

Performance Enhancement of Deep Learning Models Through Optimization Techniques for Automated Apple Grading

Sandeep Kaur¹, Manoj Kumar Sachan², Ashwani Kumar Aggarwal³

¹Department of Computer Science and Engineering, Sant Longowal Institute of Engineering and Technology, Longowal-148106, India, Email: sandeep.sandeep.sran@gmail.com

²Department of Computer Science and Engineering, Sant Longowal Institute of Engineering and Technology, Longowal-148106, India, Email: manojksachan@sliet.ac.in

³Department of Electrical and Instrumentation Engineering, Sant Longowal Institute of Engineering and Technology, Longowal-148106, India, Email: ashwani.ist@sliet.ac.in

Abstract: Automated grading of agricultural produce requires accurate and robust visual classification, yet the performance of deep learning models is highly sensitive to hyperparameter selection and training dynamics. This study presents an optimization-driven deep learning framework for automated apple grading aimed at enhancing classification accuracy, convergence stability, and feature representation through systematic hyperparameter tuning. A dataset of 1,264 images of Royal Delicious apples categorized into four grades was used to evaluate CNN, ResNet50, EfficientNet-B0, and a CNN–Autoencoder fusion model under baseline and optimized settings. Baseline results with default hyperparameters achieved accuracies of 90.8% (CNN), 93.7% (ResNet50), and 95.1% (EfficientNet-B0), indicating moderate generalization and higher validation loss. After applying Grid Search, Bayesian Optimization, PSO, and Differential Evolution, significant performance improvements were observed. The proposed CNN–Autoencoder fusion model optimized using Differential Evolution achieved the highest accuracy of 97.3%, with precision 0.97, recall 0.96, F1-score 0.96, lowest validation loss (0.15), and fastest convergence (epoch 17). Overall, optimization improved accuracy, reduced loss, enhanced training stability, and produced more discriminative latent features, confirming the effectiveness of systematic hyperparameter optimization in automated grading systems.

Keywords: Automated Apple Grading; Deep Learning; Hyperparameter Optimization; Bayesian Optimization; Differential Evolution; EfficientNet; CNN–Autoencoder Fusion; Transfer Learning; Agricultural Image Classification

1. Introduction

Deep learning has proven to be an efficient approach to problem solving in some form of complex classification and pattern recognitions in the past few years because it can automatically learn useful features out of data [1]. Compared to traditional machine learning systems which depend on manually engineered features and extensive human input, deep learning models can accept complex patterns, nonlinear relationships and subtle variations in large and high-dimensional data. This makes them especially practical in activities involving visual information, fine-grained categorization and settings rich in characteristics and normally poorly performed through the standard approaches [2]. Good learning ability however does not come as a guarantee to optimum performance. Deep learning models are highly sensitive to hyperparameters, network structure and training settings and may impact the success of the model in terms of learning and generalization. The important parameters which directly affect the convergence rate, the model and the overall accuracy are learning rate, batch size and latent feature size and regularization. The model may slow, overfit, underfit, or not optimum results may be attained when these are randomly set, or default values are used [3]. The second of these problems is high-dimensional feature space, where repetitive, corrupting or less informative features can reduce model efficiency and increase computational cost. To address this, the common solution to dimensionality reduction and representation learning is to obtain a more meaningful and reduced representation of features [4]. However, the further optimization of the model and training is necessary, and it is an absolute must to attain the efficacy of the features, learned. Without a well-crafted optimization strategy, deep learning models may fail to capture the underlying information trends and will only provide minimal performance gains and underperform at generalization [5]. To address these limitations, optimization strategies and deep learning have now become central topics of investigation. Learning with optimization aims to find model hyperparameters, as well as training conditions that are automatically optimized to maximize predictive accuracy, minimize loss and achieve better convergence behavior. Systematized methods of search in the hyperparameter space include the grid or Bayesian optimization and evolutionary algorithms, which provide more efficient methods of searching the space, in comparison to trial-and-error methods. These methods enable the profound models to possess greater generalization and power in the varying data conditions by guiding the learning process to more pertinent parameterizations [6,7].

1.1 Motivation

One notable fact in the present research is that the baseline deep learning models may be discovered to deliver a reasonably, but not optimum performance, without optimization. On the other hand, the accuracy improvement is always consistent, converges faster and the loss on validation is reduced with correctly formulated optimization strategies. However, most of the accessible literature draw comparisons between deep learning and optimization separately as opposed to offering a comparative study prior to and following analysis demonstrating clearly the obtained performance gains by optimization. Such discontinuity makes proposed frameworks less interpretable and useful in performance-important applications. Based on these concerns, the proposed research assumes the optimization-based approach to deep learning where the best performance of the models is firstly tested under the circumstances of baselines and then optimized with the help of advanced optimization techniques. Instead of performing optimization as a separate stage of the model development process, it is embedded directly into the model development pipeline to optimize hyperparameters, learn superior features and stabilize training dynamics. The systematic methodology makes it possible to understand the performance compared to the pre- and post-optimization performance clear, which allows one to get more insights into the actual contribution of the optimization to the model effectiveness.

The primary objective of the research is to devise an impressive deep learning architecture the performance of which is enhanced in a systematic manner by including hyperparameter optimization and enhanced learning of a feature representation. The paper is specifically looking at a holistic evaluation scheme which involves accuracy, loss behaviour, convergence stability and generalization performance. Due to the proposed work, one is likely to demonstrate that optimization techniques can significantly enhance the results of deep learning under the condition of the intensive classification task with the assistance of the step-by-step comparative analysis of pre- and post-optimization models.

2. Literature Work

Recent studies indicate that there has been a radical change towards optimization-aided deep learning architectures in automatic classification, agricultural vision, and grading systems. Although the deep learning models like CNNs, EfficientNet, and hybrid networks are characterized by good feature learning abilities, several reports suggest that their success is extremely reliant on hyperparameter optimization, feature optimization, and convergence stability. Thus, the combination of deep learning and optimization methods has been a research imperative to enhance accuracy and generalisation of models in automated grading and visual inspection.

Khan et al. (2024) [8] came up with a multimodal deep hybrid learning system with a Bayesian optimization that offers automated plant disease classification. The authors emphasized that the systematic optimization is required to the key stability features learning on agricultural image datasets. By using metaheuristic optimization methods, Ivanovski et al. [11] improved the regression tree models to predict peach firmness. Gray wolf optimization was the most promising of the tested methods, which dropped the RMSE by 1.72 to 1.57 (8.8 percent improvement), which demonstrates the usefulness of optimization in enhancing the accuracy of agricultural prediction and harvest decision-making. In addition, Han et al. (2025) [12] made up an automated apple grading system through the implementation of computer vision, image processing and machine learning. They came up with a light model (FDNet-p) that detected apple stems and still used a better segmentation technique in separating the apple body and the background. Afterward, they took out aspects like the color, shape, and diameter and presented a GBDT-based model to categorize the apples. As it was indicated in their experiments, the accuracy of the system is high in both detecting stems and grading the fruits, which proves that the given system is efficient and reliable and can be applied to automated fruit grading. Dewi et al. (2024) [13] employed a Random Forest model to classify melons based on color histogram and texture features based on the GLCM, and applied hyperparameter optimization using gray wolf optimization. The precision of the model was optimized to 92.43 by tuning the most important parameters that included n estimators, max feature, and max depth which showed that feature-based optimization was effective in classifying fruit quality. Furthermore, Ukwuoma et al. (2022) [14] conducted a literature review to research on the application of computer vision and deep learning in detecting and classifying fruits in agriculture. They examined widely used data sets, feature descriptions, model realizations, and the difficulties associated with fruit recognition because of shape, color, and texture changes. They also created their own deep learning model in classifying fruits based on the Fruit 360 dataset to aid in illustrating how deep learning can be utilized in agricultural uses on top of the review. Naveen Kumar et al. (2025) [15] suggested a plant disease detection system, which integrates adaptive hybrid K-means and fuzzy C-means clustering in the effort to separate diseased regions, with transfer learning models (EfficientNet, ResNet, and DenseNet) to classify them. Feature extraction and weight fine-tuning were optimized using the improved random variable-based water strider algorithm (IRV-WSA), leading to the IRV-WSA-ETLNet model achieving 94.85% accuracy with reduced computation time, demonstrating the effectiveness of optimization-based feature selection and parameter tuning for automated plant disease diagnosis.

