

Investigating the Effects of Batch Normalization on Multi-Level Image Classification

Getnet Adugna Mitiku(MSC)

getnetadugna9@gmail.com

University of Gondar

Yirga Yayeh Munaye(Phd)

Injibara University

Eshetie Teka Mitiku(MSC)

University of Gondar

Asnakew Wodaynew Abate(MSc)

University of Gondar

Awoke Berara Anagaw(MSc)

University of Gondar

Zerihun Melkamu Tensae(MSc)

University of Gondar

Getachew Alemu Baye(MSC)

University of Gondar

Research Article

Keywords: BN, MobileNetV3-Small, multi-level classification, medicinal plants, convergence, generalization

Posted Date: October 18th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-8337801/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Abstract

Batch Normalization is widely used in convolutional neural networks to enhance convergence and generalization. However, its position relative to the activation function remains underexplored in lightweight models for multi-level image classification. This study investigates these factors using the MobileNetV3-Small architecture trained on a dataset of 15,100 high-resolution images of medicinal plant parts, hierarchically labeled into 5 superclasses and 52 subclasses. Two model variants were evaluated with BN before activation, and with BN after activation, all trained under identical conditions using the Adamax optimizer and a fixed learning rate schedule. Results show that BN before activation enables faster convergence and higher accuracy, while BN after activation improves subclass-level generalization and robustness to overfitting. But, the model without BN exhibits poor performance and pronounced overfitting. These findings highlight the significance of BN placement in lightweight architectures and offer practical guidance for designing efficient hierarchical classification systems in resource-constrained environments.

1. Introduction

Deep learning has significantly advanced visual recognition tasks, enabling efficient feature extraction and representation learning in image classification, object detection, and semantic segmentation. Convolutional neural networks (CNNs)[1] have become the dominant architecture to their hierarchical feature learning capability and parameter-sharing efficiency [2]. However, the training of deep CNNs often suffers from internal covariate shift [1], vanishing gradients, and slow convergence, especially when dealing with complex and hierarchical datasets [3]. Batch Normalization(BN), introduced by [4], has become a standard technique to address these challenges by normalizing intermediate activations and stabilizing the learning process. Despite its widespread adoption, the effect of BN placement, whether applied before/after the activation function, remains an open research question, particularly for lightweight architectures and hierarchical classification[5]. Traditional implementations apply BN before the nonlinearity, assuming improved gradient flow and reduced ICS. However, several studies have indicated that post-activation normalization can yield competitive/superior results by controlling activation distribution, regularization strength[6],[7]. Understanding this behavior is crucial, as BN placement influences both the optimization dynamics and the generalization ability of neural networks. Recent works have examined BN from various theoretical and empirical views. A study by [8]demonstrated that BN smooths the optimization landscape, improving gradient predictability and allowing higher learning rates. Research by [9] reformulated BN as an equivalent convex optimization problem, revealing its implicit regularization effect.[10]Proposed separating variation modes in batch-normalized models to accelerate convergence. Similarly, [11] introduced MimicNorm, showing that the mean of weights and the last BN layer can mimic the dynamics of standard BN. These findings emphasize that BN impacts not only convergence speed but stability and representation learning, depending on its integration within the architecture. Lightweight models have gained significant attention in resource-constrained environments [11, 12]. But, limited research has explored the sensitivity of these models to BN configuration, especially when used for multi-level classification. Hierarchical

classification introduces complexity by modeling inter-class dependencies across multiple abstraction levels with superclasses and subclasses of specific species [13]. Effective normalization is therefore critical for maintaining discriminative feature consistency across these levels. In this study, we investigate the impact of BN placement on multi-level image classification using the MobileNetV3-Small architecture. We used a custom dataset of 15,100 medicinal plant images, organized hierarchically into plant parts and species. The analysis focuses on how BN placement affects convergence speed, training stability, gradient behavior, and classification accuracy, systematically comparing identical hyperparameters and training conditions. This work provides new insights into normalization sensitivity in lightweight CNNs and contributes to optimizing architectural design for resource-efficient hierarchical image recognition. Our contributions includes, we construct a multi-level classification framework to recognize medicinal plant parts; designing and evaluating BN-before/after configurations in MobileNetV3Small, providing a comparative analysis of convergence dynamics and performance on a real-world medicinal plant dataset; and offering practical insights into BN placement strategies for lightweight CNNs in imbalanced and hierarchical classification situations.

