Pomegranate Disease Classification using Advanced Deep Learning Models

Josue Sanchez
Department of Computer Science
California State University
Northridge, CA, USA
josue.sanchez.741@my.csun.edu

Jorge Enriquez
Department of Computer Science
California State University
Northridge, CA, USA
jorge.enriquez.571@my.csun.edu

Abhishek Verma
Department of Computer Science
California State University
Northridge, CA, USA
abhishek.verma@csun.edu

Abstract—**Early detection and classification of pomegranate diseases are crucial for maximizing crop yield and quality. This research study attempts to address the challenge of accurately identifying and classifying pomegranate diseases by leveraging advanced machine learning models. For this task, we selected three vision models with different architecture, complexity, and disk size requirements. The selected models where: DaViT-Base, a vision transformer model; EfficientNetV2M, a convolutional neural network; and MobileOne-S4, a lightweight model optimized for mobile devices. The methodology used involved training these models on a dataset of annotated pomegranate images, which was split into four disease categories and one healthy category. This was followed by a thorough evaluation of each model to determine the accuracy and potential deployment on mobile devices or drones. All three models showed exceptional results: DaViT-Base – 99.28%, EfficientNetV2M – 99.54%, MobileOne-S4 – 99.15%. The results show that lightweight models such as the MobileOne-S4 are viable options for the task of pomegranate disease classification, and implementing similar models has the potential to significantly enhance agricultural disease management. Future work will focus on exploring real-time classification.**

Keywords—Pomegranate Disease Classification, Machine Learning, Deep Learning, Vision Models, Image Classification, DaViT-Base, EfficientNetV2M, MobileOne-S4

I. Introduction

The production of pomegranates faces many challenges including various diseases that affect both yield and quality in agriculture. Early detection of these diseases can reduce the impact they have on production. Traditional techniques used are often inefficient and labor intensive. Using machine learning models can provide an efficient, accurate, and scalable solution for identifying and classifying these diseases.

While there has been some success using Convolutional Neural Networks for this type of task, challenges remain. Processing images taken under various conditions with different illumination, occlusion, and with complex backgrounds is a significant hurdle to overcome. Additionally, distinguishing between diseases with similar symptoms is another challenge faced by these models.

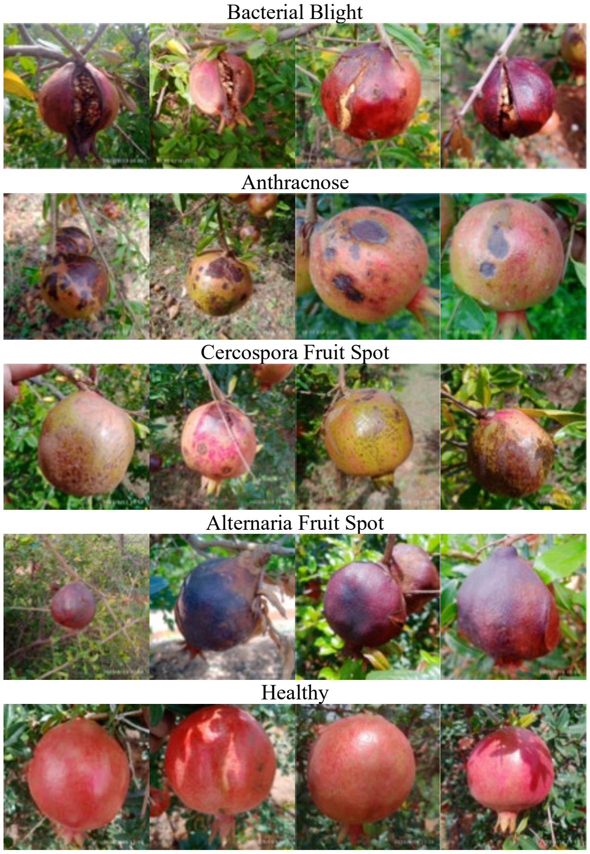
Our research explores the use of various machine learning models, each with different architecture and complexity, for the pomegranate disease classification task. It provides a comprehensive analysis of their performance focusing not only on accuracy metrics, but also on the effort and time needed to achieve the results. The methodologies and results presented in this paper provide a foundation for future researchers to apply machine learning models for disease classification tasks in other fruits or agriculture products. Additionally, the paper discusses the feasibility of training and deploying a model on a portable device to detect diseases in pomegranates.

While we were able to achieve a high level of accuracy and excellent results on all metrics across all models, the time needed to train each model varied significantly. We chose 3 different model architetures. The DaViT-Base model, a transformer model, achieved a validation accuracy of 99.28%, and took 6 hours and 9 minutes to converge. The EfficientNetV2-M model proved to be the best performing model, reaching a validation accuracy of 99.54% and taking a total of 3 hours and 4 minutes to converge. Lastly, the MobileOne-S4 model, a lightweight model designed for low latency, reached a validation accuracy of 99.15% and took 3 hours and 22 minutes to converge. These results show that lightweight models are a feasible option for detecting and classifying pomegranate diseases.

