International Journal of Computing and Digital Systems
2024, VOL. 17, NO. 1, 1–10

http://dx.doi.org/10.12785/ijcds/XXXXXX

A Development of Deep Learning-Based Lemon Quality Detection Through Peel Analysis

Nurulshamiza Zulkifli[1], Tajul Rosli Razak[2,*] and Mohammad Hafiz bin Ismail[3]

[1,2]*School of Computing Sciences, College of Computing, Informatics and Mathematics, Universiti Teknologi MARA, Shah Alam, Malaysia*
[3]*School of Computing Sciences, College of Computing, Informatics and Mathematics, Universiti Teknologi MARA, Arau, Malaysia*

Received 15 August 2024, Revised 15 October 2024, Accepted xxx xxx 2024

Abstract: This paper addresses the critical challenge of lemon quality detection by introducing a novel deep learning system that analyzes lemon peel characteristics for precise quality assessment. With the rising global demand for high-quality produce, the agricultural industry requires more reliable and efficient methods for evaluating fruit quality. This study presents a convolutional neural network (CNN) trained on a comprehensive dataset of lemon images sourced from the Kaggle platform, enabling the classification of lemons into high or low quality based on specific peel attributes. Visualization tools have been integrated to enhance the interpretability and practical application of the system, allowing users to understand better the model's output and the factors influencing quality classification. Utilizing the waterfall model within the software development life cycle (SDLC), the research encompasses essential phases such as data preprocessing, model development, graphical user interface (GUI) implementation, system functionality testing, and a thorough evaluation of the system's performance. The results indicate that the proposed system significantly enhances the accuracy and reliability of lemon quality detection, addressing the limitations of existing methods, which often lack precision and consistency. This innovation represents a significant advancement in agricultural technology, with potential applications extending beyond lemons to other crops requiring stringent quality control. This research provides valuable insights by integrating deep learning methodologies and visualization tools into the farming sector. It lays the groundwork for future developments to refine and expand these techniques. The paper concludes by discussing the system's advantages and limitations, offering a roadmap for future research that could lead to even more sophisticated approaches to agricultural quality assessment.

Keywords: Convolutional Neural Network, Deep Learning, Classification, Lemon Detection.

1. INTRODUCTION

Lemons, renowned for their economic, nutritional, and health benefits, stand among the primary citrus varieties alongside oranges and mandarins [1]. The variation in lemon characteristics, including shape, size, color, texture, and juice content, plays a crucial role in consumer preferences [2]. The lemon peel's color is essential, significantly influencing purchasing decisions. However, despite their widespread consumption, the full potential of lemons, particularly their nutrient-rich peels, remains underappreciated [3].

In recent years, with the development of agricultural planting technology, the control technology of lemon scabs, skirt rot, and pest damage often encountered by lemons has significantly improved [4]. In this study, we propose using deep learning techniques to classify lemons based on their peel quality and develop a robust and accurate model to automate the lemon detection process.

However, several problems have been found, such as inefficient tools for citrus orchards and the inhibition of HLB citrus areas [5]. Another issue is using traditional methods like manual grading or sensory evaluation. This conventional method is seen as problematic due to its limitations and weaknesses. One notable concern is that some human eyes may lack the sensitivity for accurate assessments. Therefore, some researchers [6] utilised machine vision and image processing technologies because the human eye lacks sensitivity. Hence, there is a need to develop an efficient system for lemon quality detection to benefit farmers, wholesalers, retailers, and consumers by ensuring the quality of lemons and reducing the economic losses due to low-quality products.

In response to the growing demand for reliable and efficient quality assessment methods, this study by [7]proposes developing a deep learning-based system for lemon quality detection. Deep learning, a branch of artificial intelligence,

E-mail address: tajulrosli@uitm.edu.my

has garnered attention for its ability to extract meaningful patterns and features from complex datasets. Its flexibility and scalability make it well-suited for tasks like fruit quality assessment, where intricate visual features play a crucial role.

Motivated by the need for more accurate and efficient quality assessment methods, we focus on developing a deep learning-based system for lemon quality detection, specifically emphasising peel analysis. Leveraging a convolutional neural network (CNN) trained on a comprehensive dataset of lemon images, the system aims to accurately classify lemons into good and bad-quality categories. By tackling the vital problem of accurately detecting lemon quality, this paper helps to improve farming methods by incorporating deep learning techniques. The paper emphasises the significance of evaluating lemon quality and the transformative power of deep learning techniques in farming.

The remaining sections of the paper will delve into the background of lemon quality detection, the proposed deep learning-based method, the experimental setup and results, and finally, the conclusion summarising the findings and discussing future directions.

2. BACKGROUND

This section aims to provide essential background information about the study subjects: lemon quality detection, deep learning approaches, and other similar works.

A. Lemon Quality Detection

Deep learning techniques have emerged as powerful tools for analyzing lemon peel characteristics, significantly enhancing quality detection in the citrus industry. These methods leverage advanced algorithms to automatically extract features from images of lemon peels, which is crucial for identifying quality attributes such as color, texture, and surface defects. For instance, convolutional neural networks (CNNs) have been successfully employed to classify citrus diseases and defects, demonstrating their ability to learn from raw image data without the need for extensive manual feature extraction [8], [9], [10]. This capability is particularly beneficial in lemon peel analysis, where subtle variations can indicate quality issues.

Applying deep learning in this domain improves the accuracy of quality detection and facilitates the early identification of potential problems, such as diseases or spoilage. Research has shown that deep learning models can achieve high classification accuracy, even with limited datasets, by employing data augmentation and transfer learning [11], [12]. This is particularly relevant for lemon peel analysis, where various factors, including environmental conditions and handling practices, can compromise the quality of the fruit. Producers can implement more effective quality control measures by utilising deep learning, ensuring that only high-quality lemons reach consumers.

In addition, the recognition of fruit and crop epidermis typically involves three primary research stages: conventional digital image processing, CNN, and feature processing within neural networks [13]. Machine vision technology, renowned for its rapid detection speed and non-destructive nature, is widely utilised for fruit surface detection, minimising product damage. Moreover, as reported by [14], transfer learning has demonstrated success in various domains, including agriculture, where it is employed for tasks such as image classification based on visual characteristics. Despite challenges like limited training time, model accuracy, and a lack of research on lemon defect identification, integrating deep learning and transfer learning offers a promising approach to addressing this issue, bringing forth new perspectives and potential solutions.

