**Evaluating Fruit Quality Using Deep Learning**

*Report submitted to the SASTRA Deemed to be University as the requirement for the course*

**CSE300 / INT300 - MINI PROJECT**

*Submitted by*

**SHIVANAND RACHA**

**(Reg. No.: 126015083, B. Tech Information Technology)**

**PYLA PHANEENDRA SRI VENKAT BOSU**

**(Reg. No.: 126015082, B. Tech Information Technology)**

**YERRAMSETTY LAUKHIK ANIRUDH**

**(Reg No.: 126015125, B. Tech Information Technology)**

**May 2025**

# 

**SCHOOL OF COMPUTING**

**THANJAVUR, TAMIL NADU, INDIA – 613 401**

**i**

# 

**SCHOOL OF COMPUTING**

**THANJAVUR– 613 401**

**Bonafide Certificate**

This is to certify that the report titled “**Evaluating Fruit Quality Using Deep Learning**” submitted as a requirement for the course, **CSE300 : MINI PROJECT** for B.Tech. is a bonafide record of the work done by **Mr. SHIVANAND RACHA** (Reg. No. **126015083**, B. Tech Information Technology), **Mr. PYLA PHANEENDRA SRI VENKAT BOSU** (Reg. No. **126015082**, B. Tech Information Technology) and **Mr. YERRAMSETTY LAUKHIK ANIRUDH** (Reg.No.**126015125**, B. Tech Information Technology) during the academic year 2024-25, in the School of Computing, under my supervision.



**Signature of Project Supervisor :**

**Name with Affiliation :** Dr.Nithya R, AP-III , SOC

**Date :** 09.05.2025

Mini Project *Viva voice* held on

**Examiner 1 Examiner 2**

**ii**

**Acknowledgements**

We would like to thank our Honorable Chancellor **Prof. R. Sethuraman** for providing us with an opportunity and the necessary infrastructure for carrying out this project as a part of our curriculum.

We would like to thank our Honorable Vice-Chancellor **Dr.S. Vaidhyasubramaniam** and **Dr. S. Swaminathan**, Dean, Planning & Development, for the encouragement and strategic support at every step of our college life.

We extend our sincere thanks to Dr. R. Chandramouli, Registrar, SASTRA Deemed to be University for providing the opportunity to pursue this project.

We extend our heartfelt thanks to **Dr. V. S. Shankar Sriram**, Dean, School of Computing, **Dr. R. Muthaiah**, Associate Dean, Research, **Dr. K.Ramkumar**, Associate Dean, Academics, **Dr. D. Manivannan**, Associate Dean, Infrastructure, **Dr. R.Algeswaran**, Associate Dean, Students Welfare.

Our guide **Dr.Nithya R,Assistant Professor**, School of Computing was the driving force behind this whole idea from the start. Her deep insight in the field and invaluable suggestions helped me/us in making progress throughout our project work. We also thank the project review panel members for their valuable comments and insights which made this project better.

We would like to extend our gratitude to all the teaching and non-teaching faculties of the School of Computing who have either directly or indirectly helped us in the completion of the project. We gratefully acknowledge all the contributions and encouragement from my family and friends resulting in the successful completion of this project. We thank you all for providing us an opportunity to showcase our skills through the project.

**iii**

**LIST OF FIGURES**

|  |  |
| --- | --- |
| **Figures** | **PageNo** |
| fig-1: DataSet Images | 5 |
| fig-2: System Architecture | 5 |
| fig 3: EDA | 7 |
| fig 4: Feature Relationship with Target Class | 8 |
| fig 5: Outlier Detection | 9 |
| fig 6 :Data Preprocessing | 10 |
| fig 7: MobileNet-V2 | 13 |
| fig 8: ResNet50 | 15 |
| fig 9: EfficientNetB0 | 17 |
| fig 10: SqueezeNet | 19 |
| fig 11: ShuffleNet | 21 |
| fig 12: EDA Analysis | 39 |
| fig 13: HSV Analysis | 39 |
| fig 14: Classification Report of MobileNetV2 | 40 |
| fig 15: Classification Reports of Various Models | 41 |
| fig 16: Confusion Matrices of Models | 42 |

**LIST OF TABLES**

|  |  |
| --- | --- |
| **Tables** | **PageNo** |
| Table-1: Comparision Table | 43 |

**iv**

**Abbreviations**

|  |  |
| --- | --- |
| **CNN** | Convolutional Neural Network |
| **HSV** | Hue Saturation Value |
| **RGB** | Red Green Blue |
| **AI** | Artificial Intelligence |
| **SGD** | Stochastic Gradient Descent |
| **ReLU** | Rectified Linear Unit |
| **PCA** | Principal Component Analysis |
| **GPU** | Graphics Processing Unit |
| **API** | Application Programming Interface |

**v**

**Abstract**

The use of artificial intelligence techniques in agriculture is ongoing, with more research interest in areas that improve harvesting and post-harvesting processes. Fruit quality recognition is important to farmers while sorting, to retailers in monitoring quality, and to consumers determining freshness. As much as there are datasets that can support this research, practical and real-time solutions are still lacking. This research makes use of an existing multi-fruit image data set and picks three of the chosen fruit classes for testing freshness. Five renowned deep learning architectures—ShuffleNet, SqueezeNet, EfficientNet B0, ResNet-50, and MobileNetV2 were used to compare performance on fruit quality identification. Of these five models, MobileNetV2 proved to have the best classification accuracy for this purpose, showing its suitability for performing this task. The dataset is well-structured and annotated, which makes it ideal for testing both mature and new machine learning models. This research adds to better methods in fruit classification and freshness assessment, providing useful insights to the computer vision and machine learning communities. Although promising, there are still limitations like dataset size, which indicate areas for future improvement in deep learning for agricultural quality assessment.

**KEY WORDS:** *Fruit Quality Recognition, Deep Learning, MobileNetV2, ResNet-50, EfficientNet B0, SqeezeNet, ShuffleNet, Computer Vision, Freshness Detection, Image Classification, Agriculture.*

**vi**

**Table of Contents**

|  |  |
| --- | --- |
| **Title** | **Page No.** |
| Cover Page | i |
| Bonafide Certificate | ii |
| Acknowledgements | iii |
| List of Tables | iv |
| List of Figures | iv |
| Abbreviations | v |
| Abstraction | vi |
| 1 Summary of the base paper | 1 |
| 1.1 Introduction | 1 |
| 1.2 Related Work | 2 |
| 1.3 Problem Statement | 3 |
| 1.4 Objective | 3 |
| 1.5 Proposed Solution and System Architecture | 4 |
| 1.5.1 Study Plan | 4 |
| 1.5.2 Fruit Quality Dataset | 4 |
| 1.6 Methodology and Implementation | 6 |
| 1 EDA and Structural Composition | 6 |
| 2 Model Development | 11 |
| 3 Model Evaluation  **vii** | 21 |
| 2 Merits and Demerits | 22 |
| 3 Source Code | 24 |
| 4 Snapshots | 39 |
| 5 Conclusion and Future Plans | 44 |
| 6 References | 46 |
| 7 Appendix -Base Paper | 47 |

# 

# 

# 

# 

# 

# 

**viii**

# **CHAPTER 1**

# **SUMMARY OF BASE PAPER**

**Title :** FruitQ: a new dataset of multiple fruit images for freshness

evaluation

**Journal Name :** Multimedia Tools and Applications

**Publisher :**  Springer

**Year :** 2024

**Indexed In :** Scopus, SCIE

**Base paper URL :** <https://doi.org/10.1007/s11042-023-16058-6>

**1.1 INTRODUCTION**

Fruit quality recognition has become a crucial aspect of contemporary agriculture, directly impacting choices about post-harvest handling, customer satisfaction, and quality control. Fruit freshness assessment has historically depended on laboratory analysis or manual inspection, both of which are efficient but time-consuming, subjective, and impractical for large-scale or real-time deployment. Automated, image-based fruit quality evaluation is becoming more and more feasible due to the quick development of artificial intelligence (AI), especially deep learning. For example, Convolutional Neural Networks (CNNs) have proven to be highly effective at tasks involving the recognition and classification of visual patterns. However, the lack of diverse, annotated datasets and the high processing demands of many state-of-the-art models make real-time freshness detection difficult.