The paper follows the following structure: Section II reviews related work on agricultural disease detection and classification using machine learning. Section III offers a thorough description of the image dataset used, the distribution of images per class, and the challenges the dataset presents. Section IV explains the methodology used in this research including a detailed review of each model selected. In Section V, we outline the setup used in the experiments. Finally, Section VI discusses the results from the experiments, while Section VII presents the conclusion and suggestions for future work.

II. Related Work

Our study builds upon existing research on pomegranate diseases, aiming to address specific limitations identified in previous studies. By introducing new models that utilize state-of-the-art image classification technologies, we anticipate achieving higher accuracy, including reduced latency, in diagnosing these diseases. This advancement promises to enhance the practical application, through effectiveness, of pomegranate disease detection strategies.



Bacterial Blight

Anthracnose

Cercospora Fruit Spot

Alternaria Fruit Spot

Healthy

Figure 1: Sample images from the Pomegranate Fruit Disease Dataset for Deep Learning Models

The following articles conclude the need for a model occupying a Convolutional Neural Network. The "Diagnosis of Pomegranate Plant Diseases Using Neural Network" study utilizes a Multilayer Perceptron (MLP) neural network for classification. Various techniques are applied to prepare the data for the MLP, including K-mean clustering and Gray-level co-occurrence matrix. Six different conditions in pomegranates are analyzed, and their resulting accuracies are as follows: Healthy Fruit & Healthy Leaves - 100%, Leaf Spot - 87.5%, Bacterial Blight - 85.71%, Fruit Spot & Fruit Rot - 83.33%. [1]

The authors of "Automated Detection and Classification of Pomegranate Diseases Using CNN and Random Forest" use a Convolutional Neural Network as a feature extraction tool rather than a classification. A random forest algorithm is used for the final classification, yielding the following results on the five different diseases: Aspergillus Fruit Rot - 98%, Bacterial Blight - 98%, Anthracnose - 97%, Cercospora - 97% and Psuedocercospora Punicae - 97%. This averages around 97.4%. However, there is room for further improvement. [2]

The publication "Recognition and Classification of Pomegranate Leaves Diseases by Image Processing and Machine Learning Techniques" classifies the different types of Pomegranate leaf diseases using a multi-class SVM. The images are transformed to improve color contrast and then are pattern-matched using K-means. The multi-class SVM is then used to classify the images. The average accuracy of the model was 98.07%. [3]

In the paper "A Deep Learning Approach for Multiclass Orange Disease Classification," Orange disease classification uses SVM, K-Nearest Neighbor, Random Forest, and a Convolutional Neural Network (MobileNetV2). The analysis of these different models on the same dataset showed an outperformance by the MobileNetV2, highlighting the significant improvement that modern CNN models have. [4]

"Apple Leaf Disease Detection: Machine Learning & Deep Learning Techniques" analyzes four apple leaf diseases using various models. SVM, Random Forest, Naive Bayes, K-Nearest Neighbor, and Decision Tree are the machine learning models used, with Sequential and VGG-16 being the deep learning models utilized. The paper concludes that the CNN model VGG-16 is the most accurate of the 7, with an accuracy of 97.23% with a pre-trained model. The model is pre-trained on ImageNet. The paper states that the model architecture is an excellent tool for agricultural applications, with computational demands as a drawback to its application. [5]

In the paper "Performance Analysis of Fruit Quality Detection using Computer Vision and Object Detection," a model is trained to identify fruits and their quality. The Yolo7 architecture is used to classify fruits such as Apples, Lemons, Oranges, and Pomegranates, InceptionV3 is used to detect the quality of the fruit, and Ripeness is determined by different architectures: Inceptionv3, ResNet50, and VGG19. [6]

In "Fast Object Detection of the Quadcopter Drone using Deep Learning," the idea of a quadcopter drone with object detection capabilities is explored. The model integrates MobileNet and Single Shot Detector to localize and detect objects. The paper explores implementing an object detection model, which builds on an image classification model. The drone ran an average of 14 FPS while detecting objects. [7]

Our research differs from these other works as it utilizes three different neural network models, one of which runs at a reduced latency, optimal for drone implementation. Due to advancements in image detection technologies, our research contains higher accuracy than these related works. These new architectures promote efficiency in training time. However, this comes with the utilization of higher hardware.

III. Dataset Description

The Pomegranate Fruit Diseases Dataset for Deep Learning Models dataset [8] was selected for this research. This dataset consists of high-resolution color images of pomegranate fruits that exhibit one of four distinct diseases: Bacterial blight, Anthracnose, Cercospora fruit spot, Alternaria fruit spot, or have been deemed healthy. These diseases are all prevalent in the regions of Ballari, Bangalore, and Bagalkote among others in India, which is a leading global producer of pomegranates.