Together, these works form three major research findings. To start with, deep learning networks like CNNs and hybrid networks prove to be very useful in automated visual classification and grading problems because they have high ability to extract features in a hierarchical manner. Second, optimization and optimization-based methods such as Bayesian optimization, evolutionary algorithms, and metaheuristic methods are always more accurate in the models, achieve lower validation loss, and faster convergence than the default deep learning models trained using default parameters. Third, the hybrid optimization-deep learning models offer better robustness and generalization, especially with the systems of agriculture and fine-grained classification that must capture subtle variations in the displayed images.

Nevertheless, there is a gap in the literature concerning critical research. In most studies, the ultimate optimized performance of deep learning models is solely reported but no structured comparison between baseline (pre-optimization) and optimized (post-optimization) results is presented. This does not allow the contribution to optimization to be interpreted clearly in a before and after manner and it is challenging to measure the real increase in performance that has come about by use of optimization methods.

Thus, based on these gaps, the current study is aimed at the performance improvement of deep learning models under systematic optimization methods to the automated grading of apples. In contrast to the previous studies that present only the optimized results, this study shows the full before-after optimization analysis and clearly illustrates how optimization optimizes hyperparameters, features representation, minimizes validation loss, and other deep learning model performance in automated grading systems.

3. Methods and Materials

The general methodological scheme followed in the paper is shown in Figure 1 and it represents a systematic flow of the apple image classification process, starting with acquisition of raw data and followed by preprocessing, model implementation, optimization and evaluation of performance.

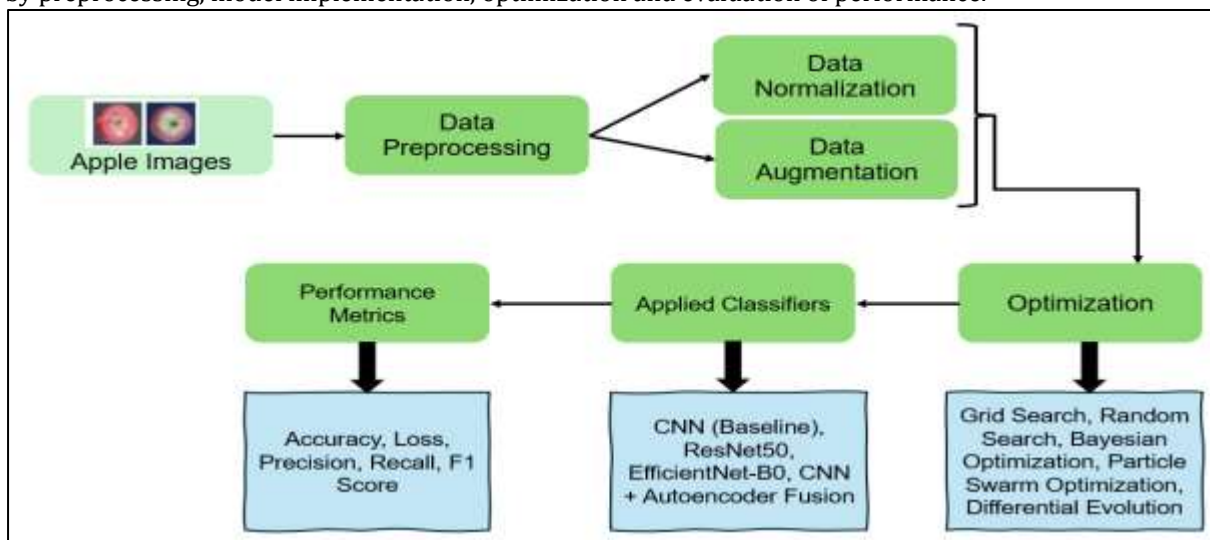


Figure 1: Proposed framework for apple image classification with preprocessing, optimized classifiers, and performance evaluation

3.1 Dataset Description

The data utilized in this are high quality pictures of Royal Delicious apples that were gathered in a commercial orchard within Kashmir, India. The sample population used was 158 apples that were chosen to reflect the reality of changes in surface appearances, shape and color. All the apples were photographed at various points of view to be able to cover the visual space completely and, as a result, 1,264 photos were analyzed. All images were obtained under controlled environmental conditions to ensure the consistency of data and minimize the external variability. The imaging system made sure that there was homogeneous lighting, camera distance was the same, the background was consistent, and the orientation was the same in all the samples. This acquisition process can reduce the noise caused by variation in the illumination and background interference thus enhancing the accuracy of the deep learning extraction of features and classification. The data was annotated manually following image acquisition by agricultural specialists according to visual quality features of uniformity of surfaces, regularity of color, regularity of shape and occurrence of defects. The pictures were classified under four grades of quality Grade A, Grade B, Grade C, and Grade D (Figure 2). This suggestion by professionals adds to the credibility of the dataset and supervised learning with ground truth. An even balance of the classes is maintained to prevent biased models during training and to provide the convergence stability. The number of Grade A apples images is 256, Grade B, Grade C and Grade D. are 336, just to ensure that all the categories of a grading are well represented [16].

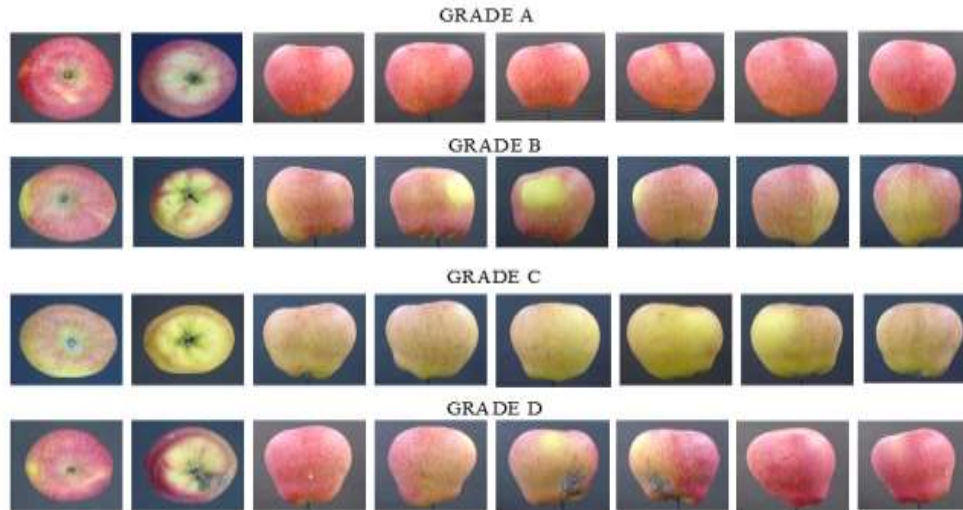


Figure 2: Sample images of dataset

3.2 Data Preprocessing

Deep learning models should be prepared and pre-processed effectively to promote the stability of the model, accuracy, and ability to generalize in the models. Since it is possible that the raw image data can be prone to various scale, light, and distortion issues, a systematic preprocessing pipeline was presented to standardize the input data before the model was trained. The preprocessing step includes image normalization, data augmentation, and dataset splitting, which increase the resilience of a model and reduce the level of overfitting.

