Plant Disease Detection and Classification using Deep Learning

1. Introduction & Problem Statement

Agriculture is a major sector that impacts society and the environment while defining the economy of a nation, this necessitates precise plant disease identification for timely intervention [10]. Plant diseases exhibit a wide range of symptoms that challenge manual categorization and often escape the human eye [11] [12]. CNNs can learn from these variations and accurately classify diseases, even in early stages not visible to the human eye [10]. However, the application of CNNs for plant disease recognition is influenced by variations in images such as lighting conditions, growth stages, and environmental changes, which impacts accurate disease identification. To solve this problem, we will assess three different architectures across three datasets related to plant diseases, employing both training-from-scratch and transfer learning methodologies. Throughout the development process, we will address technical challenges in pre-processing techniques, handling imbalanced datasets, training, and optimizing hyper-parameters, while balancing model size and performance. We will be evaluating our models on Accuracy, Precision, Recall, and Cross Entropy loss.

2. Methodology

2.1. Dataset and Data Augmentation

We have selected the *Cassava Dataset* [1], which contains 5,656 images categorized into 5 classes, *Crop Pest and Disease Dataset* [2] which includes 16,704 images having 15 different classes, and *PlantVillage Dataset* [3] with 35,320 images categorized into 30 classes. We also identified around 50 erroneous images in the Crop Pest and Disease Dataset, which were omitted for our study. We have allocated 80% to training, 10% to validation, and 10% to testing across the selected datasets.

Our solution pipeline begins with data preprocessing as the datasets exhibit varying geometric and photo-metric properties. First, we resized all the images to an input size of (224 x 224 x 3), which is compatible with the chosen architectures. Additionally, we have performed some geometric transforms such as Horizontal Flip and Vertical Flip on 10% of the images. We have also applied photo-metric transforms by adjusting the brightness and contrast of the images by a factor of 0.2. Finally, we have applied normalization on the images using ImageNet mean and standard deviation values [5].

2.2. CNN Architectures

Our study focuses on utilizing three CNN architectures, specifically chosen for their balance between size and performance. Each architecture is described below:

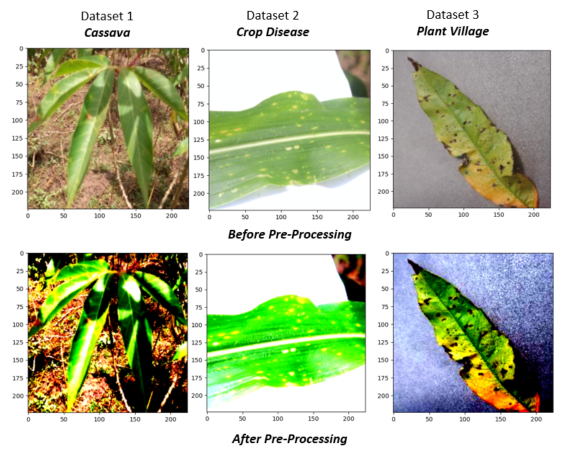

Figure 1. Image samples before and after data augmentation

2.2.1 GoogleNet

GoogleNet is a 22-layer (without pooling layers) Deep CNN architecture, that achieved state-of-the-art performance in the 2014 ImageNet Competition [6]. The core building block of this model is the Inception module, which incorporates multiple filter sizes (1x1, 3x3, 5x5) within a single module. This enables the network to learn features of different scales in the images, improving its ability to capture complex patterns. GoogleNet heavily utilizes 1x1 convolutions for dimensionality reduction, a global average pooling layer for spatial reduction, and auxiliary classifiers to address the vanishing gradient problem to improve overall training stability.

2.2.2 EfficientNet

EfficientNet B0 serves as a strong baseline model comprising around 66 layers making it relatively lightweight compared to larger variants in the series. The core building block of this model is MBConv blocks, inspired by MobileNet V2, which utilizes the squeeze-to-excitation technique to focus on informative features [7]. EfficientNet stands out for its compound scaling approach, which effectively adjusts the resolution, depth, and width of the network to capture complex features, maintaining efficiency without compromising accuracy.

2.2.3 MobileNetV2

MobileNetV2 is a deep CNN with heavy influence in embedded vision applications where accuracy and performance efficiency are crucial. The backbone of this model is depthwise and pointwise convolutions that significantly reduce computations with minimal overheads while combining the per-channel filters [8]. MobileNet V2 utilizes inverted residuals with linear bottlenecks to maintain representational power while keeping the model lightweight.

