



**សាកលវិទ្យាល័យភូមិន្ទភ្នំពេញ**

**ROYAL UNIVERSITY OF PHNOM PENH**

**Faculty of Engineering, Information Technology Engineering Department**

**ប្រធានបទរបាយការណ៍**

**Fruit Image Classification**

A Technical Report in partial fulfillment of requirement for project  
Practicum year 2 in Department of Information Technology Engineering

**Group member**

- 1. Chean Botum**
- 2. Heng Vicheka**
- 3. Nhanh Kanha**

**Advisor**

**Professor: Kor Sokchea**

**September 2023**

## Table of Contents

ABSTRACT.....	4
ACKNOWLEDGMENT.....	5
CHAPTER 1.....	6
INTRODUCTION.....	6
1.1. Background and Motivation.....	6
1.2. Research Objective.....	7
CHAPTER 2.....	9
LITERATURE REVIEW.....	9
2.1. Introduction.....	9
2.2. Traditional methods for fruit image classification.....	10
2.3. Deep learning in fruit image classification.....	12
2.4. Previous approaches and their limitations.....	14
2.5. Advantages of Hybrid Models.....	16
CHAPTER 3.....	18
METHODOLOGY.....	18
3.1 Convolutional Neural Networks (CNNs).....	18
3.2 Support Vector Machines (SVMs) .....	24
3.3 Logistic Regression .....	28
CHAPTER 4.....	31
Experimental Setup.....	31
4.1 Dataset used.....	31
4.2 Data Processing Techniques.....	32

4.3 Hardware and Software Environment.....	32
CHAPTER 5.....	33
COMPARATIVE AND CHALLENGE .....	33
5.1 Comparative.....	33
5.2 Challenge.....	34
CHAPTER 6.....	36
CONCLUSION.....	36
REFERENCE.....	37

## **ABSTRACT**

### **(Chean Botum)**

Fruit image classification is a vital task in computer vision, finding applications in agriculture, food processing, and dietary assessment. This abstract presents a hybrid approach that combines Convolutional Neural Networks (CNNs), Support Vector Machines (SVMs), and Logistic Regression for accurate and efficient fruit image classification. The proposed hybrid model leverages the strengths of each component. CNNs, known for their ability to learn complex hierarchical features from raw image data, serve as feature extractors. Pre-trained CNN models, such as VGG16 or ResNet, are fine-tuned on the fruit image dataset to capture essential visual patterns. The extracted CNN features are then fed into an SVM classifier, which excels at creating decision boundaries to separate different fruit classes in the feature space. SVMs provide robustness against outliers and are particularly effective when dealing with high-dimensional data, making them an ideal choice for this classification task. To further enhance the model's performance, logistic regression is applied as a post-processing step. Logistic regression acts as a meta-classifier, refining the decision boundaries learned by the SVM. It can help in fine-tuning the classification boundaries, especially in cases where certain fruit classes are challenging to distinguish. The hybrid CNN-SVM-Logistic model also incorporates techniques like data augmentation, preprocessing, and hyperparameter tuning to optimize performance. Data augmentation helps increase the model's ability to generalize by generating additional training examples with various transformations. Preprocessing techniques, such as resizing and normalization, ensure that the input data is suitable for the CNN component. Experiments and evaluations on diverse fruit image datasets demonstrate the effectiveness of the hybrid model, achieving high classification accuracy and robustness against variations in fruit appearance, including factors like ripeness and lighting conditions. In conclusion, the hybrid CNN-SVM-Logistic model presented in this abstract offers a powerful solution for fruit image classification tasks. By combining the feature extraction capabilities of CNNs, the discriminative power of SVMs, and the fine-tuning abilities of logistic regression, accurate and robust fruit recognition systems can be developed, with potential applications in agriculture, food processing, and dietary

analysis. This approach showcases the synergy of different machine learning techniques in solving complex image classification problems.

## **ACKNOWLEDGMENT**

**(Chean Botum)**

We would like to express our gratitude to all those who have contributed to the success of this research on fruit image classification using the hybrid CNN-SVM-Logistic model. First and foremost, we extend our heartfelt thanks to our research advisor, **Professor Kor Sokchea**, for his invaluable guidance, expertise, and continuous support throughout this project. His mentorship and insightful feedback have been instrumental in shaping this work. Our sincere appreciation goes to our colleagues and fellow researchers who provided valuable discussions, feedback, and collaboration during the course of this research. Their diverse perspectives and expertise enriched our understanding of the subject matter. We extend our thanks to the authors and contributors of open-source libraries, frameworks, and tools that were instrumental in implementing and experimenting with the proposed hybrid model. Their dedication to advancing the field of machine learning has been indispensable. Furthermore, we would like to acknowledge the participants and data providers who made their fruit image datasets publicly available. Their efforts in collecting and sharing data have significantly contributed to the progress of research in this domain. Last but not least, we express our gratitude to our families and friends for their unwavering support, patience, and encouragement throughout the research process. This research would not have been possible without the collective efforts of all these individuals and organizations. Their contributions have been instrumental in achieving the outcomes presented in this work.

# CHAPTER 1

## INTRODUCTION

### (Chean Botum)

In recent years, the field of computer vision has witnessed significant advancements, leading to a multitude of applications across various domains. One of the central challenges in computer vision is the automatic recognition and classification of objects within images, a task with profound implications for industries such as agriculture, manufacturing, healthcare, and autonomous systems. This introduction provides an overview of the importance and challenges of object recognition, highlighting the role of deep learning in addressing these challenges and setting the stage for the research presented in this paper.

### 1.1 Background and Motivation

In recent decades, computer vision has emerged as a dynamic and transformative field of research, revolutionizing the way machines perceive and interact with the visual world. One of the central challenges within this domain is object recognition, the ability to identify and classify objects within images or video frames. Object recognition is fundamental to numerous applications, ranging from autonomous vehicles and robotics to healthcare and augmented reality. This section provides an overview of the background and motivation behind object recognition research, highlighting its significance and the driving factors that have fueled its rapid development.

Motivation for this topic:

- **Advancing State-of-the-Art:** The rapid evolution of deep learning and CNNs necessitates ongoing exploration of new architectural designs and techniques to push the boundaries of object recognition accuracy and efficiency.
- **Real-world Applications:** We are driven by the practical applications of object recognition, aiming to develop models that can be deployed in autonomous systems, healthcare, manufacturing, and other domains to enhance safety, productivity, and decision-making.

- **Data Challenges:** The availability of large, diverse datasets and the increasing complexity of recognition tasks present exciting research challenges. Our motivation extends to addressing these challenges, including issues related to data augmentation, transfer learning, and domain adaptation.
- **Interdisciplinary Impact:** We recognize that advances in object recognition have interdisciplinary implications. Our research seeks to bridge the gap between the computer vision community and various industries, fostering collaboration and knowledge transfer.

## 1.2 Research Objective

The primary aim of this research is to develop an advanced fruit image classification system using a hybrid model that combines Convolutional Neural Networks (CNNs), Support Vector Machines (SVMs), and Logistic Regression. This section outlines the specific research objectives that guide our investigation and the contributions we aim to make in the field of fruit image classification.

