sustainability

MDPI

Article

The Design and Evaluation of an Orange-Fruit Detection Model in a Dynamic Environment Using a Convolutional Neural Network

Sadaf Zeeshan [1,*], Tauseef Aized [2] and Fahid Riaz [3,*]

[1] Mechanical Engineering Department, University of Central Punjab, Lahore 54590, Pakistan
[2] Mechanical Engineering Department, University of Engineering and Technology, Lahore 54890, Pakistan
[3] Mechanical Engineering Department, Abu Dhabi University, Abu Dhabi 59911, United Arab Emirates
* Correspondence: sadaf.zeeshan@ucp.edu.pk (S.Z.); fahid.riaz@adu.ac.ae (F.R.)

Abstract: Agricultural robots play a crucial role in ensuring the sustainability of agriculture. Fruit detection is an essential part of orange-harvesting robot design. Ripe oranges need to be detected accurately in an orchard so they can be successfully picked. Accurate fruit detection in the orchard is significantly hindered by the challenges posed by the illumination and occlusion of fruit. Hence, it is important to detect fruit in a dynamic environment based on real-time data. This paper proposes a deep-learning convolutional neural network model for orange-fruit detection using a universal real-time dataset, specifically designed to detect oranges in a complex dynamic environment. Data were annotated and a dataset was prepared. A Keras sequential convolutional neural network model was prepared with a convolutional layer-activation function, maximum pooling, and fully connected layers. The model was trained using the dataset then validated by the test data. The model was then assessed using the image acquired from the orchard using Kinect RGB-D camera. The model was then run and its performance evaluated. The proposed CNN model shows an accuracy of 93.8%, precision of 98%, recall of 94.8%, and F1 score of 96.5%. The accuracy was mainly affected by the occlusion of oranges and leaves in the orchard's trees. Varying illumination was another factor affecting the results. Overall, the orange-detection model presents good results and can effectively identify oranges in a complex real-time environment, like an orchard.

Keywords: convolutional neural network (CNN); fruit detection; orange fruit

Citation: Zeeshan, S.; Aized, T.; Riaz, F. The Design and Evaluation of an Orange-Fruit Detection Model in a Dynamic Environment Using a Convolutional Neural Network. *Sustainability* **2023**, *15*, 4329. https://doi.org/10.3390/su15054329

Academic Editor: Spyros Fountas

Received: 2 January 2023
Revised: 1 February 2023
Accepted: 20 February 2023
Published: 28 February 2023

1. Introduction

Detection systems play a crucial role in the design of fruit-harvesting robots. The robot should be able to detect the fruit on the tree and locate it properly. Inaccurate fruit detection can greatly reduce the robot's success rate. Fruit localization and detection has been proven to be one of the most challenging tasks for the fruit-harvesting robots. The most common problem faced in robot fruit detection is variations in illumination and occlusion of fruit on the trees. Illumination variation makes the fruit appear differently in the day and at night. It can be difficult to view fruit when dense leaves overlap. Further, fruit clusters make recognition difficult. Hence, robotic perception can pose many difficulties, especially in complex environments like fruit orchards. It is noted that in most previous studies, the datasets used do not depict the real-time data. In most datasets, single fruit images, augmented using various techniques, are used to train the data. However, in reality, the fruit on the tree often displays variations in illumination, occlusion of fruits, as well as fruit overlapping with various branches and leaves. The lack of real-time data for training often leads to more false detection.

Computer vision is commonly used in the design of fruit-harvesting robots. RGB-D cameras have proved to be of great significance in improving vision sensors, by providing the substantial and improved information needed for image recognition and localization.

Recently, machine-learning algorithms have improved the performance of fruit detection, recognition, and pose estimation using an end-effector. Machine-learning algorithms use deep-learning to train and test the data based on a dataset of fruit. A sequential model is created with layers and an assigned weightage to train and test the data for recognition of the assigned fruit.

2. Literature Review

The use of computer vision has yielded remarkable results, especially in the field of agriculture. Fruit detection is one area that has seen immense improvement. Earlier research used methods of computer vision that involved manual extraction of the features of the fruit and used machine learning to classify image features. Further, earlier fruits were identified based on a single feature, like colour, shape, or texture, using classification algorithms. Image preprocessing was carried out along with the use of a classifier. Recently, advancements in deep-learning methods saw beneficial use of, and implementation in, the field of object detection. The ability of convolutional neural networks to abstract useful features from an input image makes it a better choice for fruit detection. However, the preparation and availability of a diversified dataset can be challenging. Many researchers have used deep-learning methods for the detection of different fruits. CNN models are commonly used for fruit and plant detection in the agricultural sector [1]. Common CNN models used for fruit detection include Resnet, AlexNet, GoogLeNet and VGG. Other deep-learning models like YOLO and SSD have also been used to detect objects.

Many CNN models have been tested for detecting various fruits. Apples have been tested using VGG [2] with an F1 score of 0.791; AlexNet with 11 layers [3] was tested with an accuracy of 92.5% and detection was also carried out using two CNN models [4]. Strawberries have been detected using ResNet, with a detection accuracy of 95.78% [5] and 94% [6]. Kiwis were detected using VGG, with a harvesting success of 51% [7] and a detection rate of 90.7% [8]. Similarly, sweet peppers were detected using Resnet [9] and multiple fruits were detected using VGG with an F1 score ranging from 0.791 to 0.94 [10]. Tomatoes were detected using ResNet with a detection accuracy up to 93% [11], whereas oranges were detected using ResNet with an accuracy of 97.5% [12]. Other fruits detected using CNN models include grapes [13], papayas [14], lemons [15], bananas [16], melons [17], cucumbers [18], and others.