3. Attempts at Solving the Problem

3.1. Training & Evaluation

We have trained 9 models from scratch using these architectures over 3 datasets. Additionally, we have also trained 2 models using Transfer Learning. For our training, we have used Cross Entropy as the Loss Function and Stochastic Gradient Descent (SGD) as the Optimizer. A learning rate scheduler, *ReduceLROnPlateau* has been used to reduce the learning rate when a metric is not improving [9]. All of our models have been run on 20 Epochs, batch size of 32, and a Learning Rate of 0.001, with an exception of 16 as batch size for MobileNetV2 on the Cassava Dataset. Initially, we used 10 Epochs but later shifted to 20 Epochs considering the scope of improvement. In transfer Learning, we have chosen GoogleNet and MobileNetV2 on the Crop Pest and Disease dataset as we wanted to verify the benefits of the pre-trained weights with other parameters unchanged.

For our evaluation, we wanted a comprehensive analysis of our training pipeline. We used Precision, Recall, Cross Entropy Loss, Accuracy, and ROC as our performance metrics. As we are doing multi-label classification, we saw that the dataset can be imbalanced so we decided to use weighted average recall for a fair evaluation. We monitor the model performance at every iteration to identify any possible over-fitting and under-fitting issues. This approach ensures that the model is learning and if necessary, we tweak the pre-processing pipeline or hyperparameters.

3.2. Results

Table 1 illustrates the performance of the chosen architectures on the previously unseen images from the 3 datasets when trained from scratch.

Dataset	Metrics	GoogleNet	EfficientNet	MobileNetV2
Cassava	Accuracy (%)	64.77	68.31	60.88
	Precision (%)	57.59	59.38	51.72
	Recall (%)	64.77	68.31	60.88
Crop Pest & Disease	Accuracy (%)	79.88	76.82	71.73
	Precision (%)	80.24	77.22	72.14
	Recall (%)	79.88	76.82	71.73
Plant Village	Accuracy (%)	99.12	98.44	98.21
	Precision (%)	99.16	98.48	98.32
	Recall (%)	99.12	98.44	98.21

Table 1. Trained from Scratch: Performance of 3 Architectures on different Datasets

We observe that, on the Cassava dataset, EfficientNet performs best compared to other network architectures, achieving an accuracy of 68.31% and a precision of 59.38%. Similarly, on the Crop Pest and Disease dataset, GoogleNet outperforms other architectures with an accuracy of 79.88% and a precision of 80.24%. On the Plant Village dataset, the three architectures achieve high accuracy and precision, with GoogleNet performing slightly better than EfficientNet and MobileNetV2. This high performance can likely be attributed to the simpler backgrounds in the images compared to the other datasets. Less background noise can make it easier for the models to focus on the key embedded features of the plants, leading to better performance. Overall, the performance of the 3 architectures varies across datasets, with GoogleNet and EfficientNet generally performing better than MobileNetV2. EfficientNet shows consistent competitive performance across all datasets, making it a good choice for solving our problem. Despite its computational efficiency, MobileNetV2 tends to perform slightly lower compared to the other two architectures on our datasets.

Dataset	Metrics	GoogleNet	MobileNet V2
Crop Pest & Disease	Accuracy (%)	88.08	87.18
	Precision (%)	87.99	87.60
	Recall (%)	88.08	87.18

Table 2. Transfer Learning: Performance of GoogleNet & MobileNet V2 Architectures on Crop Pest & Disease dataset

Table 2 shows the performance of GoogleNet and MobileNetV2 on the Crop Pest and Disease dataset using the Transfer Learning approach. We observe that GoogleNet outperforms MobileNetV2, achieving an accuracy of 88.08% and a precision of 87.99%.

4. Future Improvements

Out of the three datasets, we observe high performance from all the architectures on the plant village dataset giving us no scope for further improvement. The future improvements are possible on the other two datasets. Considering the results obtained, we aim to improve EfficientNet's performance on Cassava dataset. As a next step, will be employing random search technique to find the optimal configuration for learning rate and batch size. There is also a scope of data augmentation specific to the dataset selected for improvement to tackle the variations encountered during inference. We aim to do this using segmentation which would allow us to precisely localize and highlight the regions of interest in the images and removing the background noise as observed from the results obtained on the plant village dataset. We also aim to implement visualization techniques like t-SNE to gain insights into how the model is making prediction by looking at the distribution of instances in the feature space. In addition to this, we will add F1-score as an evaluation metric for a more comprehensive understanding of the model's performance. Improving the model for F1-score would make our model more robust to class imbalances.