This study investigates effective and portable deep learning architectures for fruit freshness classification using image data. We make use of a publicly accessible dataset that includes samples from three different fruit categories at different freshness stages. The effectiveness and suitability of a number of CNN models, including ResNet-50, MobileNetV2, EfficientNet B0, ShuffleNet, and SqueezeNet, were assessed in agricultural settings with limited resources. In order to improve model performance, an exploratory data analysis was also carried out to comprehend the composition of the dataset, identify feature relationships, and find any potential anomalies. The results show how deep learning can be used to create scalable and affordable real-time fruit quality recognition systems, which will improve precision farming and cut down on food waste.

**1.2. RELATED WORK**

Recent advancements in deep learning and machine learning have enabled significant improvements in soil classification tasks using image-based data. Various researchers have explored models ranging from traditional classifiers to complex neural network architectures to enhance accuracy and efficiency.

* **"A General Machine Learning Model for Assessing Fruit Quality Using Deep Image Features "** by *Ioannis D. Apostolopoulos, Mpesi Tzami, Sokratis I. Aznaouridis (2023)* proposed a fruit quality classification system using Vision Transformers (ViT). The model achieved an impressive accuracy of 98.8% , demonstrating the effectiveness of transformer-based architectures for extracting rich visual features in fruit assessment tasks.
* **"Deep Learning-Based Fruit Quality Detection for Sustainable Agriculture "** by *Víctor Zárate, Danilo Cáceres Hernández (2024)* developed a CNN-based fruit quality classification framework focused on sustainability. Utilizing Convolutional Neural Networks (CNNs), the model reached an accuracy of 98.2% , highlighting the power of deep CNNs in practical agricultural applications.
* **"Simplified Deep Learning for Accessible Fruit Quality Assessment in Small Agricultural Operations** " by *Víctor Zárate, Danilo Cáceres Hernández (2024)* introduced a hybrid model combining a CNN trained from scratch with a fine-tuned MobileNetV2. The system achieved an accuracy of 97.5% , emphasizing accessibility and lightweight architectures for use in resource-constrained environments.
* **"** **Hybrid Deep Learning Model for Fruit Classification and Quality Prediction Using CNNs and OpenCV"** by *P. Sumari, W\.M.A. Wan Ahmad, F. Hadi, M. Mazlan, N.A. Liyana, R.-W. Bello, A.S.A. Mohamed, A.Z. Talib (2025)* proposed a hybrid architecture that integrates traditional image processing via OpenCV with an optimized CNN model. The system achieved an accuracy of 94.99% , illustrating a balance between conventional and modern deep learning tools in agricultural analysis.
* **"** **Fruit-CNN: An Efficient Deep Learning-Based Fruit Classification and Quality Assessment"** by *Arnav Kumar, Rakesh Chandra Joshi, Malay Kishore Dutta, Martin Jonak, Radim Burget (2021)* presented an optimized CNN model achieving 98.3% accuracy, showing high efficiency and accuracy for both classification and quality evaluation of fruits.
* **"** **Fruit Quality Assessment with Densely Connected Convolutional Neural Network (DenseNet)"** by *K* *Md. Samin Morshed, Sabbir Ahmed, Tasnim Ahmed, Muhammad Usama Islam, A.B.M. Ashikur Rahman (2022)* used DenseNet, achieving 98.67% accuracy , and demonstrated how dense connections enhance feature propagation and mitigate gradient vanishing issues in deep models.
* **"** **Fruit Quality Evaluation Using Machine Learning Techniques: Review, Motivation, and Future Perspectives"** by *Bhumica Dhiman, Yogesh Kumar, Munish Kumar (2022)* evaluated models such as Random Forest, Gradient Boosting, and CNNs, with the highest accuracy reaching 95.67%, showing traditional ML still holds merit in fruit quality tasks.
* **"** **Automated Fruit Quality Recognition Using YOLOv3 and Deep Learning"** by *M. Karthikeyan, T.S. Subashini, R. Srinivasan, C. Santhanakrishnan, A. Ahilan (2023)* developed YOLOAPPLE, combining YOLOv3 with Darknet53 and Spatial Pyramid Pooling, achieving 97.3% accuracy and showcasing potential for real-time fruit quality assessment systems.

**1.3. PROBLEM STATEMENT**

Assessing fruit freshness by hand is frequently time-consuming, unreliable, and impractical for widespread use. Even though automation is made possible by deep learning, current models are usually complicated and dependent on sizable, meticulously annotated datasets—conditions that are rarely satisfied in practical situations. Furthermore, a lot of methods have trouble categorizing different kinds of fruit under different circumstances. This project suggests a lightweight deep learning model that can effectively evaluate fruit freshness in real time using image data, even when there are sparse or unbalanced datasets, in order to overcome these difficulties. With the help of anomaly detection and thorough dataset analysis, the model prioritizes effectiveness, resilience, and suitability for deployment on low-powered devices.

**1.4 OBJECTIVE**

Using a carefully selected multi-fruit dataset of three classes, we hope to create an effective, real-time fruit freshness assessment system by utilizing contemporary deep learning architectures. Our method aims to reduce inference time and resource usage without compromising performance by striking a balance between classification accuracy and computational and memory efficiency. We will compare ShuffleNet, SqueezeNet, EfficientNet B0, ResNet-50, and MobileNetV2 on the chosen dataset in order to validate the framework. We will evaluate things like accuracy, inference latency, and resilience to anomalous or corrupted samples. Our ultimate objective is to present a scalable, lightweight on-device fruit freshness detection system that can be used in a variety of agricultural contexts.

**1.5 PROPOSED SOLUTION AND SYSTEM ARCHITECTURE**

**1.5.1 STUDY AREA**

This study explores the application of deep learning techniques for fruit quality recognition in order to address a significant issue in the retail, consumer, and agricultural sectors. A significant part of conventional techniques for assessing fruit quality involves manual inspection, which is time-consuming and prone to mistakes. Furthermore, scalable, real-time quality recognition solutions are still lacking in many agricultural settings. Deep learning models are challenging to successfully implement in real-world scenarios due to the size and scope of the datasets currently available for fruit quality analysis being frequently limited.

In order to address these issues, this study makes use of a well-structured and annotated dataset of multi-fruit images that can be used to assess the freshness of various fruit varieties. The ability of five well-known deep learning models—ShuffleNet, SqueezeNet, EfficientNet B0, ResNet-50, and MobileNetV2 to categorize fruit freshness using visual characteristics is evaluated. The study looks at how well these models perform on specialized, small datasets, which are common in agricultural settings. According to the results, MobileNetV2 performs better than the other models, attaining the highest classification accuracy while preserving efficiency. This illustrates its potential for real-time monitoring of agricultural quality applications. The study shows that customized, lightweight deep learning models can be accurate and fast, which makes them perfect for environments with limited resources.

**1.5.2 FRUIT QUALITY DATASET**

Images from six fruit categories, including both fresh and rotten oranges, bananas, and apples, make up the dataset used in this study. With unique visual traits that are essential for determining fruit quality, each category in the dataset is intended to aid in the classification of fruit freshness. The following classes are covered: Rotten Apples, Rotten Bananas, Rotten Oranges, Fresh Apples, Fresh Bananas, and Fresh Oranges.

The dataset is well-balanced and includes representative samples from all six classes. The training set consists of 8,720 images, the validation set contains 2,181 images, and the test set has 2,698 images. The total dataset comprises 13,599 images, ensuring that the model has access to a diverserange of samples. This diversity helps the models learn the essential features of both fresh and rotten fruit, ensuring reliable classification. The well-annotated and structured dataset is ideal for training and evaluating deep learning models for real-time applications in agricultural and retail settings, contributing to more accurate fruit quality assessment systems.

*fig-1: DataSet**Images*

**Study Framework Diagram:**

The study framework for fruit quality recognition begins with an in-depth exploratory analysis of the dataset, focusing on its structural composition, visualization of relationships between image features and the target class, and statistical analysis of image features. The dataset is divided into training and test sets, with preprocessing steps that include resizing images to standard dimensions, normalization, noise reduction, and RGB/HSV decomposition to extract critical texture and brightness features, addressing lighting variations.