B. Deep Learning Approach

This section presents an analysis focusing on the different deep-learning approaches used to detect lemon quality based on lemon peel. The comparison is conducted among CNN, VGG-16, and YOLOV3 techniques. Table I presents a comprehensive overview of the advantages and disadvantages associated with each method.

Based on the detailed comparison presented in Table I, it becomes evident that the most appropriate and effective deep learning approach for the lemon quality detection system is the CNN. When trained on large datasets, CNN has demonstrated exceptional performance, resulting in an outstanding testing accuracy rate of 99.84%.

Convolutional Neural Networks (CNNs) have emerged as a powerful tool for detecting and classifying the quality of lemons through image analysis, mainly focusing on the characteristics of lemon peels. The effectiveness of CNNs in this domain can be attributed to their ability to automatically extract features from images, eliminating the need for manual feature engineering, a significant advantage in image classification tasks [17].

The application of CNNs extends beyond mere classification; they are also effective in detecting defects and assessing the overall quality of citrus fruits. For example, a study focused on detecting various peel conditions in citrus fruits using hyperspectral imaging and CNNs underscores the potential of these networks to improve the marketability of fruits by identifying defects early in the supply chain [18]. Additionally, CNNs have been successfully employed in classifying lemons based on visual features extracted through transfer learning, further validating their robustness in recognising quality parameters [19].

In summary, recent research supports CNN's effectiveness in detecting and classifying lemon quality. It demonstrates their ability to analyse image data efficiently and accurately. CNNs' application in various studies illustrates the potential for CNNs to revolutionise quality assessment in the agricultural sector, particularly for citrus fruits.

TABLE I. Compares each deep learning approach (CNN, VGG-16, and YOLOV3)

Deep Learning	Advantages	Disadvantages	Best Accuracy	Citation
CNN	When used in farming automation, CNN can enhance crop harvest and improve output and productivity when appropriately designed. CNN is also known to perform predictions relatively faster than other algorithm CNN uses relatively little processing compared to other image classification algorithms.	In each split, the same number of samples from each class that CNN needs to convert will be a stack of alternated Conv2D (with relu activation) and MaxPooling2D layers. CNN has a bigger image and a more complex problem to be solved.	99.84%	[15]
Geometry Group 16-based (VGG16-based)	Compared to CNN, VGG-16 has three steps to build a model to get specific accuracy, and VGG-16 gets high accuracy in lemon recognition by using brightness compensation and classification models. VGG-16 uses cross-validation to represent model performance more accurately.	In the classification of lemon defect, it is hard for VGG-16 to extract defective features that are very complex. VGG-16 needs more complex and robust models, but powerful models usually take time to train.	95.44%	[4]
YOLOV3	YOLOV3 can enhance the propagation of features and needs fewer parameters, which helps to achieve higher accuracy and speed. The YOLOV3 method has superior detection performance and can more efficiently recognize and locate lemons in a natural environment.	YOLOV3 leads to the loss of some vital feature information, affecting the accuracy and speed of the detection when using DarkNet53 as the backbone network. YOLOV3 achieves lower accuracy compared to CNN andVGG-16.	90.60%	[16]

C. *VIsualization Approach*

The visualization approach is crucial in interpreting research outputs across various fields, enhancing comprehension, communication, and complex data analysis. By transforming raw data into visual formats, researchers can uncover patterns, trends, and insights that may not be readily apparent in textual or numerical forms. This capability is vital in data-intensive disciplines such as genomics, where visualization aids in hypothesis generation and interpretation of findings [20]. For instance, it emphasizes that visual tools serve as a bridge between algorithmic methods and the cognitive abilities of researchers, thereby facilitating a deeper understanding of genomic data.

Moreover, the effectiveness of visualization in research output interpretation is supported by the assertion that visual representations significantly enhance cognitive processing. Various visualization techniques can reveal hidden structures within scholarly data, allowing researchers better to understand the relationships and networks in their data sets [21]. This aligns with the findings of those who argue that visualizations provide overviews of general patterns and trends, thereby enabling the discovery of underlying structures within complex data [22]. The cognitive interpretability of visualizations is further highlighted by those who discuss how the design and presentation of visual data can influence interpretation outcomes, underscoring the importance of thoughtful visualization strategies [23].

3. PROPOSED MODEL ARCHITECTURE

This study proposes three layers that work together in the system. It describes how these parts communicate with the other parts. The system architecture is graphically shown in Figure 1.

The architecture described involves several components and processes (as seen in Figure 1). First and foremost, the front end serves as a web-based interface through which the user interacts with the system. It is responsible for presenting the application's interface to the user and handling their interactions. For instance, popular frameworks such as React and Vue.js are employed to build dynamic user interfaces that can efficiently handle user interactions and update the display without requiring a full page reload [24]. These frameworks support component-based development, which allows developers to create reusable UI components, thus improving the maintainability and scalability of the application [25].

In this study, the front end is typically built using three leading technologies: HTML, CSS, and JavaScript. The backend is usually developed using various programming languages like Java, Python, or JavaScript. The web server is a software component that runs on the backend. It receives requests from users' browsers, processes them, and sends back the appropriate responses. The server communicates with the front end using Hypertext Transfer Protocol (HTTP) requests and responses.

Moreover, these web-based applications, like CNN, involve deep learning to train and test the data. This model processes data and makes predictions based on learning from a training dataset. Once the data is trained, the system moves on to classification. Here, models are used to classify or categorise new or unseen data based on the patterns and relationships they have learned during the training phase. Finally, the system produces the result, a description or any other output.

4. MODEL DEVELOPMENT

The model development architecture had several layers: convolution, max pooling, flattening, and dense layers. Each layer served a distinct purpose in the learning process. Figures 2 and 3 show the code architecture of a convolutional neural network (CNN) model.