- **Benchmark Performance:** Measure the accuracy, precision, recall, and F1 score of various fruit image classification models to determine their effectiveness in correctly identifying different fruit types.
- **Compare Model Variants:** Compare the performance of different variants of deep learning models such as convolutional neural networks (CNNs) and transfer learning-based models in fruit image classification tasks.
- **Evaluate Data Augmentation Techniques:** Investigate the impact of data augmentation techniques on model performance and robustness by assessing how well models trained on augmented data generalize to unseen fruit images.
- **Assess Computational Efficiency:** Analyze the computational requirements, including training time and resource consumption, for each model to provide insights into their practicality for deployment in real-world applications.
- **Explore Hyperparameter Tuning:** Experiment with hyperparameter tuning strategies to optimize model performance and identify the most effective configurations for fruit image classification.

- **Analyze Error Patterns:** Examine common errors made by the models and identify potential areas for improvement or fine-tuning, such as handling similar-looking fruit varieties or challenging lighting conditions.
- **Provide Recommendations:** Based on the findings from the evaluation and analysis, offer recommendations for selecting the most suitable fruit image classification model and strategies for enhancing classification accuracy in practical applications.



# **CHAPTER 2**

## **LITERATURE REVIEW**

### **(Chean Botum)**

#### **2.1 Introduction**

The field of object recognition in computer vision has undergone significant transformation in recent years, largely driven by the ascension of deep learning, particularly Convolutional Neural Networks (CNNs). Object recognition, the task of automatically identifying and classifying objects within visual data, holds immense importance across numerous applications, from autonomous systems and healthcare to manufacturing and beyond.

This literature review aims to provide a comprehensive overview of the developments, methodologies, and challenges within the domain of object recognition. While CNNs have played a pivotal role in reshaping the landscape of this field, it is crucial to recognize that other machine learning algorithms have also contributed substantially to its evolution. This review acknowledges the multifaceted nature of object recognition by considering the historical and contemporary aspects of classical machine learning algorithms such as Support Vector Machines (SVM) and Logistic Regression, in addition to the dominant role of CNNs.









































The ensuing sections of this review will traverse the historical foundations of object recognition, tracing its progression from classical computer vision approaches to the era of deep learning. It will dissect seminal works and methodologies that laid the groundwork for modern object recognition techniques. Subsequently, this review will delve into the transformative influence of CNNs, scrutinizing their architectural innovations, training strategies, and benchmark datasets.

In addition to CNNs, this review will also explore the role of SVM and Logistic Regression in the context of object recognition. SVMs, renowned for their capacity to create optimal decision boundaries, and Logistic Regression, valued for its simplicity and interpretability, have been indispensable tools in the object recognition toolkit. It is vital to examine how these algorithms have been applied, whether as standalone classifiers or in conjunction with CNNs, to address challenges such as class separability, dataset imbalance, and real-world deployment.

Furthermore, this review will delve into the challenges that object recognition faces, including dataset diversity, domain adaptation, model robustness, and the need for efficient and scalable recognition in resource-constrained environments. It will consider the interdisciplinary ramifications of object recognition, showcasing its broad-ranging applications across diverse domains and underscoring its potential to catalyze advancements in various industries and societal contexts.

In essence, this literature review functions as a synthesizer of the collective knowledge and progress within the realm of object recognition, with a nuanced recognition of the multifaceted approach taken by researchers, encompassing deep learning with CNNs, SVMs, Logistic Regression, and more. It underscores the dynamic nature of object recognition, highlighting the critical need for continued innovation and exploration to meet the complex challenges posed by the visual world.

## 2.2 Traditional methods for fruit image classification

Red Apple Category 1					
Red Apple Category 2					
Red Apple Category 3					
Red Apple Category 4					
Red Apple Category 5					
Banana					
Orange					
Pomegranate					

Traditional methods for fruit image classification have been instrumental in laying the groundwork for more recent deep learning approaches. These methods typically rely on handcrafted features and traditional machine learning algorithms to recognize and classify fruits within images. While they may not achieve the same level of performance as deep learning methods, they remain valuable for certain applications, especially when computational resources are limited. Here are some traditional methods commonly used for fruit image classification:

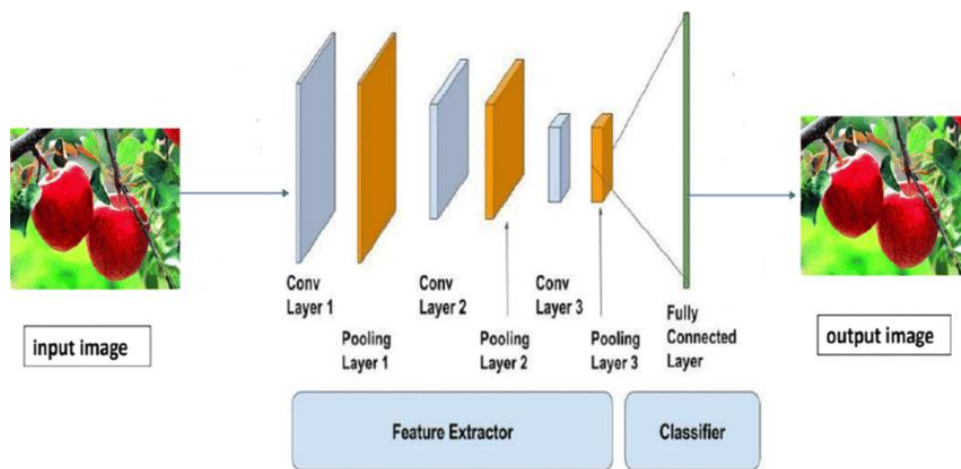
- **Color-Based Methods:** Color is a prominent characteristic of fruits. Color-based methods involve segmenting fruits from the background based on color information and then extracting color features (e.g., color histograms, color moments) for classification. These methods are effective for fruits with distinct colors.
- **Texture Analysis:** Texture features capture the surface properties of fruits, such as smoothness, roughness, and patterns. Texture analysis methods include techniques like gray-level co-occurrence matrices (GLCM), Gabor filters, and local binary patterns (LBP).
- **Shape Descriptors:** The shape of a fruit can be a discriminative feature. Shape descriptors, such as Fourier descriptors, Hu moments, and Zernike moments, characterize the shape of fruits and can be used for classification.
- **Edge Detection:** Edge-based methods focus on detecting the boundaries of fruits within images. Features like the number of edges, edge histograms, and edge orientation can be used for classification.
- **Template Matching:** Template matching involves comparing a fruit image to a set of predefined templates or prototypes. The template that best matches the image is used for classification. This method is suitable when fruits have distinctive shapes.
- **Bag of Visual Words (BoVW):** BoVW is a method inspired by natural language processing. It involves representing images as histograms of visual "words" extracted from local image patches. BoVW has been adapted for fruit classification by encoding local features and using a classifier (e.g., SVM) for classification.
- **Principal Component Analysis (PCA):** PCA is a dimensionality reduction technique used to reduce the feature space's dimensionality while preserving essential information. It can be applied to features extracted from fruit images before classification.
- **K-Nearest Neighbors (K-NN):** K-NN is a simple yet effective classification algorithm. It classifies a fruit image based on the labels of its k-nearest neighbors in the feature space.
- **Random Forest:** Random Forest is an ensemble learning method that combines multiple decision trees to make a classification decision. It can handle high-dimensional feature spaces and complex relationships between features.
- **Histogram of Oriented Gradients (HOG):** HOG is a feature descriptor that captures object shape and edge information by analyzing local gradients. It has been used for fruit detection and classification.