References

[1] Mwebaze, Ernest, et al. *"iCassava 2019 Fine-Grained Visual Categorization Challenge Dataset."* Google Research, Artificial Intelligence Lab Makerere University, National Crops Resources Research Institute. Accessed [Access Date], https://tensorflow.google.cn/datasets/catalog/cassava. 1

[2] Mensah Kwabena, Patrick, et al. *"Dataset for Crop Pest and Disease Detection."* Mendeley Data, vol. 1, 2023, doi:10.17632/bwh3zbpkpv.1. https://data.mendeley.com/datasets/bwh3zbpkpv/1 1

[3] Hughes, D. P., and M. Salathé. *"An open access repository of images on plant health to enable the development of mobile disease diagnostics through machine learning and crowdsourcing."* CoRR, vol. abs/1511.08060, 2015, http://arxiv.org/abs/1511.08060. https : / / www . tensorflow . org / datasets / catalog/plant_village 1

[4] Mohanty, Sharada P., David P. Hughes, and Marcel Salathé. *"Using deep learning for image-based plant disease detection."* Frontiers in plant science 7 (2016): 1419.

[5] J. Deng, W. Dong, R. Socher, L. -J. Li, Kai Li and Li Fei-Fei, *"ImageNet: A large-scale hierarchical image database,"* 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 2009, pp. 248-255. 1

[6] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... and Rabinovich, A. (2015). *"Going deeper with convolutions"*. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9). 1

[7] Tan, Mingxing, and Quoc Le. *"Efficientnet: Rethinking model scaling for convolutional neural networks."* International conference on machine learning. PMLR, 2019. 1

[8] Sandler, Mark, et al. *"Mobilenetv2: Inverted residuals and linear bottlenecks"*. Proceedings of the IEEE conference on computer vision and pattern recognition. 2018. 1

[9] Paszke, Adam, et al. *"Pytorch: An imperative style, high-performance deep learning library"*. Advances in neural information processing systems 32 (2019). 2

[10] Punam Bedi, Pushkar Gole, Plant Disease Detection Using Hybrid Model Based on Convolutional Autoencoder and Convolutional Neural Network, *Artificial Intelligence in Agriculture*, Volume 5, 2021, Pages 90-101, ISSN 2589-7217. 1

[11] Hassan, S.M.; Maji, A.K.; Jasiński, M.; Leonowicz, Z.; Jasińska, E. Identification of Plant-Leaf Diseases Using CNN and Transfer-Learning Approach. *Electronics* 2021, *10*, 1388. 1

[12] Shelar, Nishant; Shinde, Suraj; Sawant, Shubham; Dhumal, Shreyash; Fakir, Kausar. Plant Disease Detection Using CNN. *ITM Web Conf.* 2022, 44, 03049.