Fruit was also detected by researchers using YOLO and SSD models. These detect images in a single algorithm run and performed faster than the models mentioned above. Detection of oranges [19], cherries [20], apples [21], and others showed promising results using YOLO. Furthermore, fruits like mangoes [22], pineapples [23], and others were also detected using the SSD method. While the latter models are considered faster, the CNN models mentioned above, like ResNet, are far more accurate.

Fruit detection using CNN models has been implemented in various environments. Much fruit detection is carried out in a controlled environment like laboratories or greenhouses. The detection of tomatoes [24,25], cucumbers, and other fruits is carried out in a greenhouse; however, some fruit detection is carried out in fields [10,26,27]. Fruit detection is easier in a controlled environment such as a greenhouse. Detecting fruit in an orchard presents more of a challenge as the environment, illumination, and other factors play a major role in the true performance of the model. Hence, fruit detection in a complex real-time environment is encouraged to depict the true accuracy of the model. Thus, offline training and online testing should be adopted to obtain more reliable results.

In previous works on fruit detection using CNN models, many researchers obtained datasets from image repositories like Fruit-360, COCO, OpenImages, AGROSEG dataset, etc. These datasets contain images of single or multiple fruits placed on a table or a structured environment. Sakib et al. [25] used Fruit-360 as a dataset for fruit detection, whereas Duong et al. [28] used ImageNet as the dataset for their research. Further, Padhila et al. [26] used the OpenImages dataset for tomato detection. Very limited studies have been conducted on real-time datasets, such as Sa et al. [10], who used a customized dataset

with the network initialized by a pretrained ImageNet dataset. Similarly, Ganesh et al. [12] also used a customized dataset from orchard in Citra, FL, USA. In addition, Vasconez et al. [27] used real-time pictures of apples from orchards in California as fruit-detection datasets. Hence, there is a need for a more diversified dataset that includes images of fruit in real-time scenarios, not limited to a particular orchard, but using images of several orchards from around the world. This will make the dataset more universal as different orchard images may vary by topography, weather, and environment.

Similarly, many other researchers have used various datasets for fruit detection, taken from the internet. Most of the above datasets have images of a single fruit or fruits taken in a controlled environment. In reality, the fruit being detected is in an orchard, with varying illumination, fruit occlusion, and overlaps of leaves and fruit. Hence, it is important to have a dataset based on real-time orchard images. There are very few studies using real-time images. Real-time datasets can help improve the model's performance and can improve its robustness.

Furthermore, most CNN models for fruit detection are trained, verified, and tested offline. This means the process of training and testing the CNN model using a dataset is stored on a local machine or a remote server, rather than using live data from a camera or other sensor in real-time. Another approach uses offline training and online testing. This process refers to training a model using a dataset stored on a local machine or a remote server, then using it to make predictions on new, unseen data in real-time using a camera on an agricultural machine. There are very limited data available for online testing in a real-time system.

Hence, this study fills the following gaps:

- Unlike single fruit images used in many previous datasets, this study provides a diversified dataset with real-time images that have occlusion, illumination variation, and noise that replicates the real-time scenario of the fruit position in the orchard. This also ensures the results obtained by the fruit-detection model are authentic representations of a real-time situation. Furthermore, this helps to make the model more robust.
- Limited studies are available for offline training and online testing based on real-time datasets; this study fills this gap of evaluating data for a real-time system. A model that is trained only in a controlled environment may not perform well in a real-time scenario of an agricultural field with varying lighting, fruit occlusion, and other conditions. Many existing fruit-detection CNN models are not designed to work in real-time, which makes them unsuitable for use in many real-time applications such as fruit-harvesting robots.

This study will help improve overall detection, identify potential issues with the model, and make adjustments accordingly. It provides a more robust model using a diversified dataset and an offline training and online testing strategy that makes the fruit-detection model more useful and reliable for real-time applications in the orchard.

This paper contributes to scientific knowledge and novelty in the following aspects:

a. The study uses a diversified and universal real-time dataset of oranges. This dataset is not limited to fruit from one orchard in a region, such as Florida or Spain only, as in previous papers. The diversified dataset has most images from orchards in Sargodha, Punjab in Asia, as well as other orchards from Australia, Spain, Florida, and other orchards of different regions around the world. Some images are taken from Google internet searches, based on real-time orchard scenarios as already mentioned. The orchards vary in different parts of the continent based on their topography, environment, and weather. This study contributes to a global and diversified real-time dataset that can be used in any part of the world.
b. Previous studies that use customized datasets, like deep fruits by Sa et al. [10], have a pretrained CNN model that was already trained on an ImageNet dataset. This study involves no pretrained model. Instead, it uses a fresh model with complete training,

verification, and testing of a given dataset, based on fine-tuning the hyperparameters until the best performance is reached, hence preventing overfitting.

c. Lastly, where many fruit-detection models are only tested offline, this study also conducts online testing in the orchards of Sargodha, Punjab. Real-time testing contributes to the model's actual, authentic results. These results could contribute significantly to future studies in the orchards of Asia, when used with agricultural machines, like a fruit-harvesting robot, to detect fruit before picking.

This paper is divided into different sections. The Introduction emphasizes the need of building and testing a fruit-detection model in complex real-time environments for a true model performance evaluation. The Methodology section explains the steps carried out in the research. Next, image collection for datasets, model building, and performance indicators are discussed. This is followed by an evaluation of the model in the Results and Discussion. Lastly, the study is summarized in the Conclusion.

