Scientific Research Publishing

Journal of Computer and Communications, 2020, 8, 10-22
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Plant Disease Detection with Deep Learning and Feature Extraction Using Plant Village

Faye Mohameth[1], Chen Bingcai[1,2], Kane Amath Sada[1]

[1]School of Computer Science & Technology, Dalian University of Technology, Dalian, China
[2]School of Computer Science & Technology, Xinjiang Normal University, Urumqi, China
Email: tetofayeman@gmail.com, cbc9@qq.com, kaneamathsada@gmail.com

How to cite this paper: Mohameth, F., Bingcai, C. and Sada, K.A. (2020) Plant Disease Detection with Deep Learning and Feature Extraction Using Plant Village. *Journal of Computer and Communications*, **8**, 10-22.
https://doi.org/10.4236/jcc.2020.86002

Received: March 27, 2020
Accepted: June 14, 2020
Published: June 17, 2020

Abstract

Nowadays, crop diseases are a crucial problem to the world's food supplies, in a world where the population count is around 7 billion people, with more than 90% not getting access to the use of tools or features that would identify and solve the problem. At present, we live in a world dominated by technology on a significant scale, major network coverage, high-end smart-phones, as long as there are great discoveries and improvements in AI. The combination of high-end smart-phones and computer vision via Deep Learning has made possible what can be defined as "smartphone-assisted disease diagnosis". In the area of Deep Learning, multiple architecture models have been trained, some achieving performance reaching more than 99.53% [1]. In this study, we evaluate CNN's architectures applying transfer learning and deep feature extraction. All the features obtained will also be classified by SVM and KNN. Our work is feasible by the use of the open source Plant Village Dataset. The result obtained shows that SVM is the best classifier for leaf's diseases detection.

Keywords

Plant Diseases Detection, Feature Extraction, Transfer Learning, SVM, KNN

1. Introduction

Crop disorders have always been one of the major concerns for farmers. It can be a considerable threat to the farming production capacity. However, being able to detect the true source of the problem with precise and accurate diagnosis could be of a great help in the field of agriculture.

Lately, there have been several researches in this area. Some of them have studied traditional architectures which sometimes are no longer up to date, or used features that are slow and time-consuming. Mohanty *et al.* [1] applied transfer learning approach by using pre-trained Alex Net to classify new categories of images. It was able to classify 26 different diseases in 14 crop species using 54,306 images with a classification accuracy of 99.35%. Google Net is deeper than the Alex Net with 22 layers and consists of an inception module, which is designed using a network approach. Brahimi *et al.* [2] used Alex Net and Google Net to classify eight different tomato diseases. Being able to use an automated computational system that diagnoses and therefore detects the diseases would be a significant help and relief for the agriculturist who is asked to perform such diagnoses task through optical observation of leaves of infected plants. Building a system or a platform accessible via a mobile device could be of immeasurable help to farmers who do not have access to the necessary resources and logistics.

This idea can be extended for plant disease detection systems to manage and monitor wirelessly in a large-scale agriculture production with the use of drones for surveillance, the use of sensors for managing the quantity of water, as well as fertilizers and light necessary for a qualitative production outcome. It takes many resources and an exponential computing power to be able to collect the data, send it to the server, and then the server analyzes the data and makes a decision in real time. Therefore, we should ask ourselves: How to make it happen? Which model should we use? Which classifier would be the fastest for our task? To this end, a system capable of detecting diseases found in wheat through photo taken by a mobile phone was created [3]. Previously, the first networks used in machine learning used to be shallow and composed by one single layer of neurons, which made it almost impossible to reach a high level of accuracy, however, the paper "Deep Learning" (2015, LeCun *et al.*) [4] unveiled the fact that deep learning could be a significant tool for computational network with several layers to learn features as data representations. Those researches were considerable in the state-of-art covering many areas as object detection and recognition [5] as well as speech recognition [6]. More and more papers are published with methods applying Deep Learning in agriculture with the purpose of diagnosing plant diseases [7]. As research in this field is usually conducted following one architecture or one specific classifier on a specific database, it might be challenging putting together multiple architectures and then compare them in order to find out which one of them is more suitable for a given task, which one offers more accuracy with a given classifier. Among those networks are VGG from the Oxford's Virtual Geometry Group (Too *et al.*), [8] as well as Resnet Inception and Dense Net (Durmus *et al.*), [9] which classified healthy and disabled tomato leaves characterized with nine (09) different diseases using Squeeze Net and Alex Net. The use of feature extraction in the detection of Cassava diseases was a great success in 2010 (Aduwo *et al.*), [10] (Abdullakasim *et al.*, 2011), [11] (Mwebaze and Owomugisha, 2016) [12].

