Indonesian Journal of Electrical Engineering and Computer Science
Vol. 35, No. 3, September 2024, pp. 1556~1564
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v35.i3.pp1556-1564

Deep transfer learning classification of apple fruit diseases

Shaimaa Kamal Loutfy[1], Kamel Hussein Rahouma[2]
[1]Department of Electrical and Computer Engineering, Higher Institute of Engineering and Technology, New Minia, Egypt
[2]Department of Electrical Engineering, Faculty of Engineering, Minia University, Minia, Egypt

Article Info

Article history:

Received Dec 25, 2023
Revised Mar 11, 2024
Accepted Apr 16, 2024

Keywords:

Apple diseases
Classification
Data augmentation
Deep convolution
Neural networks

ABSTRACT

This paper applies deep convolution neural networks (DCNN) to apple fruit disease classification. Twelve DCNN methods (SqueezeNet, GoogleNet, InceptionV3, DenseNet201, ReaNet50, ResNet101, Xception, InceptionResnetV2, EfficientnetB0, AlexNet, VGG16, and VGG19) have been used. These methods have been trained to classify apples into four categories: normal, blotch, rot, and scab. A dataset of 5179 images, including 3472 for normal, 171 for blotch, 1166 for rot, and 370 for scab, has been used. A practical test on 120 images (30 for each category) has been applied. Seven of these DCNNs—InceptionV3, DenseNet201, ResNet101, ResNet50, GoogleNet, AlexNet, and VGG16—have the best accuracy. InceptionV3 is the highest. It has achieved an accuracy of 100% for all categories. The used dataset is unbalanced and small. So, it's necessary to use data augmentation to overcome any overfitting that may cause. After applying data augmentation, the dataset is balanced and contains 13888 images (3472 for each category). The seven DCNNs are retrained by the balanced dataset and retested by the same 120 images. All DCNN's accuracy has enhanced except InceptionV3, which has decreased. On the other hand, RasNet101 has achieved an accuracy of 100% for all categories. Therefore, ResNet101 has been recommended for apple fruit disease classification.

This is an open access article under the CC BY-SA license.

Corresponding Author:

Shaimaa Kamal Loutfy
Department of Electrical and Computer Engineering, Higher Institute of Engineering and Technology
New Minia, Minia, Egypt
Email: shklas@gmail.com

1. INTRODUCTION

Apple is an important fruit. It ranks second in fruit production in the world [1]. According to several research, visual exploration with hand approaches is the most commonly utilized method in fruit disease identification. However, it is a time-consuming and difficult process [2]. So, there is an urgent to involve modern technologies such as artificial intelligence (AI). AI includes machine learning (ML) and deep learning (DL). These technologies are faster, less expensive, and more efficient [3]. Whereas DL is an advanced form of ML. It allows systems to train themselves and improve classification accuracy [4]. Also, it is capable of extracting relevant features from raw data without the need for hand-crafted features or prior knowledge [5]. DL methods have been successfully implemented in a wide range of real-world applications, including the agricultural domain [6], [7]. The DL methods can also have various problems such as light scattering, brightness, contrast irregularities, and similarities between the defective lesion area and the normal area. Recently, these problems have been dealt with by using pre-trained deep convolution neural networks (DCNN). Pretrained DCNN methods were created with thousands of classes in very different categories to overcome all of these problems [8]. DCNN can execute non-linear features, automatically detecting the essential features without any human supervision [9]. Pretrained DCNNs such as GoogleNet, AlexNet, VGG, ResNet, and other models are used in fruit disease classification.

Many studies have been performed based on DL during the past few years for apple disease identification [7]. Mohanty *et al.* [10] used two pre-trained DL models, AlexNet and GoogLeNet, for plant disease classification. The model was applied with a dataset containing 54,306 images of diseases and healthy plant leaves. The dataset was collected from the PlantVillage dataset. This model achieved an accuracy of 99.35%. However, the accuracy decreased to 31% when the original dataset was changed. Wang *et al.* [11] classified healthy apple leaves and black rot of apple leaves into three stages. Pretrained DL models (VGG16, VGG19, InceptionV3, and ResNet50) are used. VGG16 model has achieved 90.4% as the highest performance result. Furthermore, Alharbi and Arif [6] applied five different models (Model-1, Model-2, Model-3, Model-4, Model-5) of convolutional neural network (CNN) to classify healthy and three apple diseases (blotch, rot, and scab). The used dataset contains 3200 images (800 images for each category). Data augmentations were applied to the dataset that generated 3200 images for each category. All models showed good classification accuracy on more than 90% of testing images. Model 5 achieved the best accuracy; it gave 99.17%. Al-Shawwa and Abu-Naser [12] introduced a DCNN method that classified 13 different apple species. This method was applied to a dataset that contains 8554 images. The trained model achieved an accuracy of 100%. Ngugi *et al.* [13] illustrate a review of image processing techniques applied for plant leaf disease recognition. Ten DL models (AlexNet, GoogleNet, InceptionV3, SqueezeNet, ResNet101, VGG16, ShuffleNet, Xception, InceptionResnetV2, MobileNetV2, and DenseNet201) are applied with the Plant Village dataset. The authors recommended DenseNet-201 as the best plant disease detection. On the other hand, ShuffleNet and SqueezeNet are suitable for real-time mobile applications.