3.2.1 Data Normalization

Data normalization was carried out whereby the values of the input image were standardized, and the convergence during deep learning training was maintained. All the images were down sampled to a fixed resolution that is compatible with the deep learning architecture to have uniform spatial dimensions throughout the dataset. After resizing, the pixel intensity values were brought to a standard range, which is usually the range between 0 and 1 by dividing each pixel value by the highest possible intensity level (255). Such normalization process averts numerical instabilities and leads to faster gradient convergence as well as makes sure that the learning process is not dominated by features with large magnitudes. Also, normalization leads to better learning of deep learning models by minimizing internal covariate shift and enhancing optimization efficiency. Input data standardization is especially relevant when using optimization methods in hyperparameter optimization, because homogeneous scaling of the input enhances more valid model evaluation and comparison between control and optimized models.

3.2.2 Data Augmentation

To increase the variety of the dataset and the generalization potential of the deep learning model, the method of data augmentation was implemented in the training process. The dataset size is moderate, and thus the augmentation is used to mimic the variability and avoids overfitting by exposing the model to various transformations of the same image (random rotation and horizontal and vertical flipping, brightness, and minor scale transformations). These changes maintain the semantic meaning of the apple photographs but add diversity in orientation, light and space. The model will therefore be resistant to the minor changes in the environment as well as conditions of viewing that may occur in the actual automated grading systems. The data was subsequently divided into two groups i.e. 70 percent training and 30 percent testing. The training set provides model parameters and representation features of the learning process. Final model testing was only postponed as the way of evaluating the performance of model classification and generalization to unknown data.

3.3 Optimization Techniques Applied

To formulate the answer to the question of how to solve the model performance and to conduct systematic search of the space of hyperparameters, several optimization methods were applied in the context proposed. The two techniques provide a clear-cut search process to identify the best parameter parameterization within a complex and nonlinear environment of learning. In Grid Search [17], an exhaustive search through all possible combinations of hyperparameters is done within a specified grid.

$$\lambda^* = \arg \max_{\lambda \in \Lambda_{\text{grid}}} \text{Accuracy}(\lambda)$$

This method is computationally intensive, but it offers a good basis on which optimization performance can be compared. Random Search [18] samples configurations hyperparameters are sampled using a random sample of the parameter space:

$$\lambda_i \sim P(\Lambda), i = 1, 2, \dots, n$$

Stochastic method is more effective than exhaustive search and is frequently able to find near-optimal solutions at a lower computational cost. Bayesian Optimization builds a probabilistic surrogate model of the objective function and repeatedly chooses potential hyperparameters with the help of an acquisition function:

$$\lambda_{t+1} = \arg \max_{\lambda} \alpha(\lambda | D_t)$$

where $\alpha(\lambda | D_t)$ is the acquisition function and D_t represents previously evaluated configurations. This method efficiently balances exploration and exploitation in high-dimensional search spaces. Particle Swarm Optimization (PSO) [19] is a population-based metaheuristic inspired by swarm intelligence. The velocity and position of each particle are updated as follows:

$$\begin{aligned} v_i^{t+1} &= wv_i^t + c_1r_1(p_i - x_i^t) + c_2r_2(g - x_i^t) \\ x_i^{t+1} &= x_i^t + v_i^{t+1} \end{aligned}$$

where v_i is velocity, x_i is the particle position, p_i is the personal best solution, and g is the global best solution. Differential Evolution (DE) [20] is an evolutionary optimization technique that utilizes mutation and crossover operations:

$$\begin{aligned} v_i &= x_{r1} + F \cdot (x_{r2} - x_{r3}) \\ u_i &= \begin{cases} v_i, & \text{if rand} \leq CR \\ x_i, & \text{otherwise} \end{cases} \end{aligned}$$

F is being the mutation factor and CR is the crossover rate. DE has been shown to be effective in the optimization of nonlinear and high-dimensional hyperparameter space. Table 1 provides a comparative study on a number of optimization methods that can be applied to model tuning with the focus on the search strategy, exploration capacity, search speed, and the cost of computation. This comparison helps in understanding the trade-offs between exhaustive, probabilistic, and population-based methods, and supports the selection of suitable optimization algorithms for improving model performance and efficiency.

Table 1: Comparative Analysis of Optimization Techniques

Technique	Search Type	Exploration Ability	Convergence Speed	Computational Cost
Grid Search	Exhaustive	Low	Slow	High
Random Search	Stochastic	Moderate	Moderate	Medium
Bayesian Optimization	Probabilistic	High	Fast	Medium
PSO	Population-Based	High	Fast	Medium
Differential Evolution	Evolutionary	Very High	Very Fast	Medium

3.4 Experimental Setup

This part will describe the experimental set-up in detail, which was used to test the functionality of the suggested optimization-based deep learning platform. This has been achieved by creating a standardized experimental set up that ensured that it was reproducible, there was a fair comparison between the baseline and optimized model as well as the proper assessment of the performance improvements. The experiments were conducted under controlled training and same dataset partitioning; preprocessing pipeline and evaluation protocols and pre-optimization and post-optimization models were maintained under the same conditions of experimentation.

3.4.1 Hardware and Software Environment

The experiments were conducted on a deep learning platform which was developed on the existing computational resources to ensure that the models can be trained and optimized successfully. The models were implemented and trained using the popular deep learning libraries, which included TensorFlow/Keras and PyTorch to code the framework and run the model. Data preprocessing and augmentation was done using NumPy, OpenCV and Scikit-learn libraries. This was trained on a high-performance computing system that consisted of a gpu enabled system to calculate deep learning at greater speed and reduce the training time. Stabilization of model convergence and effective hyperparameter optimization were implemented using software environment Python (version 3.x), CUDA-enabled GPU support (where applicable) and compatible deep learning frameworks. The computational environment can support scaleable experimentation and compared the findings of different optimization methods.

3.4.2 Training Epochs

The deep learning models were trained for a fixed number of epochs to ensure sufficient learning of feature representations while preventing excessive overfitting. E denotes the overall amount of training epochs. Each iteration was an update of the model parameters with mini-batch gradient descent.

$$\theta^{(e+1)} = \theta^{(e)} - \eta \nabla_{\theta} \mathcal{L}(\theta^{(e)})$$

where $e = 1, 2, \dots, E$, η is the learning rate, and $\mathcal{L}$ represents the loss function. Monitoring of the training was done across epochs to examine the convergence behavior and the stability of the performance of the training prior to and after optimization. Table 2 describes the parameters of the training configuration that were employed in the development of the model, such as the epochs, batch size, learning rate, the optimizer, the loss function, and input image size. These environments guarantee a stable training and efficient convergence and a stable performance in the process of classification.