In this work, we synthesize three different CNN models (ResNet 50, Google Net, VGG-16) already used in the previous works and apply them with two different classifiers (SVM and KNN). The final result determines which of these models responds better with a higher accuracy among the other models, according to a given classifier and a given data set in the question of plant disease detection. We assume that deep feature extraction and transfer learning techniques will help us solve our task; another contribution is related to the evaluation of the proposed architectures regarding their lower computational complexity, which is the goal we seek for further mobile implementation.

2. Materials and Methods

Discussing Deep Learning requires us to explain the nature of it, as well as to provide detailed explanation of the algorithms, networks and data sets we are going to work with.

2.1. Pre-Trained CNN Models and Deep Learning Network

If Machine Learning could be a class, Deep Learning would be a subset of its class; it has already proved its effectiveness in multiple areas and it is known for using multiple hidden layers to extract features from its raw input; with each level of layers assigned to detect a different given shape: edges, faces, digits hand-written, etc.

VGG16 is a model Conv Net originally proposed by K. Simonyan and A. Zisserman in their paper "Very Deep Convolutional Network for Large-scale Image Recognitio" [13]. It is based on Image Net, a Data set of 14 million images, a top-5 test of 92.7% of accuracy. The model is known for improving Alex Net considerably by replacing the large kernel-sized filters with multiple 3 * 3 kernel-sized filter. This architecture is the 1st runner-up of ILSVR2014 in classification task. Many modern image classification models are built on VGG architecture. ResNet could be comprehended as one of the best networks in classification area producing higher accuracy than all the previous networks in presence of increased depth. It was introduced by Microsoft as a residual learning framework to overcome the degradation of accuracy found in some networks which were thought to be related to over fitting problem, ResNet compared to VGG16 has fewer filters and is a network of less complexity. In the paper, "Going Deeper with Convolutions" (Szegedy *et al.*) [14] Google Net is described as an incarnation of Inception architecture. The network layers can be variable regarding the machine learning network counting model, however it has an overall of 100 layers. **Table 1** shows the network parameters count in millions.

Table 1. Networks parameters count in millions.

Network	Parameters count (in millions)
VGG16	138
Google Net	6.7
Resnet 50	25.5

2.2. Classifiers

In this sub-section we review two (02) different traditional classifiers: SVM and KNN and their use in extracting features.

2.2.1. Support Vector Machine

In the paper "Support-Vector Networks" (Cortez *et al.* 1995) [15] SVM is described as a novel learning machine that classifies two groups of problems. It is a non-linear vector. SVM solves the problem of separating classes without making errors. One of the advantages of SVM is that it is simple to apply due to its geometric interpretation, unlike ANN's. Additionally, SVM is less inclined to over fitting. Neural Network usually suffers from back-propagation, SVM, however, can solve the core problem by achieving important improvements (Rychetsky, 2001), [16] which make it fit for our classification problem. Nonetheless, we compare it with KNN in various networks following deep learning or feature extraction. The final result summarizes the best classifier for disease detection.

2.2.2. K-Nearest Neighbor

K-nearest Neighborhood is a simple classification algorithm which can be used to solve regression problems. It is easy to use and interpret but as the scale of the data increase in use, it might show its major downside of becoming substantially slow.

KNN operates through identifying the ranges between a request and all the instances throughout the data set, picking the designated number of examples (K) closest to the request, therefore deciding on the most recurrent label (throughout the classification case) or averaging the labels (throughout the regression case).

