*AgriEngineering*

MDPI

*Article*

# A Convolutional Neural Network Algorithm for Pest Detection Using GoogleNet

**Intan Nurma Yulita [1,*], Muhamad Farid Ridho Rambe [1], Asep Sholahuddin [1] and Anton Satria Prabuwono [2]**

[1] Department of Computer Science, Faculty of Mathematics and Natural Sciences, Universitas Padjadjaran, Sumedang 45363, Indonesia
[2] Faculty of Computing and Information Technology in Rabigh, King Abdulaziz University, Rabigh 21911, Saudi Arabia; aprabuwono@kau.edu.sa
* Correspondence: intan.nurma@unpad.ac.id

**Abstract:** The primary strategy for mitigating lost productivity entails promptly, accurately, and efficiently detecting plant pests. Although detection by humans can be useful in detecting certain pests, it is often slower compared to automated methods, such as machine learning. Hence, this study employs a Convolutional Neural Network (CNN) model, specifically GoogleNet, to detect pests within mobile applications. The technique of detection involves the input of images depicting plant pests, which are subsequently subjected to further processing. This study employed many experimental methods to determine the most effective model. The model exhibiting a 93.78% accuracy stands out as the most superior model within the scope of this investigation. The aforementioned model has been included in a smartphone application with the purpose of facilitating Indonesian farmers in the identification of pests affecting their crops. The implementation of an Indonesian language application is a contribution to this research. Using this local language makes it easier for Indonesian farmers to use it. The potential impact of this application on Indonesian farmers is anticipated to be significant. By enhancing pest identification capabilities, farmers may employ more suitable pest management strategies, leading to improved crop yields in the long run.

**Keywords:** pest detection; GoogLeNet; Convolutional Neural Network; mobile application



**Citation:** Yulita, I.N.; Rambe, M.F.R.; Sholahuddin, A.; Prabuwono, A.S. A Convolutional Neural Network Algorithm for Pest Detection Using GoogleNet. *AgriEngineering* **2023**, *5*, 2366–2380. https://doi.org/10.3390/agriengineering5040145

Academic Editors: Ray E. Sheriff and Chiew Foong Kwong

Received: 27 September 2023
Revised: 2 December 2023
Accepted: 6 December 2023
Published: 8 December 2023



## 1. Introduction

Food is one of the most fundamental requirements for maintaining human health and vitality. Almost anywhere in Indonesia, a wide variety of crops may be grown due to the country's abundance of arable land [1]. When it comes to plant exports, Indonesia is a major player [2]. Due to its tropical location, Indonesia frequently experiences floods, droughts, and other forms of severe weather. The changing patterns of precipitation and temperature brought on by global warming also threaten plant growth and crop output. Traditional, inefficient farming practices are still used by some Indonesian farmers. Another factor that might prevent farmers from boosting output is a lack of access to cutting-edge agricultural technologies like organic fertilizer, improved seeds, enough irrigation, and safe pesticides. Social disparities in Indonesian society, which may be experienced by farmers in particular locations, may also have an impact on agricultural output. Small farmers' inability to run their farms efficiently is largely attributable to their lack of knowledge. Poor harvest outcomes are to be expected under these conditions.

The situation of farmers in Indonesia is strongly connected to the challenge of identifying pests in agricultural settings, which in turn can have an effect on crop yields that are not at their full potential [3]. The education level of many farmers in Indonesia is rather poor, and they have limited access to agricultural training and knowledge [4–6]. As a consequence of this, they might not have an appropriate understanding of how to properly detect and control agricultural pests. Smaller farms may not have the financial means to invest in or learn to utilize sophisticated software tools for pest identification. As a result, it

may be challenging to see pests in their early stages. Without prompt and proper treatment, a pest infestation on one plant will spread to others in the area [7–11]. In a typical crop, a single rat may harm anywhere from 11 to 176 plants in a single night [12]. During the breeding season, a single rat can destroy up to 26 stalks in a single night, leaving behind only a few rows of plants. It takes a mouse only 21 days to have her own litter of 6–8 young. Obviously, the severity of this problem might increase dramatically if it is not addressed promptly and effectively. Depending on the surrounding environment and the care offered to the plant, there may be a wide variety of pests that target a single plant type [13].

Several ancillary technologies [14] are already in use in the agriculture industry, including drones. Farmers may utilize drones to keep an eye on crops from above and protect them from pests by spraying pesticides over large areas [15–18]. However, without special detecting equipment, drones cannot identify which pests are present on a farm, and not every farmer can afford to buy them. Pest detection apps for smartphones that are already ubiquitous in Indonesia are a necessity.

It is important to be able to identify the numerous plant pests that occur. Machine learning using supervised model approaches is one approach [19–25]. This technique needs data that already have a previous label. In machine learning, there are also various methods of image classification. One effective method is the use of CNNs. The Pest Region-CNN end-to-end detection model, otherwise known as Pest R-CNN, was created by Du et al. for the maize pest Spodoptera frugiperda [26]. It tracks their consumption of maize leaves using the Faster R-CNN model. The model classifies invasion intensity into juvenile, minor, moderate, and severe using UAV-acquired high-spatial-resolution RGB ortho-images. The model demonstrated the efficacy of deep learning object detection in pest monitoring with a mean average accuracy of 43.6% on the test dataset. Their model's detection accuracy improved by 12% and 19% over Faster R-CNN and YOLOv5, respectively. Jiao et al. developed a CNN-based multi-class pest identification approach for complicated scenarios [27]. Their research introduces adaptive feature fusion within feature pyramid networks to extract deeper pest characteristics. To reduce information loss in the highest-level feature map, an adaptive augmentation module was designed. Finally, a two-stage region-based CNN (R-CNN) refined the predicted bounding boxes to determine the pest types and locations in each image. Compared to SSD, RetinaNet, FPN, Dynamic R-CNN, and Cascade R-CNN, their method achieved 77.0% accuracy, which was the highest.

The key to lowering the amount of produce that is lost is finding plant pests as early, correctly, and effectively as possible. In this scenario, apps that run on mobile devices provide a significant and pertinent solution. With the help of smartphone applications, farmers are able to conveniently perform routine checks on their crops. They can capture images of plants and detect pests using their mobile devices, making it simpler for them to record and monitor everyday agricultural conditions. Mobile applications have been developed with user interfaces that are straightforward and simple to operate. Because of this, they may be used without trouble by farmers who have varied levels of education and prior experience with technology. A number of applications have been developed to help farmers detect pests. Prabha et al. developed an Android mobile application that utilizes a Deep CNN (DCNN)-based artificial intelligence system for the purpose of insect identification in maize production [28]. The utilization of CNN models on mobile devices yields advantages for all stakeholders, with particular benefits being offered to farmers and agricultural extension experts, since they enhance their accessibility. A mobile application for Android was developed for the purpose of detecting fall armyworm infection in maize crops. This application incorporated the advice provided by Tamil Nadu Agricultural University's Integrated Pest Management (TNAU IPM) capsules. This mobile software offers guidance on effectively managing the issue of fall armyworm infestation. Chen et al. presented an application for the identification of scale pests in Taiwan using image-based methods [29]. The Coccidae and Diaspididae species, known as mealybugs, are the predominant pests of scale insects in Taiwan. These species have the potential to cause significant

harm to plants and pose a substantial threat to agricultural productivity. Therefore, the identification of scale pests holds significant importance within the agricultural sector of Taiwan. You Only Look Once v4 (YOLO v4), single-shot multi-box detectors (SSDs), and faster region-based CNNs (Faster R-CNNs) are some of the deep learning algorithms used to detect and precisely localize scale pests inside an image. A smartphone application has been created that utilizes a trained scale pest detection model. This application aims to assist in the identification of pests in agricultural settings, hence enabling farmers to effectively apply targeted pesticides and minimize crop losses.