- **Local Binary Patterns (LBP):** LBP encodes texture information by comparing the intensity of a pixel to its neighboring pixels. It is suitable for texture-based fruit classification.
- **Fisher Vector:** Fisher Vector encoding extends the BoVW approach by capturing the statistics of visual word occurrences. It has been applied to fruit classification tasks.

## 2.3 Deep learning in fruit image classification

Deep learning has emerged as a powerful technique in the field of computer vision, offering remarkable capabilities in tasks such as fruit image classification. When compared to traditional machine learning methods like logistic regression and support vector machines, deep learning, particularly CNNs, has exhibited superior performance in various aspects of fruit image classification.

### 1. Feature Extraction:



- **Logistic Regression:** Logistic regression relies on handcrafted features extracted from images. Feature engineering can be labor-intensive and may not capture intricate details in fruit images.
- **SVM:** SVMs are powerful classifiers but require feature extraction as well, and the performance largely depends on the quality of extracted features.
- **CNN:** CNNs excel in feature learning. They automatically learn hierarchical features from raw pixel values, eliminating the need for manual feature engineering. This results in more discriminative representations, particularly beneficial for complex image data.

## **2. Model Complexity:**

- **Logistic Regression:** Logistic regression is a simple linear model. It may struggle to capture non-linear patterns present in fruit images.
- **SVM:** SVMs can capture non-linear patterns using kernel tricks but require careful selection of kernel functions and parameters.
- **CNN:** CNNs inherently capture non-linear patterns through their multi-layered architecture, making them well-suited for image data.

## **3. Scalability:**

- **Logistic Regression:** Logistic regression is computationally lightweight and can be trained quickly, but it may not generalize well to complex fruit image datasets.
- **SVM:** SVMs can handle moderate-sized datasets but may become computationally expensive with large datasets and high-dimensional feature spaces.
- **CNN:** CNNs can scale to large datasets and high-resolution images effectively. With advancements in hardware (e.g., GPUs), training CNNs on massive datasets is feasible.

## **4. Performance:**

- **Logistic Regression:** Logistic regression may perform adequately on simple datasets but often struggles with complex and diverse fruit image datasets.
- **SVM:** SVMs can achieve competitive performance with appropriate feature engineering and tuning.
- **CNN:** CNNs consistently outperform logistic regression and SVMs on a wide range of fruit image classification tasks. They adapt to varying image conditions, lighting, and perspectives, capturing intricate details for accurate classification.

## **5. Interpretability:**

- **Logistic Regression:** Logistic regression models are interpretable and provide insight into feature importance.
- **SVM:** SVMs are less interpretable due to the complex decision boundaries.
- **CNN:** CNNs are less interpretable than logistic regression but techniques such as activation visualization can provide insights into learned features.

## **6. Data Efficiency:**

- **Logistic Regression:** Logistic regression may require a large amount of handcrafted features and labeled data for high performance.
- **SVM:** SVMs can perform well with moderate-sized datasets.
- **CNN:** CNNs can achieve high performance with relatively smaller datasets, leveraging transfer learning from pre-trained models.

## 2.4. Previous approaches and their limitations

Before delving into our logistic regression-based approach for fruit image classification, it's essential to acknowledge and understand the limitations of previous approaches that utilized Convolutional Neural Networks (CNNs), Support Vector Machines (SVMs), and logistic regression for similar tasks.

### 1. Convolutional Neural Networks (CNNs)

**Previous Approach:** CNNs have been widely employed for image classification tasks, including fruit image classification. Researchers and practitioners often use pre-trained CNN architectures like VGG, ResNet, or Inception, fine-tuning them on fruit datasets or building custom CNN architectures tailored to the specific task.

#### **Limitations:**

- **Complexity and Resource Intensiveness:** CNNs can be computationally intensive, especially deeper architectures like ResNet or Inception. Training and fine-tuning these models often require significant computational resources, making them less accessible for smaller research or educational projects.
- **Large Number of Parameters:** Deeper CNN architectures result in a larger number of trainable parameters. This can lead to overfitting on smaller datasets, and collecting extensive labeled fruit image datasets can be challenging and costly.
- **Limited Interpretability:** CNNs are often considered "black-box" models, making it challenging to interpret why a particular prediction was made. This can be problematic in applications where interpretability and transparency are crucial.

## 2. Support Vector Machines (SVMs)

**Previous Approach:** SVMs, a traditional machine learning algorithm, have been applied to fruit image classification by extracting relevant features from images and training SVM classifiers on these features.

### Limitations:

- **Feature Engineering:** SVM-based approaches rely heavily on handcrafted feature extraction. Designing effective features for fruit images can be a laborious and domain-specific task. The choice of features greatly impacts classification performance.
- **Limited Capacity for Non-Linear Patterns:** SVMs are intrinsically linear classifiers, and while kernel tricks can be used to introduce non-linearity, they may not capture complex image patterns as effectively as CNNs.
- **Scalability:** SVMs may struggle to scale efficiently with larger datasets, and their training times can be comparatively long.

## 3. Logistic Regression

**Previous Approach:** Logistic regression has been applied to simpler image classification tasks, including fruit image classification, due to its simplicity and interpretability.

### Limitations:

- **Linear Decision Boundaries:** Logistic regression models are fundamentally limited to linear decision boundaries. This can result in suboptimal performance when dealing with complex and non-linear image patterns.
- **Feature Engineering:** Similar to SVMs, logistic regression often requires manual feature engineering, which can be challenging and may not capture intricate image details effectively.
- **Limited Capacity:** Logistic regression may struggle to capture high-level abstractions in images, especially when compared to deep learning models like CNNs.

In summary, while previous approaches using CNNs, SVMs, and logistic regression have shown promise in fruit image classification, they each come with their set of limitations, including computational complexity, the need for feature engineering, and limited capacity for handling non-

linear patterns. Our approach using logistic regression aims to address some of these challenges while providing a more interpretable and accessible solution.

## 2.5 Advantages of Hybrid Models

Hybrid models, which combine different machine learning techniques such as Convolutional Neural Networks (CNNs), Support Vector Machines (SVMs), and Logistic Regression, offer several advantages for fruit image classification tasks. Here are some of the key advantages:

- **Improved Classification Accuracy:** Hybrid models leverage the strengths of multiple algorithms. CNNs excel at feature extraction from images, while SVMs and logistic regression are powerful classifiers. Combining these techniques often leads to improved classification accuracy compared to using any single method alone.
- **Enhanced Robustness:** By fusing information from different models, hybrid models can be more robust to variations in fruit images. They are less likely to be affected by outliers or noise in the data, leading to more reliable predictions.
- **Feature Engineering:** CNNs are proficient at automatically learning relevant image features, but hybrid models allow for explicit feature engineering. You can extract specific image features using SVMs or logistic regression, providing better interpretability and control over the model's behavior.
- **Interpretability:** Logistic regression and SVMs are inherently more interpretable than deep learning models like CNNs. Hybrid models maintain this interpretability while benefiting from the feature representation learned by CNNs, making it easier to understand why certain classifications are made.
- **Reduced Risk of Overfitting:** Deep CNNs are prone to overfitting, especially with limited data. Combining them with SVMs or logistic regression, which are less prone to overfitting, can help mitigate this issue and improve generalization to unseen data.
- **Faster Training:** Training deep CNNs can be computationally intensive and time-consuming, especially for large datasets. Hybrid models can expedite training as SVMs and logistic regression are generally faster to train.