Previous studies applied DCNN methods to classify apple leaf diseases, not apple fruits. Therefore, the aim here is to apply these DCNNs on the fruit itself. This study collected most of the pre-trained DCNNs that were mostly used in previous research. It applied 12 different pre-trained DCNNs (SqueezeNet, GoogleNet, InceptionV3, DenseNet201, ReaNet50, ResNet101, Xception, InceptionResnetV2, EfficientnetB0, AlexNet, VGG16, and VGG19) to classify apples into 4 categories; normal, blotch, rot and scab. The pre-trained DCNNs are trained on a dataset that contains 5,179 images which included 3,472 for normal, 171 for blotch, 1,166 for rot, and 370 for scab. The trained models are tested using 120 images which included 30 images for each category. It noted that the dataset is unbalanced where it has some classes containing more instances than others and it can exhibit bias towards these classes. Thus, it may ignore the minority classes. For this reason, the evaluation matrices could be affected. On the other hand, the DCNNs could achieve better classification accuracy on large-scale datasets. So, the data augmentation technique has been applied to increase the dataset size. the new balanced dataset contains 13888 images which includes 3472 images for each category. The best 7 DCNNs obtained methods from the training process that was carried out before are retrained using the balanced dataset.

The remaining sections are organized as follows: section 2 proposed the used methods and explained the DCNN architecture. Section 3 provides the practical experiments and dissection. Finally, the conclusion will be introduced in section 4.

2. METHODS

The proposed study in this paper is divided into two main stages: train and test stage. The training stage has pre-trained 12 DCNNs (SqueezeNet, GoogleNet, InceptionV3, DenseNet201, ReaNet50, ResNet101, Xception, InceptionResnetV2, EfficientnetB0, AlexNet, VGG16, and VGG19) on a dataset containing 5,197 images (3,472 for normal, 171 for blotch, 1,166 for rot, and 370 for scab). The dataset images are different in size. However, the input layer in each DCNN must be in a specific size as mentioned in Table 1. So, it's necessary to apply a resizing process. Then, the 12 DCNNs were trained to generate their trained models. The block diagram of this stage is shown in Figure 1. The procedure of this stage has been described by Algorithm 1.

The test stage has tested the trained models using 120 images (30 for each category). Each image has been resized. Then it has been classified by the trained models. Finally, the performance measurements (accuracy, sensitivity, specificity, and F1-Force) have been calculated. Figure 2 shows the block diagram of the test stage. Algorithm 2 describes the steps of this stage. According to the results, the InceptionV3 model has an accuracy of 100% for all categories. Which could be a result of overfitting. Overfitting could happen when the categories in the dataset are unbalanced. Thus, it is necessary to apply data augmentation to balance the dataset. After applying data augmentation on the dataset, the balanced dataset contains 13,888 apple images that include 3,472 images for each category. Seven of the trained models (InceptionV3, DenseNet201, ResNet101, ResNet50, GoogleNet, AlexNet and VGG16) which achieved the highest accuracy were selected to retrain them with a balanced dataset. Figure 3 proposed the block diagram of the training stage with the balanced dataset. Algorithm 3 describes the steps of the training stage with the balanced dataset. Finally, the test stage has reapplied on the seven selected models.

In this section, we describe the dataset that has been used. Then explain the data augmentation process to balance the dataset. Then describe the DCNN structure. Finally, we describe the performance measurements that evaluate the proposed study.