To apply KNN we first need a suitable value of K, as the classification success relies heavily on that value. The KNN approach is, in such a way, biased with K. There are several possibilities to use the K value, however running the algorithm multiple times by different K values and picking the best result is more efficient. To make KNN less dependent on K's preference, (Wang and Guo) [7] suggested drawing attention at varying sets of closest neighbors instead of one set of closest k-nearest neighbor.

2.3. Data Set

Our Data set is open-sourced and contains approximately 54,000 images of healthy leaves and disease cases classified by 14 species and diseases into 36 categories. Plant Village is a US based, non-profit initiative by Peen State University and Switzerland-based EPFL. A large validated data set is needed in order to establish a reliable image classifier system (Sharada and Mohanty *et al.* 2016) [1]. Such large database had not existed until lately, smaller data sets had not been available to the public as well. To tackle that issue a project was created, named Plant Village and it has started gathering dozens of thousands of plant images, disabled as well as healthy (Hughes and Salath'e, 2015) [17]. The details of our data set are in **Table 2**.

Table 2. The detailed tab of the data set with species, categories and total of images.

	Species Diseases	Categories Healthy	Amount of images
Apple	3	1	33,172
Blueberry	0	1	1502
Cherry	1	1	1906
Corn	3	1	3852
Grape	3	1	4063
Orange	1	0	5507
Peach	1	1	2657
Bell pepper	1	1	2475
Potato	2	1	2152
Raspberry	0	1	371
Soybean	0	1	5090
Squash	2	0	1835
Strawberry	1	1	1565
Tomato	9	1	18,162

3. Proposed Method

Transfer learning as well as deep feature extraction is implemented using the classifiers on the data set. Hence, we make a brief explanation of the following techniques. Detailed schemes of our architectures are given below in Figure 1 and Figure 2.

3.1. Transfer Learning

Transfer learning seeks to enhance target learner's efficiency in targeted areas through passing the information found in related but distinct root areas. It is now a prominent and yet exciting field of machine learning given the large implementation opportunities [18]. One of the mean reasons of its high ranked usage is related to the fact that it is easy to take benefit of its speed during the training time. Transfer learning is also by far more convenient to implement than any CNN architecture with random defined weight [19], its architecture in this work allows us to fine-tune for better accuracy by replacing the last 3 conv layers by our own.

3.2. Deep Feature Extraction

Feature extraction is a fast and efficient method to take advantage of features learnt by a pre-trained neural network. It propagates the input image to a very specified layer of our own (fully connected) defining it as the output feature. Feature extraction process is therefore simple to apply following the architecture of the pre-trained network used the layer to take in consideration might vary but still follow the same process; an image is initiated as an input image with its

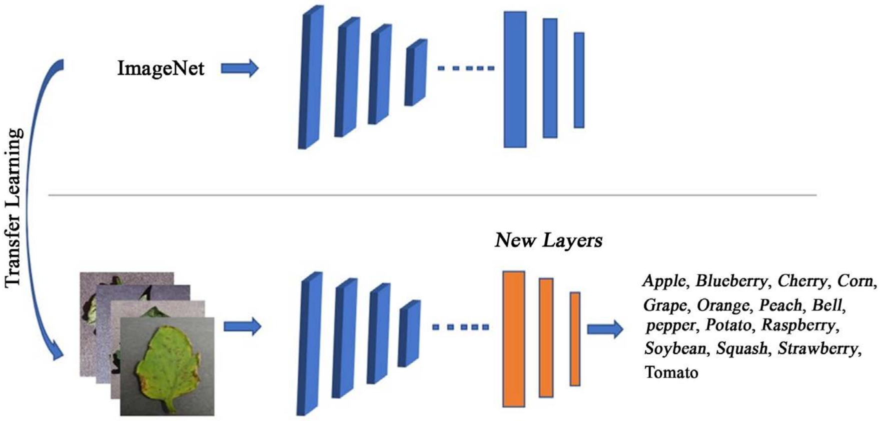
Transfer Learning
ImageNet
New Layers
Apple, Blueberry, Cherry, Corn, Grape, Orange, Peach, Bell, pepper, Potato, Raspberry, Soybean, Squash, Strawberry, Tomato

Figure 1. A brief representation of transfer learning with the data set image.

