Foliar Disease Classification

Bingze Dai
ECE, UC San Diego
bdai@eng.ucsd.edu

Tian Qiu
ECE, UC San Diego
tiq014@eng.ucsd.edu

Kai Ye
ECE, UC San Diego
k1ye@eng.ucsd.edu

***Abstract*—Apple tree diseases seriously affect their health and productivity. Accurate diagnosis of diseases is essential to early treatment and can reduce yield loss. Traditional methods rely on field scouting which is inefficient. In this project, we used machine learning methods to classify apple tree diseases. We used histogram of gradients (HOG) and pretrained CNNs to extract features. We applied SVM and transfer learning with four deep learning models, VGG, DenseNet, ResNet and GoogLeNet. Among these methods, SVM holds poor accuracy of merely over 50%, while transfer learning models achieve accuracies over 90%, with GoogLeNet achieving the highest accuracy of 94%.**

***Index Terms*—Plant disease, image classification, SVM, transfer learning**

I. Introduction

We human beings are fed by agricultural crops. However, even if we tried our best improving crop yields, there are still millions of people starving. One of the main reasons is failure to diagnose crop-impacting diseases. Misdiagnosis of these diseases can result in misuse of chemicals leading various consequences, such as increased costs of input and emergence of resistant pathogen strains. Currently, crop disease diagnosis is mainly relied on farmer scouting, which is inefficient due to the difference of diagnosing experience. Although several machine learning methods have been proposed aiming at improving the efficiency of crop disease detection, some factors such as light conditions of crop images and disease variations affect the detection accuracy.

To help farmers assess apple leaves health successfully, the team decide to train an apple leaves disease identification model to distinguish between apple leaves that are healthy, those that have apple rust, those which are infected with apple scab, and those with multiple diseases. The input to our algorithm is images of apple leaves. The team use SVM and four deep learning methods, including VGG, DenseNet, ResNet and GoogLeNet to classify the input images to four classes. Model perforamnce is evaluated by prediction accuracy and confusion matrix. At the end, the team compare the performance of each model to find best model for this task.

Source code has been put on github: [1]

II. Related Work

Many machine learning techniques have been used to identify a plant infection using leaf images [2], including SVM [3], shallow self-built CNN [4], CNN with LVQ Algorithm [5], and also several deep learning models such as GoogLeNet and AlexNet [6], AlexNetOWTBn [7], Overfeat and VGG [8]. Shanwen Zhang et al. [9] have proposed a global pooling dilated convolutional neural network (GPDCNN) to classify the diseases of cucumber plant leaves. The strength of this approach can be understood as a combination of the advantages of global pooling and dilated convolution. By using this method, they can detect up to six diseases of cucumber leaves. Besides that, Shanwen Zhang et al. [10] also used a three-channel convolutional neural network (TCCNN) for classifying diseases in vegetable leaves. Yuan Yuan et al. [11] have proposed an idea of transfer learning by using two machine learning networks, VGGNet and AlexNet, for detection of crop disease. This method can be used to successfully classify eight crop diseases. Artizai Picon et al. [12] have developed an adapted Deep Residual Neural algorithm based on networks for classifying multiple diseases. The authors used an extensive dataset which include approximately 8200 images of crops from Germany and Spain. Qiaokang Liang et al. [13]have proposed a deep learning network called Plant Disease Diagnosis and Severity Estimation Network, for detecting diseases in plant leaves. The method is considered as the state-of-the-art method by the team, as this method can be used not only to detect the diseases but also to know the severity of the diseases.

Based on the discussion above, we can see that numerous studies on crop leaf disease detection have been achieved with a considerable accuracy. The team plan to attempt several machine learning methods to determine the most ideal method to classify apple tree diseases.

III. Dataset

A. Dataset Introduction

The dataset we used are from the Kaggle plant pathology 2020-FGVC7 [14], Identify the Category of Foliar Diseases in Apple Trees. There are 1821 labelled training images of apple tree leaves and 1821 unlabeled test images. The images are categorized into four classes, including “healthy”,“scab”, “rust”, and “multiple diseases”. The distribution of four classes is shown in Fig. 1. From the pie chart we can see that class “multiple diseases” takes only a very small portion of the whole dataset and the other three classes hold approximately same portion while the “Rust” class has the most number, the “Scab” class is the second most and the “healthy” class takes the third position.