Table 1. Input layer size of pre-trained DCNN models

Networks	Size
SqueezeNet	27×227
GoogleNet	24×224
InceptionV3	299×299
DenseNet201	224×224
ResNet50	224×224
ReseNet101	224×224
Xception	299×299
InceptionRes-netV2	299×299
EfficientNet- B0	224×224
AlexNet	227×227
VGG16	224×224
VGG19	224×224

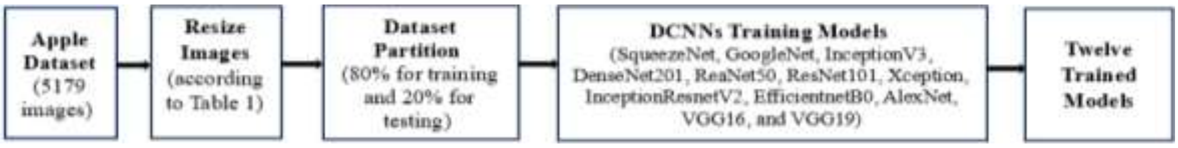

Figure 1. The block diagram of the train stage

Algorithm 1. Describes the train stage

```
Step 1: Read apple images from the dataset.
Step 2: Resize images according to DCNN's input layer size.
Step 3: Split dataset to 80% for training and 20% for testing.
Step 4: Extrate features for each one of the 12 DCNN models.
Step 5: Optimize and fitting functions that are used to train DCNNs.
Step 6: Generate the 12 DCNN Models.
```

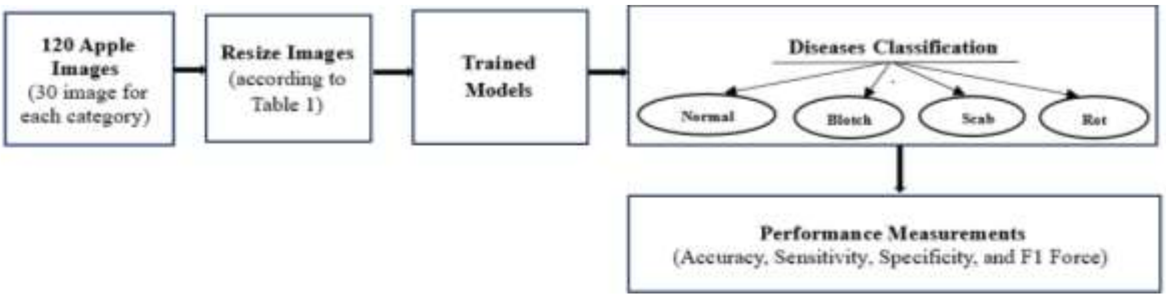

Figure 2. The block diagram of the test stage

Algorithm 2. Describes the test stage

```
Step 1:  Read 120 apple images.
Step 2:  Resize images according to DCNN's input layer size.
Step 3:  Applied the test images to each trained model that was generated from the training
         stage.
Step 4:  Classify images into 4 categories (Normal, Blotch. Rot, and Scab).
Step 5:  Calculate the system performance.
```

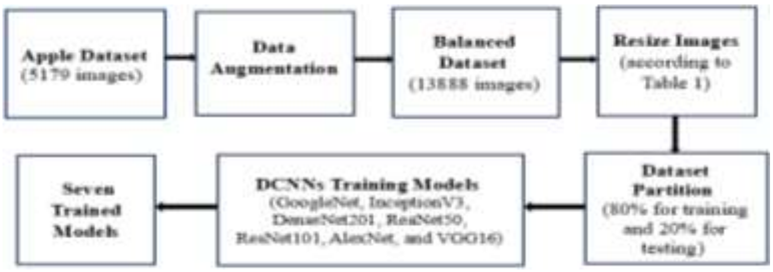

Figure 3. The block diagram of the test stage using a balanced dataset

Algorithm 3. Describes the training stage using a balanced dataset
```
Step 1: Read apple images from the dataset.
Step 2: Applied data augmentation to increase dataset images and balance the dataset.
Step 3: Resize images according to DCNN's input layer size.
Step 4: Split dataset to 80% for training and 20% for testing.
Step 5: Extrate features for each one of the 7 DCNN models.
Step 6: Optimizing and fitting functions that are used to train DCNNs.
Step 7: Generate the trained models.
```

2.1. Dataset

The used dataset contains 5,179 apple fruit images. This dataset was divided into 4 categories, including 3,472 for normal, 171 for blotch, 1,166 for rot, and 370 for scab. This dataset was collected from the Google Images website and datasets from GitHub. It is difficult to obtain abundant images of diseases due to the random occurrence of apple diseases. Therefore, the number of collected apple images of blotch and scab are far less than that of other types of diseases.

2.2. Data augmentation

The performance of DCNN is highly dependent on the amount of data used for training a model. Data augmentation is a technique used to generate photos for the training process and balance the different dataset classes. This technique increases the size of the dataset by applying some of the traditional augmentation methods, such as brightness transformation, horizontal and vertical shift, cropping, shearing, flipping, and zooming of training images, rotation, and affine transformations [14].