Despite the existence of several related applications in other countries, there is a distinct necessity for tailored mobile-based applications designed specifically for Indonesian farmers. The compatibility between applications produced in other countries and the local language may not always be guaranteed. The development of a mobile application tailored to Indonesian farmers would facilitate the incorporation of local language into the program's content and user interface. This approach would enhance comprehension and acceptance among farmers, hence facilitating their utilization of the application. There is a poor general level of English proficiency in the whole population of Indonesia [30]. Farmers who have completed elementary and secondary schooling will also find this to be a difficult situation [31]. As a result, developing a mobile application tailored to the needs of Indonesian farmers has the potential to address the distinct obstacles encountered within the agricultural sector in Indonesia.

This study also employs a CNN architecture, namely GoogleNet, for the purpose of identifying pests within mobile applications. GoogleNet makes use of a technology called Inception blocks, which integrate a variety of convolutional algorithms and filter sizes into a single layer. This results in a reduction in the number of computing processes and parameters that are necessary, making the model simpler. When compared to other CNN architectures that were available at the time, such as AlexNet or VGG, this model featured a significantly lower total number of parameters [32,33]. This enabled a more effective utilization of the RAM of the mobile device. In addition, this model includes methods such as L2 regularization and dropout to avoid overfitting [34]. This can assist in retaining model performance on small datasets without the need for a high number of parameters by reducing the likelihood of the model being overly accurate. The fact that GoogleNet was developed with hardware capabilities in mind, such as the Single Instruction Multiple Data (SIMD) instructions that are present on many recent mobile devices [35], is another key aspect to take into consideration. This helps to maximize the usage of the resources available in the hardware. Traditional convolution is rendered obsolete as a result of the development of depth-wise separable convolution. Because of this, the number of operations and parameters that are necessary is decreased, which results in increased computing efficiency. The use of a large number of convolutional layers of relatively modest size helps to extract information in an effective manner and also enables dimensionality reduction. All of the aforementioned criteria contribute to GoogleNet's ability to achieve an optimal performance on mobile devices with limited resource capacities. Because of this, it is an excellent option for mobile apps, which place a premium on computing efficiency as well as model size. This aligns with the varied agricultural landscape of Indonesia.

## 2. Materials and Methods

### *2.1. Data Collecting*

Insects and other organisms that cause damage to plants are the focus of this research. The information was obtained from the Kaggle website, which may be viewed at the following address on the internet: https://www.kaggle.com/simranvolunesia/pest-dataset (accessed on 21 February 2022). The data that were used cover nine different kinds of plant-destroying insects, including aphids, armyworm, beetles, bollworms, grasshoppers, mites, mosquitoes, and sawflies. Stem borers were also included in the list. There are two types of data in the dataset provided. The data included 2700 images, with each

class contributing 300 images. These data were used to train the model, with 1800 data as training data, and the remaining 900 data as validation data. The data were distributed evenly in terms of number throughout each class so that the process of learning could go more smoothly with the weighting model. The second data group comprised the taking of 450 images, with each class contributing 50 images. This second data group was used to test the model.

### *2.2. Data Preprocessing*

In the first step of the data preparation stage, the data were separated into two categories: training data and validation data. Following the distribution of the data, the images were subjected to preprocessing. Image preprocessing was performed so that the model was able to detect characteristics in images with greater ease as a result of the work that was performed. Image data relating to plants and pests were examined in this study. The data were eventually utilized as input data in the classification process once they had been processed via the preparation step. Data preprocessing consisted of two main stages, namely image resizing and image augmentation. Image resizing was performed to change the input image to a size of 224 × 224, with a size of 3 bits per pixel. Some deep learning operations are more efficient when the size of the image is a multiple of certain numbers, such as 32, 64, or 128. The GPU can process data more efficiently with an image size of 224 × 224 pixels [36]. Numerous deep learning models have been trained (pre-trained) with datasets containing images of size 224 × 224 [37]. Utilizing pre-trained models is facilitated by employing the same size during detection. Also, certain deep learning frameworks, such as TensorFlow, have pre-trained models that support 224 × 224 image sizes directly [38]. This facilitates the implementation of these models within applications.

Image augmentation was only carried out on the training data, because a lot of data were needed for the model training process [39–42]. The process was carried out to expand the variety of images that were used by the model using the module tf.image in TensorFlow. This stage was carried out to increase the amount of data by changing the shape of the image. The augmentation process carried out involved rotating the image, shifting the image based on the width and height of the image, applying shearing to the image, enlarging the image, and flipping the image horizontally.

- Width Shift

  This was carried out to shift the image width to the left and right to provide variations for the model to learn. The width shift range value given was 0.2.

- Height Shift

  This was carried out to shift the image height up and down to provide variations for the model to learn. The height shift range value given was 0.2.

- Shears

  This was carried out to tilt the image regarding the $x$-axis and $y$-axis to provide variations for the model to learn. The value of the shear range given was 0.2.

- Zoom

  This was carried out to enlarge or reduce the image to provide variations for the model to learn. The value of the given zoom range is 0.2.

- Horizontal Flip

  This involved flipping an image vertically or horizontally to provide variations for the model to learn. It was assisted using external tools in the form of an image data generator library that had been previously provided by Tensorflow to make the process easier.

### *2.3. Classification*

One deep learning approach is CNN, which is based on artificial neural networks (ANN) [43–45]. The process data at both the input layer and the hidden layer then output the results to the output layer. The difference between CNNs and ANNs is that the former focuses more on recognition than the latter. A CNN has a structure consisting of various layers that form a stacked network [46,47]. Convolutional layers, pooling layers, and fully connected layers are the three main layers that produce a CNN after being combined. The CNN layer is very important. This layer continuously uses filters on the data in the input layer, and then produces the output. In the pooling layer process, the output convolutional layer has its dimensions and parameters reduced [48–50]. Maximum pooling takes the largest value from the pooling filter used. Average pooling takes the average value from the pooling filter used. The pooling process helps reduce the computational complexity of the model and prevents overfitting. Multilayer perceptron applications typically use fully connected layers. The goal of convolutional or pooling layers is to increase the dimensionality of data that can be classified linearly. To achieve this goal, data that have been previously processed in this layer will be flattened first before being inserted into this layer.