3. Methodology

A dataset of 2000 images under different light conditions, angles, and noise was created using images from the internet and the orchard. Collected data were preprocessed by cleaning, resizing, and converting it to a suitable format. Images were annotated using a LabelImg tool and the labelled image was stored in two separate files. Data were then split into 80% training data and 20% validation and test data. The model training used the CNN model followed by verification of data. The model was fine-tuned and improved by adjusting the parameters of the model structure. A CNN model was built using a Keras sequential model. The layers mainly comprised the convolutional layers, a Relu activation layer, maximum pooling layer, flattening layer, and fully connected layer. A softmax classifier was used to find the output in probability vector form. Backpropagation was used to further improve the neural network weights to make better predictions. Next, a Kinect RGB-d camera was used to create actual image from the orange orchard and test the model. Real-time testing was done on an unseen dataset to evaluate performance in real-time conditions. The model loss and accuracy of the model were obtained. From the values of true positives, true negatives, false positives, and false negatives the precision, recall, and F1 score were calculated. Finally, the effectiveness of the detection system was determined. Figure 1 illustrates the methodology for detecting oranges using the CNN model.

3.1. Image Collection for the Dataset

The first process in fruit detection by machine learning was image collection. Image collection is the task of retrieving image of the fruit through a vision sensor. The most common vision sensor used for fruit detection is a camera. The camera is responsible for converting the captured light to digital information called pixels, using a complementary metal–oxide–semiconductor (CMOS) or charge-coupled device (CCD sensor).

The camera used for fruit detection was an Azure Kinect RGB-D. It contains a CMOS sensor with resolution of 12 megapixels (12 MP). Its dimensions were 103 × 39 × 126 mm and it weighed 440 g. It is an RGB camera with a resolution of 1920 × 1080, with 30 frames per second (FPS). It uses a time of flight (TOF) sensor with a depth resolution of 512 × 424 that determines the depth of the target fruit up to 5 m. Additionally, it includes an accelerometer and gyroscope (IMU) that aids orientation and spatial tracking. It is worth noting that partial data of fruit orchards were also obtained from Google to provide a diversified dataset.

3.2. Model Building

To prepare a CNN model for the fruit-harvesting robot, the dataset of fruit was first prepared. The dataset was created by obtaining real-time images of fruit from the Google website and from the orange orchards, using a camera, in JPG format. Data augmentation was used to flip and rotate the images. In addition, illumination changes were used to increase the diversity of the images. A total of 2000 images were stored in the dataset. The

images were resized and the pixel value was normalized. Next, the labels were converted to a one-hot encoding format. Libraries like TensorFlow and Keras were used for training the model, while Pandas, Numpy, and Matplotlib were used for handling, evaluating, and visualizing the data. Annotations of the dataset were carried out by the LabelImg tool. Next, data were split to ensure model training and evaluation were carried out on different data, preventing overfitting of the model. Training consisted of 80% of the data, while validation and testing comprised 20%.

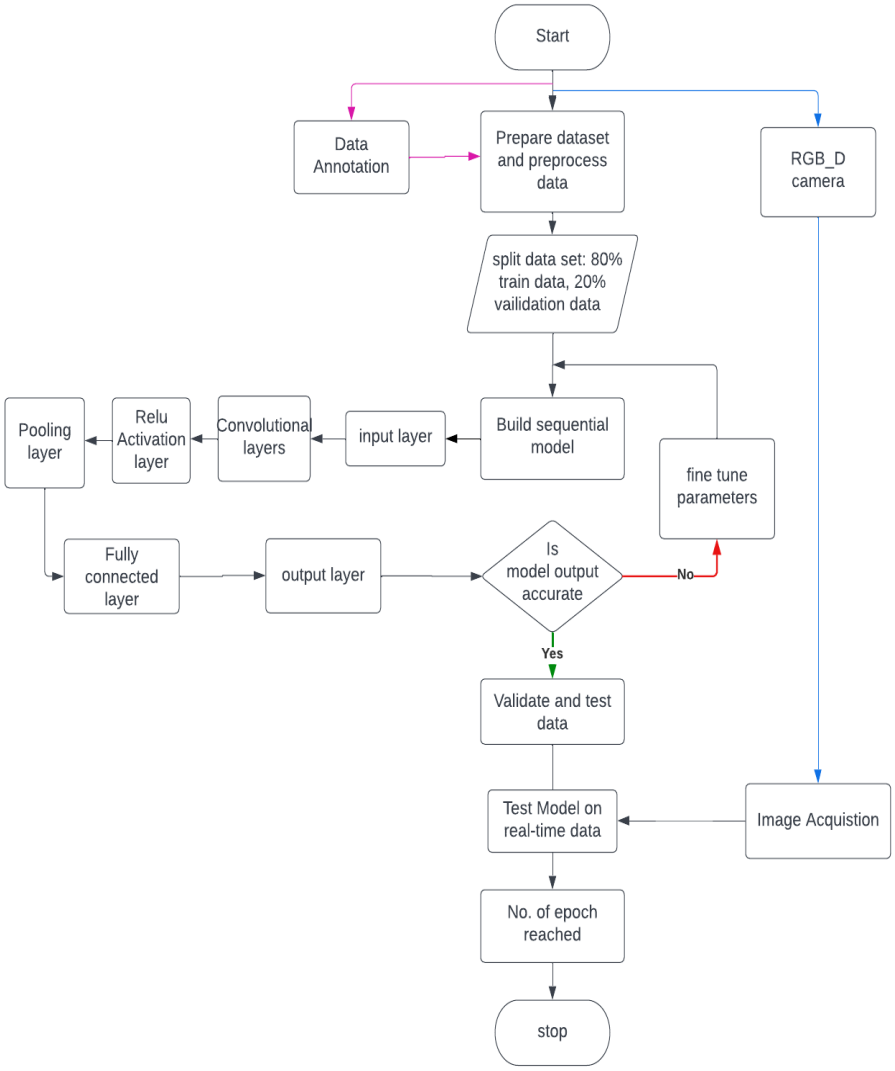

Figure 1. Flowchart of the methodology of an orange-detection CNN model.