2. Related Works

Deep learning has revolutionized the field of computer vision by achieving remarkable performance in various tasks. Many real-world applications involve multi-level image classification contexts. These frameworks simultaneously predict coarse/fine-grained labels, leveraging semantic relationships between superclass and subclass categories. The introduction of BN by[3] significantly alleviated these issues by normalizing intermediate activations within each mini-batch, enabling higher learning rates, faster convergence, and improved generalization. BN standardizes layer inputs to have zero mean and unit variance, stabilizing gradient flow and improving numerical stability during training. It also provides an implicit regularization effect, reducing overfitting, enhancing generalization performance without additional computational cost[12]. Researchers[13] further revealed that BN's benefits extend beyond mitigating ICS; it smooths the optimization landscape, making gradients more predictable and improving convergence behavior. Subsequent research has investigated the effect of BN placement and its variants to well understand how normalization relates to network nonlinearity. Conventional architectures via VGG, ResNet, and MobileNet typically place BN either before or after the activation function. However, the effect of this placement remains an open research question. Studies by [8]studied normalization placement in CTR prediction models and concluded that the correct positioning of normalization layers significantly influences gradient flow, convergence rate, and predictive performance.

Another researcher[7]proposed MimicNorm, a normalization strategy that reproduces BN dynamics without relying on mini-batch statistics, achieving competitive accuracy with faster convergence. Furthermore, [5] examined how separating modes of variation through BN accelerates convergence in compact CNNs, emphasizing the importance of normalization in lightweight architectures.

A study by [8] confirmed that normalization enhances inter-class separability and intra-class compactness, leading to improved discrimination in fine-grained recognition tasks. Lightweight

architectures such as MobileNetV2 and MobileNetV3 integrate BN within both depthwise and pointwise convolutional blocks to stabilize training and maintain accuracy under resource constraints [14]. Nevertheless, BN's dependence on batch statistics can hinder its performance in small batches. To address this limitation, alternative normalization techniques such as LN [5] and GN [14] have been introduced. IN[15] as a key improvement for real-time neural style transfer. The authors demonstrate that replacing BN with IN in feed-forward style transfer networks significantly enhances visual quality and stylization consistency. This normalization allows the model to focus purely on stylistic transformation without being influenced by illumination or contrast variations across images. Through experiments, the paper shows that IN leads to more stable and aesthetically pleasing results, bridging the gap between slow optimization-based style transfer and fast feed-forward methods. The paper [16] WN proposes an alternative to BN to improve optimization efficiency in deep networks. Unlike BN, which relies on batch statistics to normalize activations, WN normalizes weights by separating their magnitude and direction. The finding show improved convergence speed and stability across supervised and reinforcement learning tasks. However, BN remains the most widely used normalization method due to its efficiency and strong compatibility with CNN architectures. A research gap remains in understanding how BN placement before versus after the activation function affects convergence stability, gradient behavior, and accuracy in multi-level classification. Prior studies mainly focus on single-level tasks or large-scale models, with limited attention to hierarchical classification and lightweight architectures. This study addresses the gap by examining BN placement in the MobileNetV3-Small architecture for multi-level image classification, using a medicinal plant dataset. It evaluates the effects of BN positioning on convergence speed, generalization, and feature learning, offering new insights into optimization behavior and training stability in compact CNNs.

3. Research framework

This study investigates the effect of BN placement relative to the activation function on multi-level image classification using a lightweight CNN architecture. The overall framework involves dataset preparation, data preprocessing, model construction with two BN placement configurations, training and validation using identical hyperparameters, and performance evaluation through convergence, stability, and accuracy analysis.