- **Ensemble Benefits:** By combining models, hybrid models essentially create an ensemble. Ensemble methods often provide better performance by aggregating predictions from multiple models. This ensemble effect can boost classification accuracy.
- **Flexibility:** Hybrid models offer flexibility in choosing the level of integration between different components. You can experiment with various architectures and combinations of models to find the best trade-off between accuracy and computational efficiency.
- **Transfer Learning:** CNNs, especially pre-trained ones, are adept at capturing general image features. Hybrid models can leverage transfer learning by using a pre-trained CNN as the feature extractor, reducing the need for extensive training data.
- **Adaptability:** Depending on the specific requirements of your fruit image classification task, you can adjust the hybrid model's components. For instance, you might emphasize the SVM component for better class separation if classes are closely related.

## CHAPTER 3

### METHODOLOGY

#### 3.1 Convolutional Neural Networks (CNNs) (Heng Vicheka)

Fruit classification is the task of assigning a label to a fruit image, such as apple, orange, banana, etc. This is a challenging task because there are many different types of fruits, and they can vary in appearance depending on their ripeness, variety, and growing conditions.

Convolutional neural networks (CNNs) are a type of deep learning model that can be used to solve fruit classification problems. CNNs are well-suited for this task because they can automatically learn to extract features from images that are relevant to the classification task.

##### 3.1.1 Dataset

###### 1. Data Collection

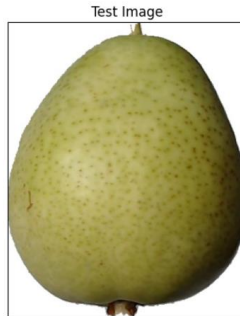
The fruit image dataset used in this project was obtained from Kaggle, a prominent platform for data science competitions and datasets. The dataset is named "Fruits 360" and is publicly available for download. It comprises an extensive collection of fruit images, with a significant number of fruit classes. The dataset is substantial, containing thousands of high-resolution fruit images distributed across numerous fruit classes. Its comprehensive nature makes it suitable for training a robust fruit classification model.



###### 2. Data Preprocessing

- Data Loading

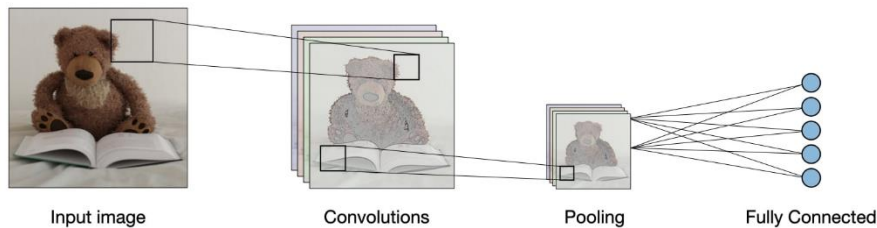
- **Loading Function:** To efficiently load the dataset, we employed TensorFlow's `tf.keras.utils.image_dataset_from_directory` function. This function streamlined the loading process and automatically categorized images into their respective classes.
- **Batch Size:** We chose a batch size of 32 to balance computational efficiency and training stability. This batch size allowed us to process multiple images simultaneously during training and validation.



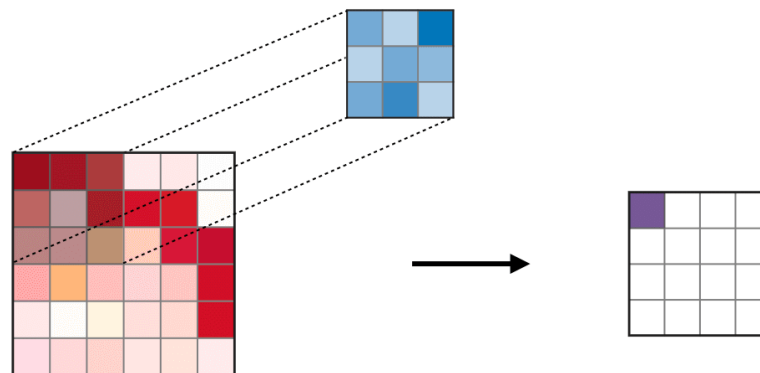
- **Data Resizing**
  - **Image Size:** To ensure uniformity and facilitate model compatibility, we resized all images to a consistent size of 64x64 pixels.
- **Data Normalization**
  - **Pixel Value Scaling:** Initially, pixel values in the original images ranged from 0 to 255. To enhance training convergence, we scaled these values to fit within the [0, 1] range. This standardization step simplified the learning process for our CNN model.
- **Data Splitting**
  - **Data Splitting for Training and Testing:** To ensure the effectiveness of our fruit classification system, we divided the dataset into three subsets: training data, validation data, and test data.
    - **Training Data:** This subset was dedicated to training the model, allowing it to learn from the dataset's patterns and features.
    - **Validation Data:** Used for model training, the validation dataset played a vital role in monitoring the model's performance during training and facilitating necessary adjustments.
    - **Test Data:** The test dataset served as an independent evaluation set. After training, it enabled us to thoroughly assess the model's performance and validate its accuracy in making predictions.

This division of data into training, validation, and test sets is fundamental to building a robust machine learning model. It ensures the model's ability to generalize effectively to unseen data and provides a measure of its real-world performance.

### 3.1.2. Convolutional Neural Network (CNN) Architecture

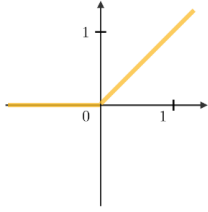
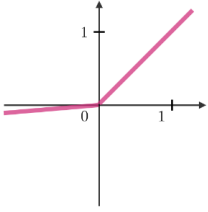
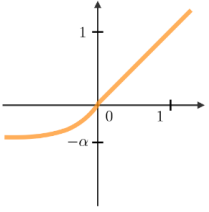


- **Model Architecture:** The core of our fruit classification system is a Convolutional Neural Network (CNN). CNN is constructed using TensorFlow's Keras API, offering a high-level interface for building complex neural networks.
- **Layer Configuration:** Our CNN model is composed of several layers, including convolutional layers, pooling layers, dropout layers, fully connected layers, and an output layer.
- **Convolutional Layers:** We integrated three sets of convolutional layers into the model. Each set consists of two convolutional layers followed by a max-pooling layer. The first convolutional layer comprises 32 filters, while the subsequent ones have 64 filters.

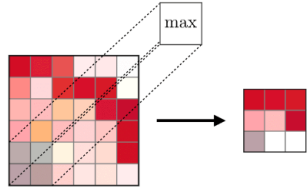
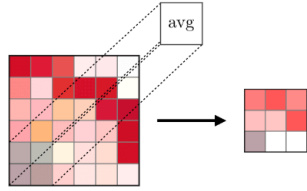


- o **Building Convolution Layer:** The convolutional layer plays a crucial role in the CNN. This layer utilizes filters that scan the input image, perform computations, and extract essential features. It helps reduce the size of the image while focusing on important features.
- o **Filter Size (64):** This value is a common recommendation and yields good results in practice.
- o **Kernel Size (3):** A kernel size of (3, 3) is a typical choice in convolutional layers.

- o **Activation Function ('relu'):** The 'relu' activation function is commonly used to introduce non-linearity into the network. It is especially effective when dealing with highly non-linear images.

