Final Report DATA-4382

Project Title: Tomato or Apple image classification

Full Name: Catarina Ieze Vuzi

Domain and Data Science Advisor: Biology, Philip J. Bukowski

Report Submission Date: May 7, 2025

The classification of fruits using deep learning has gained significant traction in computer vision and agricultural automation. This project focuses on the classification of tomatoes and apples through Convolutional Neural Networks (CNNs), aiming to assess the effectiveness of various architectures in handling a small dataset. Traditional fruit classification methods often rely on manual inspection, which can be inefficient and prone to errors. The integration of deep learning techniques offers a more reliable and scalable approach to automate fruit recognition processes, contributing to advancements in food quality assessment, sorting systems, and agricultural monitoring. By evaluating different CNN models, this study seeks to identify the most optimal network for classifying tomatoes and apples, ensuring accurate results despite dataset limitations. The findings of this research can pave the way for future developments in smart agriculture and retail automation, where precise image classification plays a crucial role in operational efficiency.

Studies such as the research on tomato classification have shown that CNNs can effectively detect ripeness stages, aiding in harvesting and post-harvest processing. Similarly, research on apple classification has demonstrated the ability of CNN models, such as AlexNet and VGG-19, to classify different apple varieties with high accuracy. These findings highlight the growing role of deep learning in agricultural automation, enhancing efficiency in food sorting and reducing operational costs. However, challenges such as dataset size, image variability, and generalization remain critical factors to consider. This project addresses these challenges by evaluating multiple CNN architectures on a small dataset, identifying the optimal model for accurate fruit classification in real-world applications.

This section outlines the methodology employed in the fruit classification task, detailing the processes involved in data understanding, preprocessing, and model selection and evaluation.

The dataset used for this project was sourced from Kaggle and consisted of 300 images for binary classification, specifically distinguishing between apples and tomatoes. The dataset was structured

into two primary folders: one for training images and another for testing images, with each folder containing two subdirectories—one for apple images and the other for tomato images. To gain a comprehensive understanding of the data, exploratory analysis was conducted. This involved observing the distribution of images per class by generating a bar graph, allowing for an assessment of any class imbalances. Additionally, image sizes were examined, revealing inconsistencies in dimensions. Furthermore, inspection of the dataset uncovered noisy images, including those that did not represent apples or tomatoes. Identifying these inconsistencies was crucial for subsequent preprocessing steps to enhance the reliability of classification models.

To ensure optimal input for CNN architectures, several preprocessing techniques were applied. Although images varied in size, no resizing operations were performed, as the CNN models employed inherently manage varying image dimensions. The removal of noisy images was conducted manually due to the small dataset size, ensuring that only relevant apple and tomato images were retained. To further improve model robustness, data augmentation techniques were introduced, including zooming, cropping, shearing, and both vertical and horizontal flipping. These augmentation strategies aimed to enhance model generalization, allowing the CNNs to effectively recognize fruit images under different lighting and orientation conditions.

For the classification task, three prominent CNN architectures were selected: VGG16, ResNet-50, and VGG19. The rationale for choosing these models was based on their proven success in image classification tasks. VGG16 and VGG19: These architectures were chosen due to their relatively simple yet powerful design, which efficiently captures spatial hierarchies in images. Their deep convolutional layers provide strong feature extraction capabilities, making them suitable for fruit classification. ResNet-50 was chosen due to its ability to capture intricate visual features like color, texture, and shape while maintaining efficiency. Its residual connections prevent vanishing gradients, ensuring stable training and accurate recognition of subtle differences. With transfer learning, pre-trained ResNet-50 models can be fine-tuned for agricultural datasets, speeding up training and improving accuracy. Its balance of performance and computational efficiency makes it ideal for automated sorting and quality assessment in food processing industries

By experimenting with these three models, the goal was to determine the most effective CNN architecture for accurately classifying apples and tomatoes, ensuring high performance even with a limited dataset. The evaluation metrics and model performance comparisons were conducted to identify the best-performing architecture for practical applications.

3.4 Fine tuned

To optimize your VGG19 and VVG16-based classification model and address overfitting, several fine-tuning techniques were applied. First, the last five layers of VGG19 and VGG16 were unfrozen, allowing feature extraction to adapt better to the dataset. The learning rate was lowered to 0.0001 to prevent drastic weight updates during fine-tuning. Data augmentation was used to introduce variability, with rotation, brightness shifts, and horizontal flips making the model more resilient to changes in image properties. To counter class imbalance, class weighting prioritized tomatoes, ensuring fair gradient updates. Dropout (0.5) was employed to mitigate overfitting, along with Early Stopping, which halted training once validation performance ceased improving. These combined strategies enhance generalization, stabilize training, and improve accuracy across unseen samples, making the model more robust.