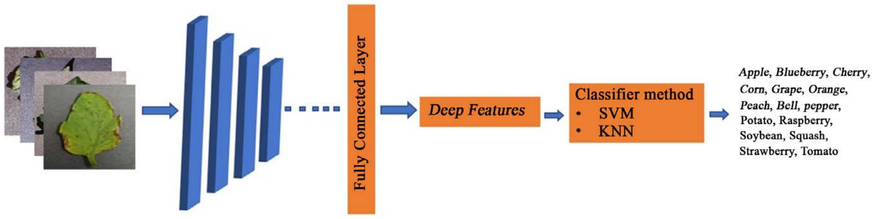
Fully Connected Layer
Deep Features
Classifier method
• SVM
• KNN
Apple, Blueberry, Cherry, Corn, Grape, Orange, Peach, Bell, pepper, Potato, Raspberry, Soybean, Squash, Strawberry, Tomato

Figure 2. A brief representation of feature extraction with the data set image.

input size defined by the pre-trained default input shape; same image is then forwarded though the network. In our case for VGG16, images will be forwarded and then propagated in the network to the last fully connected fc-6, fc-7, fc-8. The fully connected layers are of different depth while each of the first two are built with 4096 channels, the last one is of 1000 channels, the choice of these three layers is not accidental, several theories have been put forward on the question of determining which layers offers more stability and performances among them. For Resnet 50, features are also extracted from the batch size. The output is of size 7 * 7 by 2048, as a feature vector, it is flattened to 7 * 7 by 20,148. As of Google Net, the previous research have inducted feature extraction on "pool 5 drop 7 * 7 s1" layer but other results have shown that its Max pooling in practice is more reluctant to produce better performances (Huang *et al.* 2006) [20].

4. Equipment's Configurations

In this research we assessed the efficiency of three robust network neural architectures.

The techniques used were transfer learning as well as extracting feature on various layers in the network. Later on, the extracted features, as well as transfer learning were classified using Support vector machine and K-nearest Neighbor, their time of execution, F1 score, True positive and the True negative are determined.

To train and test, we used Anaconda Framework with python 3.1, Tensor-

Flow, and Google Colab. All applications were run on a GPU: 1xTesla K80, 2496 CUDA cores, compute 3.7, 12 GB (11.439 GB Usable) GDDR5 VRAM. Performances were computed using a 5-fold validation. The detailed results obtained following our architectures are below.

5. Presentation of the Results

5.1. Feature Extraction Results with Resnet 50, Google Net and VGG16

According to the results from Table 3, the fully connected-6 from VGG16 is the best layer to extract features for disease classifications using SVM as well as KNN; its time of execution is 10 minutes. Google Net shows better results with SVM on our data set, unlike KNN. However, Resnet 50 performed the best accuracy among all three of them, using SVM. VGG16 takes less training time (only 10 minutes). Resnet takes much more time to complete the training (12 minutes 21 seconds).

Besides, in Table 4 we gathered data related to the true positive rate, false positive rate and F-score for our networks and Figure 3 gives a better understanding of the obtained results.