In 2014, Google collaborated with various universities to conduct research on GoogleNet, also known as Inception V1. The winner of ILSVRC 2014 is this architectural design. Compared to winning designs from previous years, this design is more accurate. This architecture uses global average pooling and $1 \times 1$ convolution in the middle. The main aim of this architecture is to determine how optimally the local sparse structure in a convolutional vision network can be estimated and encompassed in easily accessible thick components [51–55]. This architecture's $1 \times 1$ convolution concept helps reduce the number of parameters (loads and biases) present in the architecture significantly. At the end of the network, this architecture uses an elative global average pooling method. Overall, this structure consists of twenty-two layers. To maintain computational efficiency, this architecture can be used on devices. It has low computing capabilities, as depicted in Figure 1, which explains the details of the twenty-two architectural layers. This study used the CNN method. The architecture used was GoogLeNet, which was a development of LeNet and AlexNet and had twenty-two layers. The GoogLeNet model was implemented by training the model using previously obtained data. By using a pre-existing dataset, this model predicted images of plants.

Figure 1 demonstrates the GoogLeNet architecture that was utilized in the research. The subsequent section provides an outline of the subsequent steps:

1. The input used is an image measuring $224 \times 224$.
2. The image data undergo five phases of convolution prior to reaching the fully connected layer.
3. In the initial phase, the data are subjected to $7 \times 7$ convolution and $3 \times 3$ max pooling.
4. The data are then transit through two convolution layers with dimensions of $1 \times 1$ and $3 \times 3$, followed by a max pooling layer with dimensions of $3 \times 3$.
5. In the third stage, data enter the Inception model layer twice and are processed by max pooling with $3 \times 3$ dimensions.
6. In the fourth stage, the data are first processed by an Inception model layer and then by an auxiliary classifier. The data then pass through three layers of the inception model and then the second auxiliary classifier. Then, the data pass through an Inception model layer once more prior to max pooling with $3 \times 3$ dimensions.
7. In the fifth stage, the data travel through two layers of the Inception model and then undergo $7 \times 7$-dimensional average pooling. The data then proceed to the output after passing through the fully connected layer, which is also the sixth stage of the architecture.
8. At stages 2 through 5, a $1 \times 1$ convolution layer is used to reduce the dimensions of the output.

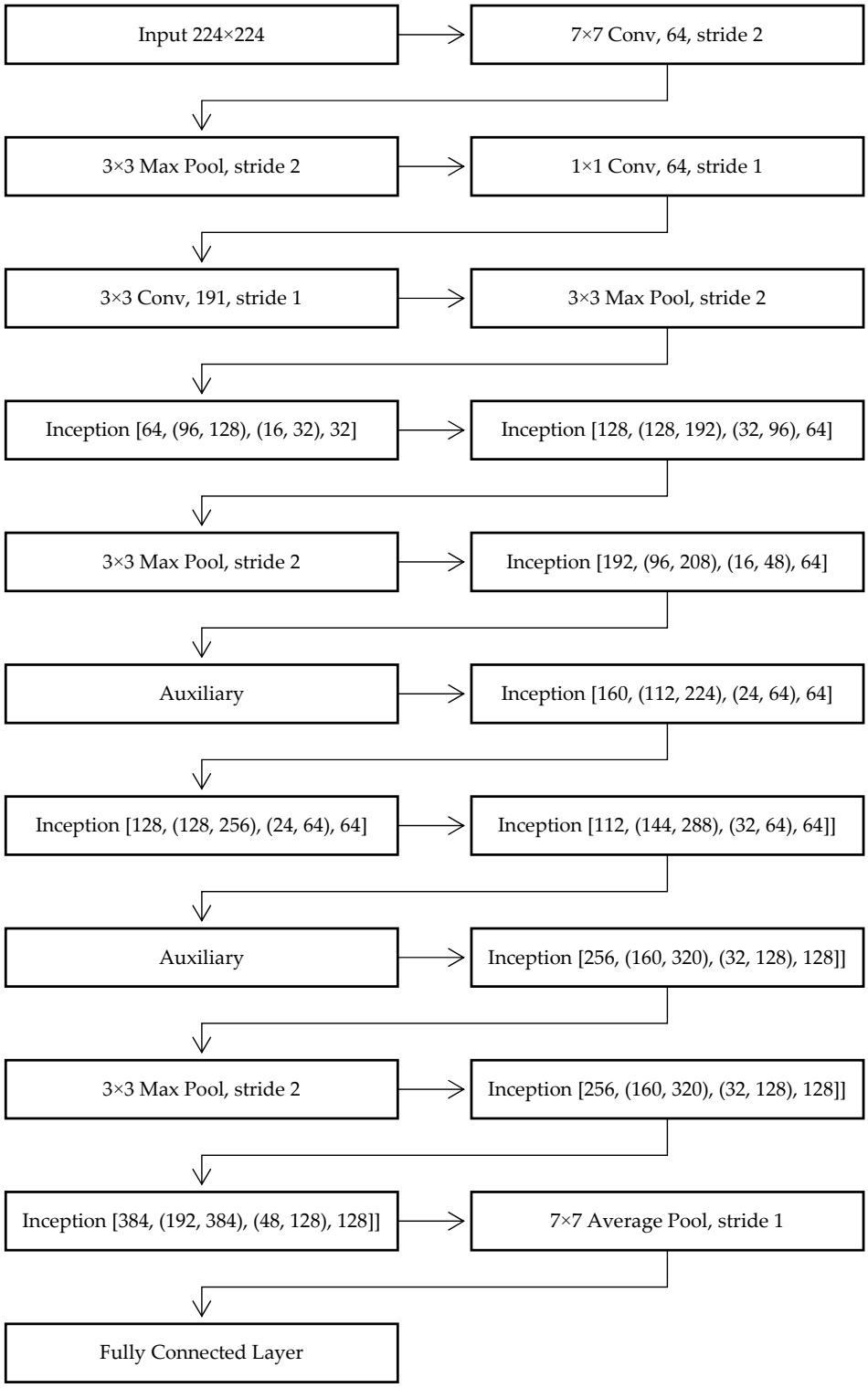



**Figure 1.** Details of each layer on GoogleNet.

An inception model and auxiliary classifier were developed for the GoogLeNet architecture in this study before creating the main model. Figure 2 illustrates the inception model's architecture. In addition, Figure 3 describes the auxiliary classifier, which was utilized by the GoogleNet model.