2.3. Deep convolution neural network (DCNN)

DCNNs are special types of artificial neural networks (ANNs) that learn hierarchical representation from the spatial information contained in digital images. DCNNs have great success in the field of image classification [15]. The structure of DCNN is shown in Figure 4. The DCNN consists of an input layer, convolution layers, pooling layers, fully connected layers, and an output layer. In this study, twelve pre-trained DCNN models will be fine-tuned. Table 2 presents the depth, size, and number of parameters in each model.

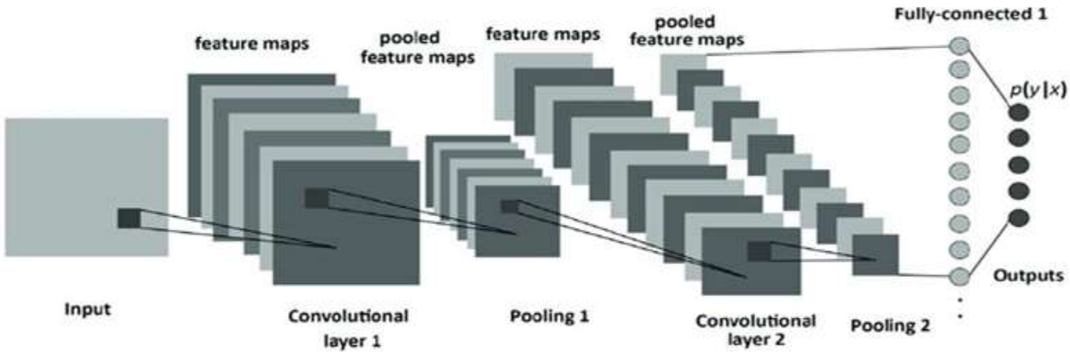

Figure 4. Structure of DCNN [16]

Table 2. Properties of pre-trained DCNN models

Networks	Depth	Size (MB)	Parameters (Million)
SqueezeNet	18	5.2	1.24
GoogleNet	22	27	7
InceptionV3	48	89	23.9
DenseNet201	201	77	20
ResNet50	50	96	25.6
ReseNet101	101	167	44.6
Xception	71	85	22.9
InceptionRes-netV2	164	209	55.9
EfficientNet-B0	82	20	5.3
AlexNet	8	227	61
VGG16	16	515	138
VGG19	19	535	144

2.3.1. Convolution layers

The convolutional layer is the core of CNN. It is responsible for the main processes of training. It contains many convolution filters which perform the convolution operation over the input image. Mathematically, two matrices are convolved using (1). In this equation, S and Z are the convolving variables. In image processing, S and Z are two matrices which represent an image and a filter mask respectively that which is represented in (2) in the discrete case. CNNs usually perform 2D convolution on images as shown in (3). After performing convolution, the feature map was generated [17].

$$(S * Z)(t) = \int_{-\infty}^{\infty} S(t-T)Z(T)dT \quad (1)$$

$$(S * Z)(n) = \int_{m=-\infty}^{\infty} S(n-m)Z(m) \quad (2)$$

$$(S * Z)(x,y) = \sum_{m=-M}^{M} \sum_{n=-N}^{N} \mathrm{S}(\mathrm{x}-\mathrm{n},\mathrm{y}-\mathrm{m})\mathrm{Z}(\mathrm{n},\mathrm{m}) \quad (3)$$

2.3.2. Pooling layer

The max pooling layer is placed after the convolution layer. It is considered as a nonlinear down-sampling. It is used to reduce the number of feature maps generated from convolution layers [18]. As shown (4) describes the max operation. Where 'a' is the feature map, 's' is the output pooled feature map, and 'Rj' is the pooling region j [19].

$$sj = \max_{i \in Rj} ai \quad (4)$$

2.3.3. Fully connected layers

It is a layer which fully connected to the output of the previous layer. This layer converts the previous layer's output to a single vector and applies weights to predict the class label. It is usually used in the CNN final stages.

2.3.4. Optimization algorithms

The DCNN model is trained by iteratively updating the parameters of all layers in the network, with the optimizer method playing a pivotal role. There are several optimization algorithms used during the training of the dataset including adaptive moment estimation (Adam), stochastic gradient descent (SGD), RMSprop, AdaGrad, and Adadelta. In this study, Adam optimizer is used to train the model. Adam optimizer is a combination of RMSprop and stochastic gradient descent. Learning rates are modified as done in RMSprop optimizer and momentum term by taking a moving average of the gradients like SGD with momentum. The main advantage of this optimizer is that it requires low memory and works well even with a slight tuning of hyper-parameters [20].

