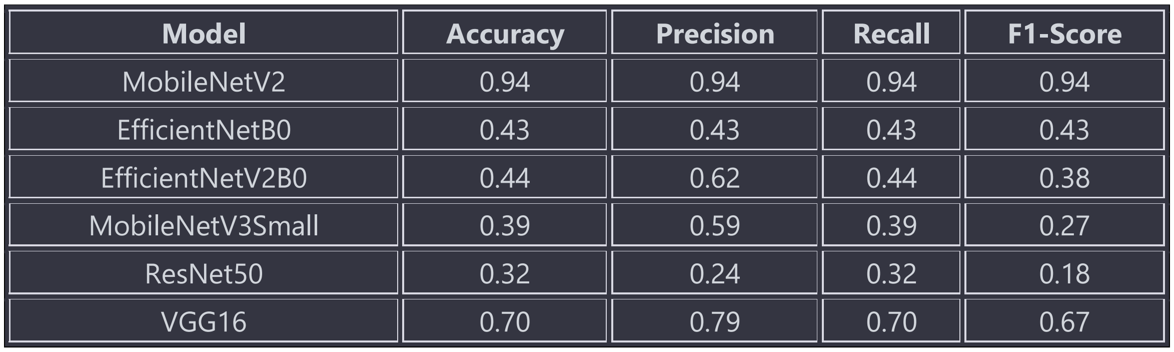
Figure 4: MobilenetV3 Small Architecture

Evaluation Metrics:

Standard evaluation metrics, including accuracy, precision, recall, and F1-score, were used to assess the performance of feature extraction models. The results were compiled into a table for comparison.

Results or Findings

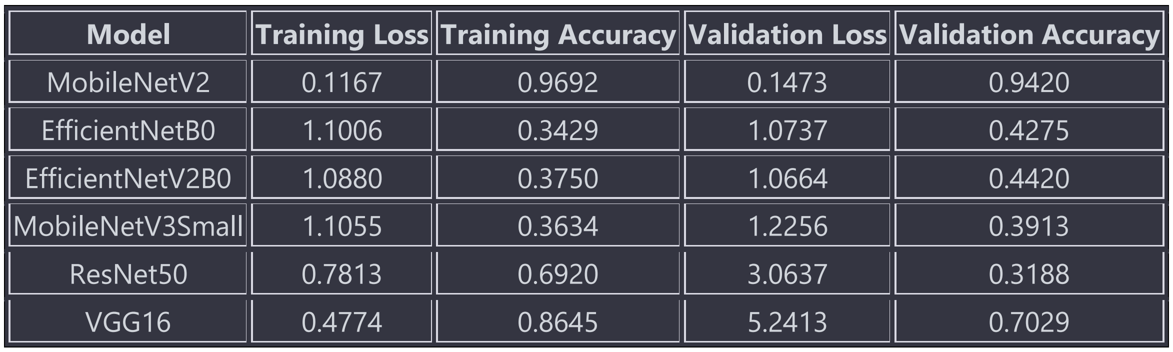
Evaluation Metrics for Feature Extraction Models after 30 epochs:



Model	Accuracy	Precision	Recall	F1-Score
MobileNetV2	0.94	0.94	0.94	0.94
EfficientNetB0	0.43	0.43	0.43	0.43
EfficientNetV2B0	0.44	0.62	0.44	0.38
MobileNetV3Small	0.39	0.59	0.39	0.27
ResNet50	0.32	0.24	0.32	0.18
VGG16	0.70	0.79	0.70	0.67

Table 1: Evaluation Metrics (macro avg) for Feature Extraction Models

Performance Metrics (Training and Validation) for Feature Extraction Models after 30 epochs:



Model	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy
MobileNetV2	0.1167	0.9692	0.1473	0.9420
EfficientNetB0	1.1006	0.3429	1.0737	0.4275
EfficientNetV2B0	1.0880	0.3750	1.0664	0.4420
MobileNetV3Small	1.1055	0.3634	1.2256	0.3913
ResNet50	0.7813	0.6920	3.0637	0.3188
VGG16	0.4774	0.8645	5.2413	0.7029