Bacterial Blight	Anthracnose	Cercospora Fruit Spot	Alternaria Fruit Spot	Healthy
966	1166	631	886	1450

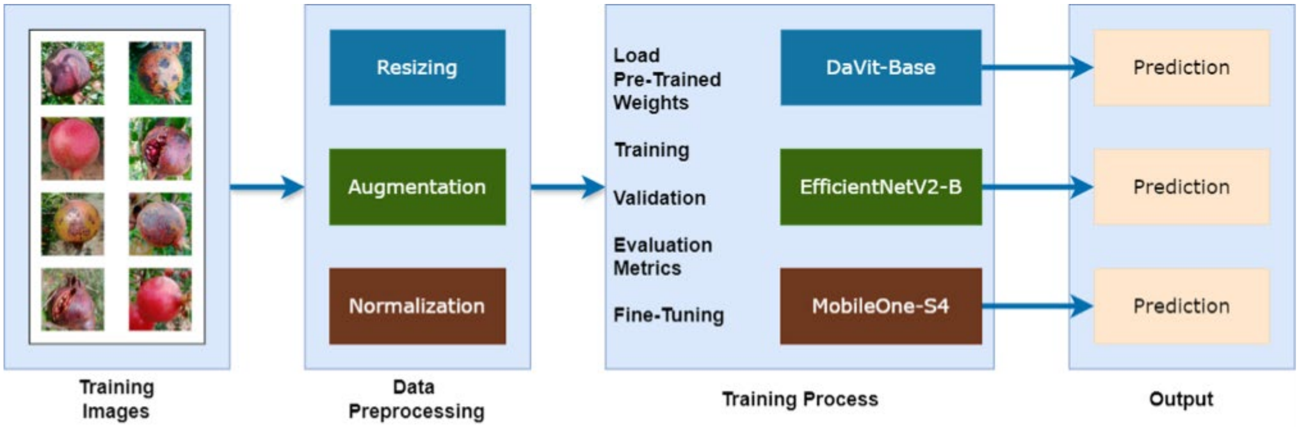


Figure 2: System Pipeline for the recognition of pomegranate diseases

The images were captured in two specific time periods: July and October 2023. They are all in JPEG format and have dimensions of 3,120 by 3,120 pixels. The dataset is organized into five directories corresponding to each of the disease or healthy categories. There is some imbalance in the dataset with the number of images in each category ranging from 631 – 1,450. The overview of the distribution of the dataset is shown in Table I. Figure 1 shows sample images from five classes.

Working with the selected image dataset does present some challenges. Aside from the imbalance in the dataset, each image was taken under different conditions. Various camera angles were used when capturing the images. There is partial occlusion in the images, and each image has different illumination conditions. Additionally, all images have a complex background with leaves, branches, fingers, dirt, other debris, and a timestamp of when the image was captured.

These challenges add to the difficulty and complexity of training a machine learning model for this task. Feature extraction becomes more challenging. Overfitting may start to appear as the model learns to recognize features that are not indicative of the disease class. Learning to address these challenges can make the model more complex, which can ultimately lead to longer training times.

IV. Research Methodology

The purpose of this research is to evaluate the effectiveness of using pre-trained models to accurately classify diseases in pomegranates. Each of the models selected for this research was trained and benchmarked using the ImageNet-21K dataset [9]. By using pre-trained models, we hope to leverage transfer learning to achieve high accuracy while using a smaller dataset. Additionally, the research explores the feasibility of using lightweight models, those designed for low latency and portability, which could be used on drones or other portable devices for this type of task. By taking this approach, we hope to address some of the common challenges in agricultural disease classification such as having a small dataset, reducing training time, and making models deployable in portable devices. Figure 2 shows system pipeline for classification.

To accomplish these objectives, three distinct models were chosen, training, and evaluated using a variety of metrics: DaViT-Base [10] EfficientNetV2-M [11], and MobileOne S4 [12]. The models were selected based on their proven performance on the ImageNet dataset [9]. Each of these models has a unique architecture, different complexity, and vary in the amount of disk space needed. These differences allow us to effectively compare the models with one another.

II. Comparative Analysis Of DaViT-Base, EfficientNetV2-M, And MobileOne-S4 Models: Computational Efficiency And Performance On ImageNet [9]

Model	FLOP	Top-1 Acc
DaViT-Base [10]	15.5 B	84.6%
EfficientNetV2-M [11]	24 B	85.1%
MobileOne-S4 [12]	2.98 B	79.4%

A. *DaViT-Base Model*

The Dual Attention Vision Transformer (DaViT) model is a transformer model that uses two self-attention mechanisms: Spatial Window Multihead Attention (SWM-SA) and Channel Group Self-Attention (CG-SA) [10]. The two attention mechanisms complement each other. The SWM-SA mechanism focuses on obtaining fine-grained local features while CG-SA finds global representations [10]. This design allows the model to be computationally efficient and effective.

SWM-SA can be represented mathematically using the following equation:

$$A_{window}(Q,K,V) = \{A(Q_i, K_i, V_i)\}_{i=0}^{N_w} \quad (1)$$

Here Q, K, and V represent the query, key, and value matrices, and d_k is the dimension of key vectors. N_w represents the number of windows.

Similarly, we can use the following to represent CG-SA:

$$A_{channel}(Q,K,V) = softmax\left(\frac{Q_g K_g^T}{\sqrt{d_{kg}}}\right) V_g \quad (2)$$

where Q_g, K_g, and V_g represent the query, key, and value matrices, and dkg is the dimension of key vectors in grouped context.

The DaViT family of models has many variations. However, we have selected the DaViT-Base model for our research. This model has 87.95 million parameters, which makes it relatively compact, and 15.5 billion floating-point operations (FLOPS) [10], which highlights its computational efficiency. Additionally, the model was able to achieve a top-1 accuracy of 84.6% on the ImageNet dataset. These characteristics enable it to be deployed in a variety of hardware platforms, including those with limited computational capacities such as phones or other portable devices.