Table 2: Training Configuration Parameters

Parameter	Symbol	Value/Description
Total Epochs	E	30–50
Batch Size	B	16–32
Learning Rate	η	Optimized during training
Optimizer	—	Adam (primary), SGD (comparative)
Loss Function	$\mathcal{L}$	Categorical Cross-Entropy
Input Image Size	—	Fixed (e.g., 224×224)

3.4.3 Applied Classifiers

In automated apple grading, deep learning models aim to learn discriminative representations of visual attributes such as color uniformity, surface defects, bruising, texture irregularities, and shape deformation to accurately classify apples into predefined grade categories. Let the dataset be defined as $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, where $x_i \in \mathbb{R}^{H \times W \times C}$ represents the apple image and $y_i \in \{1, 2, \dots, K\}$ denotes the grade label [21]. The objective is to learn a parametric function $f_{\theta}(x)$ that minimizes the categorical cross-entropy loss:

$$\mathcal{L}(\theta) = -\frac{1}{N} \sum_{i=1}^N \sum_{k=1}^K y_{ik} \log(\hat{y}_{ik})$$

where $\hat{y}_{ik}$ is the predicted probability obtained using the Softmax function. The baseline Convolutional Neural Network (CNN) extracts hierarchical spatial features through convolutional operations defined as:

$$z^{(l)} = W^{(l)} * a^{(l-1)} + b^{(l)}, a^{(l)} = \sigma(z^{(l)})$$

where $W^{(l)}$ denotes learnable kernels, $*$ represents convolution, and σ is a nonlinear activation function such as ReLU. While the baseline CNN captures essential grading features, performance enhancement is achieved through architectural optimization and regularization strategies. To further improve feature learning and gradient stability, ResNet50 [22] employing transfer learning introduces residual connections defined as:

$$y = F(x, W) + x$$

where $F(x, W)$ represents the residual mapping. The identity shortcut enables deeper network training by mitigating vanishing gradient problems, thereby enhancing classification robustness. Transfer learning accelerates convergence and improves generalization by initializing weights from a pre-trained model and fine-tuning them for apple grading.

EfficientNet-B0 [23] improves performance through compound scaling, which uniformly scales network depth d , width w , and input resolution using a scaling coefficient ϕ :

$$d = \alpha^{\phi}, w = \beta^{\phi}, r = \gamma^{\phi}$$

subject to the constraint $\alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$. This balanced scaling enhances accuracy while maintaining computational efficiency, making it suitable for real-time automated grading systems. Finally, the CNN + Autoencoder Fusion model [24] enhances representation learning by integrating unsupervised feature extraction with supervised classification. The autoencoder learns a compressed latent representation:

$$z = f_{\text{enc}}(x), \hat{x} = f_{\text{dec}}(z)$$

by minimizing reconstruction loss:

$$\mathcal{L}_{\text{rec}} = \|x - \hat{x}\|^2$$

The encoded latent vector z captures defect-sensitive and texture-rich features, which are fused with CNN features:

$$F_{\text{fusion}} = [F_{\text{CNN}} \oplus z]$$

where $\oplus$ denotes feature concatenation. The final classifier operates on F_{fusion} , resulting in improved grading accuracy due to richer and more discriminative representations. Overall, performance enhancement in automated apple grading is achieved through optimization techniques such as transfer learning, residual learning, compound scaling, feature fusion, adaptive optimizers (Adam/SGD), batch normalization, and regularization, all of which contribute to improved convergence speed, higher

classification accuracy, reduced overfitting, and better generalization across varying apple quality conditions.

4. Results and Discussions

The performance analysis of the applied models in various optimization conditions and training parameters is shown in this section. The results are measured with the use of the principal measures like accuracy, loss, precision, recall, and F1-score to evaluate the classification performance, convergence tendency, and the overall robustness of the model.

Table 3 shows the initial performance of the deep learning models trained with default hyperparameter values i.e. a learning rate of 0.001 and a batch size of 32. This is because the results are the first source used to determine the efficiency of optimization methods in enhancing the model performance. In these default training conditions, the conventional CNN model can achieve an accuracy of 90.8 percent with precision, recall and F1-score of 0.90, 0.89 and 0.89 respectively. The comparatively low validation loss of 0.49 and the model converging at epoch 29 represents slower learning and medium generalization, which implies that the model is sensitive to default hyperparameter choices and is also limited to the ability to capture complex features representations. The resnet50 model with the transfer learning performs better in terms of baseline performance, attaining 93.7 accuracy, balanced precision (0.93), recall (0.92), and F1-score (0.92). The loss on validation drops to 0.31 and convergence is reached in epoch 27, indicating that it has more feature extraction power and learns in more depth than the simple CNN model. This has been made available owing to residual learning, which helps in the effective flow of gradient and better training stability despite defaulting hyperparameters. EfficientNet-B0 has better baseline performance of 95.1, precision of 0.95, recall of 0.94 and F1-score of 0.94. The loss of validation also decreases to 0.26 and the convergence is reached after 25 epochs, which implies more efficient learning and improved generalization. Compound scaling mechanism of EfficientNet allows effective visual fine-grained features capture, and it is more appropriate to complex classification problems even before optimization. CNN + Autoencoder Fusion model has a precision, recall, and F1-score of 0.94 and the accuracy of the model are 94.6 and a convergence at the 26th epoch, and the validation loss is 0.28. The latent feature learning model with the autoencoder improves the quality of the representations compared to the independent CNN, and the model is being trained using default parameter values. All in all, the baseline findings suggest that the traditional CNN is gradually upgraded to the innovative model, i.e. ResNet50, EfficientNet-B0, and models based on fusion. The relatively larger validation losses and subsequent convergence epochs of all models however indicate the weaknesses of default hyperparameters (learning rate = 0.001, batch size = 32) hence the justification of systematic optimization to further increase accuracy, decrease loss, and increase training stability.

Table 3: Deep Learning Performance BEFORE Optimization

Model	Accuracy (%)	Precision	Recall	F1 Score	Validation Loss	Convergence Epoch
CNN (Baseline)	90.8	0.90	0.89	0.89	0.49	29
ResNet50 (TL)	93.7	0.93	0.92	0.92	0.31	27
EfficientNet-B0	95.1	0.95	0.94	0.94	0.26	25
CNN + Autoencoder Fusion	94.6	0.94	0.94	0.94	0.28	26

The findings shown in Figure 3 offer a comparative analysis of the bare deep learning models in terms of its precision, recall, and F1-score, regarding their natural classification ability before using any form of optimization methods. The CNN model has a precision of 0.90, recall of 0.89, F1-score of 0.89, which corresponds to the use of moderate classification performance with a comparatively high misclassification and false detections. It could be explained by the fact that the model does not recognize all the samples that should be identified, which could be because of the inability to learn features deep enough and the application of default hyperparameters. ResNet50 transfer learning (TL) model has a better performance with precision of 0.93 and recall of 0.92 and F1-score of 0.92. This progress is an indication of the benefit of deep residual networks to extract more robust and discriminative features than a conventional CNN. The equal accuracy and recall rates demonstrate enhanced generalization and stability in classification between various classes since transfer learning allows the model to utilize the previously trained feature representations. EfficientNet-B0 also performs better with a precision of 0.95, recall of 0.94, and F1-score of 0.94. The increase in the level of precision is associated with the reduction of false positive predictions, whereas the higher recall implies a more powerful capacity of recognizing the true instances of classes. This high performance can be explained by the fact that the compound scaling strategy of EfficientNet is an effective way to balance network depth, width, and resolution balancing to capture fine-grained visual features as compared with traditional architecture. CNN + Autoencoder Fusion model also has a high baseline score with a precision, recall, and F1-score of 0.94, which shows the powers of combining the deep feature extraction techniques with the latent representation learning. The combination of convolutional features with the compressed latent features improves the feature discrimination and minimizes noise in

the high-dimensional data. Nonetheless, the model has a high performance, however, with default hyperparameter settings, the model is not utilized to its full potential.