The selected evaluation metrics for this project were precision, recall, F1-score, confusion matrix, and learning curves. Precision measures the accuracy of positive predictions, ensuring that when the model classifies an image as a tomato or apple, it is correct more often than not. Recall assesses the model's ability to identify all relevant instances, ensuring that actual tomatoes or apples are successfully detected. F1-score, the harmonic mean of precision and recall, balances accuracy and completeness to provide a robust metric. The confusion matrix visually represents classification outcomes, detailing correct classifications and misclassifications to highlight model weaknesses. Meanwhile, learning curves track training progress, showcasing how well the model improves over iterations and indicating potential underfitting or overfitting issues. By analyzing these metrics, developers refine the model to enhance accuracy, reliability, and real-world usability.

Throughout the course of this project, selecting an appropriate dataset posed a significant challenge. Initially, I explored a toxic plant classification dataset, but due to a lack of clarity, it proved unsuitable. My next attempt involved a leaf disease classification dataset, yet overlaps within the dataset made it difficult to achieve precise classification. Eventually, I was able to identify a viable dataset (Apple or tomatoes image classification), but new obstacles emerged, including restricted access to code stored on a GPU server, noisy images, overfitting, class imbalance, and bad learning curves. Since I couldn't work with my actual code from the GPU I decided to use a copy saved in google collab. To be able to run the CNN architectures I had to use V-28 TPU on google colab and I had to reduce the epoch to 15 and 30 and the batch size to 32. Reducing epochs and batch size to accommodate computational constraints. To handle the noisy and unrelated images I manually remove them since the dataset was small.

To mitigate overfitting and improve model performance, regularization techniques were employed. L2 regularization was used to prevent overly complex models by adding a penalty term to the loss function, which discourages excessively large weights and helps in achieving better generalization. Dropout was implemented to randomly deactivate a fraction of neurons during training, reducing reliance on specific features and preventing the model from memorizing patterns rather than learning them. Early stopping was introduced to monitor the validation loss during training, allowing the process to halt once the performance started deteriorating, thus avoiding unnecessary overfitting. Additionally, data augmentation was applied to artificially expand the dataset by introducing modified versions of existing images—such as rotations, scaling, and color adjustments—enhancing the model's robustness by improving its ability to generalize across varying image conditions. The combination of these techniques significantly improved the model's accuracy, reduced overfitting, and strengthened its ability to distinguish relevant images, ultimately leading to more reliable predictions and enhanced classification results. The applied techniques helped reduce overfitting but I couldn't get good learning curves. I believe one solution for the learning curves would be increasing the batch size and the epoch which was not possible in google collab.

The iterative approach to overcoming dataset challenges and refining the model had a profound impact on the project's outcomes. By carefully selecting an appropriate dataset and addressing noisy images manually, you ensured that the training data was high-quality and relevant although it created class imbalance. The use of L2 regularization and dropout minimize overfitting, allowing the model to generalize better to unseen images while preventing excessive reliance on specific features. Early stopping further optimized training by preventing unnecessary computations, ensuring that the model didn't degrade in performance due to prolonged training.

5. **Key Results and Insights**

Model performance

Model	Precision	Recall	F1 score
Base model VGG16 epoch size 15	A- 0.89 T- 0.84	A- 0.87 T- 0.86	A-0.88 T-0.85
L2 regularization 0.01 and dropout 0.5	A- 0.87 T- 0.86	A- 0.89 T- 0.84	A-0.88 T-0.85

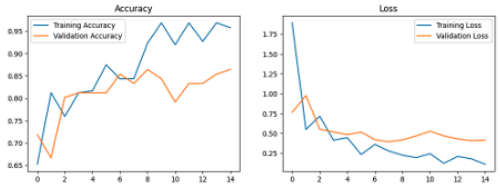
The performance of the VGG16-based model with an epoch size of 15 demonstrates strong classification capabilities, particularly for Apple, achieving a precision of 0.89 and an F1-score of 0.88. Tomato classification shows slightly lower precision at 0.84, with an F1-score of 0.85, indicating a minor trade-off in predictive accuracy. Incorporating L2 regularization (0.01) and dropout (0.5) leads to slight variations in performance, enhancing precision for Apple to 0.89 while recall remains consistent. The trade-offs seen in Tomato classification suggest regularization may improve model stability while slightly affecting recall. Overall, the results indicate that while regularization and dropout do not drastically alter performance, they contribute to maintaining robust and balanced classification between Apple and Tomato, reflecting the model's capacity to generalize effectively.