2.4. Performance measures

The performance of the apple classifier is evaluated based on the overall classification accuracy, specificity, sensitivity (recall), and F1-Force. Accuracy is the measure of the ratio of correct prediction that can be represented by (5) [21], sensitivity (recall) is defined by (6) [21], specificity is given by (7) [22]. F1-Score is the combination of Precision and Sensitivity. It is defined by (8) [23]. In the following equations, TP, TN, FP, and FN correspond to true positive, true negative, false positive and false negative values respectively.

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \quad (5)$$

$$Sensitivity\ (Recall) = \frac{TP}{TP+FN} \quad (6)$$

$$Specificity = \frac{TN}{FP+\ TN} \quad (7)$$

$$1 - Score = \frac{2TP}{2TP+FP+FN} \quad (8)$$

2.5. Details of model training

All training models were trained and tested on the same dataset. The dataset is divided into 80% for training and 20% for validation. Then, they were tested with 120 images (30 images for each category). The training parameters for all models were unified. All pre-trained models used the Adam optimizer with an initial learning rate of 1e-4, The number of epochs is equal to 5 and a batch size of 16. The performance measurements (accuracy, sensitivity (recall), specificity, and F1-Force) were calculated for each model.

3. RESULTS AND DISCUSSION

This study was performed on a personal computer. It has an Intel Core™ i7-2630QM CPU with a processing speed of 2.00 GHz. That operates on 8 GB of RAM and a NVIDIA GeForce GT 525M. The DCNN models have been trained and tested on MATLAB (R2021b) program.

3.1. Results

Most previous studies focus on using DL to classify the apple leaf diseases. While our proposed study has classified apple fruit diseases using various pre-trained DCNN models. These models were trained and tested twice. The first training and testing have been done according to Algorithm 1 and Algorithm 2 respectively as mentioned in section 2. The results of the first test are shown in Figure 5. That shows the performance measurements for the 12 trained models. Hence, the second training and testing have been done according to Algorithm 2 and Algorithm 3 respectively as mentioned in section 2. Figure 6 shows a comparison of the performance of the trained DCNN models.

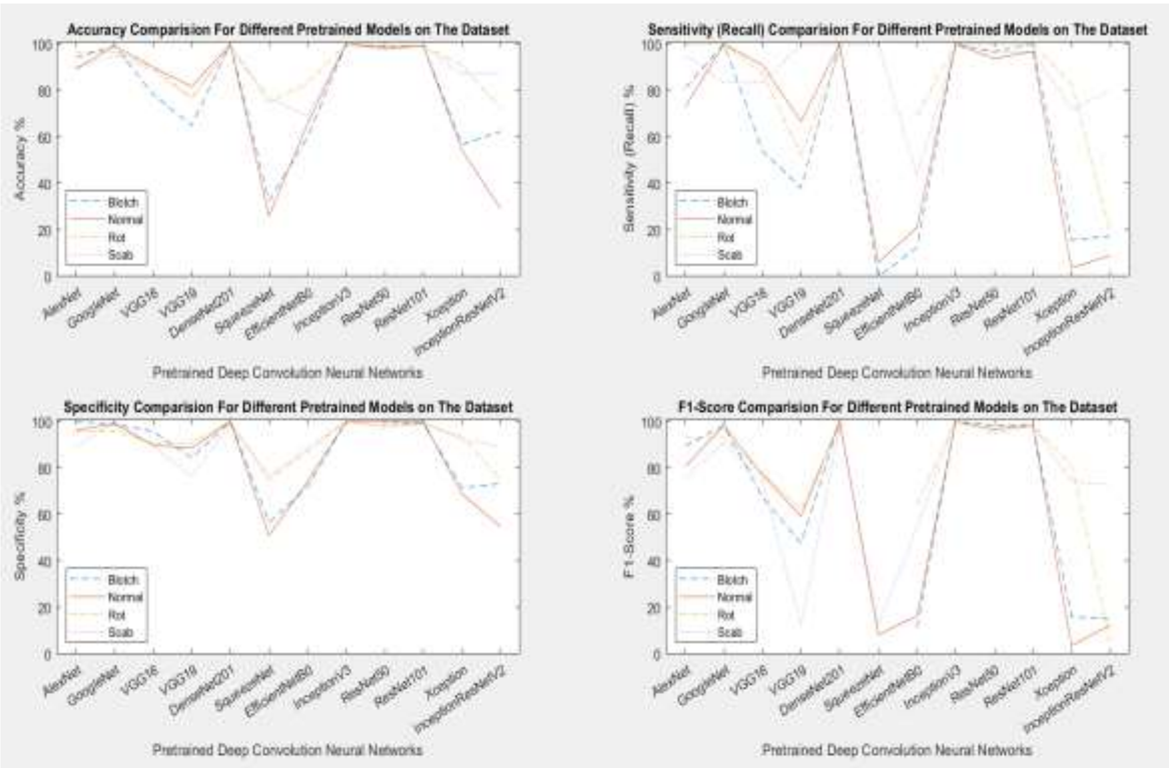

Figure 5. The accuracy comparison between the different DCNNs applied to the dataset