The hyperparameters, shown in Table 1, were tested and tuned to determine the batch size, number of hidden layers, number of neurons, and the optimizers. Selected hyperparameters for training the model included batch size of 64, 4 hidden layers and 128 neurons per layer, with RMSprop as an optimizer with a learning rate of 0.01. Further, the Relu activation function was employed to prevent overfitting and cross-entropy was used as a loss function. For the pooling layer, maxpooling was used with a pool size of 2×2. The training epoch was 100. The number of filters in convolutional layers was 64 and kernel size of filters in convolutional layers was 3×3. The stride length of 2 was kept and the output size was 128×128. A Dell workstation with 11th Generation Intel Core i5-1135G7 @ 2.40 GHz, 8 GB RAM was used to train the neural network.

Table 1. Hyperparameters used in the study to train the model.

Batch size	64
Number of hidden layers	4
Number of neurons in each layer	128
Optimization technique	RMSprop
Activation function	ReLU
Loss function	Cross-entropy loss
Kernel size of filters in convolutional layers	3×3
Number of filters in convolutional layers	64

A CNN model was prepared using the Keras sequential model. Multiple layers were added to the created model. The model was built with four hidden layers. A convolution layer, Relu activation function, and pooling layer were present in the model. The Relu activation function filtered the image. Maxpooling was performed after every 2D convolutional layer. Finally, flatten and dropout layers were added to the model. A probability of 0.5 was chosen. Flattening was used for converting the data into a one-dimensional array to be used for input for the next layer. Dropout prevents overfitting of the neural network. Then the softmax function was applied, which converted the output into a probability vector. The dense connected layers were joined with 128 neurons. The final prediction of fruit detection was made in the output layer. Figure 2 shows the maxpooling operation where the maximum value of each feature map is selected.

The convolutional layers extracted features from the input image through convolution operations. The maxpooling layer down-sampled the feature maps generated by the convolutional layers, reducing the spatial size of the data while increasing the robustness of the features. The activation function introduced non-linearity into the model, allowing it to learn the complex representations of the data. A custom layer was used to combine the feature maps into one, using an adaptive approach by the weighted average. Batch normalization activated the layers, sped up the training, and reduced overfitting. The activations from the previous layer were flattened into a vector and then passed through the fully connected layers. The fully connected layers made the image class predictions of fruits from the features extracted by the convolutional and pooling layers.

To process more data per iteration, a batch size of 64 was chosen as that resulted in faster training time compared with the batch size of 32. Further, a batch size of 64 resulted in faster convergence compared with batch size of 32. This happened because more data were processed per iteration, allowing the model to make more accurate updates. A smaller batch size converged more slowly and the training process became unstable due to the increased variance in the gradient updates. Hence, a batch size of 64 was a better choice as it offered more stability in training. The epoch number was used as the stopping criteria. The epoch number of 100 was chosen because the accuracy of model was achieved, the validation loss did not improve after a given epoch number, and the epoch number was high enough to prevent overfitting.

The model was then compiled and run to generate the output. The epoch number was also specified based on the batch size. The data were tested for various captured orange images on the tree using an RGB-D Kinect camera. Once an orange was correctly detected, the fruit was localized. Next, a bounding box was drawn on the orange and its center locations were specified. This should help the end effector to pick the fruit by giving it a trajectory to the fruit location. Next, the model loss and accuracy graphs were plotted and the CNN detection model's accuracy, precision, recall, and F1 score were determined. Figure 3 shows the CNN architecture for the fruit-detection model steps to achieve the output.

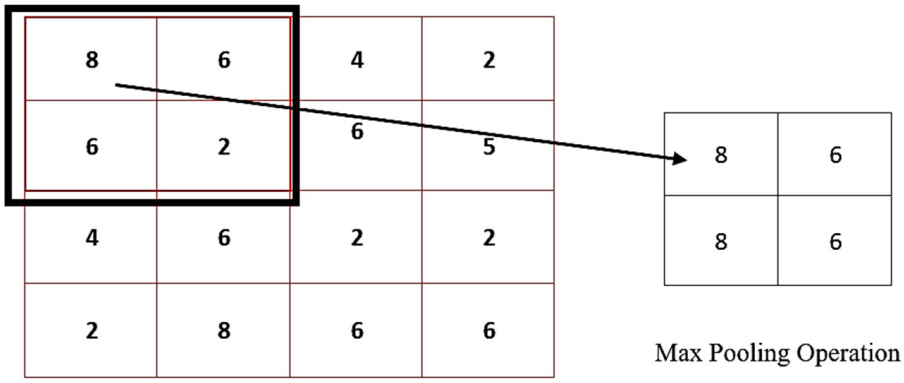

Figure 2. Process of the maximum pooling operation.