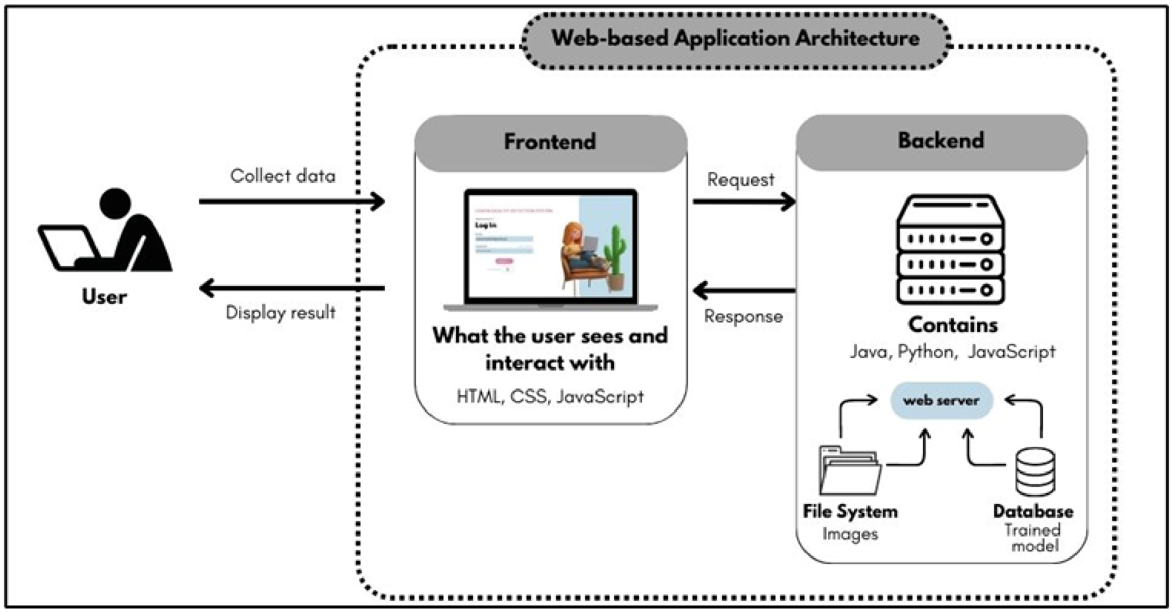
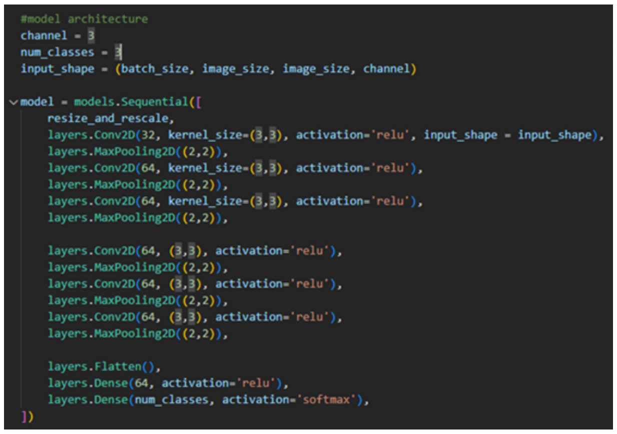


Figure 1. Proposed Architecture.



```
#model architecture
channel = 3
num_classes = 3
input_shape = (batch_size, image_size, image_size, channel)

model = models.Sequential([
    resize_and_rescale,
    layers.Conv2D(32, kernel_size=(3,3), activation='relu', input_shape = input_shape),
    layers.MaxPooling2D((2,2)),
    layers.Conv2D(64, kernel_size=(3,3), activation='relu'),
    layers.MaxPooling2D((2,2)),
    layers.Conv2D(64, kernel_size=(3,3), activation='relu'),
    layers.MaxPooling2D((2,2)),

    layers.Conv2D(64, (3,3), activation='relu'),
    layers.MaxPooling2D((2,2)),
    layers.Conv2D(64, (3,3), activation='relu'),
    layers.MaxPooling2D((2,2)),
    layers.Conv2D(64, (3,3), activation='relu'),
    layers.MaxPooling2D((2,2)),

    layers.Flatten(),
    layers.Dense(64, activation='relu'),
    layers.Dense(num_classes, activation='softmax'),
])
```

Figure 2. The model layer code.

Based on Figures 2 and 3 provide the code of the model, and the model architecture is structured as a sequence of layers. It begins with a preprocessing step utilising the "resize_and_rescale" layer, which includes data augmentation and resizing operations. The subsequent structure involves a series of convolutional layers, Conv2D with rectified linear unit or ReLU activation, and max-pooling layers, MaxPooling2D. This configuration is repeated to extract hierarchical features from the input images progressively. In subsequent stages, the convolutional layers' filters increase from 32 to 64.

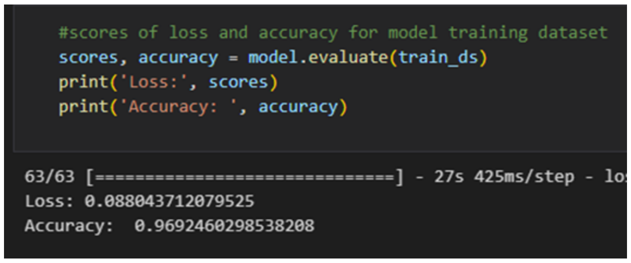
Following the convolutional layers, a flattening layer or flatten is employed to reshape the output into a one-dimensional array. Then, followed by two dense or fully connected layers with ReLU activation, contributing to the feature learning process. The final dense layer consists of "num_classes", employing a softmax activation function for multiclass classification. This last layer enables the model to make predictions and classify input images into predefined categories. Overall, this architecture is designed for image classification tasks with three classes, and the sequential arrangement facilitates a systematic flow of operations during both feature extraction and classification stages.

5. EXPERIMENTS

In this section, model experiments were carried out to check the differences between the epochs with the highest accuracy [26]. Below are the experiments that focused on the details between 10, 30, and 50 epochs.

A. *Experiment one with 10 epochs.*

Model experiment one was running with 10 epochs, and the result showed different values of training and testing. Below is the figure of the training result.



```
#scores of loss and accuracy for model training dataset
scores, accuracy = model.evaluate(train_ds)
print('Loss:', scores)
print('Accuracy: ', accuracy)

63/63 [==============================] - 27s 425ms/step - lo
Loss: 0.088043712079525
Accuracy:  0.9692460298538208
```

Figure 5. Result of training set.