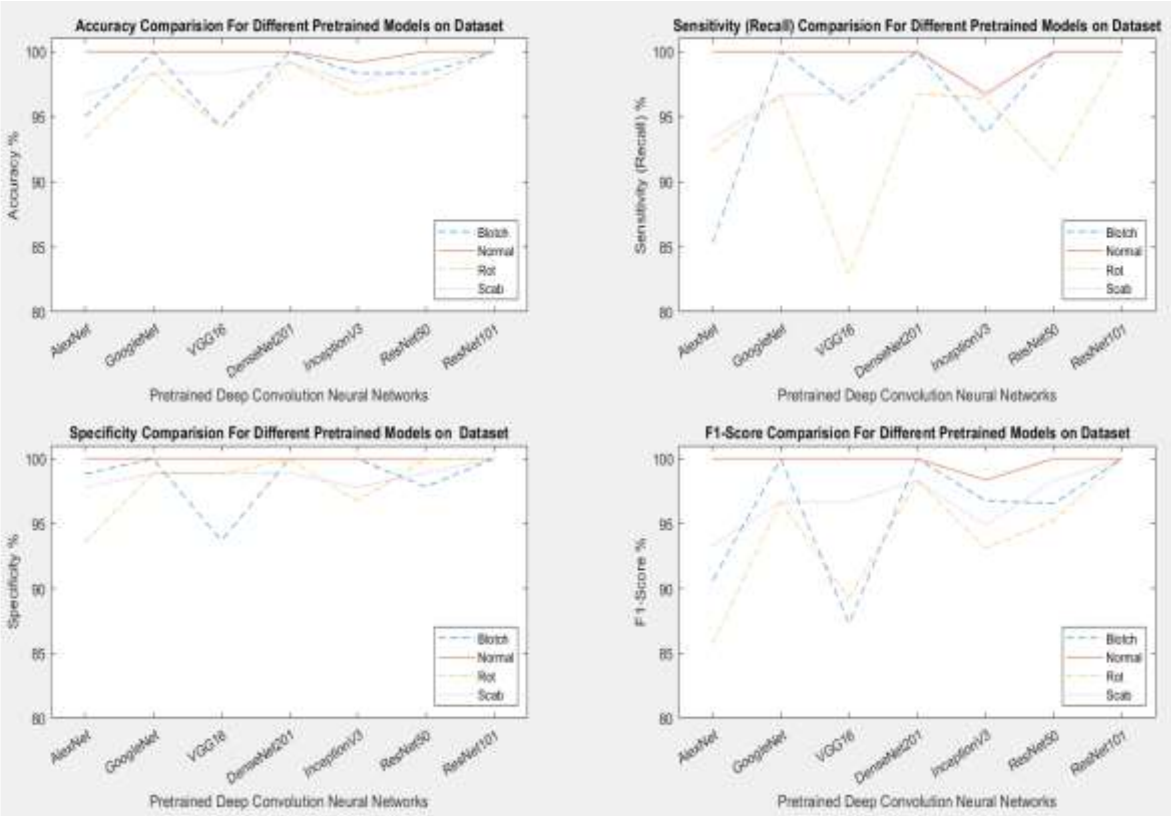

Figure 6. The accuracy comparison between the different DCNNs after applied data augmentation