1

5. Supplementary Material

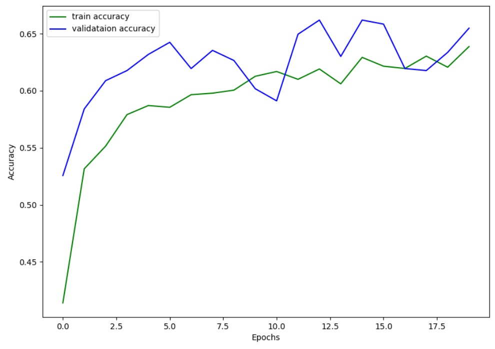

Figure 2. Accuracy of GoogleNet on Cassava Dataset

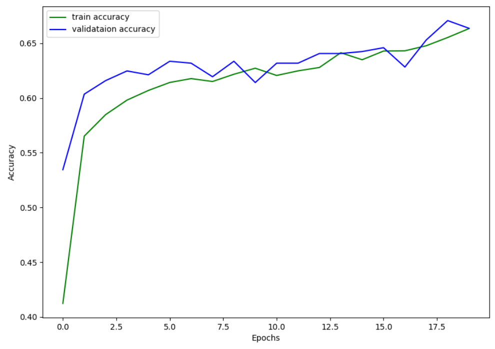

Figure 3. Accuracy of EfficientNet B0 on Cassava Dataset

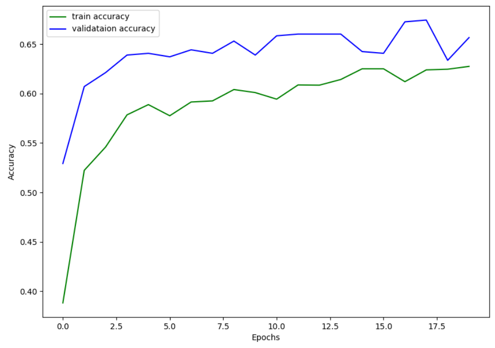

Figure 4. Accuracy of MobileNetV2 on Cassava Dataset

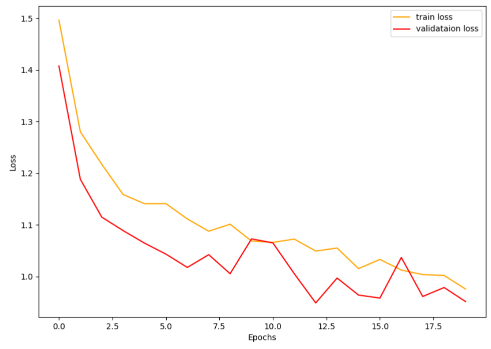

Figure 5. Loss of GoogleNet on Cassava Dataset

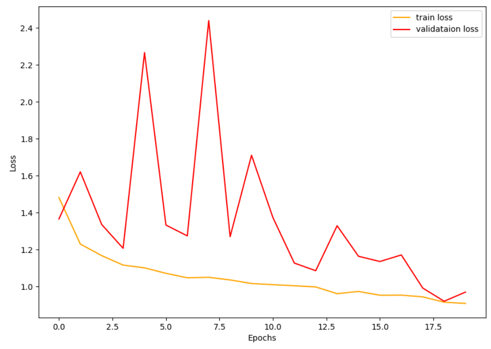

Figure 6. Loss of EfficientNet B0 on Cassava Dataset

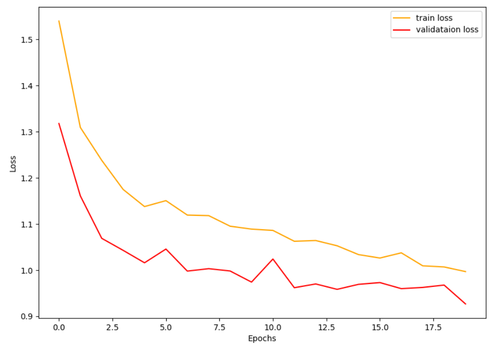

Figure 7. Loss of MobileNetV2 on Cassava Dataset

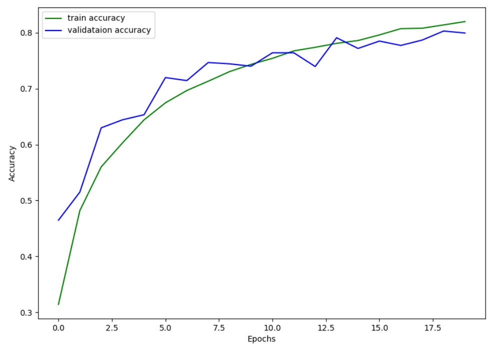

Figure 8. Accuracy of GoogleNet on Crop Disease Dataset