Scab is a fungal disease that leads to irregular shaped gray regions, and rust leads to yellowish spots on leaves. “Multiple diseases” class denotes the plant is suffering both scab and rust. By inspecting the dataset, we identified some

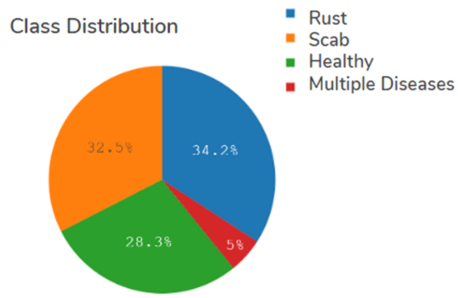


Fig. 1. Training data distribution

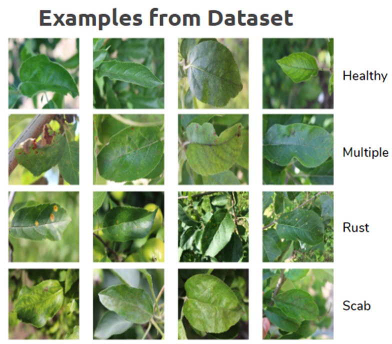


Fig. 2. Sample training images

challenges that may influence the performance of machine learning algorithm. First of all, it's quite easy for human to identify rust because there are obvious yellow spots on leaves, but the features for class 'Scab' are hard to identify. Also, it's hard to distinguish Multiple diseases from a single disease from the figures in dataset and imbalanced training samples makes this even worse. Last but not least, in this dataset the background of the images are also leaves which makes it's hard to separate the object from background.

B. Data Pre-processing

We randomly shuffle the data and split the dataset into training set and validation set with a ratio of 80:20. In order to avoid over-fitting, we apply data augmentation to preprocess the dataset by applying following methods:

- Rotate the images from -10 degrees to 10 degrees
- Flip the images horizontally
- Zoom the images from 0.9 to 1.1
- Shift the width and height of the images by range within 20% of the size.

We also resize all images to 224*224 in order to fit in the neural networks and to lower model complexity and training time.

IV. Feature Extraction

In this project, we adopted two methods to extract features from images. The first feature extraction method we used is to capture features with pre-trained neural network. Neural networks that are pre-trained on ImageNet can capture many useful features from an image. For example, in a pre-trained VGG16 network, the activations in different layers show that they can capture features like the edges of the leaf, or the veins of the leaf. Such knowledge can be transferred to related image classification problem. In our model, we add several fully connected layers after the pretrained model, to classify our input image to four classes.

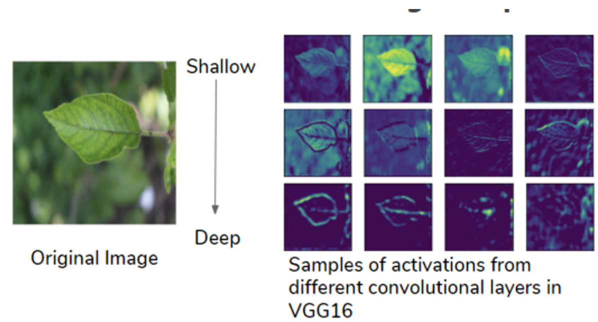


Fig. 3. Feature extraction with pretrained networks

Another feature extraction method we used is Histogram of Oriented Gradients, also known as HOG [15]. The idea behind HOG is that local object appearance and shape within an image can be described by the distribution of intensity gradients or edge directions. Therefore, we can use less features to represent the information of original image. Commonly, we use a gray image to get the HOG feature map as shown in the flowchart below.

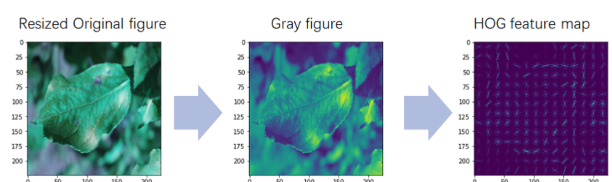


Fig. 4. Feature extraction with HOG

V. Methods

We used a total of six models from two types of machine learning methods in this project. First is traditional machine learning method, where we used SVM, and second is transfer learning with pretrained neural networks. For SVM, we used two types of data for input. One is flattened grayscale image, the other is the HOG of grayscale image. For transfer learning, we experimented with four types of pretrained models: VGG16, DenseNet, ResNet, and GoogLeNet.