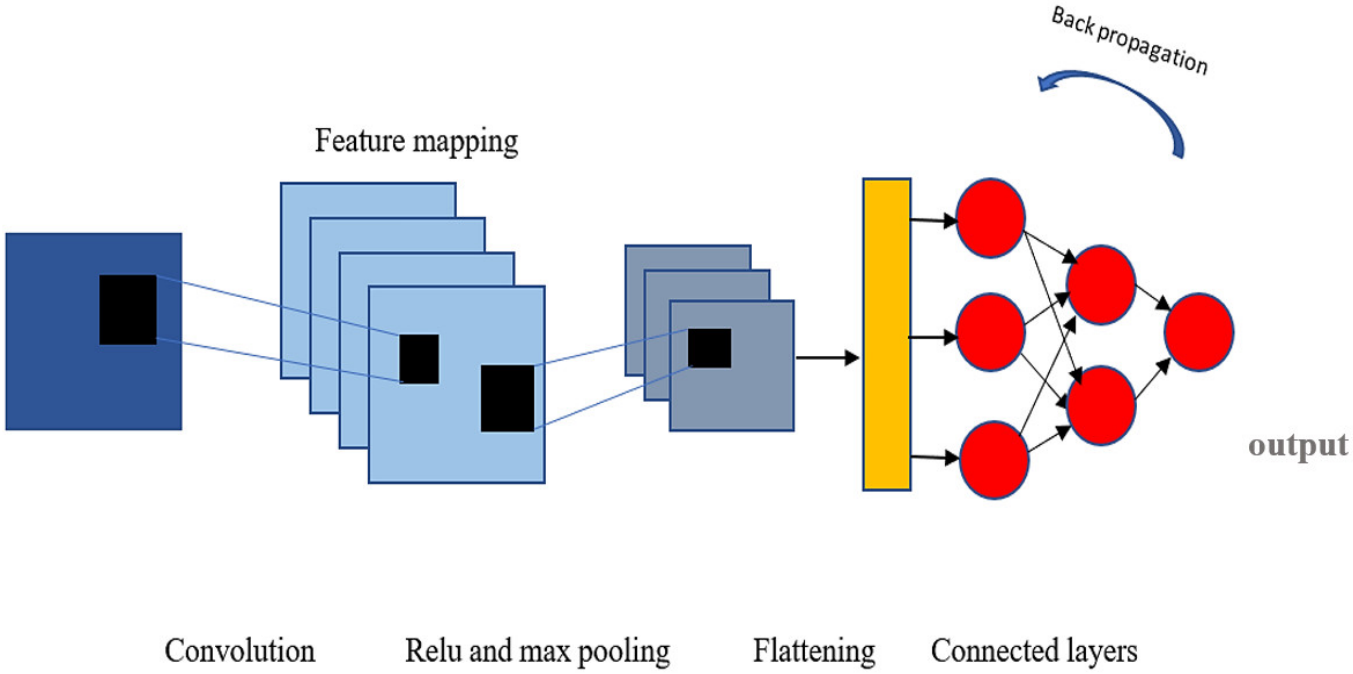

Figure 3. A convolutional neural network architecture for fruit detection.

3.3. Performance Indicators

To check the effectiveness of the object detection, certain parameters were evaluated. The most popular evaluation method for a detection model was calculation of its accuracy, precision, recall, and F1 score. These are the benchmark indicators that ensure the performance of a given model.

A confusion matrix is drawn to help visualize the positive and negative predicted data as shown in Figure 4. It gives the number of true positives, true negatives, false positives, and false negatives of our predicted data.

Based on the dataset run on the sequential model, we computed the metrices to determine the performance of the model. The parameters that determine performance of the model are accuracy, precision, recall, and F1 score.

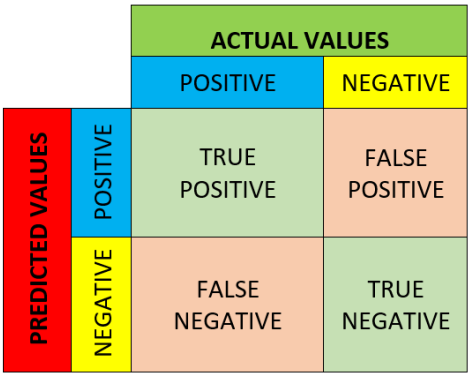

Figure 4. Confusion matrix for the CNN model.

The accuracy defines the performance of data. It helps us determine the reliability of the model to get correct results. It is calculated as:

$$Accuracy = (True_{pos} + True_{neg})/(True_{pos} + True_{neg} + False_{pos} + False_{neg}) \quad (1)$$

Precision measures the accuracy for a positive result only. However, it considers both true-positive and false-positive results in its calculation. High precision indicates that the model is more reliable and trustworthy. It is calculated as:

$$Precision = True_{pos}/True_{pos} + False_{pos} \quad (2)$$

Recall is the ratio between the number of true-positive samples and total number of positive samples. Recall tells us the reliability of getting correct results. Recall only takes into account the positive sample. If recall is high but precision is low, it indicates the model predicts correct positive results only.

$$Recall = True_{pos}/True_{pos} + False_{neg} \quad (3)$$

The F1 score is a combination of both precision and recall. It helps balance both the precision and recall score. It is also defined as the harmonic mean of precision and recall values.

$$\text{F1} - \text{Score} = (2 \times precision \times recall)/\ (precision + recall) \quad (4)$$

4. Results and Discussion

The test image was obtained from the orange orchard and run in the CNN model. The image was filtered and the orange fruits were detected. Localization was carried out by detecting the center of the image and drawing a bounding box around the detected image.

Different test images from the orchard were run and the results were computed for the accuracy and model loss. Figure 5 shows the results of orange-detection image test data. Moreover, Figure 6 shows the model loss for 100 epochs. Similarly, the accuracy of the model was computed. Figure 7 shows the accuracy percentage against 100 epochs.

Based on the test data, the precision, accuracy, recall, and F1 score were calculated for the model. Figure 8 shows the model's evaluation results for performance on real-time test data. These data were used to determine the CNN's effectiveness for orange-fruit detection. The test data were used to determine false positives, false negatives, true positives and true negatives. From these values, the precision, recall, accuracy, and F1 score were determined.