Figure 4 shows the training and validation accuracy and profound learning loss over time. The training accuracy for

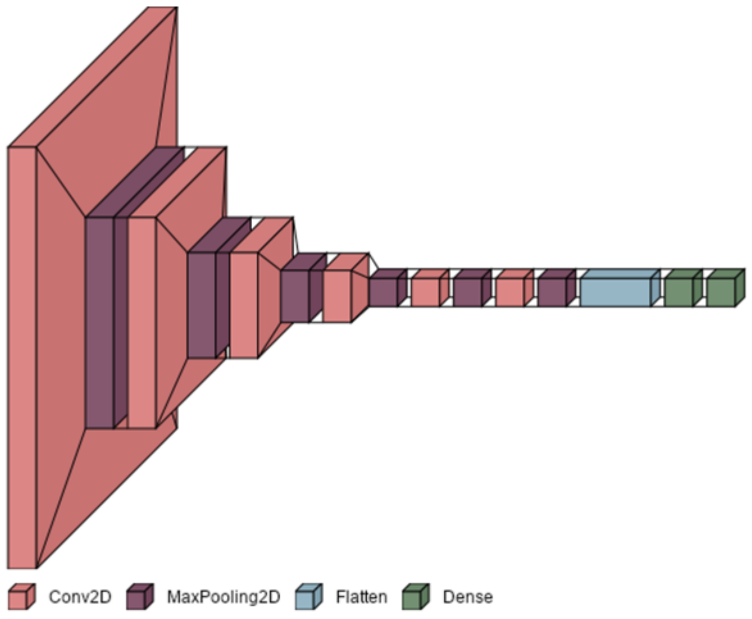


Figure 3. System model layer image.

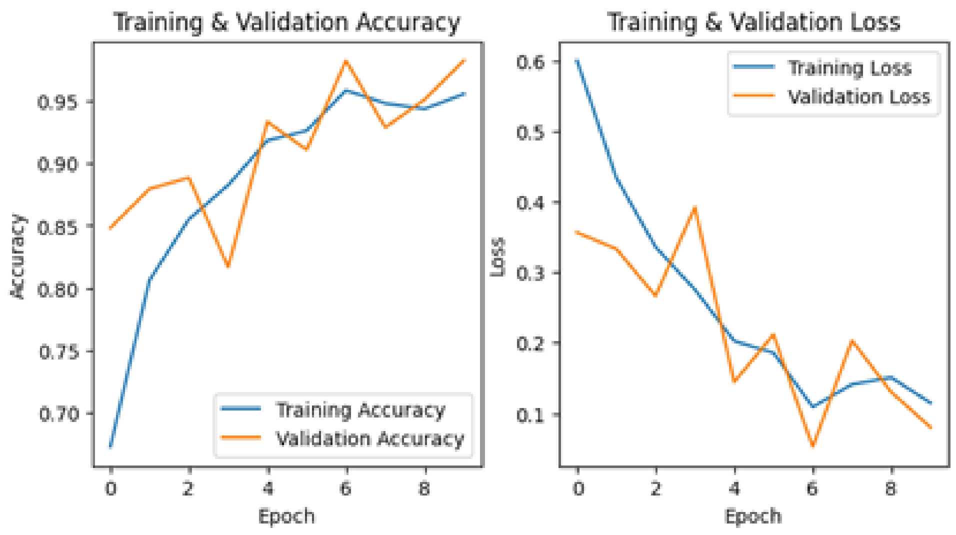


Figure 4. Graph of the training set with 10 epochs.

experiment one starts at below 0.70 and increases to about 0.95. In contrast, the validation accuracy begins at around 0.85 and rises to about 1.0. However, the training loss shows that the training and validation loss of the model over 10 epochs. The training loss starts at around 0.6 and decreases almost at 0.1.

In contrast, the validation loss starts between 0.4 and 0.3 and decreases about 0.0. The result of the training is shown in Figure 5. Figure 5 shows the training accuracy and loss results. As shown, the result of training loss was 0.088 while the accuracy was 0.968, or it was written as 96.8%.

B. Experiment one with 30 epochs.

The second experiment ran with 30 epochs, and the result showed different values of training and testing. Below is the figure of the training result.

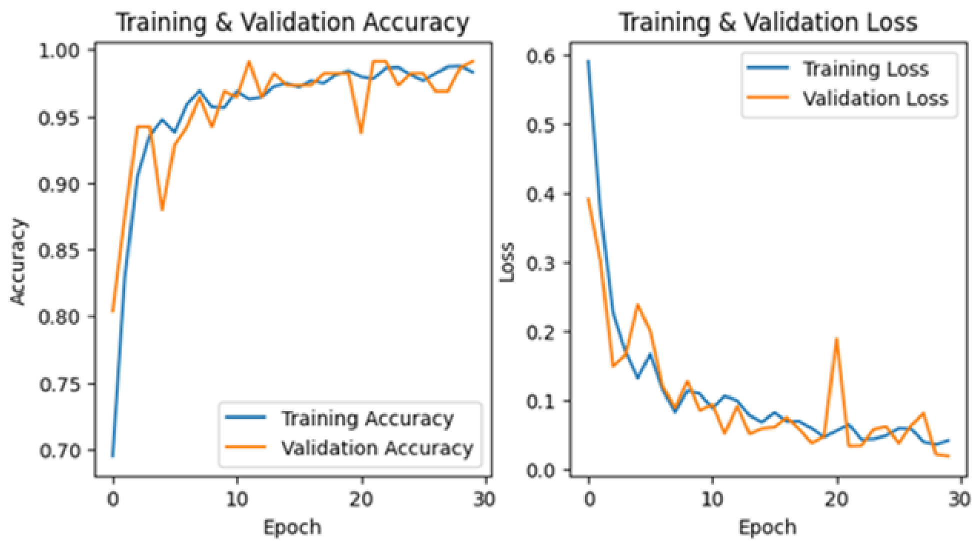
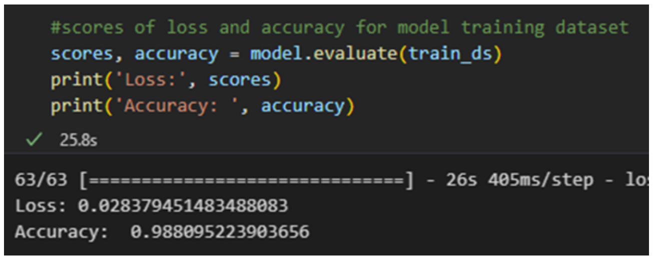
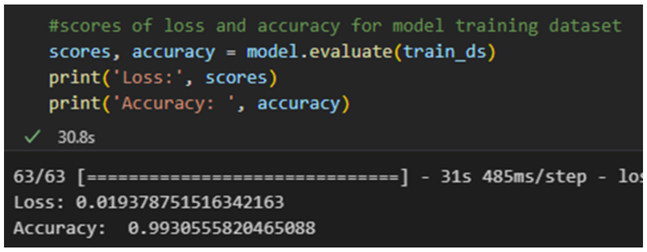