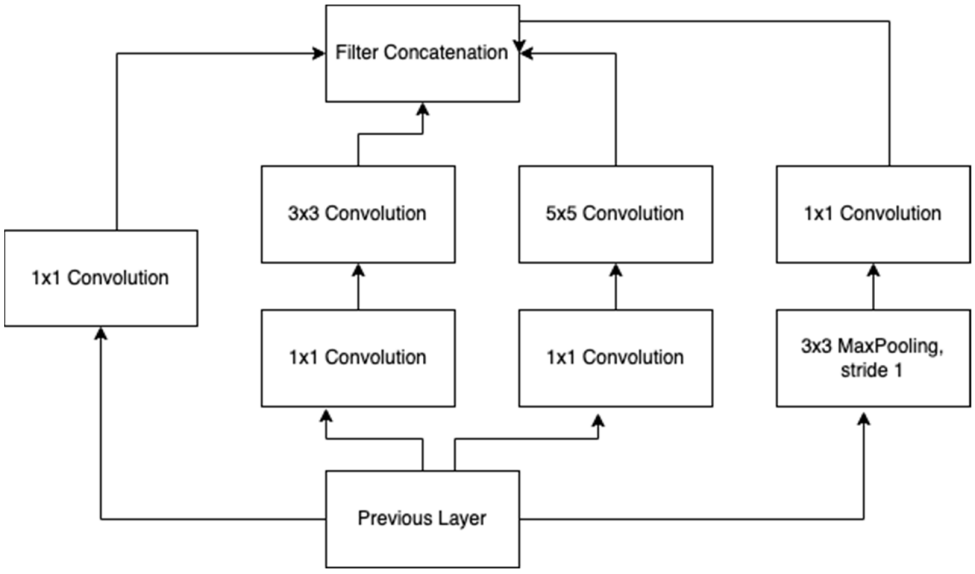



**Figure 2.** Architecture of the Inception Model.

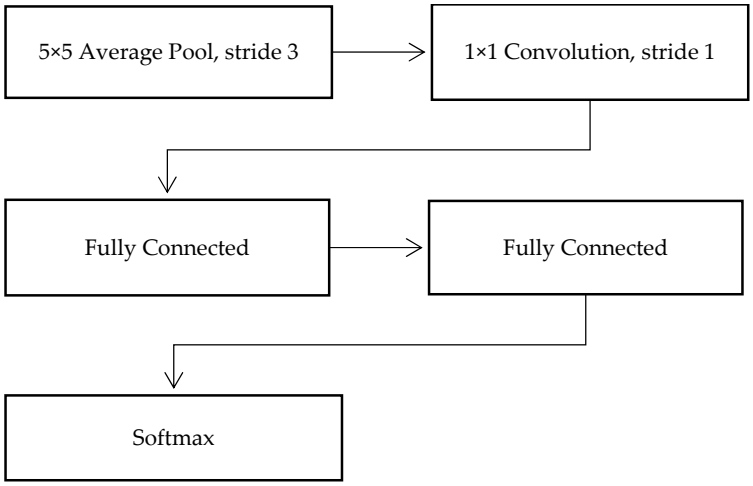



**Figure 3.** Architecture of Auxiliary Classifiers.

Before the model was implemented, hyperparameters were determined. The research was conducted by experimenting with multiple hyperparameters to determine the optimal pair of hyperparameters that yield the most accurate results from the model. This research employed the number of dense layers in the fully connected layer and various dropout values as hyperparameters. The research utilized these two categories of hyperparameters to find the optimal solution. The dropout values evaluated were 0.3, 0.4, and 0.5, and the number of dense layers was 1, 3, or 5. The hyperparameter determination procedure involved varying the number of dense layers and the dropout value on the fully connected layer to select the model with the highest accuracy. After determining the optimal hyperparameters, the GoogleNet model was applied to the applications created with the hyperparameter pair that yielded the highest accuracy.

In addition to the hyperparameters that were investigated, this study also set additional factors, including epochs, the batch size, optimizers, and the image size. The number of epochs was set to 100. The batch size, which determined the number of samples processed in each iteration, was specified as 8. The optimizer in this study was Stochastic Gradient Descent (SGD). The learning rate, denoting the step size at each iteration, was assigned a value of 0.00001. Lastly, the image size was 224 × 224. Hyperparameter

changes involved a dense layer between the flatten layer and the output layer, as well as modifications to the values inside the dropout layer.

### *2.4. Evaluation*

Model testing was carried out using data that were different from the training and validation data. This was to ensure that each model created made predictions on data that the model had not previously studied. Model validation was carried out via K-fold cross-validation (K = 3). This method helped the process of evaluating and determining hyperparameters, especially on smaller datasets. This method divided k datasets randomly. To determine the validation and training data, the rotation of each piece of data was performed [56–58]. The aim of this training was to produce different evaluation results for each fold performed. The hyperparameters were determined by taking the average value of these assessments. In addition, a confusion matrix was displayed in this study to simplify the test. The accuracy values were obtained using this matrix. This was the amount of data that had been classified correctly. During this procedure for testing models, the only model that was chosen was the one that had the highest quality validation values for each hyperparameter. This method was used to check the model with new data. The test results were used to assess the Android application system model created.

### *2.5. Implementation of Mobile Applications*

The training process was carried out using Google Colab (Google LLC, Mountain View, CA, USA). After the model evaluation process was complete, the conversion stage was carried out. Of all the hyperparameter experiments carried out, only the best model was implemented in the application. Because the application to be developed was smartphone-based, this conversion process was necessary. Tensorflow Lite is a version of Tensorflow specifically made for mobile-based applications used on the Android operating system, and was used in this study. The disparities in the accuracy and duration of detection between training in Google Colab and testing on a smartphone might arise from several sources. Some examples include hardware disparities, deployment optimization, and dependencies on frameworks and libraries. In order to address this issue, we employ frameworks such as TensorFlow Lite, which is specifically developed to enhance the efficiency of models for deployment on mobile devices. It provides tools and techniques particularly designed to enhance the speed and efficiency of inference on mobile devices. It can strive to minimize the disparities in precision and response time between the training environment in Google Colab and the testing environment on a smartphone. This process was carried out to change the format of the model obtained, which was previously in h5 format, into TFlite format. After the conversion process was complete, the model was moved to the Android application directory. It carried out the detection process using the application diagram package that is presented in Figure 4. The system structure consisted of relationships between directories in the application, as depicted in this diagram. Three main directories comprised the created directory: res, Java, and the Machine Learning Model. The res directory consisted of assets to support the visuals of the application that were used in the Java directory later. The directory included the logic that regulated the application path. Machine learning models were applied to the applications in this directory. The deployment diagram in Figure 5 explains the physical settings of the application being created. The Java classes were components that form the flow and logic of an application; resources were the material that was shown to the user. Tflite was the model used for the machine learning prediction process. This GoogleNet-installed mobile application requires an Android operating system, a minimum of 2 GB RAM memory, and a minimum of 32 GB internal storage. The camera's resolution must be at least 4 megapixels for direct image capture. When run, this application has a maximum size of 10 MB.