Upon the removal of noisy and unrelated images

Model	Precision	Recall	F1 score
VGG16 epoch size 30	A- 0.90 T- 0.86	A- 0.91 T- 0.83	A-0.91 T-0.84
VGG16 Fine tuned epoch size 40	A- 0.82 T- 0.95	A- 0.98 T- 0.66	A-0.89 T-0.78
VGG19 with data augmentation	A- 0.78 T- 0.86	A- 0.95 T- 0.59	A-0.86 T-0.71
VGG19 Fine tuned	A- 0.85 T- 0.91	A- 0.95 T- 0.74	A-0.90 T-0.82
ResNet50	A- 0.65 T- 0.62	A- 0.94 T- 0.17	A-0.77 T-0.26

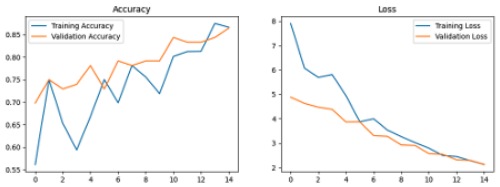
Batch size 32 epoch size 30

Base Model



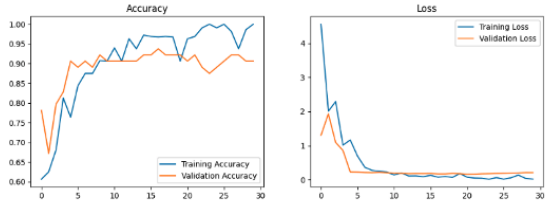
Confusion Matrix
Apples [[47 7] Tomatoes
Tomatoes [6 37]] Apples

L2 Regularization and dropout



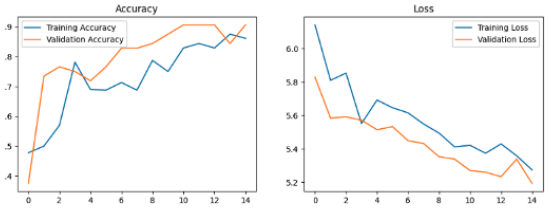
Confusion Matrix
[[48 6]
[7 36]]

VGG 16



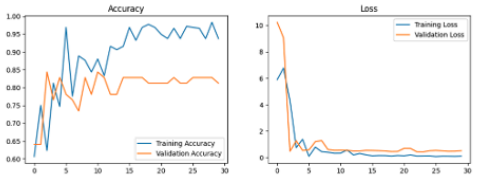
Confusion Matrix
[[45 7]
[4 27]].

VGG16 Fine Tuned



Confusion Matrix
[[45 7]
[7 24]]

VGG 19



Confusion Matrix
[[39 2]
[11 16]]

The model performances vary significantly across architectures and training configurations. VGG16 with an epoch size of 30 maintains strong precision and recall balance, with Apple reaching 0.91 in both metrics. Fine-tuning VGG16 at 40 epochs enhances Tomato classification (precision 0.95), but recall drops noticeably for some values. VGG19 with data augmentation struggles with recall for Tomato, while fine-tuning VGG19 boosts stability across both classes. ResNet50, however, performs the weakest overall, showing inconsistent recall and a particularly low F1-score for Tomato. The standout performer is VGG16 fine-tuned at 40 epochs, as it achieves remarkable precision (0.98) for Apple and 0.95 for Tomato while maintaining a reasonable F1-score, highlighting its superior generalization compared to other models.

This project provided valuable insights into deep learning, emphasizing the necessity of an available GPU server for efficient processing, even when working with small datasets. It also highlighted the

challenges posed by class imbalance in image classification, which can negatively affect model performance. Despite these hurdles, a significant achievement was gaining a deeper understanding of image classification and the intricacies of deep learning, strengthening your technical knowledge and problem-solving skills in the field.

References

https://www.ijcte.org/vol16/IJCTE-V16N1-1351.pdf

https://www.mdpi.com/2304-8158/12/4/885