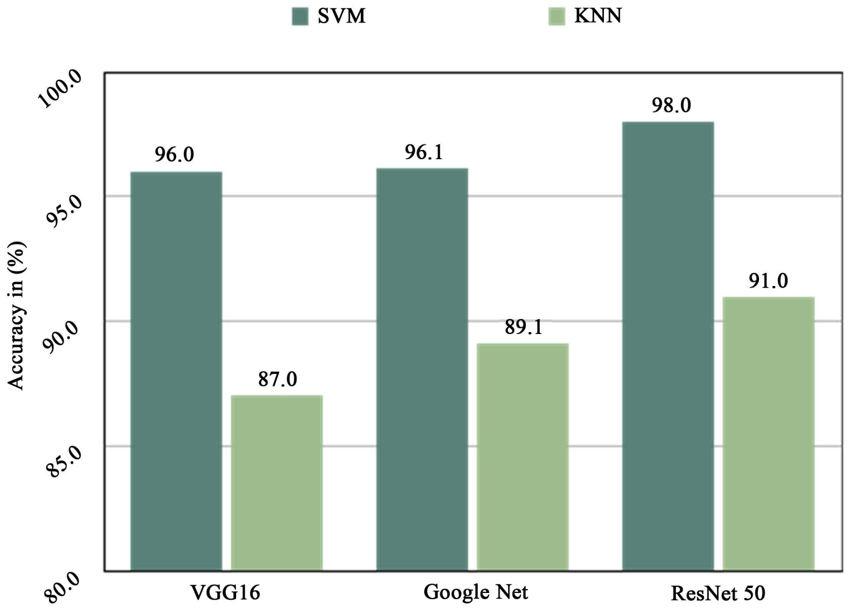

Figure 3. Graph displaying the results based in feature extraction with SVM and KNN.

Table 3. Performances of the models in accuracy and times.

	Classifier-VGG16	Google	Resnet 50	Net	
	Fc-6	Fc-7	Fc-8		
SVM	**96.0**	94.07	93.5	96.13	**98.01**
KNN	87.04	87.0	85.7	89.14	91.01
Time	10 mn	11 mn 8 s	13 mn 30 s	11 mn 41 s	12 mn 21 s

Table 4. Performance of the models according to True positive rate, False positive rate and F score measures.

Classifier	Performance measure	VGG16	Google Net	Resnet 50
SVM	True positive rate	94.01	94.30	97.50
	False positive rate	99.20	98.20	99.43
	F score	94.18	94.45	97.13
KNN	True positive rate	87.49	87.80	87.90
	False positive rate	99.45	98.60	99.02
	F score	86.36	87.90	88.04

5.2. Deep Learning Results with Resnet 50, Google Net and VGG16

Just as we defined it, transfer learning seeks to enhance target learners efficiency in target areas through passing the information found in related but distinct root areas. For instance, supposing there are two agents C and D, assuming that agent C already possesses all the knowledge related to a knows task, it would be time-consuming to train agent D from scratch to the knowledge already possessed by agent C. That is where transfer learning can be incredibly useful; with its help all the knowledge already learnt by agent C to D is transferred without the need to start from scratch. The best way to do transfer learning is through fine-tuning a pre-trained network.

However, we might have multiple options when fine-tuning related to the size of the data set; if the target data set is small we might over fit the network. In our case, we have a substantial data set, therefore we considered freezing the top layers except the 4 last layers. We fixed our batch size to 64, used data augmentation which is a technique that helps researchers to enhance the variety of a data for training models substantially, without actively obtaining new data, and we started with a learning rate of 0.001. For the optimizer, instead of Adam we used RMSprop, for the loss function we used Cross entropy. Another method defined to improve the accuracy is the use of Callbacks which can be really practical with early stopping function and the patience set to 5 in catching the best weight for each iteration. Table 5 and Figure 4 bellow will gives us more details.

The results in Table 5 shows that VGG16 is a better model than Google Net and Resnet 50, when the main goal is to classify plant diseases with the plant village data set, achieving an accuracy of 97.92, however its execution timer is higher than that of Google Net which comes second with an accuracy of 95.30 and an execution timer of 12 minutes 30 seconds. Resnet 50 achieved the top-3 best accuracy among all of the three networks in accuracy, as well as in execution timer.

The performance measures f1-score, sensitivity, and specificity, they are shown in Table 6.

Measures of the True positive rate, the False positive rate and f-score using transfer learning.