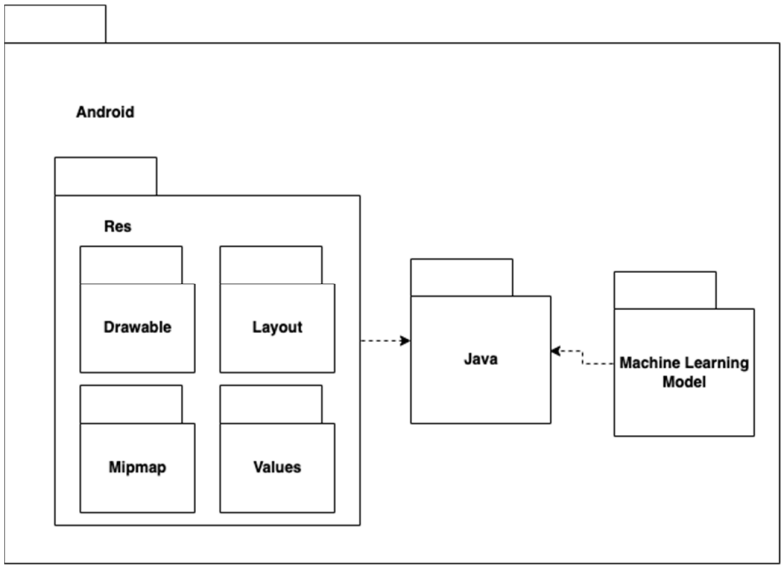



**Figure 4.** Package diagram.

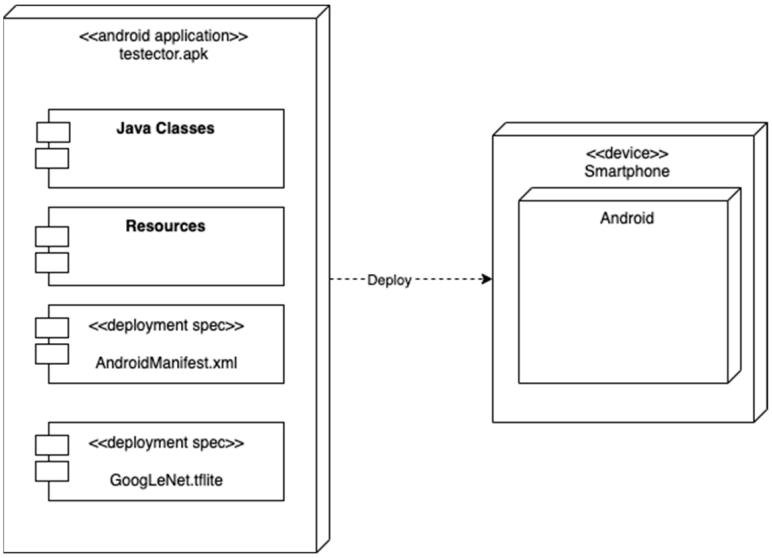



**Figure 5.** Deployment diagram.

## 3. Results

At this point, an evaluation of the hyperparameters that were employed is carried out, and its findings are analyzed. The specified hyperparameters were utilized in a total of 12 separate experiments. The metrics taken into consideration were the accuracy score from the testing to ensure that there was no overfitting or underfitting. In the testing process, the model with the best accuracy value was selected from the k-fold process that was previously carried out. This was done because K-fold produced three models and only used one of the models later to be implemented into the prototype to be made. The hyperparameters to be tested were the number of dense layers and the dropout value. The hyperparameters of dense layers had a total of 0, 1, 3, and 5, with some dropout values of 0.3, 0.4, and 0.5.

Table 1 shows that the experiment that produced the best model was the one that had the addition of five dense layers and a dropout of 0.5. Even though it had the same accuracy during testing as a model that did not experience the addition of a dense layer, and even though the dropout value was 0.5, the accuracy of the model with the addition of five dense layers during validation was 1% superior. The worst model from this experiment

was the model with the addition of three dense layers, with a dropout value of 0.3% and an accuracy of 89.33%. The best confusion matrix model is in Figure 6.

**Table 1.** Experimental results.

| Dropout | Number of Dense Layer | Number of Neurons | Accuracy |
|---|---|---|---|
| 0.3 | 0 | 0 | 92.22% |
| 0.3 | 1 | 256 | 91.33% |
| 0.3 | 3 | 512; 256; 128 | 89.33% |
| 0.3 | 5 | 512; 256; 128; 64; 32 | 93.11% |
| 0.4 | 0 | 0 | 92.88% |
| 0.4 | 1 | 256 | 91.77% |
| 0.4 | 3 | 512; 256; 128 | 92.44% |
| 0.4 | 5 | 512; 256; 128; 64; 32 | 92.22% |
| 0.5 | 0 | 0 | 93.78% |
| 0.5 | 1 | 256 | 93.33% |
| 0.5 | 3 | 512; 256; 128 | 93.55% |
| 0.5 | 5 | 512; 256; 128; 64; 32 | 93.78% |

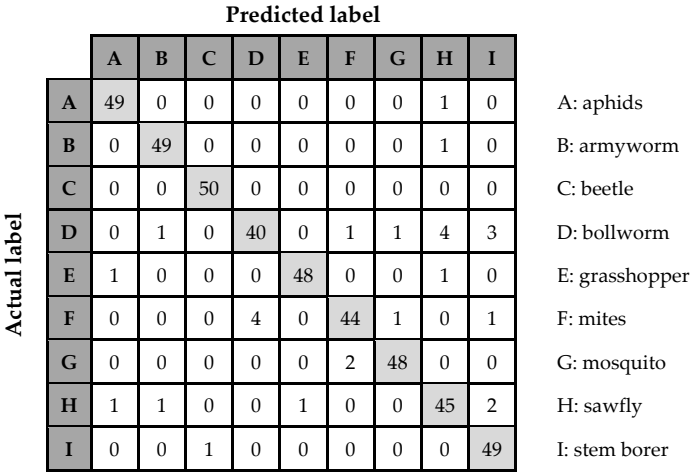
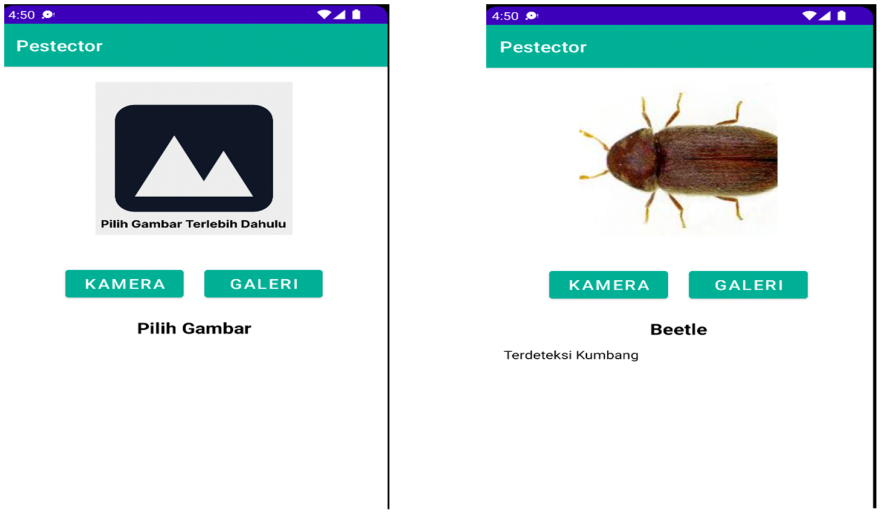



**Figure 6.** Confusion matrix.

In this case, almost all hyperparameters were stable in the range of 91–93% except for the hyperparameter pair number with 3 dense layers and a dropout value of 0.3, which had an accuracy of 89%. Models that experience the addition of five dense layers tend to be stable, with an accuracy in the range of 92–93%. With an increase in the number of dense layers, the model was able to take in deeper features in the image. However, if it took too many features from an image, overfitting occurred. If the model was too focused on a particular training dataset, it made predictions correctly when given another, similar dataset. Furthermore, in this case, the dropout value had no significant effect on producing an optimal model.