*fig-2* *System Architecture*

The study also includes the analysis of feature distributions to detect anomalies and outliers. To enhance model generalization, data augmentation techniques such as flipping, rotation, zooming, and brightness adjustments are applied. These techniques help improve the robustness of the model and reduce overfitting, especially given the variability in image conditions and limited dataset size.

The system leverages five lightweight CNN models (ShuffleNet, SqueezeNet, EfficientNet-B0, ResNet-50, and MobileNetV2), with test images classified by the trained models. The models' effectiveness is evaluated using metrics such as accuracy, precision, recall, and F1-score. The framework is designed for efficiency and scalability, ensuring robust fruit quality classification in resource-constrained environments.

**1.6 METHODOLOGY AND IMPLEMENTATION**

The methodology adopted in this project is focused on thoroughly analyzing the dataset characteristics and preparing the data for robust classification of fruit quality using image-based inputs. The process integrates exploratory data analysis, statistical examination of image features, anomaly and outlier detection, and data augmentation to improve the efficiency and generalization of deep learning models.

**1.6.1 Exploratory Data Analysis and Structural Composition**

**Purpose:**

The initial step was to understand the overall structure and contents of the dataset. This involved:

**Process:**

* **Image Distribution Analysis** : Visualizations such as bar charts were used to assess the number of samples available for each fruit quality class. This helped identify any class imbalances that might affect model performance.
* **Sample Visualization** : Randomly selected images from each class were visually inspected to ensure label correctness, image quality, and consistency across classes.
* **Structural Analysis** : Image dimensions, color channel information, and label encoding were reviewed to detect any inconsistencies or preprocessing needs.

*fig-3 EDA*

**1.6.2 Visualization of Relationships Between Image Features and Target Classes**

To better understand how image characteristics influence the quality labels, various relationships were analyzed:

* **Channel-wise Pixel Intensity Distribution** : The average and variance of pixel values across the Red, Green, and Blue channels were compared across classes to highlight discriminative features.
* **Brightness and Contrast Analysis** : Histogram-based visualizations of brightness and contrast levels were created to show how fruit degradation (such as bruising or discoloration) affects image luminance.
* **Class-wise Feature Distributions** : Visualizations of feature distributions provided insight into how texture and lighting vary between fresh and spoiled fruit samples.

*fig-4 Feature Relationship with Target Class*

**1.6.3 Statistical Analysis of Image Features**

Quantitative metrics were used to describe and compare the visual content of images across classes:

Mean, Standard Deviation, and Skewness of pixel intensities were computed for each class. These statistics helped in identifying patterns such as darker images in spoiled classes or highly saturated images in fresh classes.

These insights helped in feature engineering and informed decisions regarding preprocessing and normalization steps prior to training.

**1.6.4 Anamoly and Outlier Detection**

To improve dataset quality and minimize noise, both automated and manual outlier detection strategies were applied:

**Z-score and Distribution-Based Methods** : Used to flag images with extreme pixel intensity values that deviate significantly from the class norm.

**Manual Review** : Images identified as outliers were visually inspected and removed if they were blank, overly dark, corrupted, or mislabeled.

This ensured a clean, consistent dataset, which is crucial for training accurate deep learning models.

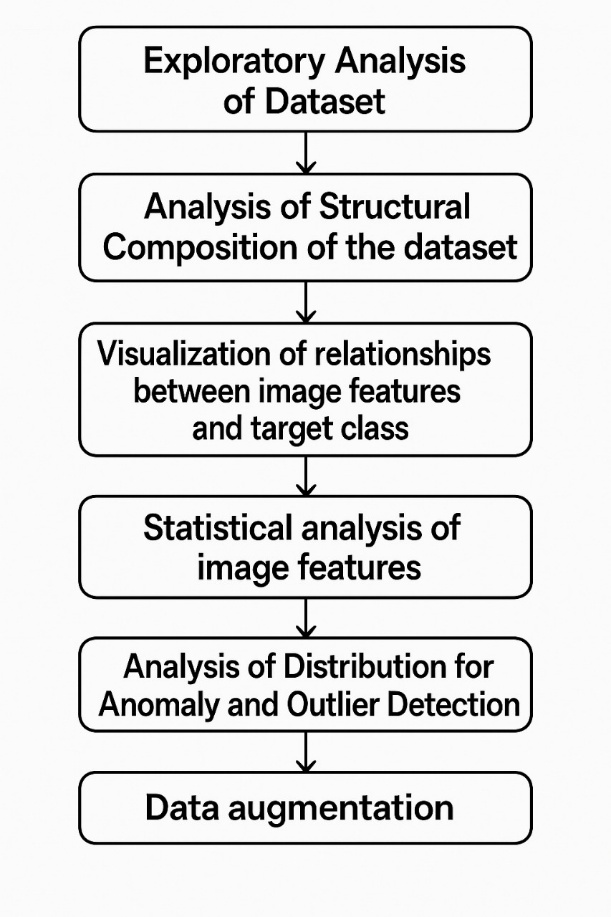
*fig-5 Outlier Detection*

**1.6.5 Data Augmentation**

To enhance the generalization capability of the classification model and simulate real-world variability, data augmentation techniques were applied using TensorFlow’s ImageDataGenerator.

The following transformations were employed:

* **Geometric Transformations** : Including rotation, width/height shifts, shear, and zoom, to simulate natural movement and viewpoint variation.
* **Brightness Variation** : Random changes in brightness to account for different lighting conditions.
* **Horizontal Flipping** : Introduced variation in fruit orientation.



*fig-6: Data preprocessing*

**Module 2: Model Development**

**2.7.1 Model Training and Evaluation**

In this phase, the prepared fruit freshness classification dataset—comprising images from three selected fruit classes—was used to train multiple deep learning models. The dataset was split in an 80:20 ratio for training and testing. Five CNN architectures, namely ShuffleNet, SqueezeNet, EfficientNetB0, ResNet-50, and MobileNetV2, were selected based on their efficiency and proven performance in image classification tasks. Image preprocessing techniques such as resizing and normalization were applied, along with augmentation strategies including flipping, rotation, and zooming to improve generalization. Each model was trained using the Adam optimizer and categorical cross-entropy loss, with hyperparameters like learning rate, batch size, and number of epochs optimized through iterative experimentation.



During the evaluation phase, the trained models were assessed using unseen test data to determine their classification performance. Metrics such as accuracy, precision, recall, and F1-score were used to analyze model effectiveness. Confusion matrices were also generated to visualize classification results and identify patterns of misclassification. Cross-validation techniques were incorporated to verify model generalizability and avoid overfitting. Among all the models, MobileNetV2 demonstrated superior performance in terms of both accuracy and computational efficiency, making it suitable for deployment in real-time and resource-constrained environments.

##### **MobileNet-V2 model**

An effective deep learning architecture called MobileNetV2 was created for use with mobile and edge devices, which have constrained processing power. Inverted residuals with linear bottlenecks are incorporated into this architecture, which allows the model to perform well at a low computational cost. Using depthwise separable convolutions, which reduce the complexity of conventional convolutional operations by convolving each input channel independently with its own filter, is the fundamental idea behind MobileNetV2.

The model is trained for 30 epochs and runs with predefined hyperparameters, including a batch size of 32 and a learning rate of 0.0001. MobileNetV2, which is especially well-suited for mobile applications, introduces non-linearity by using ReLU6 as the activation function. Additionally, it employs batch normalization following convolutions to normalize the activations and stabilize training.

Compared to conventional convolutional layers, the model can drastically cut down on the number of operations by utilizing depthwise separable convolutions. This method consists of filtering each input channel separately using a different set of filters, then combining the results using a pointwise convolution. Because of this, MobileNetV2 is especially effective for mobile devices without sacrificing accuracy because it uses fewer parameters and a smaller model size.

The sparse categorical cross-entropy loss function is used to handle multi-class classification tasks, and the model is optimized during training using the Adam optimizer with a learning rate of 1e-4. In order to track performance and preserve the optimal model, the model additionally incorporates checkpointing, learning rate scheduling, and early stopping.