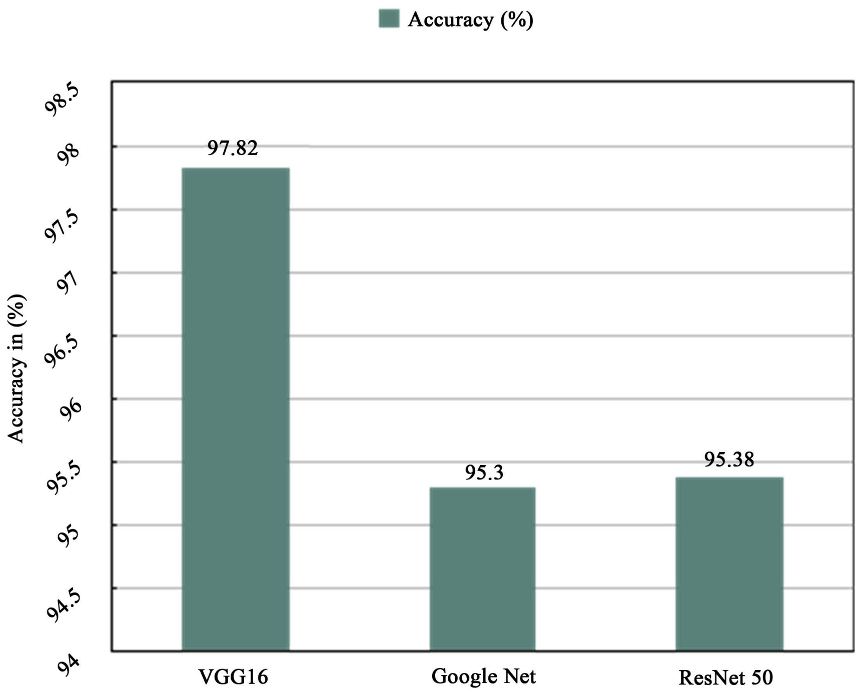

Figure 4. Results based on transfer learning.

Table 5. Performance of the models in transfer learning, in accuracy and time.

	VGG16	Google Net	Resnet 50
Accuracy (%)	**97.82**	95.3	95.38
Time	14 min 32 s	12 min 30 s	23 min 49 s

Table 6. Measures of the true positive rate, the false positive rate and f-score using transfer learning.

Performances	VGG16	Google net	Resnet 50
True positive rate	97.20	93.29	94.91
False positive rate	99.60	96.75	98.80
F-score	96.42	93.54	93.18

5.3. Performance Based on Traditional Shallow Networks

We made apply color features (CF), which are necessary in detecting plant diseases, as well as techniques such as Gamma based Feature Extraction (GFE), Histogram of Oriented Gradients (HOG) and Local Binary Pattern (LBP). Those carried out the feature extraction, and finally the extracted features were fed into the SVM and KNN classifiers in order to determine their accuracies. The results are shown below in Table 7. It can be seen from the results, that LBP achieved the best accuracy with SVM classifier 80.6%, followed by GFE 76.9% with SVM, and HOG 71.28% with SVM, and in the end, we have the color features which scored 51.03%. Table 7 and Figure 5 illustrates our results.

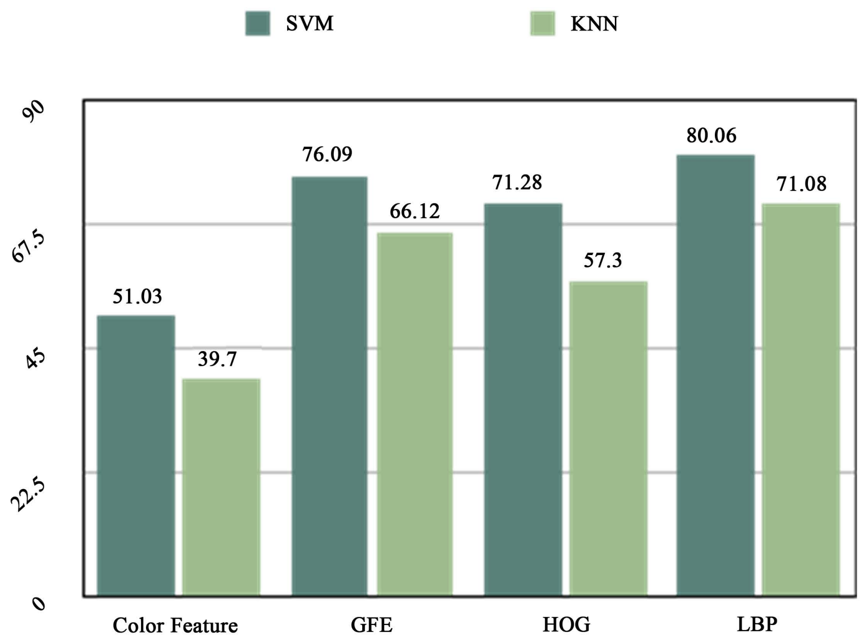

Figure 5. Results based on traditional shallow methods.