Figure 6. Graph of the training set with 30 epochs.



```
#scores of loss and accuracy for model training dataset
scores, accuracy = model.evaluate(train_ds)
print('Loss:', scores)
print('Accuracy: ', accuracy)
✓ 25.8s
63/63 [==============================] - 26s 405ms/step - lo
Loss: 0.028379451483488083
Accuracy:  0.988095223903656
```

Figure 7. Result of training set.



```
#scores of loss and accuracy for model training dataset
scores, accuracy = model.evaluate(train_ds)
print('Loss:', scores)
print('Accuracy: ', accuracy)
✓ 30.8s
63/63 [==============================] - 31s 485ms/step - lo
Loss: 0.019378751516342163
Accuracy:  0.9930555820465088
```

Figure 9. Result of training set.

Figure 6 shows the training and validation accuracy and profound learning loss over time. Likewise, in the previous experiment, the training accuracy starts at around 0.7. It increases to about 1.0, while the validation accuracy starts at around 0.8 and increases to about 0.95. However, for the training loss, it shows that the training and validation loss of the model is over 30 epochs. The training loss starts at around 0.6 and decreases almost at 0.0.

In contrast, the validation loss starts at around 0.4 and decreases to about 0.0. The result of the training is shown in the figure 7. Figure 7 shows the result of training accuracy and loss. As shown, the result of training loss was 0.028 while the accuracy was 0.988, or it was written at 98.8%.

C. Experiment one with 50 epochs.

Next, model experiment one was running with 50 epochs, and the result showed different values of training and testing. Below is the figure of the training result.

Figure 8 shows the training and validation accuracy and profound learning loss over time. The training accuracy for experiment three starts at around 0.65 and increases to about 1.0. At the same time, the validation accuracy begins at around 0.83 and increases to about 1.0, too. However, for the training loss, it shows that the training and validation loss of the model is over 50 epochs. The training loss starts at around 0.6 and decreases almost at 0.0.

In contrast, the validation loss starts between 0.4 and 0.3 and decreases about 0.0. The result of the training is shown in the figure 9. Figure 9 shows the result of training accuracy and loss. As shown, the result of training loss was 0.019 while the accuracy was 0.993, or it was written as 99.3%.

6. VISUALIZATION OF RESULTS

In this section, we have developed a lemon quality detection system and completed the overall interface design, emphasising the clarity and interpretability of the results, especially for end users. The system features a web-based interface with four key pages–Landing, Home, Detection, and Guidelines—designed for easy use and quick accessibility.

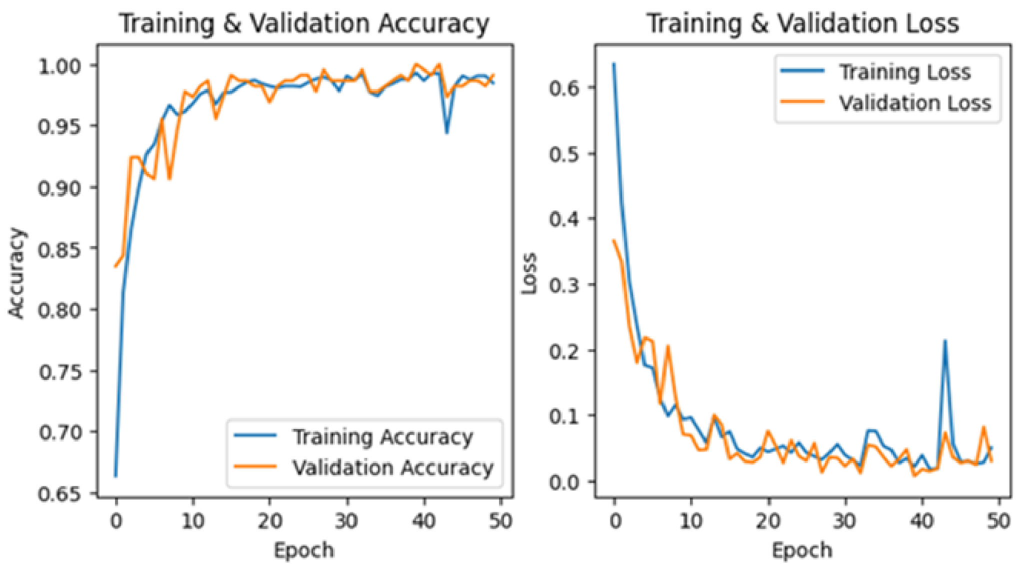


Figure 8. Graph of the training set with 50 epochs.

A. Landing Page

The landing page serves as the initial gateway to the web-based lemon quality detection system, designed to guide users seamlessly into the application. To begin using the system, users must strategically click the prominently displayed start button for easy access. As shown in Figure 10, the landing page features a minimalistic and clean design, focusing on user-friendliness and clarity.