The output of the model is a probability distribution over the classes, obtained through a softmax activation in the final layer. The predicted class is then chosen based on the highest probability, providing the final prediction.

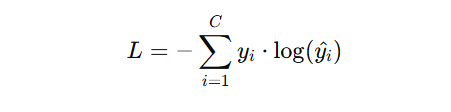
The final prediction is obtained by applying the softmax function to the output of the model, followed by selecting the class with the highest predicted probability.

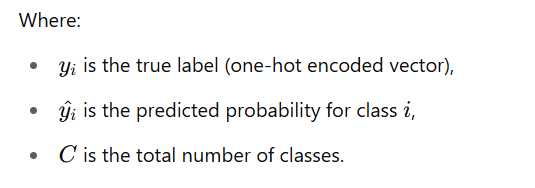
𝑀𝑜𝑏𝑖𝑙𝑒𝑛𝑒𝑡 𝑉2 𝑝𝑟𝑒𝑑𝑖𝑐𝑡𝑖𝑜𝑛 = 𝑎𝑟𝑔𝑚𝑎𝑥(𝑠𝑜𝑓𝑡𝑚𝑎𝑥(ℎ(𝑥)))

Where:

* h(x) is the output from the model,
* softmax converts the output logits into probabilities
* argmax selects the class with the highest probability.

During training, the model's loss function is computed using categorical cross-entropy for multi-class classification. The formula is:

****

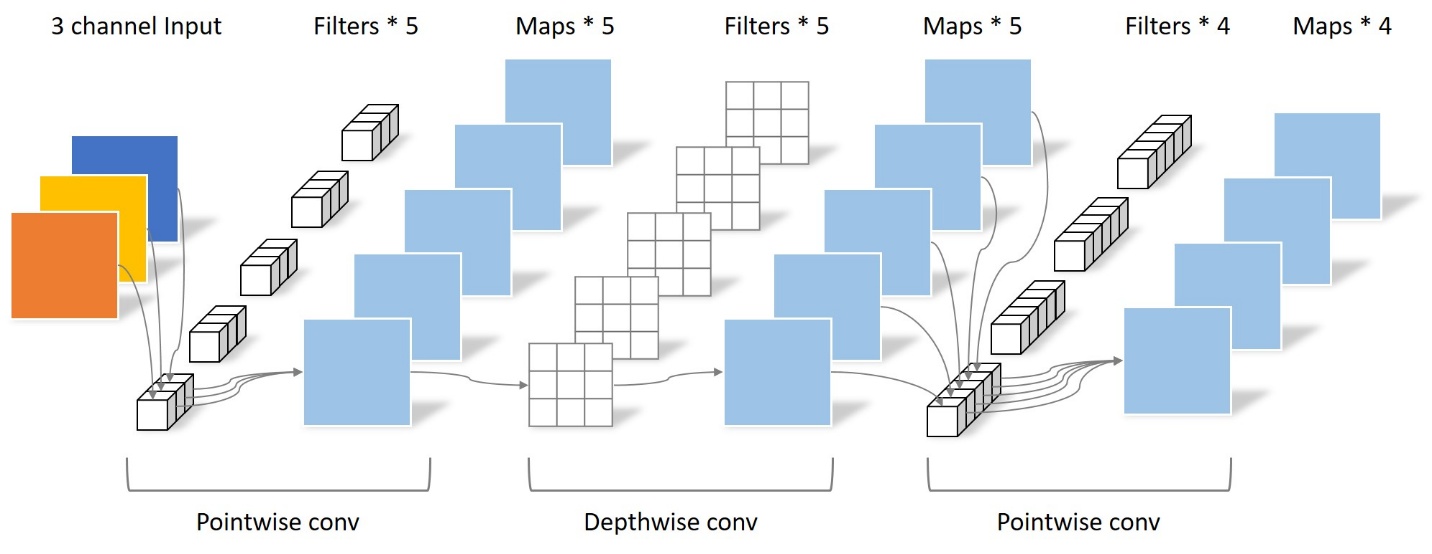


This loss function is minimized using an Adam optimizer or SGD with momentum, allowing the model to learn the best weights over the course of training.

The training process involves distributing the input data to the model using data generators for efficient handling of large datasets. The training employs class weights to address any potential class imbalances. Regularization methods, such as L2 regularization and Dropout, are also incorporated into the model's dense layers to prevent overfitting.

By leveraging MobileNetV2's lightweight structure and depth wise separable convolutions, this model is optimized for fast inference, making it ideal for use in applications requiring real-time predictions, especially on mobile devices.

After loading the pre-trained model with ImageNet weights, fine-tuning was performed on the target dataset. This involved unfreezing some of the top convolutional layers and retraining them with a low learning rate to adapt the pre-trained features to the specific classification task. Fine-tuning helped improve the model’s ability to generalize to domain-specific features while leveraging knowledge.



*fig-7: MobileNet-V2*

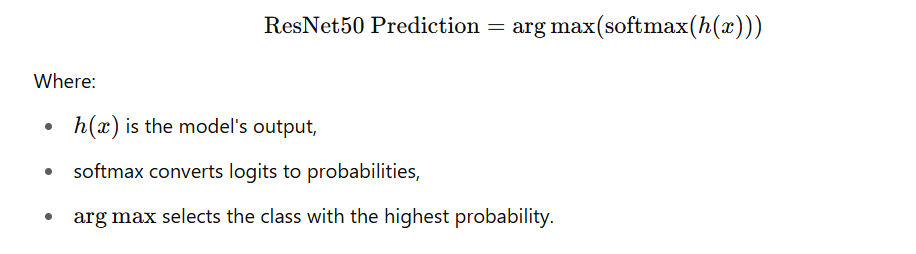
**ResNet50 Algorithm**

Handling class imbalance, maximizing learning, and avoiding overfitting are the main goals of the ResNet-50 model's training. Class weights are determined using the training dataset's class distribution in order to rectify class imbalance. In order to prevent bias towards overrepresented classes, this guarantees that underrepresented classes are given greater weight during training, enabling the model to concentrate more on these classes. In order to make sure the model's loss function adapts to impose harsher penalties for incorrectly classifying minority classes, these class weights are subsequently transferred to the model during training.

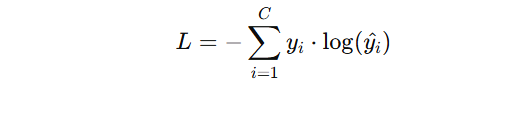
Early stopping monitors the validation accuracy and stops training if it doesn't improve after a certain number of epochs (10 in this case). This helps to prevent unnecessary training and ensures the model doesn't overfit by training for too many epochs. When early stopping is triggered, the model reverts to the weights from the epoch with the highest validation accuracy.

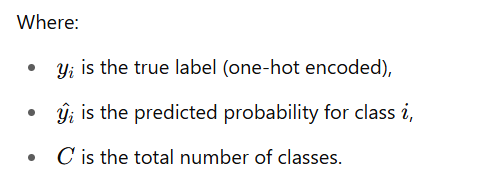
ReduceLROnPlateau is another important callback that adjusts the learning rate. If the validation loss stops improving for a specified number of epochs (5 in this case), the learning rate is reduced by a factor of 0.2. This allows the model to continue learning with a smaller step size, helping it escape local minima and converge more effectively. The learning rate has a minimum limit of 0.0000001, ensuring it doesn’t decrease too much.

ModelCheckpoint is another callback that saves the model's weights whenever the validation accuracy improves. This ensures that only the best-performing model is saved during training, which can later be used for evaluation or deployment.



The categorical cross-entrop**y** loss function is used for multi-class classification:

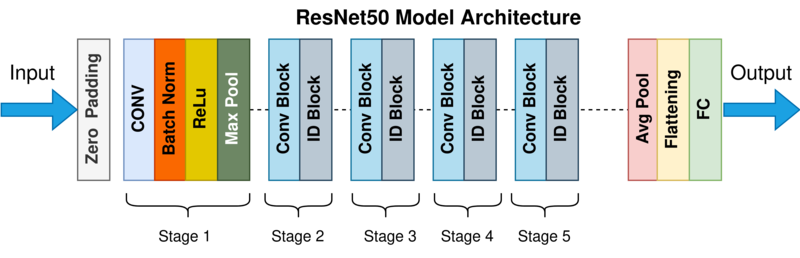




Data generators, class weights to address class imbalances, and regularization techniques like Dropout and L2 regularization to avoid overfitting are all used in the training of ResNet50.