A. Dataset Description

The datasets used in this study were a locally collected medicinal plant consisting of 15,100 high-resolution RGB images. It was hierarchically labeled with two levels; Level 1 comprises five plant part categories, while Level 2 comprises 52 species classes. The dataset exhibits natural imbalance, reflecting the real-world occurrence frequencies of different plant parts and species. Images were resized to 128×128×128 and augmented with random rotations, flips, and color jittering to improve robustness.

B. Proposed system

In this study, we proposed a well-structured DL architecture for multi-level image classification, with a focus on evaluating the impact of BN in different configurations. The system follows a structured pipeline, beginning with dataset preparation, creating a CSV file for identification purposes for each medicinal plant image, and preprocessing the dataset, which includes resizing images, normalizing pixel values, and converting images into numerical arrays. The dataset is then split into training, validation, and test sets to facilitate model evaluation. MobileNetV3-Small [9] was selected as the base model due to its efficiency and suitability for lightweight applications. The base network was extended with three fully connected layers at the top, followed by a softmax classifier for both Level 1 and Level 2 tasks. This design enabled multi-level classification by jointly learning plant part recognition and species identification. To investigate the impact of BN placement, we implemented three experimental configurations: Without batch normalization, MobilenetV3 was applied without any normalization techniques to check the performance of the base architecture. BN-Before Activation: BN layers were applied directly after convolutional layers and before the nonlinear activation function. BN-After Activation: BN layers were placed after the activation function, following the conventional MobileNetV3 implementation.

These two variants allowed us to isolate the influence of BN placement on convergence dynamics, training stability, and classification accuracy.

D. Training Procedure

The training was performed using the MobileNetV3 Small architecture under 2 BN placement configurations to systematically evaluate their impact on multi-level classification. All images were resized to 128×128 pixels and normalized to the $[0, 1]$ range. Random rotations and brightness variation data augmentation were applied to improve performance under real-world conditions. Transfer learning was adopted by initializing the backbone with ImageNet-pretrained weights, while newly added fully connected layers were trained first with the feature extractor frozen. Optimization was carried out using the Adamax optimizer with an initial learning rate of $1e-5$ and a batch size of 64, minimizing categorical cross-entropy loss at both Level outputs. After initial convergence, all layers were unfrozen for fine-tuning with a reduced learning rate to further refine feature representations while limiting overfitting.

Models were trained for 50 epochs, using early stopping based on validation loss with a patience of 10 epochs, and model checkpointing to retain the best weights. Training was conducted in a GPU-enabled environment, and identical hyperparameters were used for both configurations to ensure a fair and controlled comparison.

E. Evaluation Metrics

In this study, model performance was evaluated using hierarchical and class-wise metrics to assess both coarse-fine-grained classification. Accuracy was computed separately for the 5 superclass categories and the 52 subclass species to analyze how BN placement affects different feature levels.

Precision measures the correctness of positive predictions, Recall evaluates the ability to identify relevant subclass samples, and the F1-score balances both, making it well-suited for the imbalanced medicinal plant dataset. In addition, training and validation cross-entropy losses were analyzed to evaluate convergence speed, optimization stability, and generalization.

4. Results and Discussion of BN before the activation function

The MobileNetV3-Small model with BN placed before the activation function shows a stable and well-structured learning process over 50 epochs. At epoch 1, performance is low (loss 6.3950; subclass accuracy 0.0725; superclass accuracy 0.0495), reflecting initial feature learning. From epoch 2 onward, accuracy improves rapidly, indicating that pre-activation BN stabilizes gradients and accelerates convergence. By epoch 5, subclass and superclass accuracies rise to 0.6049 and 0.8830, and by epoch 10, they reach 0.8193 and 0.9419, with closely matched validation results, demonstrating good generalization. Between epochs 20 and 30, the model achieves stable convergence, with subclass accuracy consistently above 0.93 and superclass accuracy near 0.99, alongside smoothly decreasing losses. In the final epochs, performance saturates, reaching accuracy(0.9751,0.9986)validation, accuracies (0.9561, 0.9930) for subclass and superclass, respectively. The absence of oscillations and the low final validation loss (0.146) confirm that BN before activation promotes smooth optimization, strong generalization, and effective hierarchical feature learning. The results showing Fig. 2, placing BN before the activation function, leads to fast, stable, and well-generalized learning. In the training, accuracy, superclass training accuracy rises rapidly and reaches nearly 100% by epoch 20, while validation accuracy closely follows at 97–98%, indicating strong generalization and minimal overfitting. For the more challenging subclass task, training accuracy stabilizes around 95–96%, with validation accuracy slightly lower (92–94%), reflecting a small and expected generalization gap. The training Loss shown in the figure further confirms stable optimization.