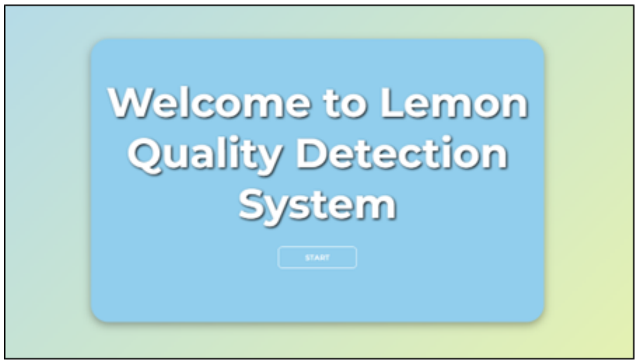
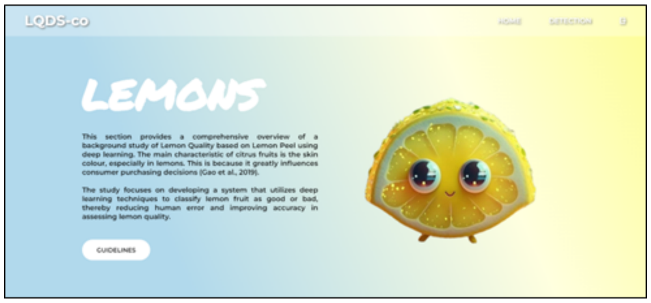


Figure 10. Landing page.

The page includes the system's title, which establishes the purpose of the application, and a well-defined start button. This button is crucial, as it directs users to the home page and signals the transition from introductory information to active system use. Users are immediately taken to the home page by clicking the start button to engage with the system's functionalities fully.

B. Home Page

The home page comprises several sections, including a header and a segment providing a comprehensive overview of lemon quality. Designed with simplicity, the page uses a clean white background with black text to ensure readability and ease of use. It also includes a 'GUIDELINES' button for users needing assistance navigating the system. The figure below illustrates the layout of the home page.



This section provides a comprehensive overview of a background study of Lemon Quality based on Lemon Peel using deep learning. The main characteristic of citrus fruits is the skin colour, especially in lemons. This is because it greatly influences consumer purchasing decisions (Gao et al., 2019).

The study focuses on developing a system that utilizes deep learning techniques to classify lemon fruit as good or bad, thereby reducing human error and improving accuracy in assessing lemon quality.

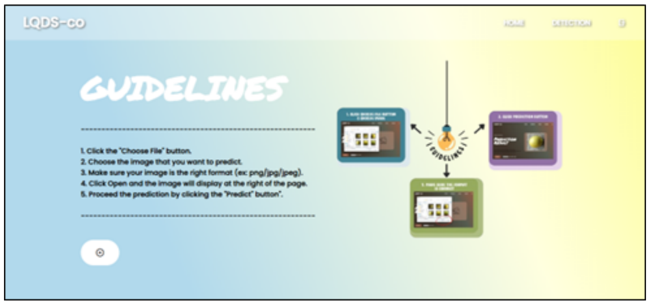
Figure 11. Home page.

As shown in Figure 11, the home page serves as an introduction to the lemon quality detection system, known as LQDS-CO. It briefly overviews the system's purpose, underlying technology, and potential benefits. The design aims to make users feel comfortable and informed immediately, providing essential information to help them understand how the system works.

The 'GUIDELINES' button is prominently placed to offer users access to step-by-step instructions on using the system, ensuring they can easily understand the process and maximise their interaction with the application.

C. Guideline Page

This section introduces the guidelines page, which ensures that users are fully prepared to use the lemon quality detection system before making any predictions. Designed with a user-centric approach, the guidelines page provides clear, step-by-step instructions that outline the entire detection process, from uploading an image for analysis to interpreting the results.



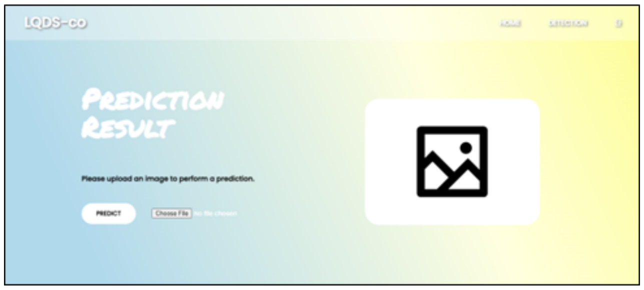
1. Click the "Choose File" button.
2. Choose the image that you want to predict.
3. Make sure your image is the right format (ex: png/jpg/jpeg).
4. Click Open and the image will display at the right of the page.
5. Proceed the prediction by clicking the "Predict" button".

Figure 12. Guidelines page.

As illustrated in Figure 12, the page is structured with simplicity and clarity, minimising ambiguity and helping users navigate the system efficiently and confidently. By offering a comprehensive overview of the necessary steps, the guidelines page is a vital reference to reduce user errors and enhance the overall experience. After reviewing the instructions, users can seamlessly return to the home page by clicking the back button, ensuring a smooth transition between the instructional content and the system's main functionality.

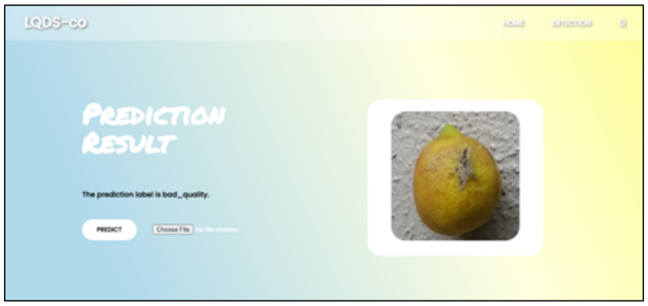
D. Detection Page

This section presents the main page of the lemon quality detection system, which serves as the core of the application by managing both the prediction of image quality and data visualisation. As depicted in Figures 13 and 14, the detection page shows the interface before and after an image is uploaded. The page is designed to guide users smoothly through the prediction process, ensuring that all necessary actions are easily accessible.



Please upload an image to perform a prediction.

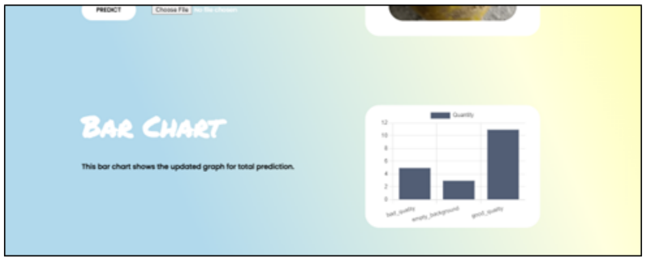
Figure 13. Before prediction.