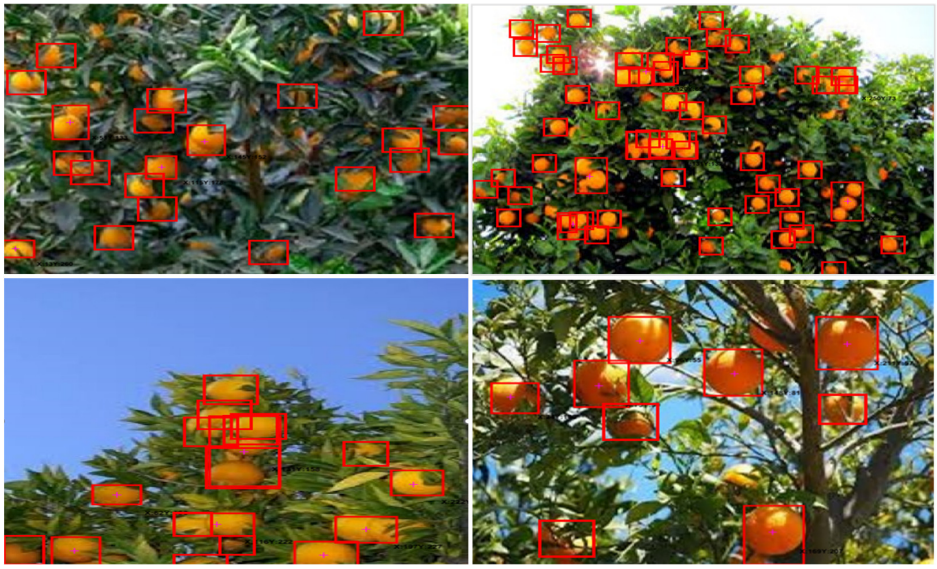

Figure 5. Detection and localization of orange fruit in an orchard.

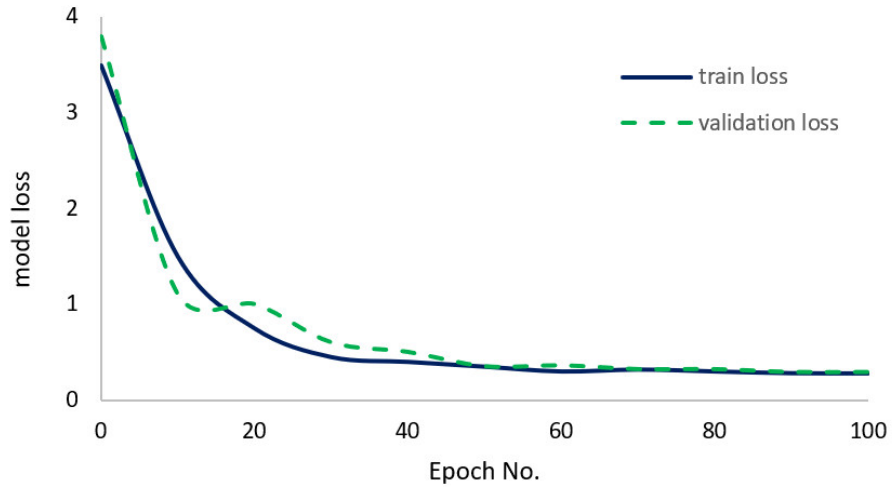

Figure 6. Model loss values for the epoch number of the orange fruit CNN model.

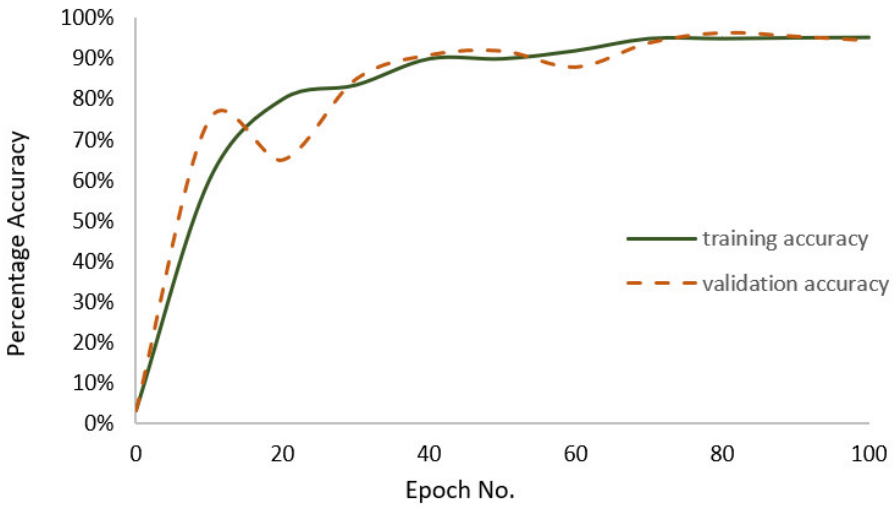

Figure 7. Accuracy percentage for the epoch number of the orange fruit CNN model.

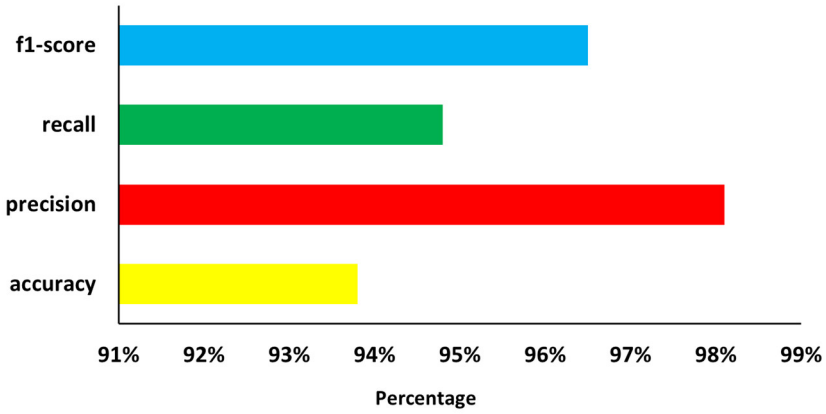

Figure 8. Performance indicators of the orange-detection CNN model.