5. Training result of BN after the activation function

With BN placed after the activation, the model shows very fast and stable convergence after an initial warm-up. Although epoch 1 starts poorly, performance improves rapidly; by epoch 6, validation subclass accuracy exceeds 92%, and by epoch 10, it reaches 95%. Peak validation performance occurs around epoch 25 (val subclass accuracy 96.3%, val loss 0.096), after which training plateaus with minimal oscillations. Superclass accuracy saturates early (99–100%), while subclass accuracy improves more gradually, reflecting the difficulty of fine-grained classification. The generalization gap is small (2–3%), indicating good generalization with only mild overfitting. The training accuracy is shown in Fig. 3, with rapid learning for both hierarchies. Superclass training and validation accuracies rise quickly and reach 0.99 by around epoch 15, with nearly overlapping curves, indicating excellent generalization and minimal overfitting at the superclass level. Subclass training accuracy increases to 0.98, while validation accuracy stabilizes around 0.95, showing only a small generalization gap, which is expected due to the finer-grained nature of subclass classification.

Training and validation losses drop sharply, indicating fast convergence and stable optimization. The close alignment of training and validation losses suggests no significant overfitting/under fitting. Subclass loss decreases slightly faster in training than validation, consistent with the small accuracy gap.

Conclusion and Future Work

This study examined how BN placement relative to the activation function affects multi-level image classification using a lightweight CNN architecture. Experiments on a 15,100 medicinal plant dataset with superclass and subclass hierarchies showed that BN before activation leads to faster convergence, smooth optimization, improved gradient stability, and smaller generalization gaps than BN after activation. Although both configurations achieved comparable final validation accuracy, pre-activation BN proved better robustness, regularization, and hierarchical feature separability, confirming BN placement as a key architectural design choice. BN is shown to act not only as a normalization mechanism but also as an implicit regularizer shaping feature learning dynamics. Future work includes extending the analysis to other normalization methods, LN, GN, and IN, combining BN placement with neural architecture search, and applying the insights to transfer learning and domain adaptation

Declarations

Conflict of interest

The authors declared that they have no known financial that could have appeared to influence the work reported in this paper.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Author Contribution

Getnet Adugna Mitiku(MSc) led the overall research design and methodology, conducted data preprocessing, model development, and experimental analysis, and wrote the main manuscript while coordinating all revisions, designed the BN placement experiments on MobileNetV3-Small, and analyzed the resulting convergence behavior.Yirga Yayeh Munaye (Phd) provided supervision, guidance, and technical review, and also assisted in refining the research framework and validating the results.Eshetie Teka Mitiku supported dataset preparation and quality inspection and assisted in running comparative model experiments, verifying the consistency of labels in the 15,100-image dataset of medicinal plants.Asnakew Wodaynew Abate (MSc) contributed to algorithm optimization and code debugging,

helped improve training stability and hyperparameter tuning, adjusted learning rate schedules, and improved early-stopping strategies. Awoke Berara Anagaw (MSc) participated in experimental validation and result verification, assisted with the literature review and related works section, and collected and summarized recent papers on BN effects in lightweight CNNs. Zerihun Melkamu Tensae (MSc), data augmentation techniques and the preprocessing pipeline, and assisted in preparing figures, tables, and accuracy comparison charts, implemented augmentation methods. Getachew Alemu Baye (MSc) assisted with software implementation and system integration, supported manuscript organization and formatting, structured the experimental results section, and formatted the tables according to IEEE guidelines. Funding The authors declare that they have no funds, grants, or other support that were received during the preparation of this manuscript.