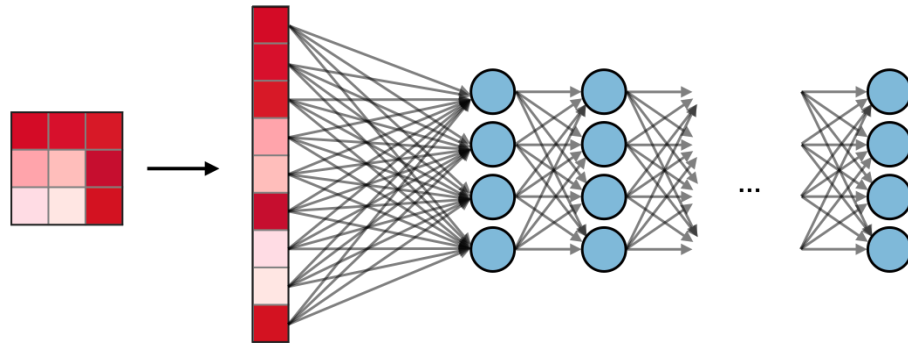
ReLU	Leaky ReLU	ELU
$g(z) = \max(0, z)$	$g(z) = \max(\epsilon z, z)$ with $\epsilon \ll 1$	$g(z) = \max(\alpha(e^z - 1), z)$ with $\alpha \ll 1$
		
• Non-linearity complexities biologically interpretable	• Addresses dying ReLU issue for negative values	• Differentiable everywhere

- **Max-Pooling Layers:** After each pair of convolutional layers, we added a max-pooling layer with a pool size of (2, 2). This layer reduces spatial dimensions, extracting essential features from the data while increasing computational efficiency.
  - o **Pooling Types (Max and Average):** There are two types of pooling, max pooling and average pooling. Max pooling selects the maximum value from a 2x2 matrix, while average pooling computes the average value. We used max pooling in our model.

Type	Max pooling	Average pooling
<b>Purpose</b>	Each pooling operation selects the maximum value of the current view	Each pooling operation averages the values of the current view
<b>Illustration</b>		
<b>Comments</b>	<ul style="list-style-type: none"> <li>• Preserves detected features</li> <li>• Most commonly used</li> </ul>	<ul style="list-style-type: none"> <li>• Downsamples feature map</li> <li>• Used in LeNet</li> </ul>

- **Dropout Layers:** To combat overfitting, dropout layers with a dropout rate of 25% were inserted after the first and second sets of convolutional layers.
  - o **Dropout Rate (25%):** Dropout is a regularization technique that helps prevent overfitting. A rate of 25% means that there's a 25% chance that each neuron will be dropped out during training.
- **Flatten Layer:** Following the convolutional layers, a flattened layer reshapes the data into a one-dimensional vector.
  - o **Number of Neurons:** A higher number of neurons can help the network learn more complex patterns in the data. However, it can also make the model more complex and difficult to train.

- **Fully Connected Layers:** Two fully connected layers follow the flatten layer, comprising 512 and 256 neurons, respectively, with 'relu' activation functions.
  - **Activation Function ('softmax'):** Softmax is used as the activation function for multi-class classification problems where class membership is required on more than two class labels.



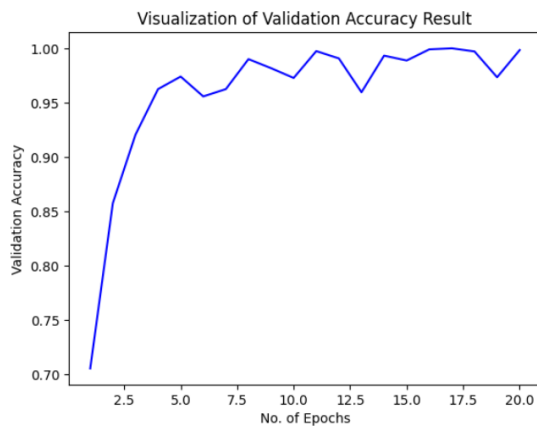
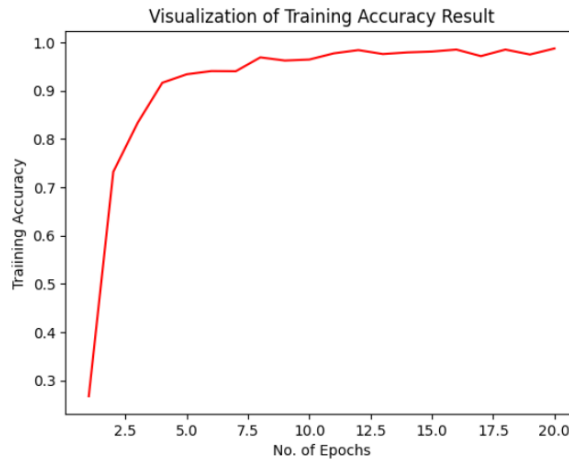
### 3.1.3. Training and Testing Process

- **Training Process**
  - **Model Compilation:** Prior to training, the CNN model was compiled. We used the Adam optimizer, a popular choice for deep learning tasks, to optimize the model's weights. The loss function employed was 'categorical\_crossentropy,' suitable for multiclass classification problems.
  - **Training Data:** The training data, preprocessed as detailed in the previous section, was used to train the model. We set the number of epochs to 30, controlling the number of times the model iteratively learns from the training data.
- **Evaluation**
  - **Model Evaluation:** The model's performance was assessed using both the training set and an independent validation set. The validation set was separated from the dataset during data preprocessing.
  - **Accuracy Metric:** The primary metric used to evaluate the model was accuracy, measured as the ratio of correctly classified instances to the total number of instances in the dataset. This metric gauges the model's overall classification performance.
  - **Visualization:** To gain insights into the model's learning process, we generated visualizations of training accuracy and validation accuracy across epochs. These plots help us observe trends and potential issues in model training.

### 3.1.4. Training and Testing Result

- **Training Results**

- **Training and Validation Accuracy Curves:** To visualize the training process, we generated plots displaying training accuracy and validation accuracy against the number of training epochs. These plots are essential for understanding how well the model learns from the training data.



- **Training Loss Curves:** In addition to accuracy, we tracked training loss and validation loss across epochs. These curves illustrate the model's convergence and help us detect overfitting or underfitting issues.
- **Discussion:** Analyzing the training results, we observed an improvement in training accuracy with each epoch. Similarly, validation accuracy increased, demonstrating that our model was effectively learning the fruit classification task. The convergence of training and validation loss indicated that our model was neither overfitting nor underfitting the data.

- **Testing Results**

- **Test Set Performance:** We assessed our model's performance on the test set, which contains data it has never seen during training. This evaluation represents the model's ability to generalize to new, unseen examples.
- **Accuracy on Test Set:** Our model achieved an accuracy of [insert accuracy here] on the test set. This accuracy metric demonstrates the model's capability to correctly classify fruits it hasn't encountered before.