Table 7. The accuracy of traditional methods.

Classifier Methods	Color Features	GFE	HOG	LBP
SVM	51.03	76.9	71.28	80.6
KNN	39.7	66.12	57.3	71.8

6. Discussions

The performance results obtained by applying feature extraction and deep learning using VGG16, Google net and Resnet 50 were evaluated, as well as the performance of traditional classifiers (Color features, GFE, HOG, LBP) which had proven themselves to be relevant in the field of images classification. We first extracted features based on each of the three models above each on a different layer; we computed their performances through classifiers (SVM and KNN) and each of them responded in a different manner, based on the method of classification used. Therefore, we cannot point at any given network as the best for a given classification task as a standard because there are numerous parameters to take into consideration. The size of the data set is one of the key points to take into account, the parameters used to initiate the network are also relevant. The results show that if we need to extract features, ResNet 50 is the best network to use compared to VGG16 and Google Net. Furthermore, a classifier is also needed to be put on top of the layers and our results shows that SVM compared to KNN is the best and the fastest classifier in the area of detecting plant diseases through a database of leaf images, it was the best in our study when it comes to feature extraction, compared to VGG16 and Google Net, however when transfer learning is needed, we would definitely recommend using VGG16 among the three of them in the area of image classification. However, the graph in Figure 4 shows that while conducting transfer learning by fine-tuning a network with a large data set, VGG16 might be the network that one can consider using compared to Google Net and ResNet 50; it produced an accuracy of 97.82%. ResNet 50 is the second top best with an accuracy of 95.38% and Google Net follows with an accuracy of 95.3%.

Regarding traditional methods' accuracy and their performances, we used Color features, GFE, HOG, LBP with our classifiers SVM and KNN. LBP displayed the best accuracy among all of them showing 80.6% with SVM, GFE had the second-best accuracy of 76.9% with SVM, Color features showed the worst accuracy for both SVM and KNN 51.03% and 39.7%, respectively.

7. Conclusions

In our research, we implemented Deep feature extraction and Deep learning techniques on the plant village data set in order to detect plant diseases. We tested three (03) deep learning models VGG16, Google Net and ResNet 50. The choice for those networks was not random. They are the most used networks by the state of art. We first extracted features using SVM and KNN, and then conducted transfer learning using fine-tune. Results were compared using accuracy percentage and time of execution. According to the models' behavior, we can state that in computer vision, extracting features is the way more efficient than transfer learning; with the use of the best classifier, it produces greater accuracy and its time of execution is shorter than that in transfer learning.

In a future project, we would like to extend this work by collecting our own data set in sub-equatorial zones where the cultivable lands might be hostile to the development and survival of plants. This will allow us to study the behavior of the plant, while detecting the main threats to its survival according to its environment. The results will be therefore compared to the Plant Village Dataset. Following that research, we will be able to detect the right environment fit for the development and survival of a plant in the range of our data set.

Acknowledgements

This work was supported in part by the Tianshan Youth Plan, Xinjiang Uygur Autonomous region under Grant 2018Q024 and in part by the Open Research Fund of Key Laboratory of Data Security, Xinjiang Normal University under Grant XJNUSY102018B01.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Mohanty, S.P., Hughes, D.P. and Salathe, M. (2016) Using Deep Learning for Image-Based Plant Disease Detection. *Frontiers in Plant Science*, **7**, 1419-1419. https://doi.org/10.3389/fpls.2016.01419

[2] Brahimi, M., Boukhalfa, K. and Moussaoui, A. (2017) Deep Learning for Tomato Diseases: Classification and Symptoms Visualization. *Applied Artificial Intelligence*, **31**, 299-315. https://doi.org/10.1080/08839514.2017.1315516