Table 2: Performance Metrics for Feature Extraction Models

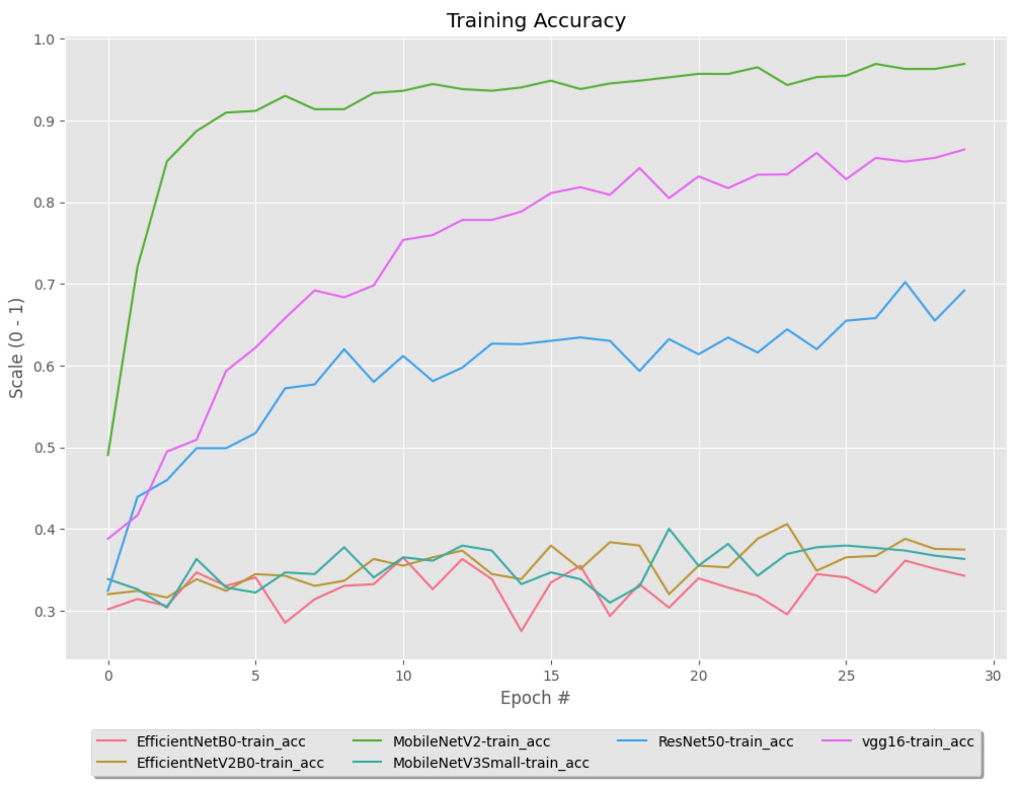


Figure 5: Performance Metrics (Training Accuracy) for Feature Extraction Model

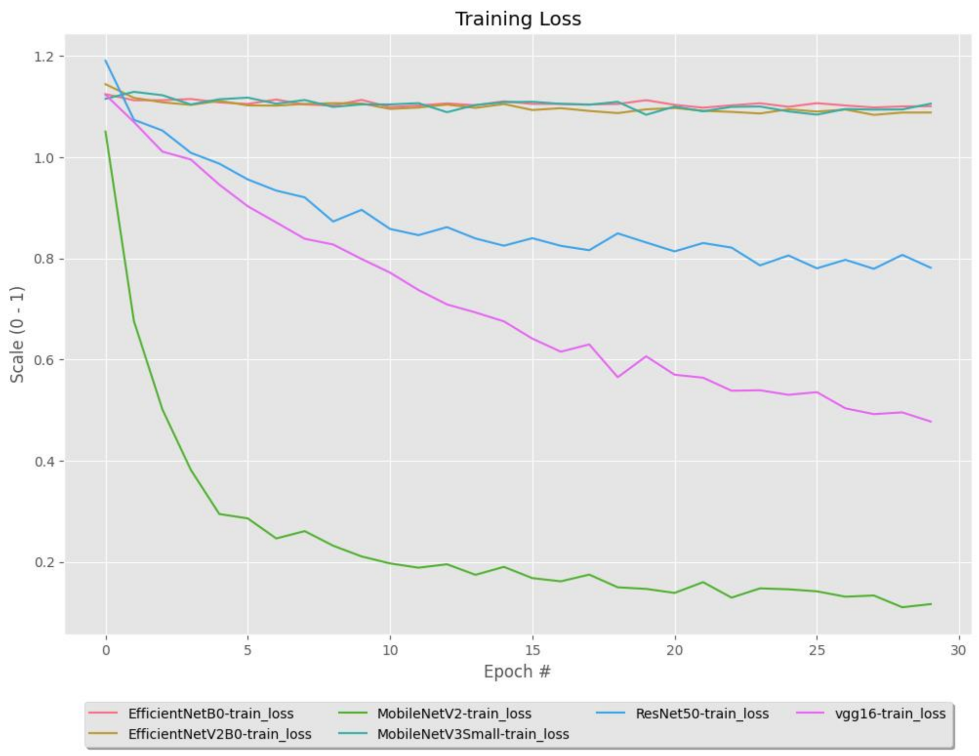


Figure 6: Performance Metrics (Training Loss) for Feature Extraction Model

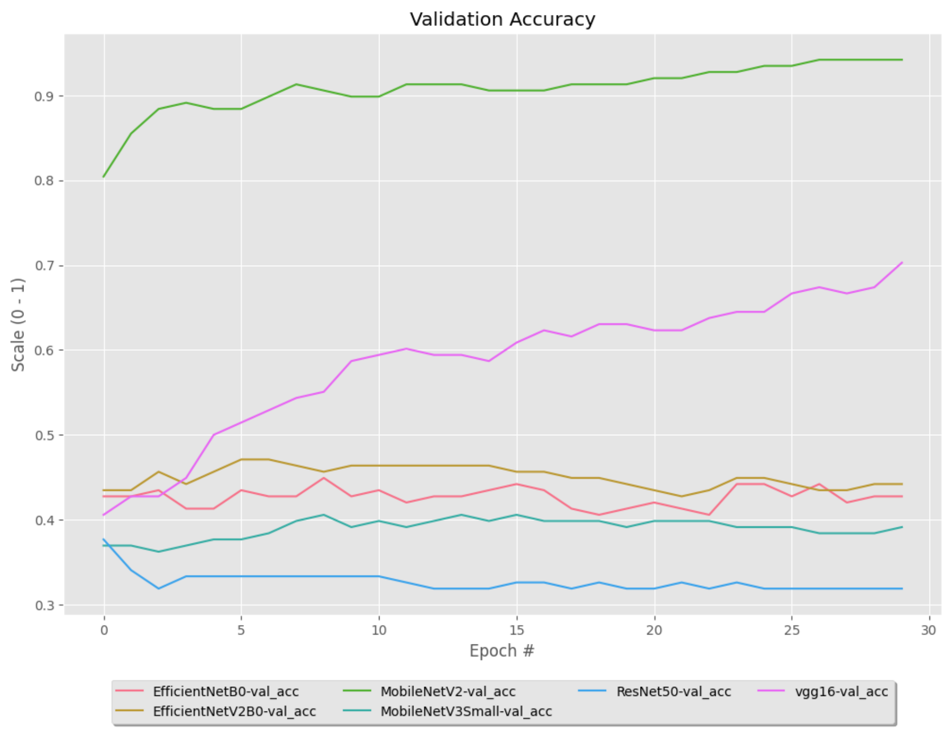


Figure 7: Performance Metrics (Validation Accuracy) for Feature Extraction Model

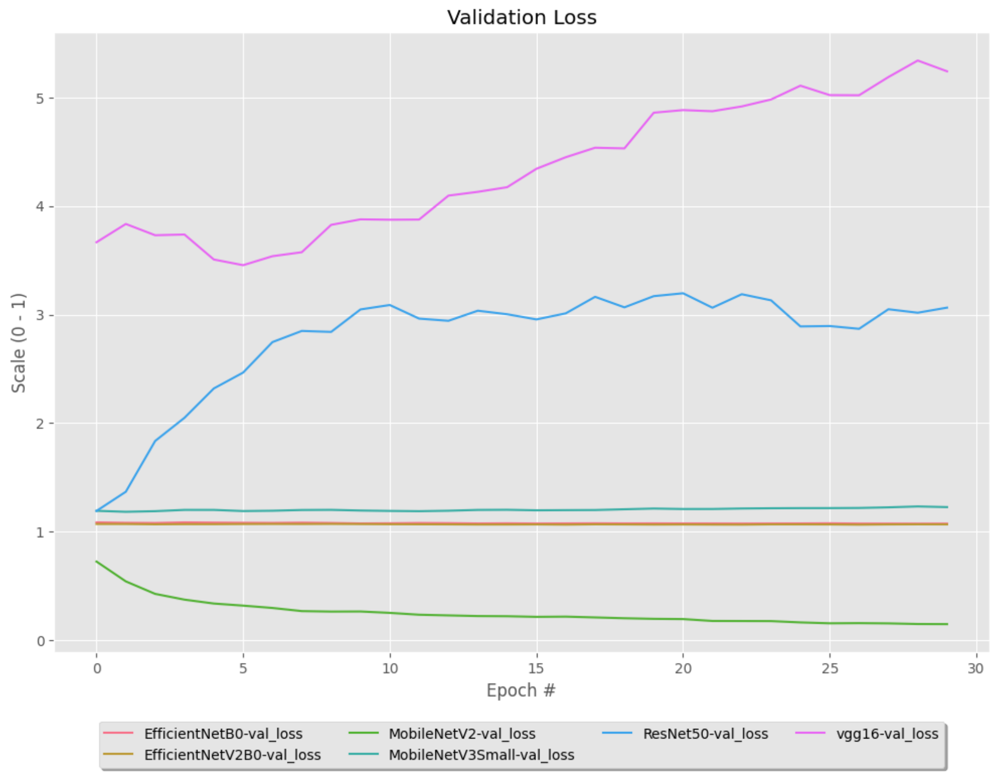


Figure 8: Performance Metrics (Validation Loss) for Feature Extraction Model

The evaluation of feature extraction models unveiled diverse performance outcomes. Notably, MobileNetV2 emerged as top performers, achieving remarkable accuracy levels, close to perfection, on the validation sets. On the contrary, models like EfficientNetB0, EfficientNetV2B0, and MobileNetV3Small faced challenges, displaying lower accuracy scores. MobileNetV2: Achieved high accuracy (96.92%) on the training set and good performance (94.20%) on the validation set. MobileNetV3Small: Demonstrated lower accuracy (36.34%) on the training set and the lowest performance (39.13%) on the validation set.