The prediction label is bad_quality.

Figure 14. After prediction.

The primary focus of the detection page is the prediction results section. This section includes a prompt—' Please upload an image to perform a prediction'—and provides two interactive buttons: 'PREDICT' and 'Choose File.' These elements create a clear call to action, encouraging users to upload an image to see if the LQDS-CO system can successfully predict its quality. The straightforward design facilitates user interaction and highlights the system's core functionality.



This bar chart shows the updated graph for total prediction.

Figure 15. Result.

Furthermore, Figure 15 displays a bar chart graph visually representing the system's predictions. This graph updates dynamically with each new image prediction, providing a real-time overview of the number of predictions made. This data visualisation feature is essential, as it not only tracks the usage of the system but also helps users better understand the analysis outcomes by presenting them in a clear, graphical format.

7. DISCUSSION

The experiments conducted, as outlined in Sections 5-A to 5-C, demonstrate a clear and consistent trend of improvement in training loss and accuracy across increasing epochs. This trend highlights the model's capacity to progressively learn and adapt as it is exposed to more training data. Notably, as the number of epochs increases, there is a marked reduction in training loss, suggesting that the model becomes more proficient at fitting the training data. This is particularly evident in Experiment 3, where the model was trained for 50 epochs. Here, we observed the most substantial reduction in training loss, reaching a minimal value of 0.019, coupled with the highest accuracy of 99.3%.

This outcome indicates that the model's performance in Experiment 3 is optimal for minimising loss and maximising accuracy.

However, while these results are promising, it is crucial to acknowledge that the high accuracy and low training loss observed in Experiment 3 may primarily reflect the model's performance on the training data. Without further validation, there is a risk of overfitting, where the model performs exceptionally well on the training set but fails to generalise to unseen data. Therefore, additional experiments, including cross-validation and testing on an independent dataset, are necessary to confirm that the model's performance is not merely a reflection of its ability to memorise the training data but can also generalise to new, unseen instances.

Including visualization in this paper enhances the findings' interpretability and practical application. By providing explicit, graphical representations of training loss and accuracy trends across different epochs, the visualizations make complex data more accessible and comprehensible to readers. This not only aids in understanding the model's performance but also allows users to see how the method operates in a practical context, bridging the gap between theoretical results and real-world application.

However, while these visualizations are valuable, they also have limitations. They may simplify complex processes, potentially leading to misinterpretations if users rely solely on them without understanding the underlying data. Additionally, the choice of data and parameters for visualization might exclude essential nuances, which could affect the model's performance in different scenarios. Therefore, while visualizations are a powerful tool for enhancing usability, they should be complemented by thoroughly examining the data and methods to ensure a comprehensive understanding.

Overall, based on the discussion, the critical contributions underscore optimizing model performance, ensuring practical applicability, and maintaining rigorous standards for interpretability and validation, as follows:

1) *Demonstration of Model Improvement:* The paper shows a clear trend of improvement in training loss and accuracy as the number of epochs increases, highlighting the model's capacity to learn and adapt over time. Experiment 3, with 50 epochs, achieved the best performance, indicating optimal training for minimizing loss and maximizing accuracy.
2) *Identification of Overfitting Risk:* The discussion acknowledges the potential risk of overfitting despite the high accuracy and low training loss, emphasizing the need for further validation, including cross-validation and testing on independent datasets, to ensure the model's generalizability.
3) *Enhancement of Interpretability through Visualization:* Including visualization features makes complex data more accessible, helping readers and users understand the model's performance trends and practical application of the method.
4) *Critical Evaluation of Visualization:* The discussion critically assesses the limitations of visualizations, cautioning against over-reliance on them without understanding the underlying data and processes. It highlights the importance of complementing visual tools with thoroughly examining data and methods to avoid potential misinterpretations.

8. CONCLUSIONS

In conclusion, we proposed comprehensive system development, which starts with meticulous data preprocessing, including splitting, resizing, and rescaling, to prepare the dataset for model training. By employing data augmentation techniques, we enhance dataset diversity, which is crucial for robust model performance. Our model development focuses on Convolutional Neural Network (CNN) layers. This is followed by designing and implementing a user-friendly graphical interface, covering various components such as the landing, home, guidelines, and detection pages. Detailed code snippets and Flask integration ensure seamless interaction between the front and back end, with functional testing confirming the system's reliability.

In the future, we plan to apply our approach to a complex real-world case study to validate our system further. This future work will provide valuable insights and advancements for researchers and practitioners, contributing significantly to developing and applying machine learning and web-based interfaces.

ACKNOWLEDGEMENTS

The authors acknowledge the support of the School of Computing Sciences, College of Computing, Informatics and Mathematics, Universiti Teknologi MARA, Malaysia, for providing the necessary facilities to conduct this study.

References

[1] X. Dong, Y. Hu, Y. Li, and Z. Zhou, "The maturity degree, phenolic compounds and antioxidant activity of Eureka lemon [Citrus limon (L.) Burm. f.]: A negative correlation between total phenolic content, antioxidant capacity and soluble solid content," *Scientia Horticulturae*, vol. 243, pp. 281–289, 1 2019.

[2] M. Ladaniya, "Fruit morphology, anatomy and physiology," *Citrus Fruit: Biology, Technology and Evaluation. Academic Press, Elsevier*, pp. 103–124, 2010.

[3] A. O. Ademosun, "Enrichment with citrus peels as a strategy for improving the health benefits and nutritional value of breakfast cereals: A review," *Human Nutrition & Metabolism*, vol. 36, p. 200239, 6 2024.

[4] H. Li, Y. Shui, S. Li, Y. Xing, Q. Xu, X. Li, H. Lin, Q. Wang, H. Yang, W. Li, and Z. Che, "Quality of fresh cut lemon during different temperature as affected by chitosan coating with clove oil," *International Journal of Food Properties*, vol. 23, no. 1, pp. 1214–1230, 1 2020.