B. *EfficientNetV2-M Model*

The EffcientNetV2-M model is a convolutional neural network that is known for its good performance on image classification tasks [11]. We selected the "M" or medium variant, since it offers a good balance of model size, speed, and accuracy. EfficientNetV2 models builds upon its predecessor. Inverted Residual Blocks, often called MBConv, and fused-MBConv layers [13] are used to enhance the model's efficiency and make it more computationally efficient. Additionally, the model limits the size of the images used to 480 by 480. This helps to address the memory overhead that comes with processing larger images as well as reduces the computation time.

The EfficientNetV2-M model was trained and benchmarked on the ImageNet-21K dataset. It reached a top-1 accuracy rate of 85.1%, which highlights its ability to extract visual features. The model was able to achieve this level of accuracy with only 52.86 million parameters and 24 billion FLOPs. The low parameter count and FLOPs make this model a viable option to deploy on a variety of devices including those with limited computational capabilities.

C. *MobileOne-S4 Model*

MobileOne prioritizes lowering the latency cost of deep-learning models to develop an efficient architecture for mobile devices. We've selected "S4", which is the largest MobileOne model with 12.91 million parameters, 2.978 billion FLOPs, and a latency of 1.86ms. On a GPU, MobileOne-S4 was able to achieve a latency of 0.95 milliseconds. MobileOne-S4 achieves this by choosing a simpler activation function (ReLU), which reduces latency at low loss of accuracy, and Architectural Blocks with no branches at inference, which reduce the amount of parallelism the model undergoes, thus reducing latency.

In ImageNet-1k benchmarks, MobileOne-S4 reached an accuracy of 79.4%. All MobileOne models were trained for 300 epochs with a 256-batch size. Since MobileOne has reduced complexity, it requires less regularization due to a reduced risk of overfitting. MobileOne's low latency and relatively high accuracy make the model optimal for mobile devices, however, is limited by the reduced complexity, as larger applications will not yield high accuracy rates.

By selecting three distinct models with different architectures and sizes, we aim to assess the trade-off between computation time and resources and performance. This will help guide future research and selections of models used in real-world deployment.

V. Experiment Setup

The training of all three models selected was performed under the same conditions utilizing the same hardware and software components. Additionally, similar modifications were made to each model to enable them to make predictions on the dataset, and each model was evaluated using the same performance metrics.

A. *Hardware and Software*

All experiments were conducted using a personal laptop computer connected to an eGPU to assist with processing the images during training and evaluation. A Dell XPS 15 equipped with a 12th Generation Intel Core i9-12900HK processor and 16 GB of RAM was used. The GPU was a NVIDIA Titan XP with 12 GB of memory.

We used a Jupyter Notebook environment for all preprocessing, training, and evaluation tasks. The models were implemented using PyTorch version 2.1.2 [14] and were acquired using the timm [15] and torchvision [16] libraries. In addition, CUDA version 12.1 was utilized to leverage the NVIDIA Titan GPU. Additional libraries such as scikit-learn [17] and seaborn [18] were used to assist in calculating performance metrics and visualizing the training progress and results.

B. *Metrics*

We employed a variety of evaluation metrics and visualization tools to measure the overall performance of the models while accounting for the class imbalance.

1) *Cross-Entropy Loss*

Cross-Entropy Loss was used as the primary criterion for optimizing the models. This loss function is suitable and widely used for multi-class classification tasks. When making predictions, Cross-Entropy returns a distribution of probabilities over all classes. Then, the difference between the predicted probabilities and the actual target values is calculated. By minimizing this value, the aim is to improve the model's ability to make accurate predictions. The formula for the cross-entropy loss for a single observation can be represented by the formula:

$$H(p,q) = -\sum_{x} p(x)\, log\, q(x) \tag{3}$$

Here, *p(x)* is the actual probability of class *x*, *q(x)* is the predicted probability distribution of class *x*, *H(p,q)* is the cross-entropy between the true distribution *p* and the predicted distribution *q* with the sum being over all classes.

2) *Performance Metrics*

III. Training and Validation Metrics at Selected Epoch for the DaViT-Base, EfficientNetV2-M and MobileOne-S4 models

Model	Epoch	Train Acc.	Val. Acc.	Train Loss	Val. Loss
DaViT-Base	28	99.97	99.28	0.001	0.031
EfficientNetV2-M	27	99.92	99.54	0.003	0.025
MobileOne-S4	31	99.55	99.15	0.013	0.037

A variety of performance metrics were used to evaluate the performance of the models. This included accuracy: a general measurement of correctness, precision: the correctness of positive predictions, recall: measures the ability to detect positive instances of each class, and F1 score: a balance between precision and recall, which is useful in situations with uneven class distributions. While accuracy was measured for both training and validation sets, precision, recall, and F1 values were only taken from the validation set as our aim was to evaluate the model's generalization on unseen data.

3) *Confusion Matrix*

To visualize the model's performance across all classes, a confusion matrix was created. The confusion matrix makes it possible to identify classes that are often confused with one another. Additionally, to address the imbalance of the dataset, a weighted confusion matrix was also created. This weighted confusion matrix allowed us to consider the frequency of each class in the dataset, which helped us to evaluate the model's performance more accurately.