3.2. Discussion

According to the results in Figure 5, Seven of the trained models have the highest accuracy. InceptionV3 model has the highest accuracy in classifying the normality and all apple diseases of the tested images. It achieved an accuracy of 100%, specificity of 100%, sensitivity of 100%, and F-1 Force of 100%. Hence, DenseNet201 and ResNet101 occupied the second rank, followed by ResNet50 and GoogleNet in the third level where the AlexNet and VGG16 could take the fourth level. The result of InceptionV3 was doubting an overfitting might have occurred. Therefore, it is necessary to balance the dataset and increase the images in the dataset. Then the Seven DCNN models were retrained with the balanced dataset. Figure 6 shows the performance comparison of these seven models. According to the results in Figure 6, we can conclude that the ResNet101 model has the best results in classifying the normality and all apple diseases of the tested images. It achieved an accuracy of 100%, specificity of 100% and sensitivity of 100%, and F-1 Force of 100%. The comparison between Figure 5 and Figure 6 shows that the use of data augmentation improves the results of GoogleNet, VGG16, ResNet50 and ResNet101 but it does not affect the results of DenseNet201. The InceptionV3 results decreased after balancing the dataset. The usage of ResNet101 on the balanced dataset achieved an accuracy of 100% for all categories improving the results in [6], [24], [25] as mentioned in Table 3. Therefore, this study recommended ResNet101 to classify apple diseases. Future research may combine our data on apple fruit and prior work on apple leaf to create a system for apple disease classification. Also, Future research may use DCNNs to develop a system capable of classifying any fruit disease.

Table 3. Comparison with previous works

Reference	Accuracy %	Sensitivity %	Specificity %
Alharbi *et al.* [6]	99.17%	-	-
Dubey *et al.* [24]	99.99%	98.34%	99.45
Tian *et al.* [25]	94.72%	-	-
In this study	100%	100%	100%

4. CONCLUSION

This paper focuses on classifying healthy apples and three common apple diseases (blotch, scab, and rot). That shows the effectiveness of using DCNN. We have applied twelve models of DCNN with the dataset to classify apple fruit diseases. Seven of the twelve models achieved the best results. The highest accuracy of 100% was achieved by the InceptionV3 model for real practical tests. The dataset is small and unbalanced, so it needs to use data augmentation. After training the best seven DCNNs with the balanced dataset, the best accuracy was achieved by the ResNet101 model which is 100%. The effect of increasing and balancing the dataset improved the accuracy of ResNet101 to 100% and increased the accuracy of GoogleNet, VGG16, and ResNet50. On the other hand, it did not change the results of DenseNet201, and it decreased the accuracy of InceptionV3. Therefore, this study recommended ResNet101 to classify apple diseases.