[5] Y. Lan, Z. Huang, X. Deng, Z. Zhu, H. Huang, Z. Zheng, B. Lian, G. Zeng, and Z. Tong, "Comparison of machine learning methods for citrus greening detection on UAV multispectral images," *Computers and Electronics in Agriculture*, vol. 171, p. 105234, 4 2020.

[6] R. Chellappa, S. Sirohey, and C. L. Wilson, "Human and Machine Recognition of Faces: A Survey," *Proceedings of the IEEE*, vol. 83, no. 5, pp. 705–741, 1995.

[7] Y. Zhang, L. Deng, H. Zhu, W. Wang, Z. Ren, Q. Zhou, S. Lu, S. Sun, Z. Zhu, J. M. Gorriz, and S. Wang, "Deep learning in food category recognition," *Information Fusion*, vol. 98, p. 101859, 10 2023.

[8] S. Janarthan, S. Thuseethan, S. Rajasegarar, Q. Lyu, Y. Zheng, and J. Yearwood, "Deep Metric Learning Based Citrus Disease Classification With Sparse Data," *IEEE Access*, vol. 8, pp. 162 588–162 600, 2020. [Online]. Available: https://doi.org/10.1109/access.2020.3021487

[9] S. Xing and M. Lee, "Classification Accuracy Improvement for Small-Size Citrus Pests and Diseases Using Bridge Connections in Deep Neural Networks," *Sensors*, vol. 20, no. 17, pp. 1–16, 9 2020. [Online]. Available: https://doi.org/10.3390/s20174992

[10] A. Elaraby, W. Hamdy, and S. Alanazi, "Classification of Citrus Diseases Using Optimization Deep Learning Approach," *Computational Intelligence and Neuroscience*, vol. 2022, 2022. [Online]. Available: https://doi.org/10.1155/2022/9153207

[11] M. Z. Ur Rehman, F. Ahmed, M. A. Khan, U. Tariq, S. S. Jamal, J. Ahmad, and I. Hussain, "Classification of Citrus Plant Diseases Using Deep Transfer Learning," *Computers, Materials &Amp; Continua*, vol. 70, no. 1, pp. 1401–1417, 2021. [Online]. Available: https://doi.org/10.32604/cmc.2022.019046

[12] M. Zhang, S. Liu, F. Yang, and J. Liu, "Classification of Canker on Small Datasets Using Improved Deep Convolutional Generative Adversarial Networks," *IEEE Access*, vol. 7, pp. 49 680–49 690, 2019. [Online]. Available: https://doi.org/10.1109/access.2019.2900327

[13] D. S. Joseph, P. M. Pawar, and R. Pramanik, "Intelligent plant disease diagnosis using convolutional neural network: a review," *Multimedia Tools and Applications*, vol. 82, no. 14, pp. 21 415–21 481, 6 2023.

[14] L. A. Aldakhil and A. A. Almutairi, "Multi-Fruit Classification and Grading Using a Same-Domain Transfer Learning Approach," *IEEE Access*, 2024.

[15] J. Y. Alzamily, S. Salim, and A. Naser, "Lemon Classification Using Deep Learning," 2020.

[16] G. Li, X. Huang, J. Ai, Z. Yi, and W. Xie, "Lemon-YOLO: An efficient object detection method for lemons in the natural environment," *IET Image Processing*, vol. 15, no. 9, pp. 1998–2009, 7 2021.

[17] A. Taner, Y. B. Öztekin, and H. Duran, "Performance Analysis of Deep Learning CNN Models for Variety Classification in Hazelnut," *Sustainability*, vol. 13, no. 12, 6 2021.

[18] P. K. Yadav, T. Burks, Q. Frederick, J. Qin, M. Kim, and M. A. Ritenour, "Citrus disease detection using convolution neural network generated features and Softmax classifier on hyperspectral image data," *Frontiers in Plant Science*, vol. 13, p. 1043712, 12 2022.

[19] Y. He, T. Zhu, M. Wang, H. Lu, Y. He, T. Zhu, M. Wang, and H. Lu, "On Lemon Defect Recognition with Visual Feature Extraction and Transfers Learning," *Journal of Data Analysis and Information Processing*, vol. 9, no. 4, pp. 233–248, 9 2021.

[20] S. Nusrat, T. Harbig, and N. Gehlenborg, "Tasks, Techniques, and Tools for Genomic Data Visualization," *Computer Graphics Forum*, vol. 38, no. 3, pp. 781–805, 2019. [Online]. Available: https://doi.org/10.1111/cgf.13727

[21] J. Liu, T. Tang, W. Wang, B. Xu, X. Kong, and F. Xia, "A Survey of Scholarly Data Visualization," *IEEE Access*, vol. 6, pp. 19 205–19 221, 3 2018. [Online]. Available: https://doi.org/10.1109/access.2018.2815030

[22] J. L. Finch and A. R. Flenner, "Using Data Visualization to Examine an Academic Library Collection," *College &Amp; Research Libraries*, vol. 77, no. 6, pp. 765–778, 11 2016. [Online]. Available: https://doi.org/10.5860/crl.77.6.765

[23] A. A. Zakharova, E. V. Vekhter, A. V. Shklyar, A. V. Krysko, and O. A. Saltykova, "Quantitative assessment of cognitive interpretability of visualization," *Scientific Visualization*, vol. 10, no. 4, pp. 145–153, 2018. [Online]. Available: https://doi.org/10.26583/sv.10.4.11

[24] C. Yang, Y. Liu, Y. Lin, and J. Yuz, "Leveraging the Power of Component-based Development for Front-End Components: Insights from a Study of React Applications (S)," *International Conferences on Software Engineering and Knowledge Engineering*, vol. 2018-July, pp. 578–581, 2018.

[25] S. Chen, U. R. Thaduri, and V. K. R. Ballamudi, "Front-End Development in React: An Overview," *Engineering International*, vol. 7, no. 2, pp. 117–126, 12 2019. [Online]. Available: https://doi.org/10.18034/ei.v7i2.662

[26] T. R. Razak, H. Jarimi, and E. Z. Ahmad, "Simplified Artificial Neural Network Configuration in R Programming for Predictive Modelling," *8th International Conference on Recent Advances and Innovations in Engineering: Empowering Computing, Analytics, and Engineering Through Digital Innovation, ICRAIE 2023*, 2023.