C. *Preprocessing*

Some preprocessing was performed on the dataset to prepare it for training. The dataset was split into training and validation sets using a 70/30 split, which is common practice. This ensured that most of the data was used for training while retaining a sizeable amount for validation. All images were resized to 224 by 224 pixels and were normalized using ImageNet's mean and standard deviation values of (0.485, 0.456, 0.406) and (0.229, 0.224, 0.225) respectively. An augmentation factor for 0.05 was used to synthesize a small but significant amount of data to add some variety to the training samples. Additional augmentations were performed on the training samples including a random rotation of ± 20 degrees, a random horizontal rotation, and color brightness, contrast, and saturation were all varied by up to 20 percent. By performing these augmentations, we hope to enhance the models' ability to learn from the training data and generalize enabling it to make more accurate predictions.

D. *Model Configuration*

An instance of each model was created using the timm [15] and torchvision libraries [15]. The models were initialized using the pretrained weights provided by the libraries. This allows us to leverage transfer learning to improve the models ' performance.

In our experiments, we elected to keep the standard configuration of model-specific parameters for each model. For the DaViT-Base model, this included keeping the predefined settings for the attention mechanisms, dimensions of each layer, the GELU activation function, and dropout, which was set to 0.0. Similarly, the EfficientNetV2-M model was left with its original scaling, SiLU activation function, and dropout rate of 0.3 per dropout layer. The MobileOne-S4 model was also used with its default settings and used the ReLU activation function. The rest of the parameters were left to the default values as set in the timm [15] and torchvision [16] libraries with the only exception being the modification of the classifier layer. To meet the requirements for the task, the classifier layer of each model was modified to be a linear layer with five output features, one for each of the five distinct classes in the dataset.

E. *Training Parameters*

A specific set of training parameters were used to ensure the results from each model can be accurately compared with each other. The parameters consisted of the batch size used, the number of epochs, the learning rate, the optimizer selected, and the scheduler that was used.

All models were trained using a batch size of 20 images for 40 or 50 epochs. This selection was made to maximize the efficiency of the hardware use, maintain an acceptable level of randomness in the gradient decent process, and provide the models with enough exposure to the training data for adequate learning and convergence.

Adam was selected as the optimizer for its computational efficiency, small memory requirements, and usability for non-convex optimization problems [19]. A learning rate of 0.001 was chosen as it is commonly used with the Adam optimizer for a wide range of tasks. Additionally, the ReduceLROnPlateu scheduler was used. This scheduler, accessed through the torch

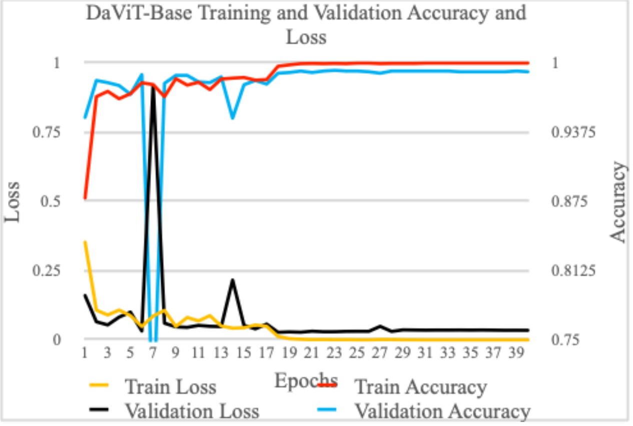


Figure 3: DaViT-Base Model Training and Validation Accuracy and Loss Graph

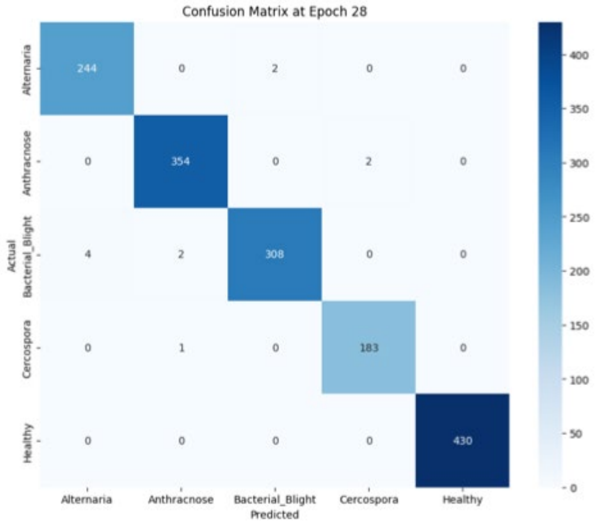


Figure 4: DaViT-Base Confusion Matrix at Epoch 28

library, dynamically adjusts the learning rate based on the validation loss allowing the weights to be adjusted more precisely.