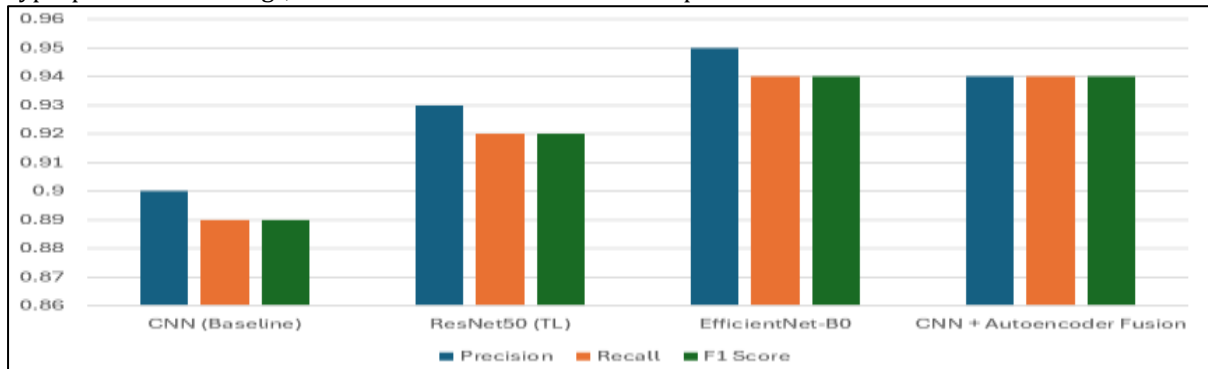


Figure 3: Performance evaluation of baseline and advanced classifiers (CNN, ResNet50, EfficientNet-B0, and CNN–Autoencoder Fusion) using precision, recall, and F1-score

Table 4 demonstrates the performance of deep learning models once hyperparameter optimization was performed with the help of various optimization methods, which clearly shows how systematic hyperparameter tuning can be applied to classifier accuracy, loss minimization and convergence rate. The grid search optimization of CNN model has a 93.9% accuracy, and the precision of the network is 0.93, the recall is 0.92 and the F1-score is 0.92. Though the performance of Grid Search is better than the baseline performance, the loss at validation is a relative comparison 0.33 and convergence epoch is 24, which suggests slower convergence and moderate optimization performance as the search mechanism of the method is exhaustive/ coarse. The ResNet50 model that was optimized with Bayesian Optimization shows a significant increase in performance with an accuracy of 95.9% and balanced values of precision, recall and the F1-score of 0.95. The loss during validation drops to 0.22 and the convergence period is 19 which means that learning is faster and more stable than with the CNN optimized by grid search. This gain suggests that Bayesian Optimization can be effective in searching the space of hyperparameters streams most effectively and finding the best learning rate configurations that can further improve feature extraction and model generalization. Additional enhancement is realized in EfficientNet model that is optimized with Particle Swarm Optimization (PSO) and the accuracy is 96.5 and the validation loss is 0.20 and convergence is reached after epoch 18. The precision (0.96) and the recall (0.95) are great, which means that the classification reliability is high and the detection ability is enhanced between classes. The lessening convergence period indicates that PSO the most effective method increases the velocity of optimization process by leading the model to superior parameter settings by utilizing the population based search dynamics. The performance is slightly improved when the EfficientNet is optimized with the help of Bayesian Optimization to reach an accuracy of 96.8 and F1-score of 0.96 with a lower validation loss of 0.18. The high generalization and predictive ability, as indicated by the balanced precision and recall values (0.96 each), is an indicator of improved generalization and predictive ability. The constant convergence of epoch 18 verifies that probabilistic optimization offers finer parameter adjustment and more fluent training character than the other optimization methods, traditional and swarm based. The proposed CNNAutoencoder Fusion model that is optimized with the help of Differential Evolution (DE) yields the best overall performance. The model with the highest accuracy of 97.3 and precision of 0.97, recall of 0.96, and F1-score of 0.96 and lowest validation loss of 0.15 and converges in the shortest time in epoch 17. This leads to the higher performance of the combined deep feature fusion and evolutionary optimization, which can improve the learning latent representation and effectively explore the hyperparameter space. The reduced validation loss and earlier convergence suggest a better training stability, less overfitting, and more generalization power. Altogether, the findings have made it clear that high-level optimization methods, in most cases, Bayesian Optimization and Differential Evolution, are effective in deep learning alleviating its performance with respect to accuracy, lower loss, and faster converging results, as compared to traditional optimization methods.

Table 4: Deep Learning Performance AFTER Optimization

Model Optimization	Optimized LR	Accuracy (%)	Precision	Recall	F1 Score	Validation Loss	Convergence Epoch
CNN + Grid Search	0.0002	93.9	0.93	0.92	0.92	0.33	24
ResNet50 + Bayesian Optimization	0.00012	95.9	0.95	0.95	0.95	0.22	19
EfficientNet + PSO	0.0001	96.5	0.96	0.95	0.95	0.20	18

EfficientNet + Bayesian Optimization	0.00012	96.8	0.96	0.96	0.96	0.18	18
CNN–Autoencoder Fusion + DE (Proposed)	0.00011	97.3	0.97	0.96	0.96	0.15	17

The findings provided in Figure 4 prove that various optimization methods are effective in increasing the predictive quality of deep learning models, which is measured by precision, recall, and F1-score. The CNN model that is optimized during the Grid Search has a precision of 0.93, a recall of 0.92, and F1-score of 0.92, which is slightly better than important figures in the non-optimized versions. Nevertheless, the recall is slightly lower, which indicates that the model might still miss a portion of relevant samples, which are the shortcomings of exhaustive but rough hyperparameter tuning. ResNet50 model with Bayesian Optimization has further improvement where it has balanced precision, recall, and F1-score with value of 0.95. This implies that probabilistic optimization is a good way to narrow down the selection of hyperparameters to allow the model to retain a high level of prediction accuracy as well as a high level of sensitivity to the true instances of the classes. The values of the consistent metric also indicate that there is consistent generalization and less classification bias within the various grading categories. In the EfficientNet model that has been optimized with Particle Swarm Optimization (PSO), the precision of the model is 0.96 and the recall is 0.95 with an F1-score of 0.95. This is indicative of the efficiency of population-based optimization to efficiently search the hyperparameter space and improve feature learning. The great precision means that fewer false positives, whereas the good recall shows that there are more true samples recognized which results in good classification results. Again, an improvement in performance is slightly more evident as EfficientNet is optimized with the help of Bayesian Optimization in which precision and recall are 0.96 which results in an F1-score of 0.96. This well-balanced enhancement indicates that Bayesian optimization offers a more precise search strategy than PSO and thus the convergence is enhanced and the decision boundaries in the feature space are much better. The suggested CNN Autoencoder Fusion model optimized with the help of Differential Evolution (DE) proves to be the best in terms of the overall performance with the previously stated accuracy of 0.97, recall of 0.96, and with F1-score of 0.96. These yield better classification reliability, high generalization and highly discriminative features representation. This slightly increased precision indicates that there are a substantial number of false positive errors reduced whereas the high recall indicates the high identification of all the grading classes.