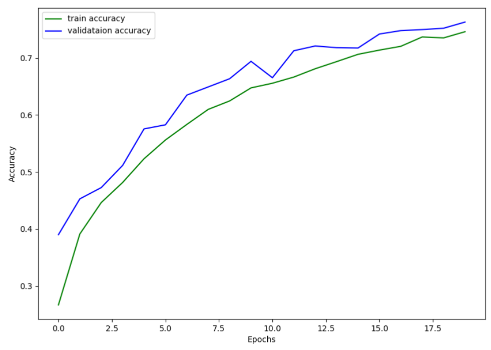

Figure 9. Accuracy of EfficientNet B0 on Crop Disease Dataset

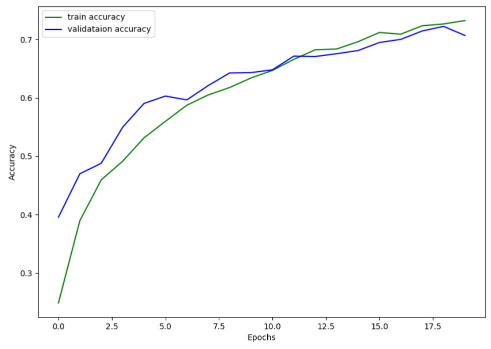

Figure 10. Accuracy of MobileNetV2 on Crop Disease Dataset

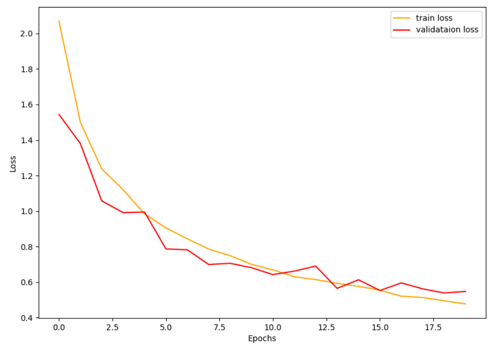

Figure 11. Loss of GoogleNet on Crop Disease Dataset

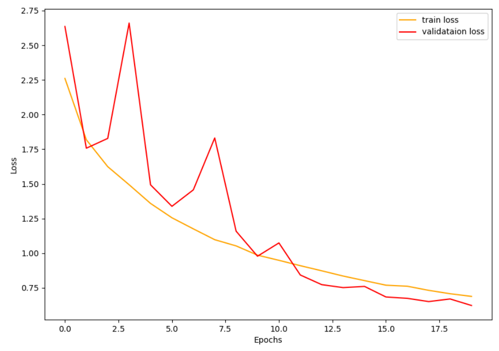

Figure 12. Loss of EfficientNet B0 on Crop Disease Dataset

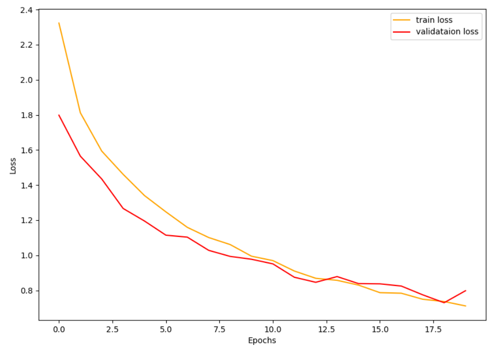

Figure 13. Loss of MobileNetV2 on Crop Disease Dataset

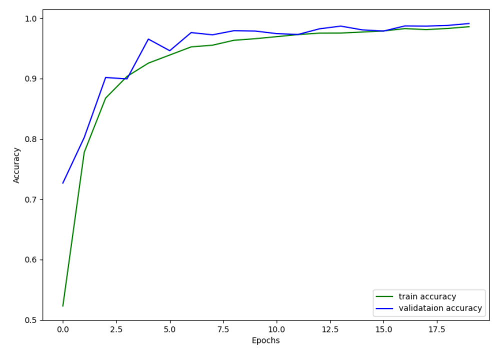

Figure 14. Accuracy of GoogleNet on Plant Village Dataset

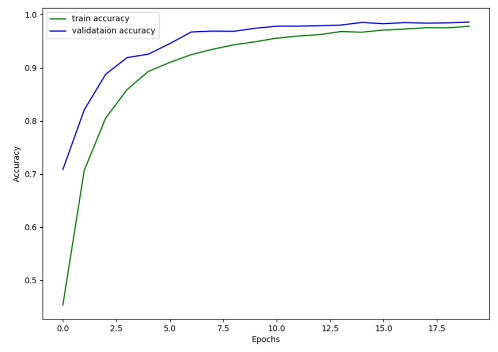

Figure 15. Accuracy of EfficientNet B0 on Plant Village Dataset