VI.Results and Discussion

IV. Precision, Recall, and F1 Score for each Disease Class Obtained by DaViT-Base Model At Epoch 28

Disease	Validation Accuracy	Precision	Recall	F1 Score
Alternaria	0.9919	0.9839	0.9919	0.9879
Anthracnose	0.9944	0.9916	0.9944	0.9930
Bacterial Blight	0.9809	0.9935	0.9809	0.9872
Cercospora	0.9946	0.9892	0.9946	0.9919
Healthy	1.0000	1.0000	1.000	1.0000

While the architecture, complexity, and computational efficiency of each model is distinct, the results obtained from the experiments show comparable performance. The performance of each model is evaluated at different epochs. These epochs were specifically chosen based on the performance of the models, indication of convergence, and signs of overfitting. This approach allows us to assess the capabilities of each model under optimal conditions. Table III compares the training and validation metrics for each model at the selected epoch.

A. *DaViT-Base Model Performance*

V. Precision, Recall, and F1 Score for each Disease Class Obtained by the EfficientNetV2-M Model

Disease	Validation Accuracy	Precision	Recall	F1 Score
Alternaria	0.9919	0.9839	0.9919	0.9879
Anthracnose	0.9944	0.9916	0.9944	0.9930
Bacterial Blight	0.9809	0.9935	0.9809	0.9872
Cercospora	0.9946	0.9892	0.9946	0.9919
Healthy	1.0000	1.0000	1.0000	1.0000

The DaViT-Base model is the more complex of the three models selected. Figure 3 shows its accuracy and loss metrics throughout the training phase, which lasted 40 epochs. Based on these metrics, we determined that the model converged at Epoch 28. At this epoch, the model showed the highest validation accuracy and maintained a minimal gap between the training and validation metrics. At Epoch 28, the model obtained an accuracy of 99.28% on the validation dataset. The graph in Figure 3 shows a stable decrease in training loss while the validation loss stabilizes. This indicates that the model was able to learn effectively and avoided overtraining. The total training time up to this epoch was 6 hours and 9 minutes.

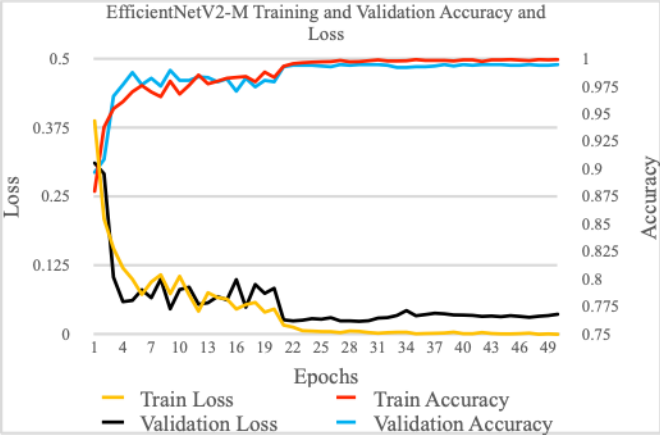


Figure 5: EfficientNetV2-M Model Training and Validation Accuracy and Loss Graph

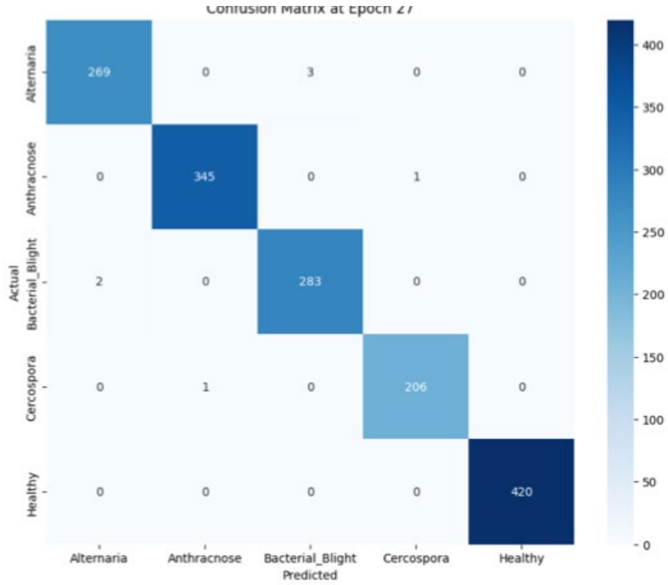


Figure 6: EfficientNetV2-M Confusion Matrix at Epoch 27

The confusion matrix for the DaViT-Base model at Epoch 28 is shown in Figure 4. The matrix validates the model's performance across all five classifications of diseases. Precision, recall, and F1 scores for each disease classification are provided in Table V. All metrics demonstrate the model performed exceptionally across all classes with 100% accuracy, precision, recall, and F1 score on the Healthy class.