Data Availability

The dataset analyzed in the current study is not publicly available due to the sensitive nature of the human participant data and ethical restrictions imposed by the relevant traditional herbalist.

References

1. M. Awais, M. T. Bin Iqbal, and S. H. Bae, "Revisiting Internal Covariate Shift for Batch Normalization," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 32, no. 11, pp. 5082–5092, 2021, doi: 10.1109/TNNLS.2020.3026784.
2. K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1–14, 2015.
3. S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *32nd Int. Conf. Mach. Learn. ICML 2015*, vol. 1, pp. 448–456, 2015.
4. E. Sari, M. Belbahri, and V. P. Nia, "How Does Batch Normalization Help Binary Training?" no. NeurIPS, 2019, [Online]. Available: <http://arxiv.org/abs/1909.09139>
5. M. M. Kalayeh and M. Shah, "Training Faster by Separating Modes of Variation in Batch-Normalized Models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 6, pp. 1483–1500, 2020, doi: 10.1109/TPAMI.2019.2895781.
6. T. Ergen, A. Sahiner, B. Ozturkler, J. Pauly, M. Mardani, and M. Pilanci, "Demystifying Batch Normalization in Relu Networks: Equivalent Convex Optimization Models and Implicit Regularization," *ICLR 2022 - 10th Int. Conf. Learn. Represent.*, no. 2016, pp. 1–36, 2022.
7. W. Fei, W. Dai, C. Li, J. Zou, and H. Xiong, "MimicNorm: Weight Mean and Last BN Layer Mimic the Dynamic of Batch Normalization," no. 2015, 2020, [Online]. Available: <http://arxiv.org/abs/2010.09278>
8. Z. Wang, Q. She, P. Zhang, and J. Zhang, "Correct Normalization Matters: Understanding the Effect of Normalization on Deep Neural Network Models for Click-Through Rate Prediction," no. 1, 2020, [Online]. Available: <http://arxiv.org/abs/2006.12753>

9. A. Howard, W. Wang, G. Chu, L. Chen, B. Chen, and M. Tan, "Searching for MobileNetV3 Accuracy vs MADDs vs model size," *Int. Conf. Comput. Vis.*, pp. 1314–1324, 2019.
10. M. Sandler, A. Howard, M. Zhu, and A. Zhmoginov, "Sandler_MobileNetV2_Inverted_Residuals_CVPR_2018_paper.pdf," *arXiv*, pp. 4510–4520, 2018.
11. T. Chen, S. Lu, and J. Fan, "SS-HCNN: Semi-Supervised Hierarchical Convolutional Neural Network for Image Classification," *IEEE Trans. Image Process.*, vol. 28, no. 5, pp. 2389–2398, 2019, doi: 10.1109/TIP.2018.2886758.
12. J. Bjorck, C. Gomes, B. Selman, and K. Q. Weinberger, "Understanding batch normalization," *Adv. Neural Inf. Process. Syst.*, vol. 2018-Decem, no. NeurIPS, pp. 7694–7705, 2018.
13. O. Elharrouss, Y. Akbari, N. Almadeed, and S. Al-Maadeed, "Backbones-review: Feature extractor networks for deep learning and deep reinforcement learning approaches in computer vision," *Comput. Sci. Rev.*, vol. 53, 2024, doi: 10.1016/j.cosrev.2024.100645.
14. Y. Wu and Kaiming He, "Group Normalization – Facebook Research," *Eur. Conf. Comput. Vis.*, no. Figure 1, pp. 3–19, 2018, [Online]. Available: <https://research.fb.com/publications/group-normalization/>
15. D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Instance Normalization: The Missing Ingredient for Fast Stylization," no. 2016, 2017, [Online]. Available: <http://arxiv.org/abs/1607.08022>
16. T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," *Adv. Neural Inf. Process. Syst.*, no. Nips, pp. 901–909, 2016.