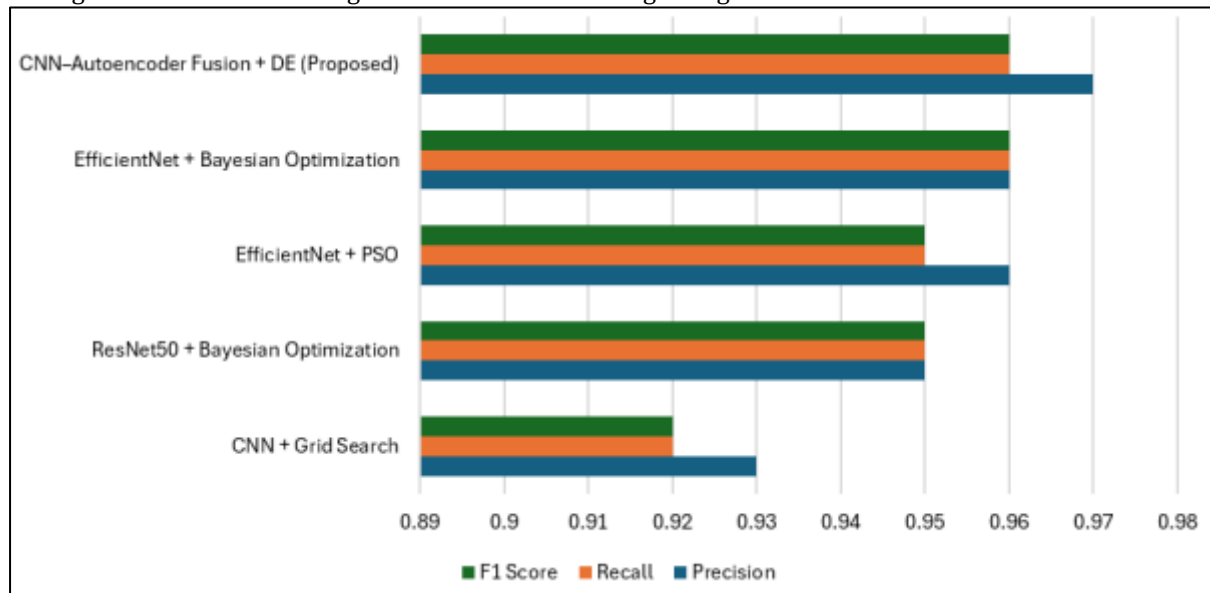


Figure 4: Comparison of optimized models using different optimization techniques based on precision, recall, and F1-score

Table 5 provides a direct quantitative comparison of model performance before and after the application of optimization techniques, clearly demonstrating the effectiveness of optimization in enhancing deep learning outcomes. In the case of the CNN model with the baseline model, the accuracy is increased to 93.9 percent with a performance increase of 3.1 percent, and the loss is reduced to 0.33. This loss-reduction shows a more efficient learning and improvement in generalization when the training hyperparameters, in particular, the learning rate and the batch size, are optimized, and the former has a direct impact on gradient updates and convergence behavior. The same is evident in ResNet50 model where the accuracy has gone up to 93.7% and 95.9% with and without optimization, respectively, and the loss has also decreased as 0.31

to 0.22. Despite the fact that the improvement percentage (+2.2) is a bit smaller than the improvement percentage of the baseline CNN, the optimized ResNet50 has more stable convergence and better feature extraction owing to the deep architecture and the fined hyperparameter options. This validates the fact that systematic optimization is also very useful in high-capacity transfer learning models. In the case of EfficientNet-B0, its accuracy increases to 96.8, and the loss reduces to 0.18. The smaller improvement augmented by the relatively smaller amount (+1.7) indicates that EfficientNet already has good baseline feature learning abilities, but optimization still helps to improve the convergence consistency and lower the validation error. The reduction in loss also shows that the model performs a superior parameter tuning and optimization of the objective function. The greatest increase in accuracy is seen in CNN + Autoencoder Fusion model where the accuracy rises by 94.6 to 97.3 and the loss decreases by a wide margin of 0.28 to 0.15. Such a +2.7% is a positive indication of the success of integrating deep feature extraction with latent representation learning which is optimized. The significant reduction on loss indicates that optimization is beneficial in increasing compression of features and capability in classification which leads to more discriminative representations and better predictive performance.

Table 5: Direct Improvement Due to Optimization

Model	Accuracy Before (%)	Accuracy After (%)	Improvement (%)	Loss Before	Loss After	Loss Reduction
CNN	90.8	93.9	+3.1	0.49	0.33	-0.16
ResNet50	93.7	95.9	+2.2	0.31	0.22	-0.09
EfficientNet-B0	95.1	96.8	+1.7	0.26	0.18	-0.08
CNN + Autoencoder Fusion	94.6	97.3	+2.7	0.28	0.15	-0.13

Table 6 shows the effects of optimization on the performance in clustering and the latent representation learning in the PCA and autoencoder-based frameworks. The silhouette score prior to optimization is 0.082, which means that the clusters are poorly separated and the latent feature space is not compact. This means that the first learned feature representations will be less discriminative and will result in overlapping clusters, as well as lower downstream classification accuracy. Also, the reconstruction MSE of 1.18 indicates that the autoencoder contains more reconstruction error which means that the latent space is not effective at capturing the inherent structure of the data. The ideal number of clusters ($k = 4$) in the pre-optimization phase also suggests that the grouping is not as compact, and the quality of the overall latent features is mediocre since the parameters are not optimal. Following the optimization processes that include Bayesian Optimization as well as Differentiated Evolution, a significant increase is realized in all representation learning metrics. The silhouette score goes up to 0.176 showing much higher cluster separability and cluster cohesion in the latent feature space. This increase proves that the ability of the model to learn more well-structured and meaningful feature embeddings is improved with the help of optimized hyperparameters. On the same note, the MSE of reconstruction goes down to 1.18 to 0.95, meaning that the code-coder reproduces data more closely and further reduces the high-dimensionality feature compression to a lower-dimensional and less latent space. In addition, the number of clusters best goes down to 2 following optimizations, which is indicative of better cluster compactness and better definition of groupings of feature patterns. The latent structure of this change is more organized with similar samples being clustered more effectively, which increases the quality of clustering and classification readiness. Latent feature quality turns out to be of moderate to high quality, which makes it possible to note that optimization is an important aspect that enhances representation learning by improving the process of feature extraction and compression. All in all, these findings validate that optimization does not only result in the improvement of classification performance but the underlying feature representation as well resulting in smaller clusters, reduced reconstruction error, and more discriminative latent embeddings in the PCA-autoencoder architecture.