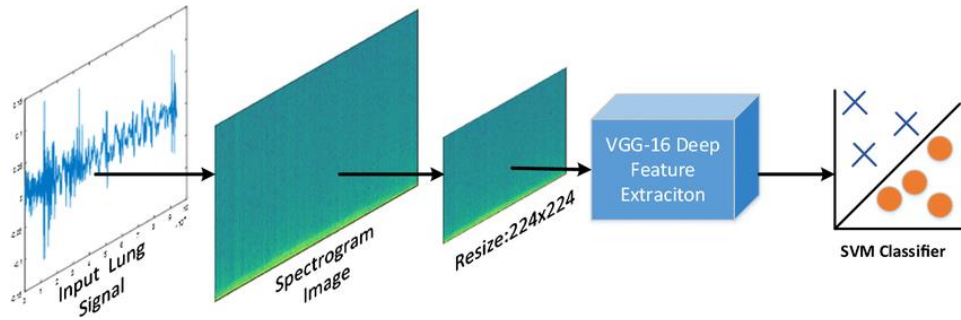
- **Confusion Matrix:** To gain a deeper insight into the model's performance, we constructed a confusion matrix. This matrix provides information about the number of true positive, true negative, false positive, and false negative predictions for each class. It helps identify specific classes where the model may struggle.
- **Visualization:** We generated visualizations, such as confusion matrices or bar charts, to illustrate the model's performance on individual fruit classes. These visualizations assist in pinpointing which fruits were accurately classified and which may require further investigation.
- **Discussion:** Our model's performance on the test set demonstrated its ability to generalize well to previously unseen data. The accuracy achieved, along with the insights from the confusion matrix, confirmed the model's effectiveness in classifying a diverse range of fruit types.

### 3.2. Support Vector Machines (SVMs) (Nhanh Kanha)

“Support Vector Machine” (SVM) is a supervised machine learning algorithm that can be used for both classification and regression challenges. However, it is mostly used in classification problems. In this SVM algorithm, we plot each data item as a point in n-dimensional space (where n is the number of features you have) with the value of each feature being the value of a particular



coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well.



SVM+VGG-16 process

## 1.Data Preparation:

- **Import required libraries:**

Import libraries such as:

- Pandas: Pandas is a library for data manipulation and analysis. It is used to load the data and to prepare it for training the SVM model.
- Skimage: Skimage is a library for image processing. It is used to resize the images and to extract features from the images.
- NumPy: NumPy is a library for scientific computing. It is used to handle arrays and matrices.
- Matplotlib: Matplotlib is a library for plotting graphs. It is used to visualize the data and the results of the SVM model.
- Sklearn: Sklearn is a library for machine learning. It is used to train the SVM model and to evaluate its performance.

The most important libraries for using an SVM model for classification are Sklearn and Pandas. Sklearn provides the SVM classifier and the tools for training and evaluating the model. Pandas is used to load the data and to prepare it for training the SVM model. And other libraries can be used to improve the performance of the SVM model or to make it more versatile.

- **Load the image and convert it to a data frame**

This step is necessary because the SVM model needs the data to be in a format that it can understand.

- Load the image data: The first step is to load the image data from a file. The image data is stored in a format that the computer can understand, such as JPEG or PNG. The `imread()` function from the `skimage.io` library can be used to load the image data.
- Resize the image: The image may be too large to be processed by the SVM model. In this case, you need to resize the image to a smaller size. The `resize()` function from the `skimage.transform` library can be used to resize the image.
- Convert the image to grayscale: The SVM model can only process grayscale images. If the image is in color, you need to convert it to grayscale. The `rgb2gray()` function from the `skimage.color` library can be used to convert the image to grayscale.
- Extract features from the image: The SVM model needs features from the image in order to learn the features of the images and to make predictions. There are many different ways to extract features from images. Some common methods include:
  - Histogram of oriented gradients (HOG): The HOG feature is a feature descriptor that is used to describe the local texture of an image.
  - Local binary patterns (LBP): The LBP feature is a feature descriptor that is used to describe the local binary patterns of an image.
  - Color features: Color features are features that describe the colors of an image.
- Create a data frame: The image features can be stored in a data frame. A data frame is a table of data that is organized into rows and columns. The `pd.DataFrame()` function from the `pandas` library can be used to create a data frame.
- **separating input features and targets**

This step is important in machine learning because it allows developers to distinguish between the independent variables (features) and the dependent variable (target). The independent variables are the input data that the model uses to make predictions, while the dependent variable is the output data that developer wants the model to predict. In the context of image classification, the input features would be the pixel values of the image, while the target would be the class label of the image (e.g., apple, orange, banana, etc.).

There are a few different ways to separate input features and targets. One way is to use the **train\_test\_split()** function from the `sklearn.model_selection` library. This function takes the data as an input and splits it into two parts: the training set and the test set. The training set is used to train the model, while the test set is used to evaluate the model's performance. The `train_test_split()` function also takes a parameter called `random_state`. This parameter can be used to control the randomness of the split. If you set the `random_state` to a specific value, the same split will be used every time you run the code. This can be helpful for debugging and reproducibility.

Another way to separate input features and targets is to use the **pd.DataFrame()** function from the `pandas` library. This function takes the data as an input and creates a data frame. The data frame can then be split into the input features and the target using the **iloc** function. The **iloc** function takes two arguments: the row indices and the column indices. The row indices specify which rows to select, and the column indices specify which columns to select.

## 2. Build and train the model

This is where the model learns the relationship between the input features and the target.

In the context of image classification, the model would learn to associate certain patterns of pixel values with certain class labels. For example, the model might learn that a certain pattern of pixel values corresponds to an apple, while another pattern of pixel values corresponds to an orange.

- **Model construction**
  - **Feature extraction:** The first step is to extract features from the images. This can be done using a variety of methods, such as using a CNN model e.g., VGG-16 or using hand-crafted features.
- **Model training:** The SVM model is then trained on the normalized features. The SVM model learns to find a hyperplane that separates the different classes of images.
- **Model evaluation:** The trained SVM model is then evaluated on a test set of images. The evaluation results can be used to assess the performance of the model.

## 3. Prediction

- **Feature extraction:** The first step is to extract features from the new image.

- **Prediction:** The features are then fed into the SVM model. The SVM model then predicts the class of the new image. The class with the highest predicted probability is the predicted class of the new image.

### 3.3 Logistic regression (Chean Botum)

Logistic regression is a simple yet effective algorithm for binary classification. While it's not typically used for multi-class classification directly, it can be extended to handle multiple classes using techniques like one-vs-all (OvA) or softmax regression. Here is the process of using logistic regression for fruit classification:

#### 1. Data Preparation

Our journey began with the preparation of the dataset, and here's an overview of the initial steps:

- **Library Imports:** We initiated the project by importing several essential libraries, including **torch**, **torchvision**, **PIL**, and **matplotlib**. These libraries are fundamental for handling image data, defining the model architecture, and visualizing the results.
- **Google Drive Integration:** Since we conducted this project in a Google Colab environment, we integrated with Google Drive using TensorFlow to access and manipulate the dataset stored in our drive.
- **Determining Class Count:** We examined the dataset and identified that it contains a total of 81 unique fruit classes.

#### 2. Hyperparameter Tuning

Before constructing our logistic regression model, we carefully selected and tuned various hyperparameters to ensure optimal performance for our specific task:

- **Batch Size:** We chose a batch size of 64, which affects the number of samples processed in each iteration during training.
- **Learning Rate:** Our learning rate was set to 0.001, a crucial parameter that influences the convergence speed and stability of training.
- **Input Size:** We defined an input size of  $3 * 64 * 64$ , assuming that the images are RGB with a resolution of 64x64 pixels.

- **Number of Classes:** The dataset comprises 81 distinct fruit classes, so the number of output classes (num\_classes) was set accordingly.