Figures

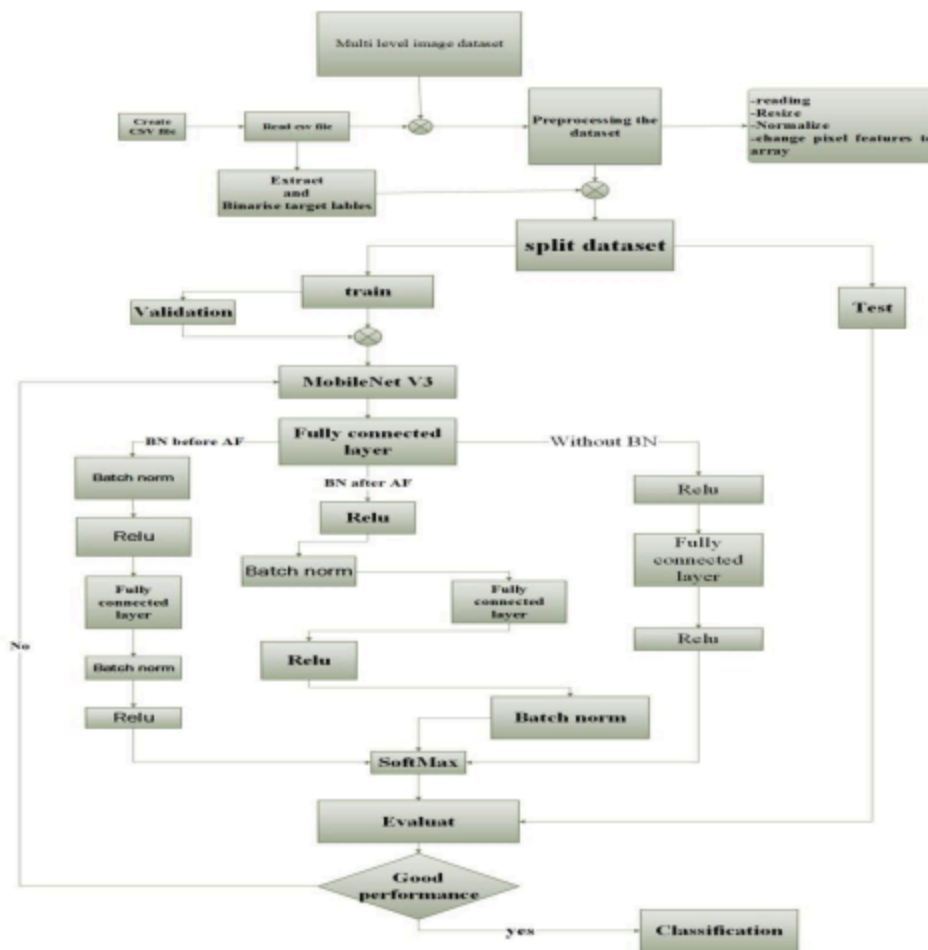


Figure 1

Proposed system

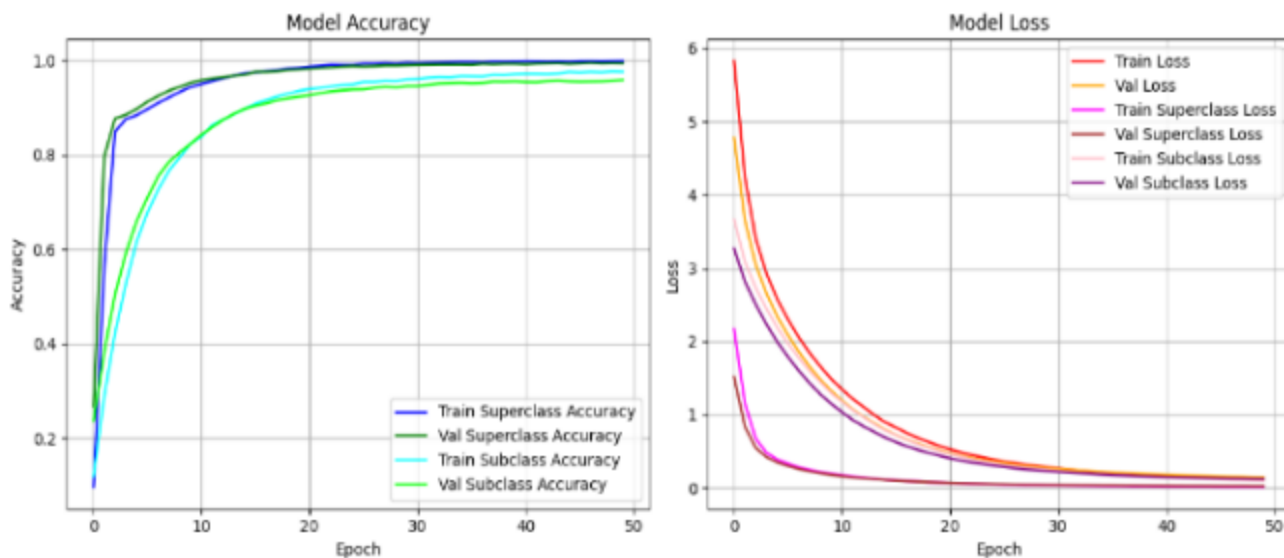


Figure 2