B. *EfficientNetV2-M Model Performance*

Table VII. Precision, Recall, and F1 Score for each Disease Class Obtained by the MobileOne-S4 Model

Disease	Validation Accuracy	Precision	Recall	F1 Score
Alternaria	0.9847	0.9923	0.9847	0.9885
Anthracnose	1.0000	0.9886	1.0000	0.9943
Bacterial Blight	0.9865	0.9832	0.9865	0.9848
Cercospora	0.9843	1.0000	0.9843	0.9921
Healthy	0.9954	0.9954	0.9954	0.9954

The EfficinetNetV2-M model went through a training phase that lasted 50 epochs. The accuracy and loss metric graphs are shown in Figure 5. From the results, we determined that the model converged at Epoch 27. Here, the graph shows that the validation loss stabilizes. At this epoch, the model obtained a

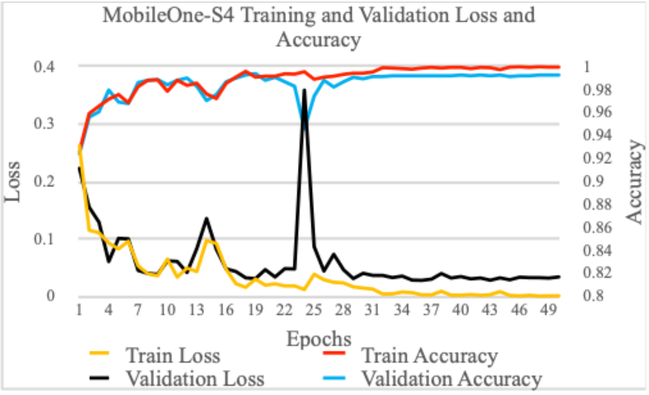


Figure 7: MobileOne-S4 Model Training and Validation Accuracy and Loss Graph

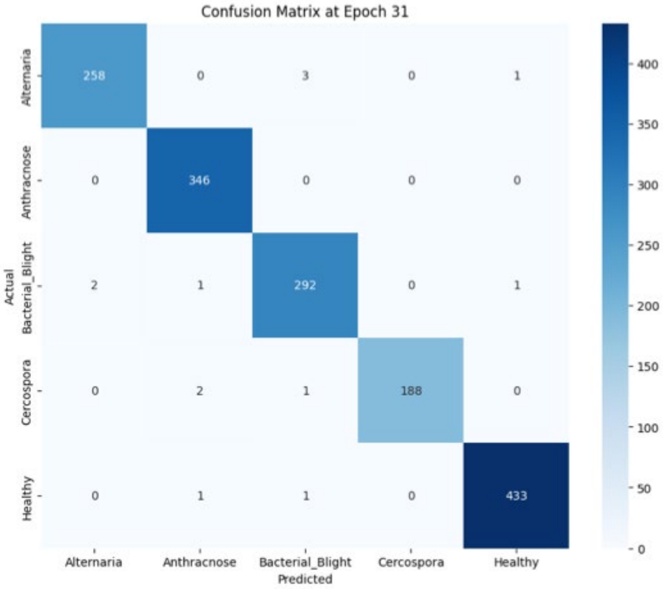


Figure 8: MobileNet-S4 Confusion Matrix at Epoch 31

validation accuracy of 99.54% and held a minimal gap between training and validation metrics. The total training time up to this epoch was 3 hours and 4 minutes.

Figure 6 shows the confusion matrix at Epoch 27 for the EfficientNetV2-M model while Table VI shows the precision, recall, and F1 score metrics. These results show that the model performed remarkably well across all classes with 100% accuracy, precision, recall, and F1 score on the Healthy class.

C. *MobileOne-S4 Model Performance*

The MobileOne-S4 was trained for 50 epochs. The training and validation accuracy and loss metrics graphs are shown in Figure 7. From the results, we determined that the model converged at Epoch 31. Here the model obtained a validation accuracy of 99.15% and held a minimal gap between training and validation metrics. The total training time up to this epoch was 3 hours and 22 minutes.

TABLE VIII. TRAINING TIME AND DISK SPACE COMPARISON OF THE DAVIT-BASE. EFFICIENTNETV2-M. AND MOBILEONE-S4 MODELS

Model	DaViT-Base	EfficientNetV2-M	MobileOne-s4
Total Training Time to Converge	6 hrs. 9 mins.	3 hrs. 4 mins	3 hrs.22 mins
Average Time per Epoch	13 mins. 11 secs.	6 mins. 49 secs.	6 mins. 32 secs.
Disk Space	331 MB	203 MB	49.9 MB
# of Params	87.95 M	52.86 M	12.92 M

The confusion matrix shown in Figure 8 and the metrics presented in Table VII show the model's performance at Epoch 31. As indicated in the results, the model performed well across all classes.

D. *Performance Comparison*

All three models performed exceptionally well on this dataset with each model attaining an overall validation accuracy of over 99%. However, the training time needed for each model to converge, and the amount of disk space needed for each model varied significantly. Table VIII compares the training times and disk spaces of each of the models.

While DaViT-Base, the transformer model, performed exceptionally well on all classes of this dataset, its training time was substantially longer than the training time required for the other models without any significant improvement in performance. The DaViT-Base Model was also the largest in size regarding disk space taking up 331 MB.

The EfficientNetV2-M, the convolutional neural network, model performed the best out of the three models trained and tested. However, the improvement in performance is marginal. The EfficientNetV2-M model accomplished these results with a shorter training time – approximately 50% less when compared to DaViT-Base – and taking up less space on the disk: 203 MB.

The lightweight model, MobileOne-S4, also demonstrated good performance on all classes of this dataset with results comparable to the other models. While it did take a few more epochs to converge, it had the fastest training time per epoch resulting in a similar overall training time as the EfficientNetV2-M model. The MobileOne-S4 model was able to achieve these results while only taking up 49.9 MB of disk space.