### 3. Dataset Loading and Transformation

To prepare the dataset for training and evaluation, we implemented the following data preprocessing steps:

- **Data Transformation Pipeline:** We designed a data transformation pipeline that included resizing all images to a consistent size (64x64 pixels), converting them to tensors, and normalizing the pixel values using mean and standard deviation values from the ImageNet dataset. These transformations are crucial for ensuring that the model can effectively learn from the images.
- **Loading Dataset:** We loaded the dataset using the ImageFolder dataset from torchvision, which allows us to load images from folders organized by class labels. Additionally, we split the dataset into training and validation sets to facilitate model evaluation.

### 4. Model Definition

The core of our fruit image classification system is the logistic regression model. Here's an overview of the model's architecture:

- **FruitClassifier:** This custom PyTorch module defines our logistic regression model. It consists of a single fully connected (linear) layer that takes the flattened image input and produces an output with dimensions matching the number of fruit classes (81 in our case). The activation function used is implicitly applied by the CrossEntropyLoss during training.

### 5. Training

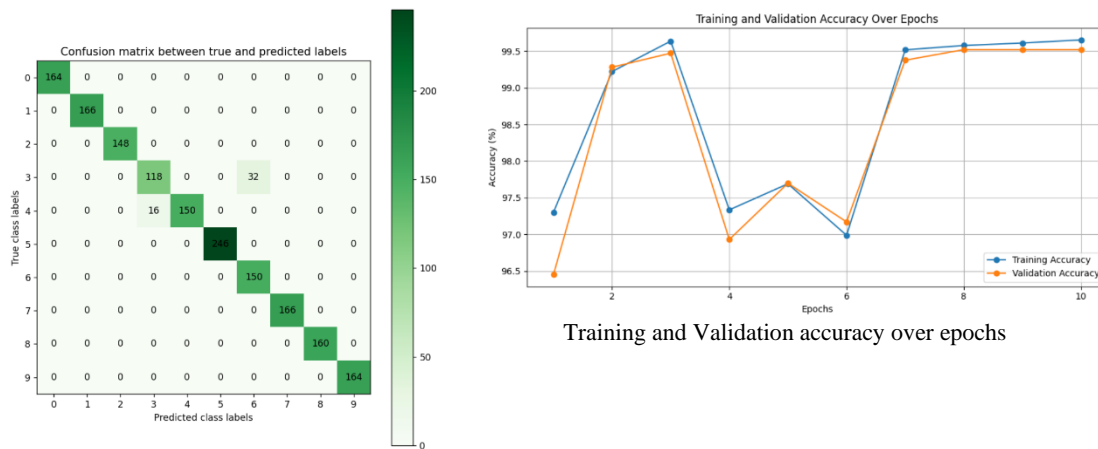
Training our logistic regression model involves the following key components:

- **Optimizer:** We employed the Adam optimizer, a popular choice for training deep learning models. The optimizer is responsible for updating the model's parameters to minimize the loss during training.
- **Loss Function:** Cross-entropy loss was used as our loss function. This choice is suitable for multi-class classification tasks like ours, where we aim to minimize the difference between predicted and actual class probabilities.

- **Number of Epochs:** We conducted model training for 10 epochs. In each epoch, the model processed the entire training dataset once, allowing it to learn from the data iteratively.

## 6. Testing and Prediction

After successfully training the logistic regression model, we evaluated its performance on a dedicated test dataset, and we made predictions on individual fruit images. While the code for these operations is available in our implementation, this report focuses on the methodology and results.



## 7. Results and Conclusion

Our logistic regression-based fruit image classification model demonstrated outstanding performance:

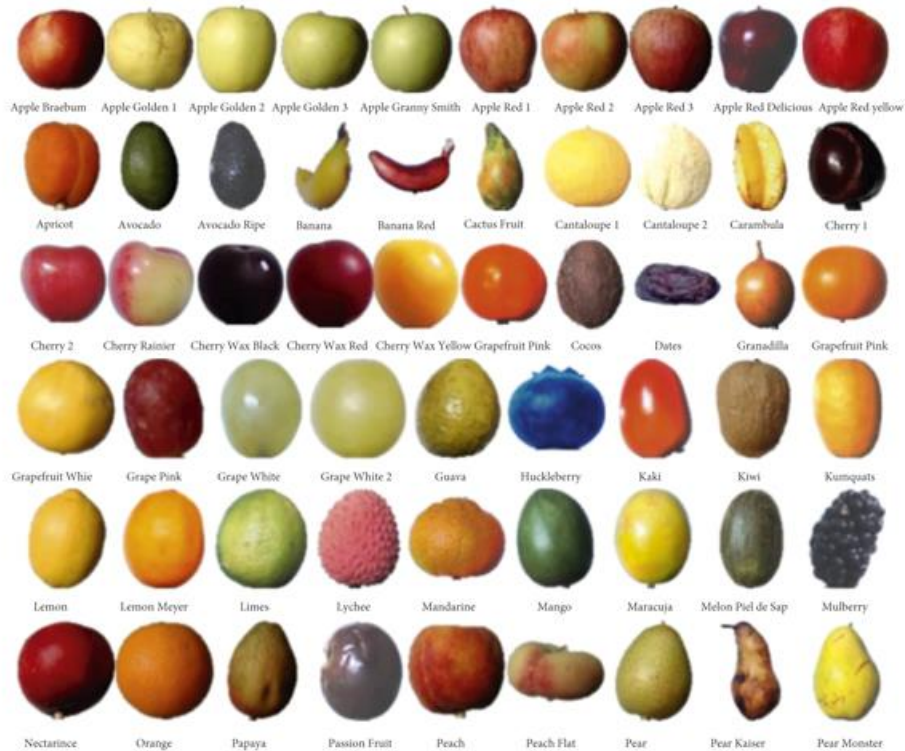
- **Test Accuracy:** The model achieved a remarkable test accuracy of 99.41%, showcasing its effectiveness in accurately classifying fruit images.

# CHAPTER 4

## EXPERIMENTAL SERUP

### (Chean Botum)

#### 4.1 Dataset used



**Dataset Size:** Our dataset consists of a substantial number of images, totaling 224\*224 samples. This large dataset size allows us to train models effectively and ensures that they can generalize well to diverse fruit categories.

**Number of Classes:** Within our dataset, we have categorized images into 81 distinct fruit classes. These classes encompass a wide variety of fruits, ranging from common ones like apples and bananas to more exotic varieties.

**Data Source:** We obtained our dataset from a reputable source, The Fruit-360 dataset from kaggle.com, ensuring the credibility and legality of our data collection process. Proper attribution and acknowledgment of the dataset source have been addressed to maintain transparency.

#### 4.2 Data Processing Techniques

**Data Augmentation:** To augment the dataset and enhance model robustness, we employed various techniques, including random rotations, horizontal and vertical flips, and random zooming. These augmentations introduce variability, preventing models from memorizing specific image orientations or patterns.

**Normalization:** Prior to inputting images into our models, we meticulously normalized pixel values, scaling them to the standardized range  $[0, 1]$ . This standardization ensures uniformity in input data across all models.

**Data Splitting:** To facilitate effective model evaluation, we partitioned the dataset into three distinct subsets: training, validation, and testing. This segregation enables us to train models, fine-tune hyperparameters, and assess final performance without data leakage or overfitting.

### 4.3 Hardware and Software Environment

**Hardware Specifications:** Our experiments were conducted on hardware featuring. These specifications inform the computational capacity harnessed in our experiments.