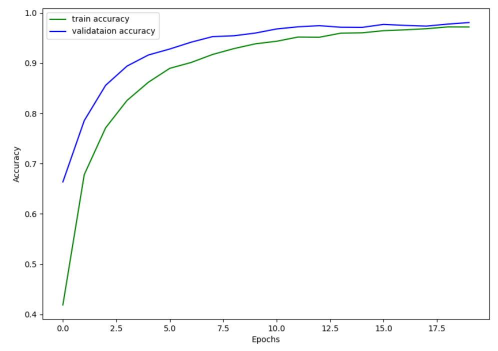

Figure 16. Accuracy of MobileNetV2 on Plant Village Dataset

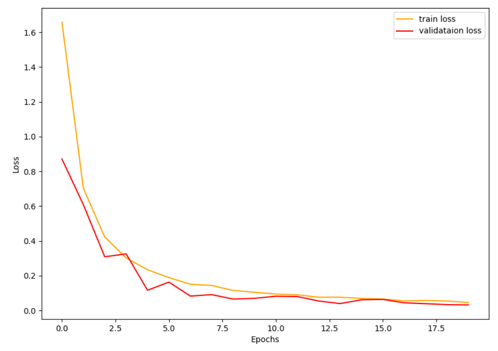

Figure 17. Loss of GoogleNet on Plant Village Dataset

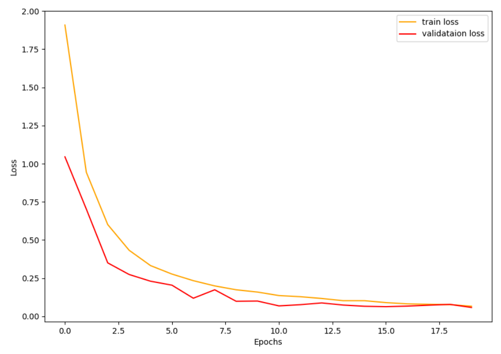

Figure 18. Loss of EfficientNet B0 on Plant Village Dataset

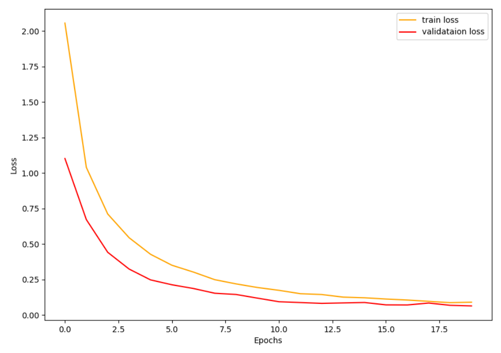

Figure 19. Loss of MobileNetV2 on Plant Village Dataset

5.1. Transfer Learning

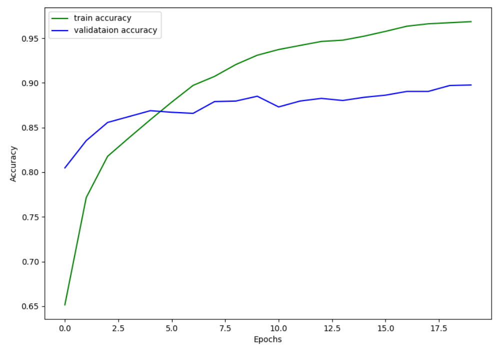

Figure 20. Accuracy-GoogleNet

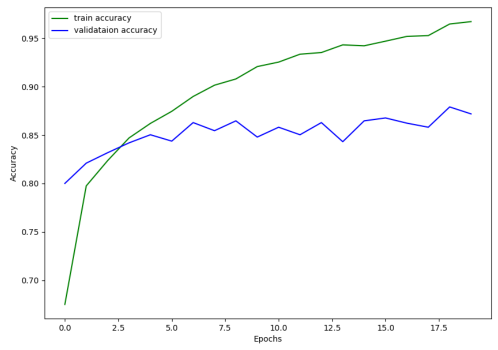

Figure 21. Accuracy-MobileNetV2

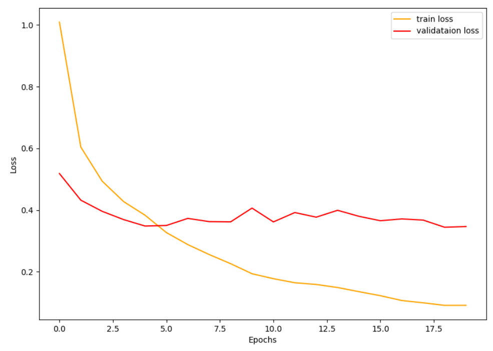

Figure 22. Loss-GoogleNet

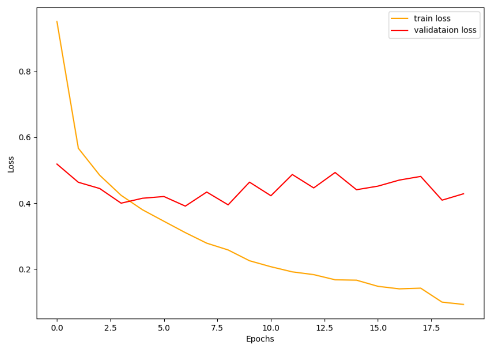

Figure 23. Loss-MobileNetV2