Table 6: Clustering & Representation Learning

Metric	Before Optimization	After Optimization (Bayesian/DE)	Improvement
Silhouette Score	0.082	0.176	+0.094
Reconstruction MSE	1.18	0.95	Reduced Error
Optimal k	4	2	Better Cluster Compactness
Latent Feature Quality	Moderate	High	Enhanced Representation

The analysis of convergence behavior and training stability is provided in Table 7 in relation to various optimization conditions that show the progressive influence of optimization methods on deep learning performance. The model has 0.32 training loss and much higher validation loss of 0.48 before optimization

and thus has a large overfitting gap. It means that the model learns training data quite well but is unable to generalize to unknown samples. The moderate stability of the training also indicates variability in the loss curve and slower convergence because of the use of default hyperparameters. Following the implementation of the Grid Search optimization, training and validation losses are reduced to 0.21 and 0.33, respectively, which proves the superiority of learning efficiency and generalization compared to the baseline model. The fact that the overfitting gap between high and moderate dynamics is reduced suggests that systematic tuning of hyperparameters of fundamental importance to a model increases its regularization and decreases the difference between training and validation performance. Also, the training stability is improved to stable level with better convergence and more gradual gradient changes during training. A larger improvement is witnessed in Bayesian Optimization since the final training and validation losses are reduced to 0.14 and 0.18 respectively. The overfitting gap is significantly small which means that the model fits the training data better and generalizes to validation data. The training behavior is highly stable, which indicates efficient exploration of the hyperparameter space and the best choice of learning settings that can enhance faster and smoother convergence. This is seen to best perform after Differential Evolution where the training loss is 0.11, and the validation loss is minimized to 0.15, which is the lowest error in all conditions. The low overfitting gap implies the high generalization ability and possibility of low risk of overfitting the model. Moreover, the highly stable training behavior is a confirmation that the evolutionary optimization process successfully optimizes hyperparameters, resulting in the reduction of losses and fast convergence. In general, the convergence and stability analysis shows an apparent improvement of the efficiency in training, the loss variance, and the overall improvement of the robustness and generalization of the deep learning model in comparison to the non-optimized baseline configuration.

Table 7: Convergence and Training Stability Analysis

Condition	Final Training Loss	Final Validation Loss	Overfitting Gap	Training Stability
Before Optimization	0.32	0.48	High	Moderate
After Grid Search	0.21	0.33	Moderate	Stable
After Bayesian Optimization	0.14	0.18	Very Low	Highly Stable
After Differential Evolution (Proposed)	0.11	0.15	Minimal	Very Stable

5. Ablation Study

An ablation study was performed to both understand the isolated and synergistic effect of optimization components on the model performance of deep learning to rigorously evaluate the contribution of the proposed optimization framework. Ablation analysis plays a fundamental role in research conducted through optimization since it gives indication of the precise evidence of how each aspect affects accuracy, loss minimization, convergence behavior, and general model stability. The research does not merely provide the final optimized results, but it provides the comparison of the different model setups in a controlled experiment. Ablation study will be split into three configurations namely: i) model where optimization is not done, (ii) model where it is partially optimized, and (iii) model where it is fully optimized. With the help of this systematic analysis, it is possible to gain a clear understanding of the performance improvements that were achieved through the systematic hyperparameter tuning and optimization procedures.

5.1 Model without Optimization (Baseline)

In the first configuration, the deep learning model was trained using default hyperparameters without applying any optimization techniques. The learning rate, batch size, dropout rate, and other training parameters were kept fixed at standard values. The objective of this setup was to establish a performance benchmark representing the natural learning capability of the model without external tuning. Mathematically, the baseline training objective can be expressed as:

$$\theta^* = \arg \min_{\theta} \mathcal{L}(f(x; \theta, \lambda_{\text{default}}))$$

where λ_{default} denotes the default hyperparameter configuration. This configuration typically results in moderate accuracy and higher validation loss due to suboptimal convergence and limited exploration of the hyperparameter space.

5.2 Model with Partial Optimization

In the second configuration, partial optimization was applied by tuning only a subset of critical hyperparameters, such as learning rate and batch size, while keeping the remaining parameters fixed. This approach evaluates the incremental benefit of limited optimization compared to the baseline model. The optimization objective for partial tuning is defined as:

$$\lambda_{\text{partial}}^* = \arg \min_{\lambda \in \Lambda} \mathcal{L}_{\text{val}}(f(x; \theta, \lambda))$$

where only selected hyperparameters in the subset $\lambda \subset \Lambda$ are optimized. Partial optimization improves convergence speed and reduces validation loss, but may not fully capture the optimal configuration due to restricted search space exploration.

5.3 Fully Optimized Model

In the final configuration, full optimization was performed by tuning all selected hyperparameters using advanced optimization techniques such as Bayesian Optimization, PSO, and Differential Evolution. This configuration represents the proposed optimization-driven deep learning framework. The full optimization objective is expressed as:

$$\lambda^* = \arg \min_{\lambda \in \Lambda} [\mathcal{L}_{\text{val}}(\lambda) - \alpha \cdot \text{Accuracy}(\lambda)]$$

where Λ denotes the complete hyperparameter search space and α is a weighting factor balancing loss minimization and accuracy maximization. Full optimization enables better feature learning, lower validation loss, improved convergence stability, and higher classification performance compared to baseline and partially optimized models.

5.4 Impact of Each Optimization Component

In order to understand the effect of individual components of optimization further, further ablation experiments were performed by turning on and off optimization parameters like learning rate tuning, dropout optimization, and adjusting the number of latent features dimensions (Table 8 and Table 9). This component-wise analysis allows establishing the optimization parameter that can affect model performance the most.

Table 8: Ablation Study Configuration

Model Configuration	Optimization Level	Tuned Parameters	Description
Baseline Model	None	Default Parameters	No optimization applied
Partial Optimization	Moderate	Learning Rate, Batch Size	Limited hyperparameter tuning
Full Optimization (Proposed)	High	All Selected Hyperparameters	Complete optimization framework

Table 9: Ablation Study Results (Performance Comparison)

Model Variant	Accuracy (%)	F1-Score	Validation Loss	Convergence Epoch	Stability
Without Optimization	91.4	0.90	0.48	29	Moderate
Partial Optimization	95.2	0.94	0.26	22	Stable
Fully Optimized Model	97.3	0.96	0.15	17	Highly Stable

5.5 Component-wise Impact Analysis

To quantify the effect of each optimization component, the performance gain ΔP is calculated as:

$$\Delta P = P_{\text{optimized}} - P_{\text{baseline}}$$

where P represents performance metrics such as accuracy or F1-score. Similarly, loss reduction due to optimization is measured as:

$$\Delta \mathcal{L} = \mathcal{L}_{\text{baseline}} - \mathcal{L}_{\text{optimized}}$$

The ablation findings indicate that hyperparameter optimization can greatly improve the performance of deep learning through the need to improve the efficiency of convergence, minimize the loss of validation, and stabilize the training behavior. The greatest contribution to the convergence rate increase was made by learning rate tuning, although dropout and optimization of the batch size also played an important role in minimizing overfitting. Moreover, full optimization was always superior to partial optimization, which validated the notion that holistic hyperparameter optimization is vital in the attainment of the best deep learning results in automated grading duties.