VII. CONCLUSION AND FUTURE WORK

Three models with different architecture, complexity, and size were trained and evaluated on the pomegranate fruit disease dataset. While each model performed exceptionally on the disease classification task, the time needed to reach these results varied. The DaViT-Base transformer model reached a high validation accuracy of 99.28% but took significantly longer to reach convergence: over 6 hours. This extra training time needed did not yield any performance benefits. The EfficientNetV2-M model achieved the best results, 99.54%

validation accuracy, and required the least amount of training time: 3 hours and 4 minutes. Lastly, the lightweight model, MobileOne-S4, reached a validation accuracy of 99.15% with only a slightly longer training time of 3 hours and 22 minutes. These results show that lightweight models, those designed for low latency and portability, are a viable option for the task of disease classification on pomegranates.

While these experiments show that lightweight models are a viable option for disease classification task, this research stops short of deploying to model on a portable device and testing the performance in a non-controlled setting. Additionally, the images used were all collected using the same equipment. Future work could explore using images taken with different equipment using different camera settings and having different resolution as well as test model performance on the field. Another area for future research could modify the model to detect pomegranates infected with a novel disease.

REFERENCES

[1] M. Dhakate and Ingole A. B., "Diagnosis of pomegranate plant diseases using neural network," 2015 Fifth National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG), Patna, India, 2015, pp. 1-4, doi: 10.1109/NCVPRIPG.2015.7490056.

[2] A. Gupta, S. Mishra, Saweksha and V. Kukreja, "Automated Detection and Classification of Pomegranate Diseases Using CNN and Random Forest," 2024 International Conference on Automation and Computation (AUTOCOM), Dehradun, India, 2024, pp. 62-66, doi: 10.1109/AUTOCOM60220.2024.10486122.

[3] Madhavan, Mangena Venu, et al. “Recognition and Classification of Pomegranate Leaves Diseases by Image Processing and Machine Learning Techniques.” Computers, Materials & Continua/Computers, Materials & Continua (Print), vol. 66, no. 3, Jan. 2021, pp. 2939–55. https://doi.org/10.32604/cmc.2021.012466.

[4] S. Kumar, A. K. Pandey, D. Raghav, G. Gupta and V. Srivastava, "A Deep Learning Approach for Multiclass Orange Disease Classification," 2024 2nd International Conference on Disruptive Technologies (ICDT), Greater Noida, India, 2024, pp. 184-189, doi: 10.1109/ICDT61202.2024.10489557.

[5] M. Sebastian, S. M S and C. M. Antony, "Apple Leaf Disease Detection: Machine Learning & Deep Learning Techniques," 2023 Intelligent Computing and Control for Engineering and Business Systems (ICCEBS), Chennai, India, 2023, pp. 1-5, doi: 10.1109/ICCEBS58601.2023.10449037.

[6] K. Deshmukh, R. Kasture, S. Bhoite, S. L. Tade and S. A. Vaishnav, "Performance Analysis of Fruit Quality Detection Using Computer Vision and Object Detection," 2023 7th International Conference On Computing, Communication, Control And Automation (ICCUBEA), Pune, India, 2023, pp. 1-8, doi: 10.1109/ICCUBEA58933.2023.10392090.

[7] W. Budiharto, A. A. S. Gunawan, J. S. Suroso, A. Chowanda, A. Patrik and G. Utama, "Fast Object Detection for Quadcopter Drone Using Deep Learning," 2018 3rd International Conference on Computer and Communication Systems (ICCCS), Nagoya, Japan, 2018, pp. 192-195, doi: 10.1109/CCOMS.2018.8463284.

[8] B. Pakruddin and R. Hemavathy, "Pomegranate Fruit Diseases Dataset for Deep Learning Models," Mendeley Data, 2023.

[9] T. Ridnik, E. Ben-Baruch, A. Noy and L. Zelnik-Manor, "ImageNet-21K Pretraining for the Masses," 05 August 2021. [Online]. Available: https://arxiv.org/pdf/2104.10972.pdf. [Accessed 01 March 2024].

[10] M. Ding, B. Xiao, N. Codella, P. Luo, J. Wang and L. Yuan, "DaViT: Dual Attention Vision Transformers," Springer Nature, 2022.

[11] M. Tan and Q. V. Le, "EfficientNetV2: Smaller Models and Faster Training," PMLR, 2021.

[12] P. K. A. Vasu, J. Gabriel, J. Zhu, O. Tuzel and A. Ranjan, "Mobileone: An improved one millisecond mobile backbone," in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2023.

[13] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov and L.-C. Chen, "Inverted Residuals and Linear Bottlenecks: Mobile Networks for Classification, Detection and Segmentation," CoRR, vol. abs/1801.04381, 2018.

[14] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in Advances in Neural Information Processing Systems 32, Curran Associates, Inc., 2019, p. 8024–8035.

[15] R. Wightman, PyTorch Image Models, GitHub, 2019.

[16] T. maintainers and contributors, TorchVision: PyTorch's Computer Vision library, GitHub, 2016.

[17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, p. 2825–2830, 2011.

[18] M. L. Waskom, "seaborn: statistical data visualization," Journal of Open Source Software, vol. 6, p. 3021, 2021.

[19] D. P. Kingma and J. Ba, Adam: A Method for Stochastic Optimization, 2017.