The results indicate that the orange-detection model has an accuracy of 93.8% and precision of 98.1%. The recall value was found to be 94.8%, whereas the F1 score was 96.45%. The results depict a good orange-fruit detection model. The results show that with a dataset based on real-time images with varying illumination, occlusion, and noise, the recall F1 score improves due to less false detection. From previous studies with conventional datasets, it was observed that although the accuracy was usually high, the F1 score value was generally less. Sa et al. [10] detected various fruit using the CNN model, obtaining a customized dataset on a pretrained model using ImageNet, with an F1 score of 0.9 for oranges. Similarly, Kattenborn et al. [29] also detected vegetables with an average F1 score of 0.91. Also, cotton detection was carried out by Lin et al. [30] with a maximum F1 score of 0.87. Very few studies have been conducted on a real-time dataset in agricultural sector. The study found that by incorporating varying illumination, images of overlapping fruit as well as images of fruit overlapping with leaves and branches, the false-detection rate decreased. Although there was not much difference in the accuracy level from previous studies, the precision, recall, and F1 score showed significant improvement. Further, online testing in the orchard verified the reliability of using real-time data for training the dataset. Moreover, the robustness of the model increased, with a good variation within the dataset. Comparison of this model with previous studies verifies that false data can be minimized by using more data depicting real-time scenarios. This study proves that for better performance, as well as the reliability of the model to be used in real-time system, a universal real-time dataset should be deployed for training, instead of single fruit images.

The study has a few limitations. In the existing study, 2000 images were used due to limitations in computer resources. Future studies should be conducted using a larger dataset to provide more diversity. The existing dataset can be requested by mutual consent of the authors and the funding agencies.

5. Conclusions

This paper proposed an orange-detection method in a complex environment, like an orchard, based on convolutional neural networks. The orange-detection method was designed to detect oranges with an orange-harvesting robot. The study suggested using a real-time dataset from diversified orchards around the world for training, using images of fruit with varying illumination, occlusion, as well as images of fruit overlapping with leaves and branches. Furthermore, in addition to offline dataset training, online testing was conducted. The real-time testing of data based on images captured from the orchard provided a better evaluation of the model to be used in agricultural field applications.

Images were annotated using the LabelImg tool. This helped prepare the dataset for images obtained from the orchard. The CNN model consisted of convolutional layers, activation functions, maximum pooling layers, a dropout layer, flattening layer, and fully connected layers. The backpropagation feature improved the weight and helped to improve the detection results. The detected orange was localized by forming a bounding box

and center point that made it easy for the fruit-harvesting robot to pick the fruit. The designed model had an accuracy of 93.8%, precision of 98%, recall of 94.8%, and F1 score of 96.5%. As compared with previous studies that mostly used datasets from the internet and library repositories, the real-time dataset provided a more robust and reliable evaluation of performance. The study showed a clear reduction in the false-detection rate of data as compared with other studies. This research proves the study was effective in establishing that using a real-time dataset improves the F1 score and is more reliable in field applications. In future, a larger dataset should be used with a high-memory computer resource to further evaluate the performance of the model in the real-time agricultural field.

Author Contributions: Conceptualization, S.Z. and T.A.; methodology, S.Z.; validation, S.Z., T.A. and F.R.; formal analysis, S.Z.; investigation, S.Z.; resources, T.A. and F.R.; writing—review & editing, S.Z., T.A. and F.R.; visualization, S.Z.; supervision, T.A.; funding acquisition, T.A. and F.R. All authors have read and agreed to the published version of the manuscript.

Funding: The authors acknowledge grants for this research project provided by the Punjab Higher Education Commission (PHEC) under the Punjab Innovation and Research Challenges Award (PICRA), under letter PHEC/ARA/PICRA/20168/5. Also, authors acknowledge a UET research grant for PhD studies to Sadaf Zeeshan, under letter ORIC/106-ASRB/1086.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Dataset can be requested by mutual consent of the authors and the funding agencies.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Tugrul, B.; Elfatimi, E.; Eryigit, R. Convolutional neural networks in detection of plant leaf diseases: A review. *Agriculture* **2022**, *12*, 1192. [CrossRef]
2. Bargoti, S.; Underwood, J. Image segmentation for fruit detection and yield estimation in apple orchard. *J. Field Robot.* **2017**, *34*, 1039–1060. [CrossRef]
3. Wu, A.; Zhu, J.; Ren, T. Detection of apple defect uisng laser-induced light backscattering imaging and convolutional neural network. *Comput. Electric. Eng.* **2020**, *81*, 106454. [CrossRef]
4. Chen, S.W.; Shivakumar, S.; Dcunha, S.; Das, J.; Okon, E.; Qu, C.; Taylor, C.; Kumar, V. Counting apples and orages with deep learning: A data-driven approach. *IEEE Robot. Autom. Lett.* **2017**, *2*, 781–788. [CrossRef]
5. Yu, Y.; Zhang, K.; Yang, L.; Zhang, D. Fruit detection for strawberry harvesting robot in non-structured enviornment based on Mask-RCNN. *Comput. Electron. Agric.* **2019**, *163*, 104846. [CrossRef]
6. Ge, Y.; Xiong, Y.; From, P.J. Instance segmenatation and localization of strawberries in farm conditions for automatic fruit harvesting. In Proceedings of the 6th IFAC Conference on Sensing, Control and Automation Technolgies for Agriculture, Sydney, Australia, 4–6 December 2019.
7. Williams, H.; Jones, M.; Nejati, M.; Seabright, M.; Bell, J.; Penhall, N.; Barnett, J.; Duke, M.; Scarfe, A.; Ahn, H.; et al. Robotic kiwifruit harvesting using machine vision, convolutional neural networks and robotic arms. *Biosyst. Eng.* **2019**, *181*, 140–156. [CrossRef]
8. Liu, Z.; Wu, J.; Fu, L.; Majeed, Y.; Feng, Y.; Li, R.; Cui, Y. Improved kiwifruit detection using pre-trained VGG16 with RGB and NIR infomation fusion. *IEEE Access* **2019**, *8*, 2327–2336. [CrossRef]
9. Zapotezny-Anderson, P.; Lehnert, C. Towards active robotic vision in agriculture: A deep learning approach to visual servoing in occluded unstructured protected cropping enviornments. *IFAC-PapersOnline* **2019**, *52*, 120–125. [CrossRef]
10. Sa, I.; Ge, Z.; Dayoub, F.; Upcroft, B.; Perez, T.; McCool, C. Deepfruits: A fruit detection system using deep neural networks. *Sensors* **2016**, *16*, 1222. [CrossRef]
11. Rahnemoonfar, M.; Sheppard, C. Deep count: Fruit counting based on deep simulated learning. *Sensors* **2017**, *17*, 905. [CrossRef]
12. Ganesh, P.; Volle, K.; Burks, T.; Mehta, S. Deep orange: Mask R-CNN based ornage detection and segmentation. In Proceedings of the 6th IFAC Conference on Sensing, Control and Automation Technologires for Agriculture, Sydney, Australia, 4–6 December 2019.
13. Barre, P.; Herzog, K.; Hofle, R.; Hullin, M.; Topfer, R.; Steinhage, V. Automated phenotyping of epicuticular waxes of grapevine berries using light separation and convolutional neural networks. *Comput. Electron. Agric.* **2019**, *156*, 263–274. [CrossRef]
14. Munasingha, L.; Gunasinghe, H.; Dhanapala, W. Identification of papaya fruit diseases using deep leraning approach. In Proceedings of the 4th International Conference on Advances in Computing and Technology, Kelaniya, Sri Lanka, 29 July 2019.