Training result of BN before the activation function

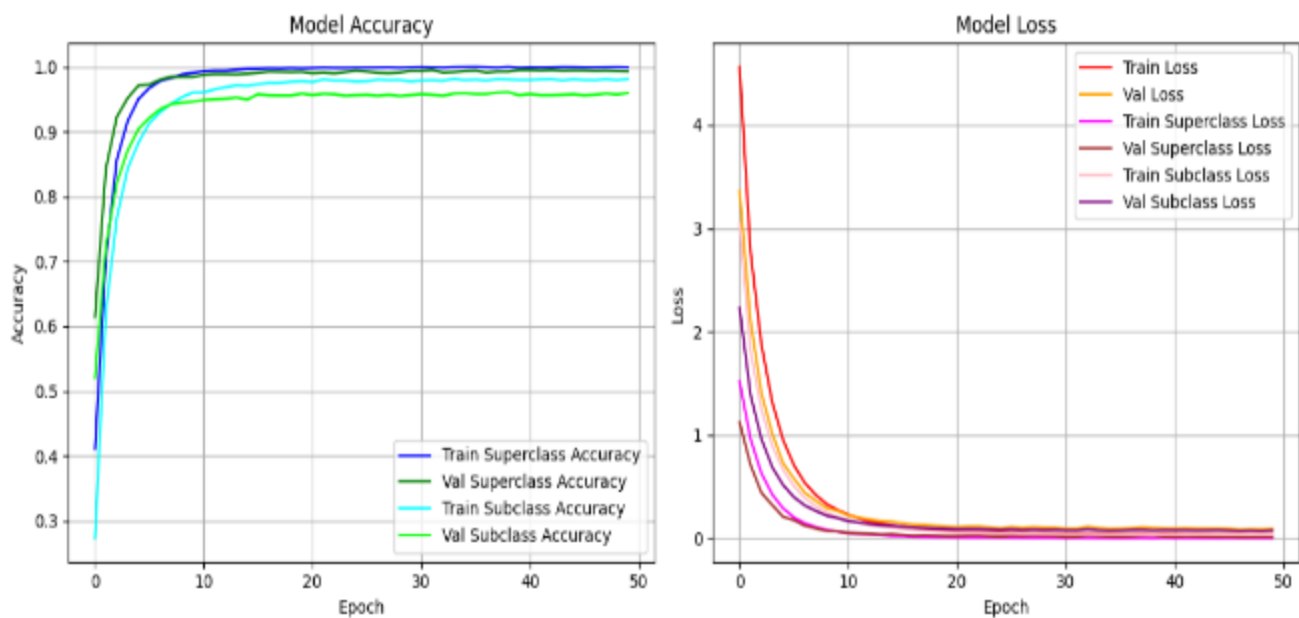


Figure 3

Training result of BN after the activation function