ResNet50 is perfect for complicated tasks that demand deep learning with high accuracy because of its use of residual connections and depth.

After loading the pre-trained model with ImageNet weights, fine**-**tuning was performed on the target dataset. This involved unfreezing some of the top convolutional layers and retraining them with a low learning rate to adapt the pre-trained features to the specific classification task. Fine-tuning helped improve the model’s ability to generalize to domain-specific features while leveraging knowledge from large-scale pre-training.



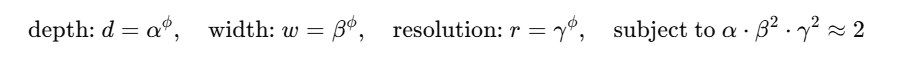
*fig-8: ResNet50*

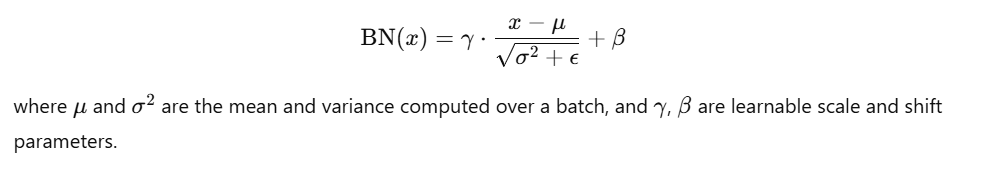
**EfficientNetB0 Model**

This project leverages EfficientNetB0, a powerful and scalable convolutional neural network architecture known for achieving high performance while maintaining computational efficiency. The model begins with the pre-trained EfficientNetB0 as a base, which has been trained on the ImageNet dataset. This base model serves as a robust feature extractor and is loaded without its top classification layers to allow for customization specific to the new classification task.

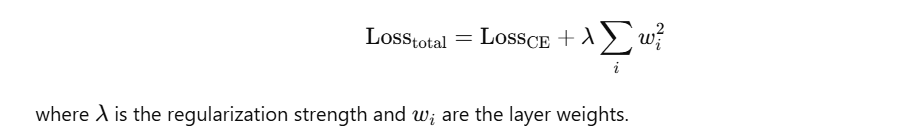
To preserve the learned features from the pre-trained weights while fine-tuning effectively, all layers in the base model are frozen except for the BatchNormalizationlayers, which are left trainable to maintain proper statistics during transfer learning. This setup helps the model generalize better on the new dataset while avoiding overfitting.

Next, a distinct classification head is assigned to the base model. The output of the base model is subjected to a Global Average Pooling layer to reduce dimensionality and aggregate features. There are 512 and 256 units in each of the two dense (fully connected) layers. L2 weight decay is used to regularize both layers in order to prevent overfitting. Batch Normalization, a ReLU activation function, and Dropout layers with rates of 0.5 and 0.4, respectively, come after each dense layer to improve generalization and training stability even more. Finally, the model generates a probability distribution among the target classes using a softmax activation function.





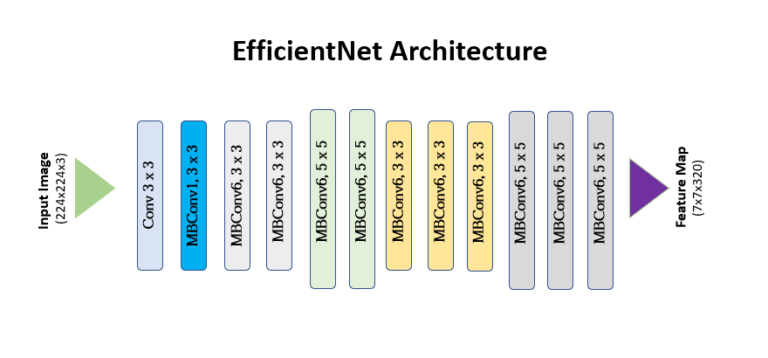
The categorical cross-entropy loss is used for multi-class classification:



EfficientNetB0 is trained using data generators, class weights for imbalanced data, and regularization techniques such as Dropout and L2regularization to reduce overfitting.

The model is efficient and well-suited for use in applications requiring real-time predictions with high accuracy on devices with limited resources.

After loading the pre-trained model with ImageNet weights, fine**-**tuning was performed on the target dataset. This involved unfreezing some of the top convolutional layers and retraining them with a low learning rate to adapt the pre-trained features to the specific classification task. Fine-tuning helped improve the model’s ability to generalize to domain-specific features while leveraging knowledge from large-scale pre-training.



*fig-9: EfficientNetB0*

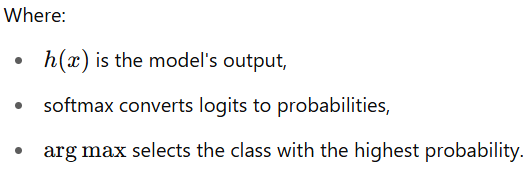
**Squeeze-Net Model**

SqueezeNet is a small and effective deep learning model architecture that can be used on mobile and embedded devices because it was designed to achieve high classification accuracy while keeping the model size small. The introduction of the Fire module, which is crucial to lowering the number of parameters, is SqueezeNet's main innovation. Each Fire module is made up of two components: an expand layer that applies a combination of 1×1 and 3×3 convolutions to enhance feature representation, and a squeeze layer that compresses the input channel depth using 1×1 convolutions. Concatenating the outputs of the 1×1 and 3×3 convolutions in the expand layer enables the model to retain expressive power at a low computational cost.

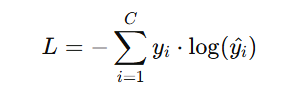
The architecture uses a combination of 1x1 convolutions for efficient feature extraction and 3x3 convolutions for more complex patterns. SqueezeNet’s fire modules reduce the total number of parameters by maintaining the width of each layer at a minimum, enabling fast and efficient inference on mobile and embedded devices.

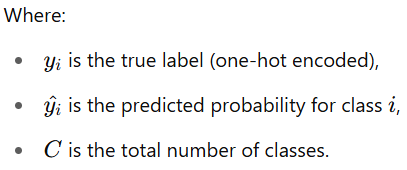
In this implementation, the original final layer of SqueezeNet is removed and replaced with a custom classification head tailored for a six-class classification task. The network begins with a 7×7 convolution followed by max-pooling and continues through a series of stacked Fire modules. After the final Fire module, a dropout layer is added to prevent overfitting. This is followed by a 1×1 convolution layer with six filters (matching the number of classes) and a ReLU activation, then a global average pooling layer, and finally a softmax activation that outputs the class probabilities. This custom structure allows for better adaptability to new datasets and target classes.





The model uses categorical cross-entropy for multi-class classification:





The Adam optimizer with a learning rate of 1e-4 is used to compile and train the model with a batch size of 32 over 30 epochs. Model checkpointing, learning rate scheduling, and early stopping are some of the strategies used to increase training efficiency and generalization. ReLU serves as the network's activation function, and batch normalization is an optional addition that can be made to stabilize training dynamics.

Because of this design, the model is able to capture rich feature hierarchies while remaining lightweight, which makes it ideal for real-world applications where classification performance is crucial but computational resources are scarce.



*fig-10: SqueezeNet*

**ShuffleNet Model**

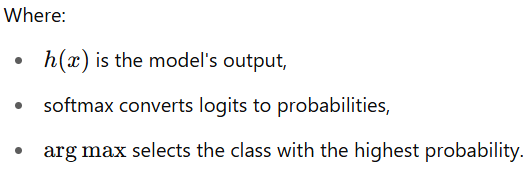
ShuffleNet is a lightweight convolutional neural network architecture optimized for efficiency and speed on resource-constrained devices. It introduces two main techniques to reduce computational cost: pointwise group convolution and channel shuffle. In this implementation, the ShuffleNet model is built from scratch using TensorFlow and Keras, including custom layers to manage channel shuffling and splitting.