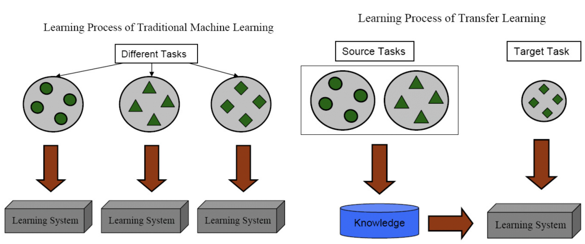
A. SVM

Support vector machine [16] is a supervised machine learning method that aims to find a hyperplane to separate a group of high-dimensional data. Originally, SVM is used to classify input into exactly two classes, but it can also be expanded to multi-class problem. The term "support vector" refers to the data point that lies closest to the hyperplane. SVM can be applied to both linearly-separable data and linearly non-separable data.

B. Transfer learning

1) Convolutional neural networks: Convolutional neural networks(CNN) are a class of neural networks that uses multiple convolution layers inside. Previous researches have shown that CNNs have the ability to capture features from image, and are therefore ideal method for image classification.

2) Transfer learning with CNN: Transfer learning is a type of machine learning technique that uses knowledge from other machine learning tasks. The intuition behind transfer learning is that different problems may be learning similar information from the data, so a new problem can utilise a good data representation learned by another model. With this learned knowledge, the new model can converge to the result faster, or even get better results. Fig. 5 [17] shows the fundamental difference between a traditional machine learning method and transfer learning method.



(a) Traditional Machine Learning (b) Transfer Learning

Fig. 5. Transfer learning process

For image classification problem, a transfer learning model is build by the following steps:

1) Extract convolution layers from a classification model trained for another task with different data. Discard the classification layer.
2) Add custom layers to form a classifier for the new problem.
3) Train the new model with new data.

3) Base models for transfer learning: In our project, four models pretrained on ImageNet are used as base models and their performances are compared.

- VGG16
  The original VGG16 [8] network has 13 convolution layers and 3 fully-connected layers. Its main contribution is exploring the effect of deep convolution layers with small filter sizes. In our model, the convolution layers are kept. A fully-connected layer of 256 outputs and an output layer of 4 outputs are appended to the base model.
- DenseNet
  DenseNet [18] consists of convolution layers that are densely connected, which encourages feature propagation through a deep neural network. It can also reduce total number of parameters since all the layers are already densely connected.
- ResNet
  Resnet [19] is a class of networks made up of small residual blocks with different depth. The purpose for the skip connection is to give a direct path for the error to back propagate through the entire network and reuse activations from previous layer until adjacent layer learns the weights which solved the problem of vanishing gradients.
- GoogLeNet
  GoogLeNet [20]is also known as Inception net. It is built by inception layers. The basic idea is that layers with a set of different size filters that cover a bigger area may handle better multiple objects scales while still keep a fine resolution for detail information from the images. Therefore, the idea is to convolve in parallel different sizes from the most accurate detailing (1x1) to a bigger one (5x5), by this way, all filters on the inception layer are learnable.

VI. Experiments and Results

A. Experiment setup

All the experiments are completed on Goolge Colab with GPU support. We used Python3 programming environment with TensorFlow 2.1 installed. The SVM algorithm is implemented with scikit-learn package. The transfer learning models are built with Keras framework.

B. Metrics

The main metric used to evaluate the models is classification accuracy on the test set. In this Kaggle competition, the true labels of test set are not released. In order to evaluate the model on the test set, we have to upload the classification result to Kaggle and the test accuracy is calculated based on a subset of all the test images.

C. Classification with SVM

SVM has accuracy only above 50%. Using HOG performs slightly better than using grayscale images. This can be explained by that SVM flattens input data so some features in the imags are lost.

TABLE I
SVM classification accuracy

Model	Val accuracy	Kaggle test accuracy
SVM with grayscale	0.326	0.516
SVM with HOG	0.419	0.549

D. VGG16

The best results is acquired by training 30 epochs with method 2, which is two-step finetuning. The learning rate is set at 10^{-5}, the optimizer used is Adam. The training result is shown in Fig. 6(a).