[3] Johannes, A. (2017) Automatic Plant Disease Diagnosis Using Mobile Capture Devices. *Computers and Electronics in Agriculture*, **138**, 200-209. https://doi.org/10.1016/j.compag.2017.04.013

[4] LeCun, Y., Bengio, Y. and Hinton, G. (2015) Deep Learning. *Nature*, **521**, 436-444. https://doi.org/10.1038/nature14539

[5] Lecun, Y., Bottou, L., Bengio, Y. and Haffner, P. (1998) Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, **86**, 2278-2324. https://doi.org/10.1109/5.726791

[6] Hinton, G. (2012) Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Processing Magazine*, **29**, 82-97. https://doi.org/10.1109/MSP.2012.2205597

[7] Guo, G., Wang, H., Bell, D., Bi, Y. and Greer, K. (2003) KNN Model-Based Approach in Classification. *Lecture Notes in Computer Science*, **2888**, 986-996. https://doi.org/10.1007/978-3-540-39964-3_62

[8] Ferentinos, K.P. (2018) Deep Learning Models for Plant Disease Detection and Diagnosis. *Computers and Electronics in Agriculture*, **145**, 311-318. https://doi.org/10.1016/j.compag.2018.01.009

[9] Durmus, H., Gunes, E.O. and Kirci, M. (2017) Disease Detection on the Leaves of the Tomato Plants by Using Deep Learning. 2017 6*th International Conference on Agro-Geoinformatics*, Fairfax, 7-10 August 2017. https://doi.org/10.1109/Agro-Geoinformatics.2017.8047016

[10] Ramcharan, A., Baranowski, K., McCloskey, P., Ahmed, B., Legg, J. and Hughes, D.P. (2017) Deep Learning for Image-Based Cassava Disease Detection. *Frontiers in Plant Science*, **8**, 1852-1852. https://doi.org/10.3389/fpls.2017.01852

[11] Abdullakasim, W., Powbunthorn, K., Unartngam, J. and Takigawa, T. (2011) An Images Analysis Technique for Recognition of Brown Leaf Spot Disease in Cassava. *Journal of Agricultural Machinery Science*, **7**, 165-169.

[12] Owomugisha, G. and Mwebaze, E. (2016) Machine Learning for Plant Disease Incidence and Severity Measurements from Leaf Images. 15*th IEEE International Conference on Machine Learning and Applications*, Anaheim, 18-20 December 2016.

[13] Simonyan, K. and Zisserman, A. (2015) Very Deep Convolutional Networks for Large-Scale Image Recognition. 3*rd International Conference on Learning Representations*, *ICLR* 2015, San Diego, 7-9 May 2015, 1-14.

[14] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A. (2015) Going Deeper with Convolutions. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Boston, 7-12 June 2015, 1-9. https://doi.org/10.1109/CVPR.2015.7298594

[15] Cortes, C. and Vapnik, V. (1995) Support-Vector Network. *Machine Learning*, **20**, 273-297. https://doi.org/10.1007/BF00994018

[16] Rychetsky, M. (2001) Algorithms and Architecture for Machine Learning Based on Regularized Neural Networks and Support Vector Approaches. Shaker Verlag GmbH, Düren.

[17] Hughes, D.P. and Salathe (2015) An Open Access Repository of Images on Plant Health to Enable the Development of Mobile Disease Diagnostics.

[18] Zhuang, F., *et al.* (2019) A Comprehensive Survey on Transfer Learning.

[19] Krahenb, P., Krahenbuhl, K., Doersch, C., Donahue, J. and Darrell, T. (2016) Data-Dependent Initializations of Convolutional Neural Networks. *International Conference on Learning Representations*, *ICLR* 2016, San Juan, 2-4 May 2016.

[20] Huang, L., Massa, L. and Karle, J. (2006) The Kernel Energy Method: Application to a tRNA. *Proceedings of the National Academy of Sciences*, **103**, 1233-1237. https://doi.org/10.1073/pnas.0510342103