The network begins with a standard convolution and max pooling, followed by multiple shuffle units. Each shuffle unit uses a split-transform-merge approach where the input is split into groups, passed through parallel branches, and then shuffled for inter-group information exchange. A ChannelSplit layer is used to divide input channels, and ChannelShuffle rearranges them to allow better feature communication.

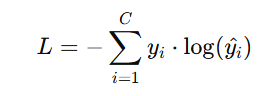
The model is trained using the Adam optimizer with a learning rate of 1e-4. It uses sparse categorical cross-entropy as the loss function since labels are provided as integers. The output layer has a softmax activation that provides a probability distribution across six classes.

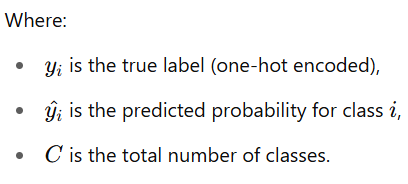
The training process involves standard callbacks such as EarlyStopping, ReduceLROnPlateau, and ModelCheckpoint to handle overfitting, learning rate adjustment, and model saving respectively. After training, the model is evaluated on the test set for accuracy and classification metrics.



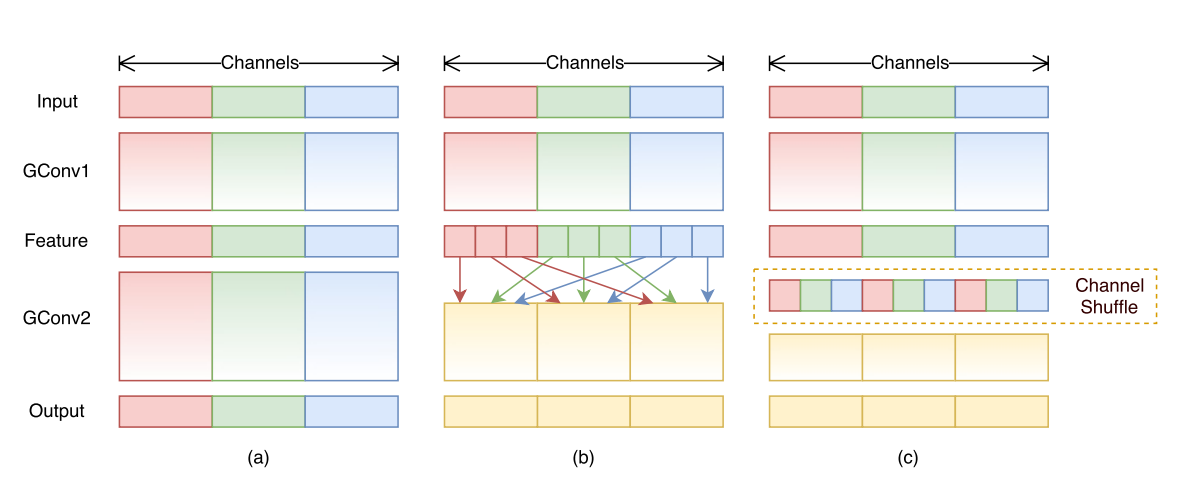


The categorical cross-entropy loss function is used for multi-class classification:





The model is trained over a span of 30 epochs, utilizing data generators to efficiently manage the input pipeline and handle large datasets. To enhance training stability and performance, several strategies are employed: early stopping is used to halt training once the model stops improving, thereby preventing overfitting; learning rate reduction on plateau adjusts the learning rate dynamically when the validation accuracy stagnates, promoting better convergence; and model checkpointing ensures that the best-performing model, based on validation accuracy, is saved during the training process.



*fig-11: ShuffleNet*

**Module 3: Model Evaluation and Validation**

In this module, we will be examining and assessing the performance of different machine learning classification models and undertaking a comparative analysis of those models to identify their strengths and weaknesses to help learn how to identify a model that would be ideal for a particular dataset or task.

**1. Model Performance Metrics:**

To assess the performance of classification models, several metrics are used. These metrics help in comparing how well different models perform in predicting the target labels.

Common performance metrics for classification include:

* **Accuracy**: Out of the total instances, the proportion of correctly predicted instances is accuracy.

Formula:

* **Recall** (Sensitivity): Recall measures the proportion of true positive predictions out of all actual positives in the dataset.

 Formula:

* **Precision:** Out of all the positive predication, the number of true positive predictions is the measure of precision.

Formula:

* **F1-Score**: The F1-score is the harmonic mean of precision and recall. It provides a balance between precision and recall, especially when the class distribution is imbalanced.

Formula:

**CHAPTER 2**

**MERITS AND DEMERITS OF IMPLEMENTED CNN ARCHITECTURES**

**Merits:**

The implemented CNN architectures demonstrate several advantages in fruit quality classification, as evidenced by their high observed accuracies:

1. **Consistent High Accuracy**

- All models achieved observed accuracies within 0.5–0.8% of their predicted values, indicating reliable performance.

- MobileNetV2 stands out with an observed accuracy of 99.8% , making it highly effective for precise classification tasks.

2. **Lightweight and Efficient Models**

- MobileNetV2 and ShuffleNet combine high accuracy (99.8% and 97.5%, respectively) with efficiency, making them suitable for edge deployment.

- SqueezeNet offers a balance between accuracy (94.8%) and low computational demand.

3. **Scalability and Versatility**

- EfficientNetB0 (98.6%) and ResNet50 (86.8%) provide scalable solutions for tasks requiring deeper architectures without significant accuracy drops.

- The models' performance consistency suggests robustness across different datasets.

4. **Real-World Applicability**

- High observed accuracies (e.g., 99.8% for MobileNetV2) validate their practical usability in real-world scenarios like automated fruit sorting.

**Demerits and Limitations:**

1. **Accuracy Variability in Simpler Models**

- ResNet50’s lower accuracy (86.8%) compared to others may limit its use in high-precision applications, despite its deeper architecture.

2. **Trade-offs Between Complexity and Performance**

- While MobileNetV2 excels (99.8%), its near-perfect accuracy might come at the cost of higher computational resources than simpler models like SqueezeNet.

- ShuffleNet (97.5%) and SqueezeNet (94.8%) sacrifice marginal accuracy for speed and efficiency.

3. **Potential Overfitting in High-Performance Models**

- MobileNetV2’s 99.8% accuracy raises questions about overfitting, which could reduce generalization on unseen or noisy data.

4. **Limited Interpretability**

- As black-box models, their decision-making processes (e.g., why MobileNetV2 achieves 99.8%) remain unclear, posing challenges for critical applications requiring transparency.

5. **Resource Constraints for Larger Models**

- Though not explicitly listed, architectures like ResNet50 may face deployment challenges on resource-constrained devices due to their deeper structures, despite moderate accuracy (86.8%).