The stages of system implementation were carried out by making a prototype application to detect pests on plants. The prototype of this application was based on the Android operating system. In this application prototype, there are four main views: pages from the main menu, pest detection, how to deal with pests, and application description. The application was shown to farmers in Indonesia so that the language of instruction is the national language, namely Indonesian. A detection page is a page containing the primary application features that will be provided to end users, as shown in Figure 7. It exhibits a preview of the detected images and their classification based on nine labels extracted from the dataset. The detection page has two options to select the type of input that will be used for detection. The two categories of input are direct capture with the mobile device's camera and taking images from the gallery. This page employs a pre-built deep-learning model for detection.




(**a**) (**b**)

**Figure 7.** The mobile application is integrated with a pest detection system utilizing GoogleNet. The purpose of this application is to answer the needs of Indonesian farmers, who predominantly communicate in the Indonesian language, by presenting its content in Indonesian. (**a**) Illustrates the user interface for inputting images to be detected. The "Kamera" function refers to the capability of capturing images by utilizing the built-in camera on a mobile device. In contrast, the "Galeri" feature facilitates the inclusion of images from the pre-existing collection into the user's mobile gallery. (**b**) Displays the program interface that exhibits the detected outcomes of the pests that were examined.

This study evaluated the application using a sample size of 36 participants. Table 2 displays the roster of inquiries. The participants in the study included 15 males and 21 females, ranging in age from 16 to 43 years old. The majority of participants were residing in Java and Bali, which are two regions in Indonesia known for their extensive agricultural areas. This survey utilized a Likert scale consisting of 10 assertions, each accompanied by a range of response options that gauged the level of agreement or disagreement with a topic. A rating of 1 signified a strong disagreement, while a rating of 5 signified a strong agreement.

**Table 2.** Survey results.

| No | Questions | Mean | Median | Variation | Min. | Max. |
|---|---|---|---|---|---|---|
| 1 | Do you think this application will be effective in detecting pests? | 4.36 | 4.00 | 0.13 | 3 | 5 |
| 2 | Will this application help in identifying pest problems in plants? | 4.31 | 4.00 | 0.15 | 3 | 5 |
| 3 | Are you satisfied with the performance of this application in detecting pests? | 4.08 | 4.00 | 0.19 | 2 | 5 |
| 4 | Do you like the features of this app? | 4.11 | 4.00 | 0.18 | 2 | 5 |
| 5 | Is this application fast and accurate in detecting pests? | 3.89 | 4.00 | 0.19 | 2 | 5 |
| 6 | Can this app detect pests better than humans? | 3.78 | 4.00 | 0.19 | 3 | 5 |
| 7 | Will this application increase efficiency or productivity in agriculture? | 4.31 | 4.00 | 0.17 | 3 | 5 |
| 8 | Will this application make a positive contribution to overcoming pest problems in plants or the surrounding environment? | 4.36 | 4.50 | 0.16 | 3 | 5 |
| 9 | Is this application easy to use? | 4.36 | 5.00 | 0.17 | 3 | 5 |
| 10 | Would you recommend this app to others, especially farmers? | 4.33 | 5.00 | 0.19 | 3 | 5 |

The value 2 appears three times in all responses in the survey. The interviews with the three individuals revealed, respectively, discontentment with the application being available only on Android and a desire for the application to offer many more functionalities to assist farmers in their work; furthermore, one respondent highlighted the lack of accuracy in real-time testing. Nevertheless, the survey findings from the 36 participants indicated a favorable outlook in terms of their contentment with the program. This is evidenced by the fact that the average score falls between the range of 3.78 to 4.36.

## 4. Conclusions

Using the GoogLeNet architecture, this research tries to apply the CNN method to develop an application that can detect pests in plants. In its implementation, this research showed that the convolution function, which extracted features pixel by pixel in the image, allowed CNN to detect pests on plants. Adding a thick layer improved the accuracy of the model; the number of thick layers made the model extract more features from the dataset, but adding too many thick layers caused the overfitting of the model. The layer dropout value of 0.5 was the most accurate but did not show significant changes between models. The model with an accuracy of 93.78% was the best in this research. By entering the GoogLeNet model into an application, an application can be created to detect pests in plants on Android devices. This application was developed on a smartphone basis so that Indonesian farmers could easily identify pests on their crops. This application is also presented in Indonesian, which is the national language of Indonesia, so it is easy to use. It is the main contribution of this research. It is hoped that this application will be a breakthrough for Indonesian farmers. By identifying pests more easily, farmers can treat pests more appropriately, which will ultimately result in better crop yields.

This research aims to help researchers and practitioners face the challenges that come with real-world implementation by covering topics such as more heterogeneous data, a focus on performance in limited-power devices, integration in end-to-end systems, security, end-user evaluation, the development of custom algorithms, and ongoing evaluation. This study has the potential to be extended further into applications that are more practical by increasing the scale of the experiment. This is also supported by an interview conducted with a respondent who is a staff member from the Plant Quarantine Agency of the Indonesian government. The suggestion is that the impact can be enhanced if the pests listed for detection also encompass the sorts of pests that currently exist in Indonesia but are still prevalent in specific regions. Due to the current process, this task is currently being performed manually by the agency. Therefore, it is possible to ensure that this research offers maximum advantages in a variety of practical applications while ensuring security, privacy, and a good influence on society if a commitment is made to following the relevant rules and standards and taking an ethical approach.

**Author Contributions:** Conceptualization, I.N.Y.; methodology, I.N.Y., M.F.R.R. and A.S.; software, M.F.R.R.; validation, I.N.Y. and A.S.; formal analysis, I.N.Y.; investigation, I.N.Y. and M.F.R.R.; resources, M.F.R.R.; data curation, I.N.Y. and A.S.; writing—original draft preparation, I.N.Y., M.F.R.R. and A.S.P.; writing—review and editing, I.N.Y. and A.S.P.; visualization, M.F.R.R.; supervision, A.S.P.; project administration, I.N.Y.; funding acquisition, I.N.Y. All authors have read and agreed to the published version of the manuscript.

**Funding:** The Associate Professor Acceleration Research 2023 initiative at the University of Padjadjaran provided funding for this project, No. Contract: 1549/UN6.3.1/PT.00/2023.

**Data Availability Statement:** Publicly available datasets were analyzed in this study. This data can be found here: https://www.kaggle.com/simranvolunesia/pest-dataset (accessed on 21 February 2022).

**Acknowledgments:** The authors would like to express their sincere gratitude to the Universitas Padjadjaran and the Directorate of Research and Community Service (DRPM). In addition, we would like to express our gratitude to the World Class Professor 2023 initiative, which is run by the Indonesian Ministry of Education, Culture, Research, and Technology. We extend our gratitude to the 36 participants who willingly dedicated their time to assessing our application and engaging in interviews about it.