15. Jahanbakhshi, A.; Momeny, M.; Mahmoudi, M.; Zhang, Y. Classification of sour lemons based on apparent defects uisng stochastic pooling mechanism in deep convolutional neural network. *Sci. Hortic.* **2020**, *263*, 109133. [CrossRef]
16. Zhang, Y.; Lian, J.; Fang, M.; Zheng, Y. Deep indicator for fine-grained classification of banana's ripening stages. *J. Image Video Proc.* **2018**, *46*. [CrossRef]
17. Tan, W.; Zhao, C.; Wu, H. Intelligent alerting for fruit-melon lesion image based on momentum deep learning. *Multim. Tools Appl.* **2016**, *75*, 16741–16761. [CrossRef]
18. Cen, H.; He, Y.; Lu, R. *Hyperspectral Imaging-Based Surface and Internal Defects Detection of Cucumber via Stacked Sparse Auto-Econder and Convolutional Neural Network*; ASABE, American Society of Agricultural and Biological Engineers: St. Joseph, MI, USA, 2016; p. 1.
19. Mirhaji, H.; Soleymani, M.; Asakereh, A.; Mehdizadeh, S.A. Fruit detection and load estimation of an orange using the YOLO models through simple approaches in different imaging and illumination conditions. *Comput. Electron. Agric.* **2021**, *191*, 106533. [CrossRef]
20. Gai, R.; Chen, N.; Yuan, H. A detection algorithm for cherry fruits based on the improved YOLO-v4 model. *Neural Comput. Appl.* **2021**, 1–12. [CrossRef]
21. Tian, Y.; Guodong, Y.; Wang, Z.; Wang, H.; Li, E.; Liang, Z. Apple detection during different stages in orchards using the improved YOLO-V3 model. *Comput. Electron. Agric.* **2019**, *157*, 417–426. [CrossRef]
22. Liang, Q.; Zhu, W.; Long, J.; Wang, Y.; Sun, W.; Wu, W. A real-time detection framework for on-tree mango based on SSD network. In Proceedings of the International Conference on Intelligent Robotics and Applications, Newcastle, NSW, Australia, 9–11 August 2018.
23. Zhang, X.; Gao, Q.; Pan, D.; Cao, P.C.; Huang, D.H. Research on spatial postioning system of fruits to be picked in field based on binocular vision and SSd model. *J. Phys. Conf. Serires* **2021**, *1748*, 042011. [CrossRef]
24. Liu, X.; Zhao, D.; Jia, W.; Ji, W.; Ruan, C.; Sun, Y. Cucumber fruits detection in greenhouses based on instance segmentation. *IEEE Access* **2019**, *7*, 139635–139642. [CrossRef]
25. Sakib, S.; Ashrafi, Z.; Sidique, M.A.B. Implementation of Fruits Recognition Classifier using Convolutional Neural Network Algorithm for Observation of Accuracies for Various Hidden Layers. *arXiv* **2020**, arXiv:1904.00783.
26. Padilha, T.C.; Moreira, G.; Magalhães, S.A.; dos Santos, F.N.; Cunha, M.; Oliveira, M. Tomato Detection Using Deep Learning for Robotics Application. In *Progress in Artificial Intelligence: 20th EPIA Conference on Artificial Intelligence, EPIA 2021, Virtual Event, September 7–9, 2021*; Marreiros, G., Melo, F.S., Lau, N., Lopes Cardoso, H., Reis, L.P., Eds.; Springer: Cham, Switzerland, 2021; Volume 12981. [CrossRef]
27. Vasconez, J.P.; Delpiano, J.; Vougioukas, S.; Cheeina, F. Comparison of convolutional neural networks in fruit detection and counting: A comprehensive evaluation. *Comput. Electron. Agric.* **2020**, *173*, 105348. [CrossRef]
28. Duonga, L.T.; Nguyen, P.T.; Sipio, C.D.; Ruscio, D.D. Automated Fruit Recognition using EfficientNet and MixNet. *Comput. Electron. Agric.* **2020**, *171*, 105326. [CrossRef]
29. Kattenborn, T.; Leitloff, J.; Schiefer, F.; Hinz, S. Review on Convolutional Neural Networks (CNN) in vegetation remote sensing. *ISPRS J. Photogramm. Remote Sens.* **2021**, *173*, 24–49. [CrossRef]
30. Lin, Z.; Guo, W. Cotton stand counting from unmanned aerial system imagery using mobilenet and centernet deep learning models. *Remote Sens.* **2021**, *13*, 2822. [CrossRef]