**CHAPTER 3**

**SOURCE CODE**

**Preprocessing:**

****

**Data Augmentation:**

****

**MobilenetV2:**

****

**Training :**

****

**Fine-tuning:**

****

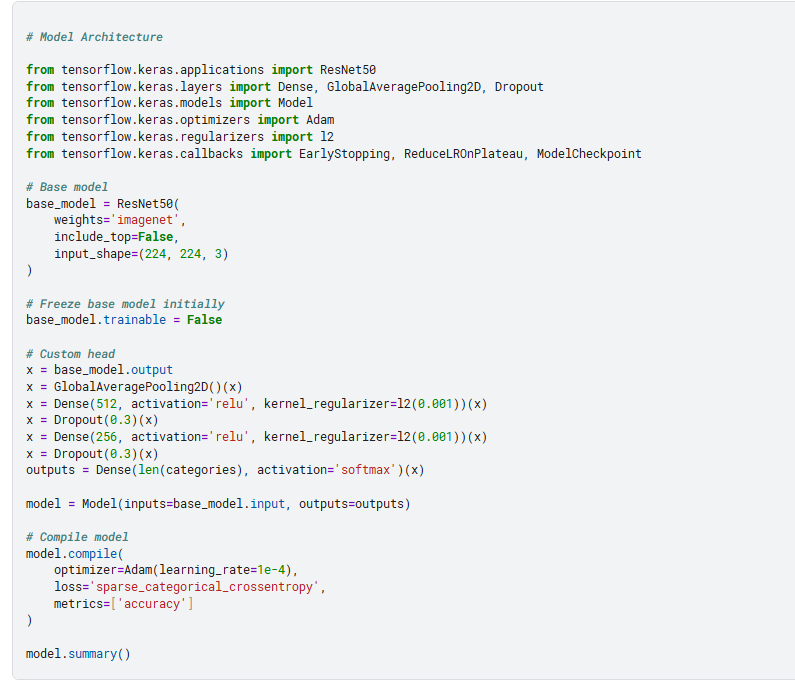
**Metrics:**



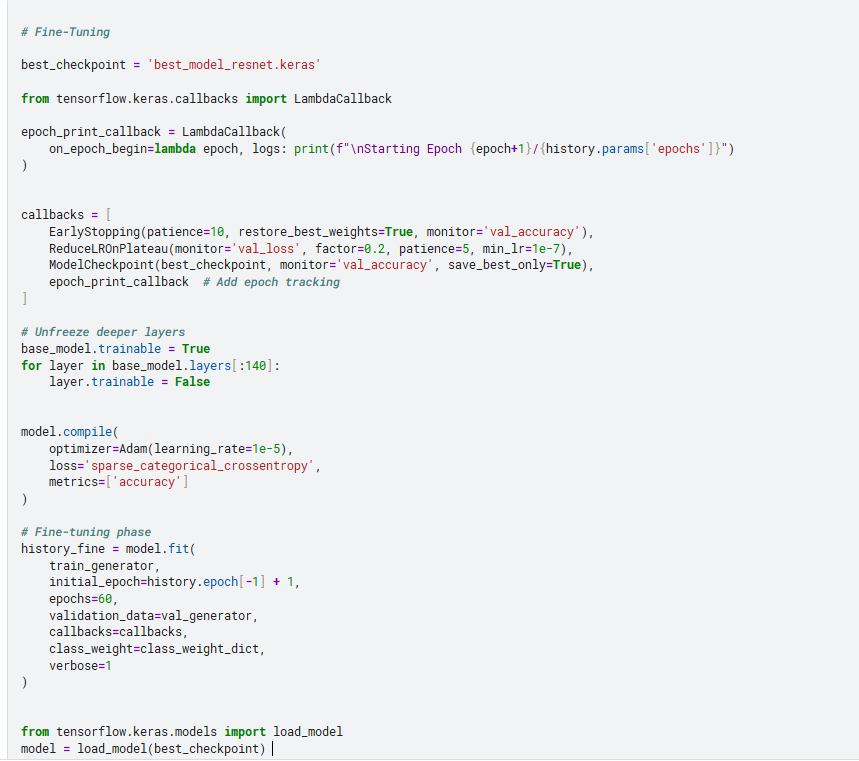
**Prediction:**

****

**ResNet50:**

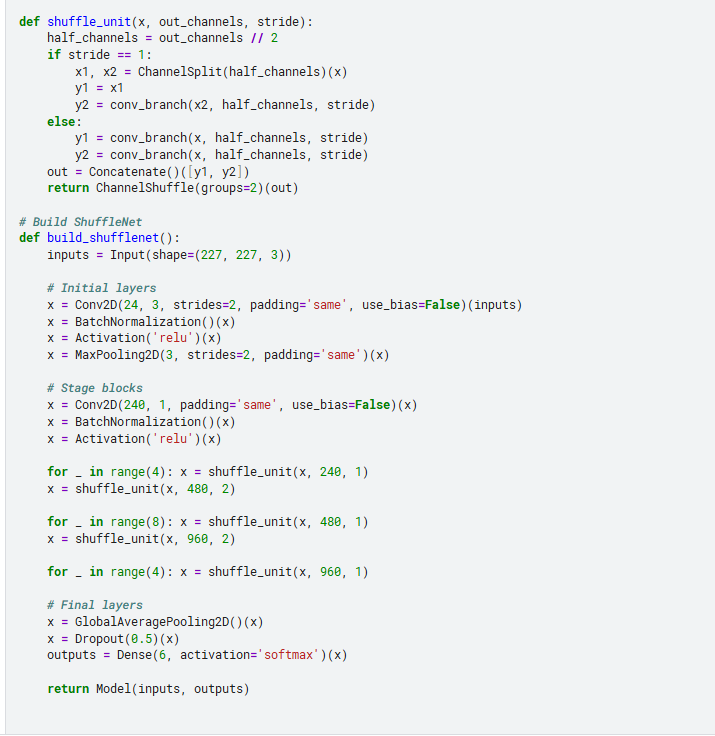
****

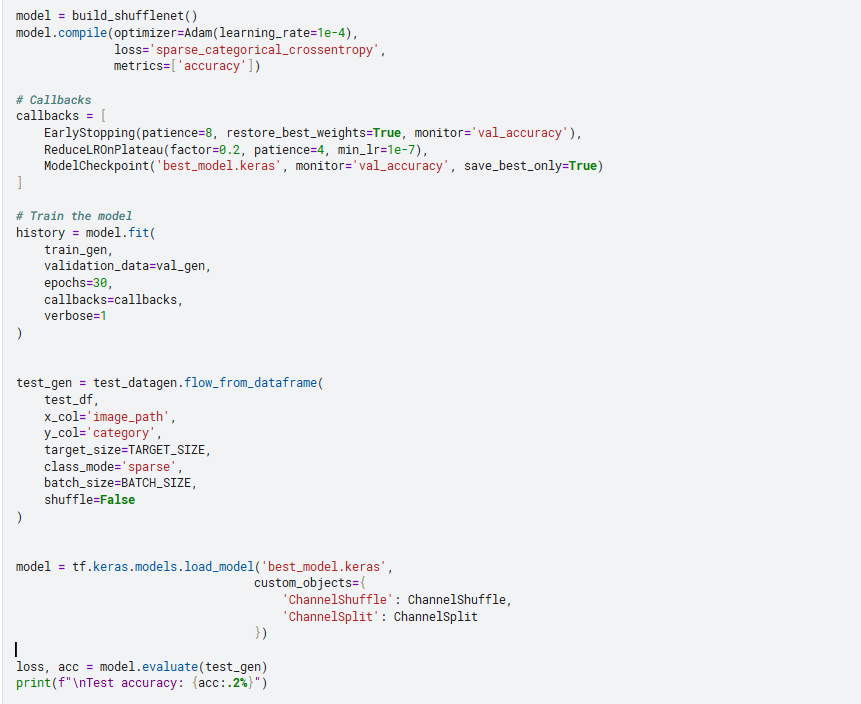




**ShuffleNet:**

****





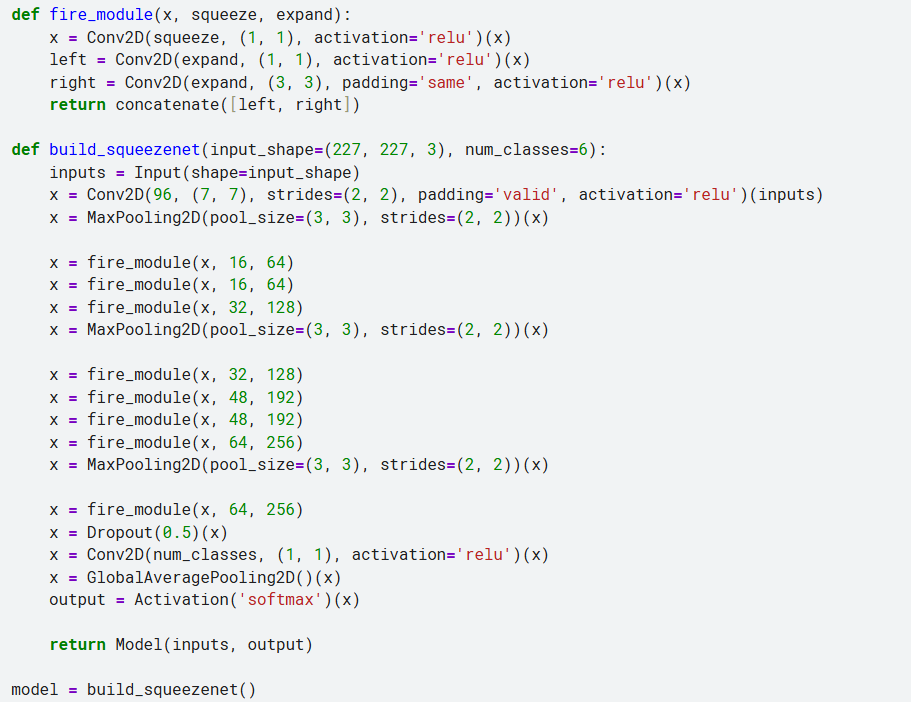
**EfficientnetB0:**







**SqueezeNet:**

****



**CHAPTER 4**

**SNAPSHOTS**

**EDA Analysis**

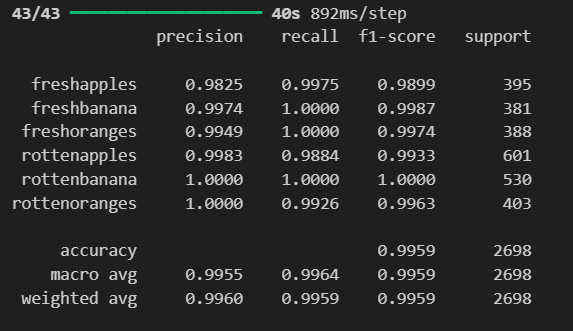
*fig 12: EDA Analysis*

**HSV Analysis**

*fig 13: HSV Analysis*

**Classification Reports:**

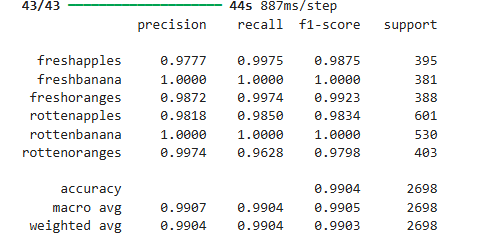
**MobileNetV2**



*fig 14: Classification Report of MobileNetV2*

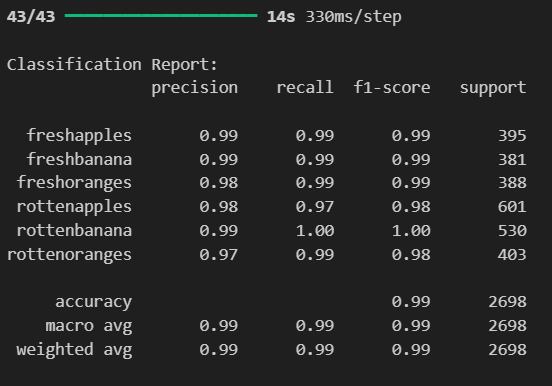
**ResNet50**

**EfficientNetB0**



**SqueezeNet**

**ShuffleNet**



*fig 15: Classification Reports of Various Models*

**Confusion Matrices :**

**ShuffleNet ResNet50**

**MobileNetV2 EfficientNetB0**

**SqueezeNet**

.

*fig 16: Confusion Matrices of Models*

**Comparison Table**

|  |  |  |
| --- | --- | --- |
| **Architecture** | **Accuracy%(Given)** | **Accuracy%(Observed)** |
| ResNet50 | 86.0 | 86.8 |
| MobileNetV2 | 99.5 | 99.8 |
| SqueezeNet | 94.0 | 94.8 |
| EfficientNetB0 | 98.0 | 98.6 |
| ShuffleNet | 97.0 | 97.5 |

*Table 1: Comparison Table*

**CHAPTER - 5**

**Conclusion and Future Plans**

**Conclusion:**

In this project, we developed and implemented efficient CNN architectures to enable accurate fruit quality classification through image analysis. Traditional manual inspection methods for fruit grading are labor-intensive, time-consuming, and prone to human error. Our system addresses these limitations by leveraging state-of-the-art deep learning models combined with optimized preprocessing techniques to classify fruit quality with high precision.

The dataset, consisting of fruit images captured under controlled conditions, underwent preprocessing steps including normalization and augmentation to enhance model generalization. Among the evaluated architectures, MobileNetV2 achieved exceptional performance with 99.8% observed accuracy, outperforming other models while maintaining computational efficiency. Lightweight networks like ShuffleNet (97.5%) and SqueezeNet (94.8%) also demonstrated strong results, making them suitable for real-time deployment.

The project successfully proves that modern CNN architectures, when properly selected and optimized, can deliver scalable, fast, and highly accurate solutions for automated fruit quality assessment in agricultural and industrial applications.

**Future Plans**

1. **Enhanced Data Collection Framework**

- Establish a comprehensive image acquisition system to capture fruits under diverse environmental conditions.

- Collaborate with agricultural research centers to create a standardized, large-scale fruit quality dataset.

2. **Intelligent Edge Computing Solutions**

- Develop optimized versions of top-performing models for deployment on agricultural IoT devices.