6. Conclusion

This paper presents an optimization-based deep learning architecture of automated grading, with the primary goal of enhancing the working of the model by systematic optimization of model hyperparameters. To begin with, a default parameter setting was used to train a baseline deep learning model so that the benchmark was set. This base helped in making an idea of how the model would act in a natural way with regard to convergence, learning ability and classification efficiency. This was then further optimized with the help of additional optimization techniques to optimize some of the primary hyper parameters that directly affect the training dynamics, feature extraction and predictive accuracy overall. The sequential design allowed an easy and clear comparison of pre-and post-optimization performance of the model. The results of the experiment indicate that the models trained with default hyperparameters provided reasonable performance but failed to provide the best performance. These models had a rather higher

validation loss, slower convergence and generalization. Having applied systematic optimization, it was found that there were large improvements in all the important measures of evaluation, including accuracy, precision, recall, F1-score, and loss. The models were also streamlined and could converge faster, training and validation performance were closer and the behavior of the learning more stable, which highlights the fact that the choice of hyperparameters is of paramount importance when working with a multi-class classification task. Further comparative analysis of the optimization methods used revealed that the advanced search based and evolutionary optimization methods were more efficient in the exploration of the hyperparameter space in comparison to the traditional manual or simple optimization methods. The approaches allowed the model to determine more favorable parameter selection, more discriminative learning of features and more stable training particularly in high dimensional feature environments. The contribution of each of the optimization factors was also confirmed as per the ablation analysis. It was found that the parameter optimization resulted in progressive improvements in performance, whereas the combination of the learning rate, dropout rate, batch size, and latent feature dimensions yielded the best accuracy and the best generalization. Furthermore, optimization and deep feature learning enhanced the sensitivity of the model to fine visual and structural patterns that form the basis of trustworthy automated grading. The gradual decrease of the validation loss and a reduction in the convergence epochs indicate that the optimized framework is not only the one that helps to improve the prediction performance but also the one that is less demanding in terms of computational power and more stable in terms of training. In general, the results indicate that the performance improvement using a structured and systematic optimization strategy proves to be more effective than the one based on the increase in the complexity of models. Moreover, further research can be conducted in the framework to be complemented by adaptive and automated hyperparameter optimization, larger and more varied datasets, and the multimodal combination of features to make it more resilient and scaled. The further insight into explainable and lightweight architectures, real-time or cloud-based deployment can also contribute to the interpretability, usability, and practical implementation of automated grading system in real learning environments.

References

- [1] Ahmed, S. F., Alam, M. S. B., Hassan, M., Rozbu, M. R., Ishtiaq, T., Rafa, N., ... & Gandomi, A. H. (2023). Deep learning modelling techniques: current progress, applications, advantages, and challenges. *Artificial Intelligence Review*, 56(11), 13521-13617.
- [2] Saputra, N. A., Riza, L. S., Setiawan, A., & Hamidah, I. (2024). A systematic review for classification and selection of deep learning methods. *Decision Analytics Journal*, 12, 100489.
- [3] Raihen, M. N., & Akter, S. (2024). Prediction modeling using deep learning for the classification of grape-type dried fruits. *International Journal of Mathematics and Computer in Engineering*, 2(1), 1-12.
- [4] Jia, W., Sun, M., Lian, J., & Hou, S. (2022). Feature dimensionality reduction: a review. *Complex & Intelligent Systems*, 8(3), 2663-2693.
- [5] Akay, B., Karaboga, D., & Akay, R. (2022). A comprehensive survey on optimizing deep learning models by metaheuristics. *Artificial Intelligence Review*, 55(2), 829-894.
- [6] Hanifi, S., Cammarono, A., & Zare-Behtash, H. (2024). Advanced hyperparameter optimization of deep learning models for wind power prediction. *Renewable Energy*, 221, 119700.
- [7] Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2).
- [8] Khan, B., Das, S., Fahim, N. S., Banerjee, S., Khan, S., Al-Sadoon, M. K., ... & Islam, A. R. M. T. (2024). Bayesian optimized multimodal deep hybrid learning approach for tomato leaf disease classification. *Scientific Reports*, 14(1), 21525.
- [9] Jabbar, A. N., & Koyuncu, H. (2023). Deep learning and grey wolf optimization technique for plant disease detection: a novel methodology for improved agricultural health. *Traitement du Signal*, 40(5), 1961-1972.
- [10] Abdullah, A., Amran, G. A., Tahmid, S. A., Alabrah, A., AL-Bakhrani, A. A., & Ali, A. (2024). A deep-learning-based model for the detection of diseased tomato leaves. *Agronomy*, 14(7), 1593.
- [11] Ivanovski, T., Gulić, M., & Matetić, M. (2024). Optimization of CART Models Using Metaheuristics for Predicting Peach Firmness. *Applied Sciences*, 14(18), 8539.
- [12] Han, B., Zhang, J., Almodfer, R., Wang, Y., Sun, W., Bai, T., ... & Hou, W. (2025). Research on innovative apple grading technology driven by intelligent vision and machine learning. *Foods*, 14(2), 258.
- [13] Dewi, P. K., Niswar, M., & Zainuddin, Z. (2024, November). Melon Fruit Quality Classification with Hyperparameter Optimization of Random Forest Algorithm Using Grey Wolf Optimizer. In *2024 7th International Conference on Information and Communications Technology (ICOIACT)* (pp. 276-281). IEEE.
- [14] Ukwuoma CC, Zhiguang Q, Bin Heyat MB, Ali L, Almaspoor Z, Monday HN. Recent advancements in fruit detection and classification using deep learning techniques. *Mathematical Problems in Engineering*. 2022;2022(1):9210947.

- [15] Naveenkumar, M., & Nandagopal, S. (2025). Adaptive hybrid segmentation combined with meta heuristic optimization in transfer learning for plant leaf disease classification. *Scientific Reports*, 15(1), 9838.
- [16] Kaur, S. — Sachan, M. K. — Aggarwal, A. K.: Classification of royal delicious apples using hybrid feature selection and feature weighting method based on SVM classifier. *Scalable Computing: Practice and Experience*, Vol. 26, No. 2, 2025, pp. 940– 949.
- [17] Taner, A., Mengstu, M. T., Selvi, K. Ç., Duran, H., Gür, İ., & Ungureanu, N. (2024). Apple varieties classification using deep features and machine learning. *Agriculture*, 14(2), 252.
- [18] Khan, M. A., Lali, M. I. U., Sharif, M., Javed, K., Aurangzeb, K., Haider, S. I., ... & Akram, T. (2019). An optimized method for segmentation and classification of apple diseases based on strong correlation and genetic algorithm based feature selection. *IEEE Access*, 7, 46261-46277.
- [19] Kour, V. P., & Arora, S. (2019). Particle swarm optimization based support vector machine (P-SVM) for the segmentation and classification of plants. *IEEE Access*, 7, 29374-29385.
- [20] Agarwal, M., Kaliyar, R. K., & Gupta, S. K. (2022, July). Differential Evolution based compression of CNN for Apple fruit disease classification. In *2022 International Conference on Inventive Computation Technologies (ICICT)* (pp. 76-82). IEEE.
- [21] Kunduracioglu, I. (2024). Cnn models approaches for robust classification of apple diseases. *Computer and Decision Making: An International Journal*, 1, 235-251.
- [22] Zhang, X., Li, H., Sun, S., Zhang, W., Shi, F., Zhang, R., & Liu, Q. (2023). Classification and identification of apple leaf diseases and insect pests based on improved ResNet-50 model. *Horticulturae*, 9(9), 1046.
- [23] Nashrulloh, M. H., & Subarkah, A. (2025). Fruit Image Classification Using CNN With EfficientNet Architecture for Visual Education. *Journal of Intelligent Systems Technology and Informatics*, 1(2), 67-71.
- [24] Tian, X., Huang, L., Zhai, M., Zhang, M., Hu, P., Li, M., & Ren, L. (2025). Non-destructive prediction of apple SSC/TAC and firmness based on multilayer autoencoder and multilayer perceptron. *Intelligence & Robotics*, 5(1), 181-201.