Discussion

Chatbot Integration:

- The OpenAI language model (LLM) chatbot successfully generated personalized recommendations based on the predicted skin type. It effectively responded to user queries and provided personalized recommendations based on the predicted skin type.
- The chatbot provided advice on skincare routines, products, home remedies, and lifestyle adjustments tailored to the predicted skin type (e.g., acne-prone skin).

CSV Product and Personalized Recommendation Integration:

- The chatbot utilizes a CSV file (chatgpt_recommendations.csv) containing product recommendations for different skin types (acne, dry, oil).
- The Python code reads the CSV file, filters recommendations based on the predicted skin type, and displays relevant products and suggestions.
- The CSV integration allows for dynamic and easily updatable recommendations. As new products or suggestions become available, they can be added or modified in the CSV file without altering the code.

Model Performance Analysis:

- MobileNetV2: This model performed consistently well on both training and validation sets, indicating its effectiveness in skin type classification.
- EfficientNetB0 and EfficientNetV2B0: These models showed lower performance, suggesting the need for further fine-tuning or exploration of more complex architectures.
- MobileNetV3Small: Despite achieving lower accuracy, it still provided valuable diversity to the ensemble of models.
- ResNet50: While the training accuracy was high, the model struggled to generalize well on the validation set, indicating potential overfitting.

- ❖ VGG16: Achieved higher training accuracy, but its performance on the validation set was slightly lower than MobileNetV2.

Data Augmentation and Transfer Learning:

- Data augmentation techniques, including rotation, brightness adjustment, and contrast stretching, were applied to enrich the dataset.
- Transfer learning using pre-trained models (e.g., MobileNetV2, ResNet50, VGG16) served as a powerful strategy for skin type classification.

Conclusion

In this analysis of skin type classification using deep learning models, several key insights and findings have emerged. The primary focus was on evaluating different pre-trained models, incorporating data augmentation techniques, and integrating personalized recommendations through OpenAI's language model (LLM). The prominent model that outperformed others in our dataset was MobileNetV2.

Model Performance:

➢ **MobileNetV2 Superiority:** MobileNetV2 demonstrated superior performance, likely owing to its lightweight architecture, which strikes an excellent balance between accuracy and computational efficiency. This model's ability to capture intricate features in dermatological images played a crucial role in its success.

➢ **Transfer Learning Advantage:** Leveraging transfer learning with pre-trained models allows the network to inherit knowledge from large datasets, proving advantageous in skin type classification, where feature extraction is essential. MobileNetV2 excelled in transferring this knowledge to our specific task.

Data Augmentation Impact:

▪ **Rescale Influence:** The inclusion of rescale=1.0/255 in data augmentation significantly impacted the model's accuracy and loss during both training and validation. This normalization step brought pixel values into the range [0, 1], ensuring consistent input scaling. However, it's noteworthy that the extent of normalization should align with the underlying model architecture and training data characteristics.

- **Normalization Considerations:** The impact of normalization underscores the importance of aligning data preprocessing steps with the chosen model's requirements. Excessive normalization or rescaling can lead to convergence issues and suboptimal model performance. The models, including MobileNetV2, were performing significantly worse in both the stages of training and validations. Hence, in the code, the rescaling part was omitted.

Personalized Recommendations:

- **CSV Integration Success:** Integrating personalized recommendations through an OpenAI language model (LLM) and a CSV file provided a user-centric approach. This allowed for dynamic and easily updatable skincare product recommendations, enhancing user engagement and experience.

- **Scalability and Flexibility:** The use of a CSV file for recommendations demonstrated scalability and flexibility. The system can be easily maintained and updated without modifying the core code, making it adaptable to evolving trends in the skincare industry.

To ensure the security of the API credentials, the OpenAI API key was revoked from environment variables in order to protect sensitive data.

REFERENCES

Saiwaeo, S., Arwatchananukul, S., Mungmai, L., Preedalikit, W., & Aunsri, N. (2023, November). Human skin type classification using image processing and deep learning approaches. *Heliyon, 9*, e21176. doi:10.1016/j.heliyon.2023.e21176

Roboflow. SkinClassification Image Dataset

https://universe.roboflow.com/skincareexperiments/skinclassification-kyxvj/dataset/1