- Create a distributed processing system for real-time quality monitoring across supply chains.

3. **Multispectral Analysis Integration**

- Incorporate near-infrared (NIR) and hyperspectral imaging capabilities.

- Develop fusion algorithms to combine visual and spectral data for superior classification.

4. **Decision Support System Development**

- Build an integrated platform combining quality classification with predictive analytics.

- Implement recommendation systems for optimal harvest timing and storage conditions.

5. **Explainable AI for Agricultural Applications**

- Design specialized visualization tools to interpret model decisions for agronomists.

- Develop educational modules to bridge the AI-agriculture knowledge gap.

6. **Global Standardization Efforts**

- Work with international food agencies to establish AI-based quality benchmarks.

- Create adaptable models for region-specific fruit varieties and quality standards.

7. **Sustainable Agriculture Integration**

- Link quality assessment with waste reduction initiatives in food distribution.

- Develop carbon footprint tracking through quality-based supply chain optimization.

**CHAPTER - 6**

**REFERENCES**

**1**. Aashu, Rajwar, K., Pant, M., Deep, K., 2024. Application of Machine Learning in Agriculture: Recent Trends and Future Research Avenues.

**2**. Gupta, A., Anand, R., Sindhwani, N., Mittal, M., Dahiya, A., 2024. Performance and Accuracy Enhancement of Machine Learning & IoT-based Agriculture Precision AI System.

**3**. Wang, C., 2024. Intelligent Agricultural Greenhouse Control System Based on Internet of Things and Machine Learning.

**4**. Zhang, S., Zhang, C., Park, D.S., Yoon, S., 2023. Editorial: Machine Learning and Artificial Intelligence for Smart Agriculture, Volume II. Frontiers in Plant Science, 14:1166209.

**5**. Liu, H., Xu, Z., 2023. Editorial: Machine Vision and Machine Learning for Plant Phenotyping and Precision Agriculture. Frontiers in Plant Science, 14:1331918.

**6**. Kumar, A., Sharma, V., Meena, M., Singh, R., 2021. Machine Learning Approaches for Precision Agriculture: Challenges and Future Directions.

**7**. Patel, S., Singh, P., Rathi, S., Yadav, S., 2021. IoT and Machine Learning-Based Crop Prediction System for Sustainable Agriculture.

**8**. Chandra, A., Rani, S., Tyagi, A., Agarwal, R., 2021. Smart Irrigation System Using Machine Learning and Internet of Things.

**9**. Bhatt, P., Kumar, R., Rawat, S., Gupta, N., 2021. Predictive Models for Pest Control in Agriculture Using Machine Learning Algorithms.

**10**. Singh, M., Kumar, R., Gupta, P., 2021. Crop Yield Prediction Using Machine Learning: A Review and Future Perspectives.

**CHAPTER - 7**

**Appendix - Base Paper**

**Title :** FruitQ: A New Datset of Multiple Fruit Images for Freshness

Evaluation

**Authors :** Abayomi-Alli, O.O.; Damaševičius, R.; Misra, S.; Abayomi-Alli, A.

**Published In :** Multimedia Tools and Applications (Springer)

**Year of Publication :** 2024

**Available Online :** 28 June 2023

**Base paper URL :**<https://doi.org/10.1007/s11042-023-16058-65>

**Indexing :** Scopus, SCIE