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

## References


1. Rozaki, Z.; Wijaya, O.; Rahmawati, N.; Rahayu, L. Farmers' disaster mitigation strategies in Indonesia. *Rev. Agric. Sci.* **2021**, *9*, 178–194. [CrossRef]
2. Gandharum, L.; Mulyani, M.E.; Hartono, D.M.; Karsidi, A.; Ahmad, M. Remote sensing versus the area sampling frame method in paddy rice acreage estimation in Indramayu regency, West Java province, Indonesia. *Int. J. Remote Sens.* **2021**, *42*, 1738–1767. [CrossRef]
3. Yatoo, A.M.; Ali, M.N.; Baba, Z.A.; Hassan, B. Sustainable management of diseases and pests in crops by vermicompost and vermicompost tea. A review. *Agron. Sustain. Dev.* **2021**, *41*, 7. [CrossRef]
4. Nuryitmawan, T.R. The Impact of Credit on Multidimensional Poverty in Rural Areas: A Case Study of the Indonesian Agricultural Sector. *Agriecobis J. Agric. Socioecon. Bus.* **2021**, *4*, 32–45. [CrossRef]
5. Safitri, K.I.; Abdoellah, O.S.; Gunawan, B.; Suparman, Y.; Mubarak, A.Z.; Pardede, M. The Adaptation of Export-Scale Urban Farmers Amid the COVID-19 Pandemic in Bandung Metropolitan. *Qual. Rep.* **2022**, *27*, 1169–1196. [CrossRef]
6. Agustina, K.K.; Wirawan, I.M.A.; Sudarmaja, I.M.; Subrata, M.; Dharmawan, N.S. The first report on the prevalence of soil-transmitted helminth infections and associated risk factors among traditional pig farmers in Bali Province, Indonesia. *Vet. World* **2022**, *15*, 1154. [CrossRef]
7. Thao, L.Q.; Cuong, D.D.; Anh, N.T.; Minh, N.; Tam, N.D. Pest Early Detection in Greenhouse Using Machine Learning. *Rev. D'intelligence Artif.* **2022**, *36*, 209–214. [CrossRef]
8. Wang, X.; Liu, J.; Zhu, X. Early real-time detection algorithm of tomato diseases and pests in the natural environment. *Plant Methods* **2021**, *17*, 43. [CrossRef]
9. Gupta, N.; Slawson, D.D.; Moffat, A.J. Using citizen science for early detection of tree pests and diseases: Perceptions of professional and public participants. *Biol. Invasions* **2022**, *24*, 123–138. [CrossRef]
10. Pocock, M.J.; Marzano, M.; Bullas-Appleton, E.; Dyke, A.; De Groot, M.; Shuttleworth, C.M.; White, R. Ethical dilemmas when using citizen science for early detection of invasive tree pests and diseases. *Manag. Biol. Invasions* **2020**, *11*, 720–732. [CrossRef]
11. White, R.; Marzano, M.; Fesenko, E.; Inman, A.; Jones, G.; Agstner, B.; Mumford, R. Technology development for the early detection of plant pests: A framework for assessing Technology Readiness Levels (TRLs) in environmental science. *J. Plant Dis. Prot.* **2022**, *129*, 1249–1261. [CrossRef]
12. Rempelos, L.; Baranski, M.; Wang, J.; Adams, T.N.; Adebusuyi, K.; Beckman, J.J.; Brockbank, C.J.; Douglas, B.S.; Feng, T.; Greenway, J.D.; et al. Integrated soil and crop management in organic agriculture: A logical framework to ensure food quality and human health? *Agronomy* **2021**, *11*, 2494. [CrossRef]
13. Braga, M.P.; Janz, N. Host repertoires and changing insect–plant interactions. *Ecol. Entomol.* **2021**, *46*, 1241–1253. [CrossRef]
14. Yulita, I.N.; Amri, N.A.; Hidayat, A. Mobile Application for Tomato Plant Leaf Disease Detection Using a Dense Convolutional Network Architecture. *Computation* **2023**, *11*, 20. [CrossRef]
15. van der Merwe, D.; Burchfield, D.R.; Witt, T.D.; Price, K.P.; Sharda, A. Drones in agriculture. *Adv. Agron.* **2020**, *162*, 1. [CrossRef]
16. Dutta, G.; Goswami, P. Application of drone in agriculture: A review. *Int. J. Chem. Stud.* **2020**, *8*, 203–218. [CrossRef]
17. Ahirwar, S.; Swarnkar, R.; Bhukya, S.; Namwade, G. Application of Drone in Agriculture. *Int. J. Curr. Microbiol. Appl. Sci.* **2019**, *8*, 2500–2505. [CrossRef]
18. Rejeb, A.; Abdollahi, A.; Rejeb, K.; Treiblmaier, H. Drones in agriculture: A review and bibliometric analysis. *Comput. Electron. Agric.* **2022**, *198*, 107017. [CrossRef]
19. Tannous, M.; Stefanini, C.; Romano, D. A Deep-Learning-Based Detection Approach for the Identification of Insect Species of Economic Importance. *Insects* **2023**, *14*, 148. [CrossRef]
20. Gondal, M.D.; Khan, Y.N. Early pest detection from crop using image processing and computational intelligence. *Int. J. Sci. Res. Eng. Manag.* **2022**, *6*, 59–68. [CrossRef]
21. Hadipour-Rokni, R.; Asli-Ardeh, E.A.; Jahanbakhshi, A.; Esmaili paeen-Afrakoti, I.; Sabzi, S. Intelligent detection of citrus fruit pests using machine vision system and convolutional neural network through transfer learning technique. *Comput. Biol. Med.* **2023**, *155*, 106611. [CrossRef]
22. Kasinathan, T.; Singaraju, D.; Uyyala, S.R. Insect classification and detection in field crops using modern machine learning techniques. *Inf. Process. Agric.* **2021**, *8*, 446–457. [CrossRef]
23. Barbedo, J.G.A. Detecting and Classifying Pests in Crops Using Proximal Images and Machine Learning: A Review. *AI* **2020**, *1*, 21. [CrossRef]
24. Sangeetha, T.; Mohanapriya, M. A Novel Exploration of Plant Disease and Pest Detection Using Machine Learning and Deep Learning Algorithms. *Publ. Issue* **2022**, *71*, 1399–1418.
25. Chithambarathanu, M.; Jeyakumar, M.K. Survey on crop pest detection using deep learning and machine learning approaches. *Multimed. Tools Appl.* **2023**, *11*, 1–34. [CrossRef]
26. Du, L.; Sun, Y.; Chen, S.; Feng, J.; Zhao, Y.; Yan, Z.; Zhang, X.; Bian, Y. A Novel Object Detection Model Based on Faster R-CNN for Spodoptera frugiperda According to Feeding Trace of Corn Leaves. *Agriculture* **2022**, *12*, 248. [CrossRef]
27. Jiao, L.; Xie, C.; Chen, P.; Du, J.; Li, R.; Zhang, J. Adaptive feature fusion pyramid network for multi-classes agricultural pest detection. *Comput. Electron. Agric.* **2022**, *195*, 106827. [CrossRef]