REFERENCES

[1] S. Musacchi and S. Serra, “Apple fruit quality: Overview on pre-harvest factors,” *Scientia Horticulturae*, vol. 234, pp. 409–430, Apr. 2018, doi: 10.1016/j.scienta.2017.12.057.
[2] J. G. Arnal Barbedo, “Digital image processing techniques for detecting, quantifying and classifying plant diseases,” *Springerplus*, vol. 2, no. 1, p. 660, Dec. 2013, doi: 10.1186/2193-1801-2-660.
[3] M. Alruwaili, S. A. El-Ghany, and A. Shehab, “An enhanced plant disease classifier model based on deep learning techniques,” *International Journal of Engineering and Advanced Technology*, vol. 9, no. 1, pp. 7159–7164, Oct. 2019, doi: 10.35940/ijeat.A1907.109119.
[4] Ş. Işık and K. Özkan, “Overview of handcrafted features and deep learning models for leaf recognition,” *Journal of Engineering Research*, vol. 9, no. 1, Feb. 2021, doi: 10.36909/jer.v9i1.7737.
[5] L. Li, S. Zhang, and B. Wang, “Plant disease detection and classification by deep learning—a review,” *IEEE Access*, vol. 9, pp. 56683–56698, 2021, doi: 10.1109/ACCESS.2021.3069646.
[6] A. G. Alharbi and M. Arif, “Detection and classification of apple diseases using convolutional neural networks,” in *2020 2nd International Conference on Computer and Information Sciences (ICCIS)*, Oct. 2020, pp. 1–6, doi: 10.1109/ICCIS49240.2020.9257640.
[7] A. Fuentes, S. Yoon, S. Kim, and D. Park, “A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition,” *Sensors*, vol. 17, no. 9, p. 2022, Sep. 2017, doi: 10.3390/s17092022.
[8] H. ÇETİNER, “Classification of apple leaf diseases using the proposed convolution neural network approach,” *Mühendislik Bilimleri ve Tasarım Dergisi*, vol. 9, no. 4, pp. 1130–1140, Dec. 2021, doi: 10.21923/jesd.980629.
[9] M. Ahmad, M. Mazzara, and S. Distefano, “Regularized CNN feature hierarchy for hyperspectral image classification,” *Remote Sens (Basel)*, vol. 13, no. 12, p. 2275, Jun. 2021, doi: 10.3390/rs13122275.
[10] S. P. Mohanty, D. P. Hughes, and M. Salathé, “Using deep learning for image-based plant disease detection,” *Front Plant Sci*, vol. 7, Sep. 2016, doi: 10.3389/fpls.2016.01419.
[11] G. Wang, Y. Sun, and J. Wang, “Automatic image-based plant disease severity estimation using deep learning,” *Computational Intelligence and Neuroscience*, vol. 2017, pp. 1–8, 2017, doi: 10.1155/2017/2917536.
[12] Mohammed O, Al-Shawwa, and Samy S. Abu-Naser, “Classification of apple fruits by deep learning,” *International Journal of Academic Engineering Research (IJAER)*, vol. 3, no. 12, pp. 1–7, 2019.
[13] L. C. Ngugi, M. Abelwahab, and M. Abo-Zahhad, “Recent advances in image processing techniques for automated leaf pest and disease recognition – A review,” *Information Processing in Agriculture*, vol. 8, no. 1, pp. 27–51, Mar. 2021, doi: 10.1016/j.inpa.2020.04.004.
[14] C. Shorten and T. M. Khoshgoftaar, “A survey on image data augmentation for deep learning,” *Journal of Big Data*, vol. 6, no. 1, p. 60, Dec. 2019, doi: 10.1186/s40537-019-0197-0.
[15] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.
[16] A. Khan, J. Ali Shah, K. Kadir, W. Albattah, and F. Khan, “Crowd monitoring and localization using deep convolutional neural network: a review,” *Applied Sciences*, vol. 10, no. 14, p. 4781, Jul. 2020, doi: 10.3390/app10144781.
[17] S. Albelwi and A. Mahmood, “A framework for designing the architectures of deep convolutional neural networks,” *Entropy*, vol. 19, no. 6, p. 242, May 2017, doi: 10.3390/e19060242.
[18] A. Giusti, D. C. Ciresan, J. Masci, L. M. Gambardella, and J. Schmidhuber, “Fast image scanning with deep max-pooling convolutional neural networks,” in *2013 IEEE International Conference on Image Processing*, Sep. 2013, pp. 4034–4038. doi: 10.1109/ICIP.2013.6738831.
[19] M. D. Zeiler and R. Fergus, “Stochastic pooling for regularization of deep convolutional neural networks,” *arXiv preprint arXiv:1301.3557*, 2013, doi: 10.48550/arXiv.1301.3557.
[20] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
[21] A. ÖZCAN and E. DÖNMEZ, “Bacterial disease detection for pepper plant by utilizing deep features acquired from DarkNet-19 CNN Model,” *DÜMF Mühendislik Dergisi*, pp. 573–579, Sep. 2021, doi: 10.24012/dumf.1001901.
[22] D. Maheshwari, B. Garcia-Zapirain, and D. Sierra-Soso, “Machine learning applied to diabetes dataset using quantum versus classical computation,” in *2020 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, IEEE, Dec. 2020, pp. 1–6. doi: 10.1109/ISSPIT51521.2020.9408944.
[23] A. I. Khan, S. M. K. Quadri, and S. Banday, “Deep learning for apple diseases: classification and identification,” *International Journal of Computational Intelligence Studies*, vol. 10, no. 1, p. 1, 2021, doi: 10.1504/IJCISTUDIES.2021.113831.
[24] S. R. Dubey and A. S. Jalal, “Detection and classification of apple fruit diseases using complete local binary patterns,” in *2012 Third International Conference on Computer and Communication Technology*, IEEE, Nov. 2012, pp. 346–351, doi: 10.1109/ICCCT.2012.76.
[25] Y. Tian, E. Li, Z. Liang, M. Tan, and X. He, “Diagnosis of typical apple diseases: a deep learning method based on multi-scale dense classification network,” *Front Plant Sci*, vol. 12, Oct. 2021, doi: 10.3389/fpls.2021.698474.

BIOGRAPHIES OF AUTHORS

Shaimaa Kamal Loutfy graduated from the Faculty of Engineering Computers and Systems, Minia University, in 2010. She received her master’s degree in 2018. She is currently pursuing a PhD degree with Minia University. She also works as an Assistant Lecturer with the El Minya High Institute for Engineering and Technology. She can be contacted at email: shklas@gmail.com.

Kamel Hussien Rahouma earned his B.Sc. and M.Sc. in Communications and Electronic Engineering from the Faculty of Engineering, Cairo University in June 1984, and March 1988 respectively. He earned his first doctoral degree in Communications and Electronics Engineering jointly from the University of Kent at Canterbury (England) and Minia University (Egypt) in 1996. In May 2001, he earned his second doctoral degree in Computer Science from the University of Salzburg, Austria. Currently, he is the head of the Computer Science Department, Obour Higher Institute of Administration, Computers, and Information Systems, Egypt. He can be contacted at email: kamel_rahouma@yahoo.com.