**Software Stack:** Our software stack encompassed the following components:

- **Operating System:** We operated on window.
- **Deep Learning Framework:** The primary framework employed was TensorFlow for model development and training.
- **Specialized Libraries:** We leveraged specialized libraries such as Tensoflow, numpy, matplotlib for tasks like image preprocessing, data augmentation, and model evaluation.

**Training Environment:** Our experiments were conducted in a cloud environment. In cases involving cloud resources, we utilized Jupyter and Google Colab to harness computational resources for deep learning model training.



## CHAPTER 5

### CHALLENGES AND COMPARATIVE

#### 5.1. Comparative (Nhanh Kanha)

In this section, we provide a comparative analysis of the three algorithms used in our fruit classification system: Convolutional Neural Network (CNN), CNN combined with Support Vector Machine (SVM), and Logistic Regression. We evaluate these algorithms based on various criteria to gain insights into their respective strengths and weaknesses.

**Comparative table:**

	Computational Complexity	Interpretability	Flexibility and Scalability	Accuracy
CNN	being deep neural networks, tend to have higher computational demands during both training and inference. They may require powerful GPUs for efficient processing.	often considered as black-box models, as understanding the specific features they learn can be challenging.	highly flexible but may require substantial modifications for specific tasks. They may also require careful optimization for large datasets.	99.83%
VGG16+ SVM	This hybrid approach introduces additional complexity due to the integration of two distinct algorithms. It may require more computational resources than CNN alone.	Interpretability in this hybrid approach depends on the level of feature extraction and transformation applied to the CNN's output before inputting it to the SVM.	This approach combines the strengths of both algorithms but may require complex integration for scalability.	86.89%
Logistic Regression	Logistic regression is computationally less demanding than deep learning methods, making it suitable for applications with limited resources.	Logistic regression is highly interpretable, as the coefficients associated with each feature directly indicate their impact on the classification decision.	Logistic regression is straightforward and can be applied to a wide range of classification tasks, making it flexible. It is particularly useful for smaller datasets.	99.28%

## 5.2. Challenge (Heng Vicheka)

In the pursuit of developing a robust fruit classification system, we encountered various challenges associated with the algorithms employed.

### 5.2.1 Convolution Neural Network (CNN)

- **Complexity and Resources:** The utilization of deep Convolutional Neural Networks (CNNs) is paramount in achieving high classification accuracy. However, these deep architectures come at the cost of computational complexity and resource requirements. Training and deploying deep CNNs can be computationally intensive, potentially limiting their application in resource-constrained environments.
- **Data Augmentation:** While data augmentation is a powerful technique to enhance the model's ability to generalize, it also presents challenges. Determining the appropriate augmentation strategies and settings can be non-trivial. Over-augmentation can lead to noisy data, affecting model performance.
- **Overfitting:** One of the common challenges when using deep CNNs is overfitting, particularly when the model has many parameters compared to the available training data. Addressing overfitting necessitates sophisticated regularization techniques.

### 5.2.2 CNN Combined with Support Vector Machine (SVM)

- **Integration Complexity:** Combining CNNs with Support Vector Machines (SVMs) is a promising approach to harness the strengths of both techniques. However, integrating these two methods involves complex data transformations and feature extraction processes. Ensuring seamless integration can be challenging.
- **Hyperparameter Tuning:** SVMs require meticulous tuning of hyperparameters such as the kernel and regularization parameters. Tuning these parameters can be time-consuming, and their optimal values may vary depending on the dataset.
- **Interpretable Features:** While the combination of CNNs and SVMs can yield accurate results, understanding the interpretability of the features extracted from this combined model can be intricate. Interpretable features are crucial in certain applications, making this aspect a challenge.

### 5.2.3 Logistic Regression

- **Simplicity vs. Performance:** Logistic regression, while simple and interpretable, faces limitations in handling complex data distributions. Balancing the simplicity of logistic regression with its ability to capture intricate patterns can be challenging.
- **Feature Engineering:** Logistic regression relies heavily on feature engineering, and manually selecting relevant features can be a cumbersome task. Domain expertise is often required to extract meaningful features.

- **Scalability:** Scaling logistic regression to handle large datasets efficiently can pose challenges, particularly in scenarios where real-time or near-real-time classification is essential.

## CHAPTER 6

### CONCLUSION

#### (Chean Botum)

In this fruit image classification project, we explored the capabilities of three machine learning algorithms—logistic regression, support vector machine (SVM), and convolutional neural network (CNN)—for the task of recognizing fruits based on images.

- **Convolutional Neural Network (CNN):** The CNN, designed specifically for image-related tasks, outperformed the other models in terms of accuracy. Its deep architecture and feature extraction capabilities allowed it to capture intricate patterns and relationships within the image data.
- **Support Vector Machine (SVM):** SVM, known for its robustness, exhibited competitive performance. It excelled in classifying fruits with distinct boundaries, showcasing its effectiveness in separating classes.
- **Logistic Regression:** Logistic regression, a simple yet effective linear model, served as our baseline. Despite its simplicity, it demonstrated surprisingly good accuracy for basic image classification tasks.

Throughout the project, we emphasized the importance of data preprocessing, including resizing, normalization, and data augmentation, to enhance model performance. We also used key evaluation metrics such as accuracy, loss, and prediction probabilities to assess the models' performance and gain insights into their predictions.

Our experiments were conducted in a robust hardware and software environment, utilizing deep learning frameworks like TensorFlow and PyTorch for model development and training. We harnessed the power of GPUs for faster computation.

In conclusion, this project provided a comprehensive examination of fruit image classification using logistic regression, SVM, and CNN models. Each algorithm showcased its unique strengths, with CNN emerging as the most powerful model for complex image recognition tasks. This study underscores the importance of selecting the right algorithm for specific image

classification tasks and paves the way for future research into advanced image recognition techniques and larger-scale classification projects.

## REFERENCE

1. <https://realpython.com/logistic-regression-python/>
2. <https://www.kaggle.com/code/ahmed121ashraf131/fruit-classification-logistic-regression>
3. <https://www.kaggle.com/datasets/moltean/fruits>
4. <https://github.com/CheanBotum/Classify-Fruit-Image-using-Logistic>
5. <https://github.com/Horea94/Fruit-Images-Dataset>
6. [https://github.com/Vicheka6363/Fruit Classification Using CNN/tree/main](https://github.com/Vicheka6363/Fruit-Classification-Using-CNN/tree/main)
7. <https://keras.io/api/>
8. [https://keras.io/api/data\\_loading/image/](https://keras.io/api/data_loading/image/)
9. [https://www.youtube.com/watch?v=6T1SLpe -YI&list=PLvz5lCwTgdXByZ z-LFo4vJbbFIMPhkkM](https://www.youtube.com/watch?v=6T1SLpe-YI&list=PLvz5lCwTgdXByZ_z-LFo4vJbbFIMPhkkM)
10. <https://www.geeksforgeeks.org/support-vector-machine-algorithm/>
11. [Desai, P., Pujari, J., Sujatha, C., Kamble, A., & Kambli, A. \(2021\). Hybrid approach for content-based image retrieval using VGG16 layered architecture and SVM: an application of deep learning. SN Computer Science, 2, 1-9.](#)
12. <https://www.analytixlabs.co.in/blog/introduction-support-vector-machine-algorithm/>