28. Prabha, R.; Kennedy, J.S.; Vanitha, G.; Sathiah, N.; Priya, M.B. Android application development for identifying maize infested with fall armyworms with Tamil Nadu Agricultural University Integrated proposed pest management (TNAU IPM) capsules. *J. Appl. Nat. Sci.* **2022**, *14*, 138–144. [CrossRef]
29. Chen, J.W.; Lin, W.J.; Cheng, H.J.; Hung, C.L.; Lin, C.Y.; Chen, S.P. A smartphone-based application for scale pest detection using multiple-object detection methods. *Electronics* **2021**, *10*, 372. [CrossRef]
30. Panggabean, H.; Tampubolon, S.; Sembiring, M. Indonesia's Ambivalent Language Policy on English: Cause and Effect. *Int. J. Innov. Creat. Change* **2020**, *14*, 588–605. Available online: www.ijicc.net (accessed on 16 August 2023).
31. Hasan, K.; Masriadi; Muchlis; Husna, A. Digital Farming and Smart Farming from the Perspective of Agricultural Students at Malikussaleh University 2022. In Proceedings of the 3rd Malikussaleh International Conference on Multidisciplinary Studies (MICoMS), Virtual, 30 November–1 December 2022. [CrossRef]
32. Yuesheng, F.; Jian, S.; Fuxiang, X.; Yang, B.; Xiang, Z.; Peng, G.; Zhengtao, W.; Shengqiao, X. Circular Fruit and Vegetable Classification Based on Optimized GoogLeNet. *IEEE Access* **2021**, *9*, 113599–1135611. [CrossRef]
33. Muhammad, N.A.; Nasir, A.A.; Ibrahim, Z.; Sabri, N. Evaluation of CNN, alexnet and GoogleNet for fruit recognition. *Indones. J. Electr. Eng. Comput. Sci.* **2018**, *12*, 468–475. [CrossRef]
34. Jayalakshmy, S.; Lakshmipriya, B.; Sudha, G.F. Bayesian optimized GoogLeNet based respiratory signal prediction model from empirically decomposed gammatone visualization. *Biomed. Signal Process Control* **2023**, *86*, 105239. [CrossRef]
35. Loo, M.C.; Logeswaran, R.; Salam, Z.A.A. CNN Aided Surface Inspection for SMT Manufacturing. In Proceedings of the 2023 15th International Conference on Developments in eSystems Engineering (DeSE), Baghdad/Anbar, Iraq, 9–12 January 2023. [CrossRef]
36. Panchbhaiyye, V.; Ogunfunmi, T. A Fifo Based Accelerator for Convolutional Neural Networks. In Proceedings of the ICASSP 2020—2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, 4–8 May 2020. [CrossRef]
37. Xie, Y.; Lu, H.; Yan, J.; Yang, X.; Tomizuka, M.; Zhan, W. Active Finetuning: Exploiting Annotation Budget in the Pretraining-Finetuning Paradigm. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 17–24 June 2023. [CrossRef]
38. Yapıcı, M.M.; Topaloğlu, N. Performance comparison of deep learning frameworks. *Comput. Inform.* **2021**, *1*, 1–11.
39. Saleh, A.M.; Hamoud, T. Analysis and best parameters selection for person recognition based on gait model using CNN algorithm and image augmentation. *J. Big Data* **2021**, *8*, 1. [CrossRef]
40. Shorten, C.; Khoshgoftaar, T.M. A survey on Image Data Augmentation for Deep Learning. *J. Big Data* **2019**, *6*, 60. [CrossRef]
41. Khalifa, N.E.; Loey, M.; Mirjalili, S. A comprehensive survey of recent trends in deep learning for digital images augmentation. *Artif. Intell. Rev.* **2022**, *55*, 2351–2377. [CrossRef] [PubMed]
42. Xu, M.; Yoon, S.; Fuentes, A.; Park, D.S. A Comprehensive Survey of Image Augmentation Techniques for Deep Learning. *Pattern Recognit.* **2023**, *137*, 109347. [CrossRef]
43. 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]
44. Lei, X.; Pan, H.; Huang, X. A dilated cnn model for image classification. *IEEE Access* **2019**, *7*, 124087–124095. [CrossRef]
45. Tuggener, L.; Schmidhuber, J.; Stadelmann, T. Is it enough to optimize CNN architectures on ImageNet? *Front. Comput. Sci.* **2022**, *4*, 1703. [CrossRef]
46. Lu, T.C. CNN Convolutional layer optimisation based on quantum evolutionary algorithm. *Conn. Sci.* **2021**, *33*, 482–494. [CrossRef]
47. Ardison, A.C.I.; Hutagalung, M.J.R.; Chernando, R.; Cenggoro, T.W. Observing Pre-Trained Convolutional Neural Network (CNN) Layers as Feature Extractor for Detecting Bias in Image Classification Data. *CommIT J.* **2022**, *16*, 149–158. [CrossRef]
48. Hao, K.; Lin, S.; Qiao, J.; Tu, Y. A Generalized Pooling for Brain Tumor Segmentation. *IEEE Access* **2021**, *9*, 159283–159290. [CrossRef]
49. Jang, B.; Kim, M.; Harerimana, G.; Kang, S.U.; Kim, J.W. Bi-LSTM model to increase accuracy in text classification: Combining word2vec CNN and attention mechanism. *Appl. Sci.* **2020**, *10*, 5841. [CrossRef]
50. Wang, S.H.; Tang, C.; Sun, J.; Yang, J.; Huang, C.; Phillips, P.; Zhang, Y.D. Multiple sclerosis identification by 14-layer convolutional neural network with batch normalization, dropout, and stochastic pooling. *Front. Neurosci.* **2018**, *12*, 818. [CrossRef]
51. Kora, P.; Ooi, C.P.; Faust, O.; Raghavendra, U.; Gudigar, A.; Chan, W.Y.; Meenakshi, K.; Swaraja, K.; Plawiak, P.; Acharya, U.R. Transfer learning techniques for medical image analysis: A review. *Biocybern. Biomed. Eng.* **2022**, *42*, 79–107. [CrossRef]
52. Yoo, H.-J. Deep Convolution Neural Networks in Computer Vision: A Review. *IEIE Trans. Smart Process. Comput.* **2015**, *4*, 35–43. [CrossRef]
53. Yu, H.; Yang, L.T.; Zhang, Q.; Armstrong, D.; Deen, M.J. Convolutional neural networks for medical image analysis: State-of-the-art, comparisons, improvement and perspectives. *Neurocomputing* **2021**, *444*, 92–110. [CrossRef]
54. Chen, S.L.; Chen, T.Y.; Huang, Y.C.; Chen, C.A.; Chou, H.S.; Huang, Y.Y.; Lin, W.C.; Li, T.C.; Yuan, J.J.; Abu, P.A.; et al. Missing Teeth and Restoration Detection Using Dental Panoramic Radiography Based on Transfer Learning with CNNs. *IEEE Access* **2022**, *10*, 118654–118664. [CrossRef]
55. Yilmaz, E.; Trocan, M. A modified version of GoogLeNet for melanoma diagnosis. *J. Inf. Telecommun.* **2021**, *5*, 395–405. [CrossRef]

56. Lyu, Z.; Yu, Y.; Samali, B.; Rashidi, M.; Mohammadi, M.; Nguyen, T.N.; Nguyen, A. Back-Propagation Neural Network Optimized by K-Fold Cross-Validation for Prediction of Torsional Strength of Reinforced Concrete Beam. *Materials* **2022**, *15*, 1477. [CrossRef] [PubMed]
57. Fushiki, T. Estimation of prediction error by using K-fold cross-validation. *Stat. Comput.* **2011**, *21*, 137–146. [CrossRef]
58. Wong, T.T.; Yeh, P.Y. Reliable Accuracy Estimates from k-Fold Cross Validation. *IEEE Trans. Knowl. Data Eng.* **2020**, *32*, 1586–1594. [CrossRef]