E. DenseNet

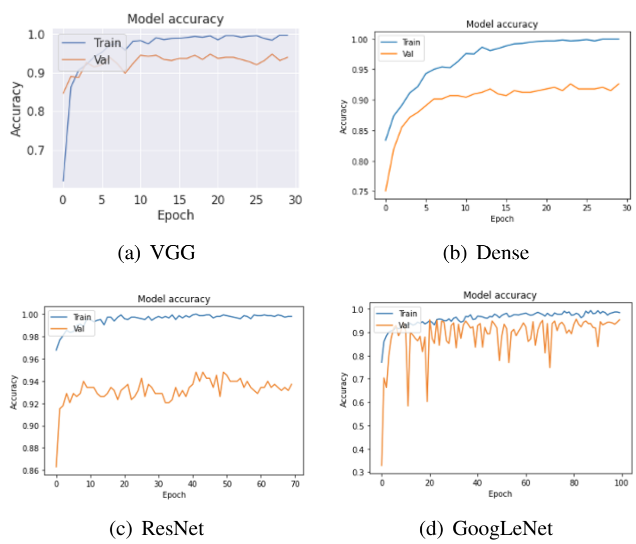
The best result is acquired by training 30 epochs with method 2. The learning rate is set at 10^{-5}, the optimizer used is Adam. The training result is shown in Fig. 6(b).

F. ResNet

The best result is acquired by training 70 epochs with method 2. The learning rate is set at 10^{-5}, the optimizer used is Adam. The training result is shown in Fig. 6(c).

G. GoogLeNet

The best result is acquired by training 100 epochs with method 1. The learning rate is set at 10^{-5}, the optimizer used is Adam. The training result is shown in Fig. 6(d).



(a) VGG (b) Dense

(c) ResNet (d) GoogLeNet

Fig. 6. Transfer learning model accuracy

H. Summary of transfer learning models

TABLE II
TRANSFER LEARNING CLASSIFICATION ACCURACY

Model	Train accuracy	Val accuracy	Kaggle test accuracy
VGG16	0.997	0.940	0.930
DenseNet	0.999	0.926	0.939
ResNet	0.998	0.937	0.936
GoogLeNet	0.984	0.953	**0.940**

All the deep learning methods worked fairly well on this problem, and they can correctly classify almost all the samples from healthy, rust and scab class. However, they all have significantly worse performance on classifying the "multiple diseases" class. Another typical error among the models is that some scab leaves will be classified as Healthy.

After looking into why multiple disease class is so hard to classify, we found some possible reasons. First, some disease features are hard to distinguish, as shown by the red circles. Second, many images have poor quality, like a large region of shadow, which makes it hard to identify diseases.

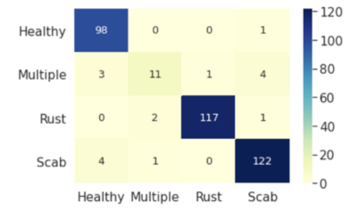


Fig. 7. Confusion matrix of GoogLeNet

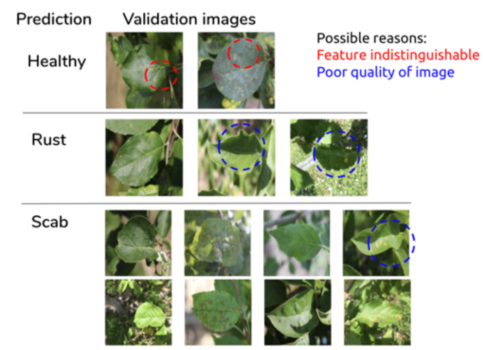


Fig. 8. Mis-classified samples from "Multiple disease class"

VII. FUTURE WORK

Several work can be done in the future to improve the model performance:

1. Try different kernels in SVM

We used linear kernel in SVM. Since the dataset might not be linearly separable, other kernels, such as polynomial, radial basis function, or sigmoid, could perform better.

2. Apply adaptive learning rate decay

Due to time limit, we used fixed learning rate in our experiment, which may not be able to achieve the best result. A better approach is to use adaptive learning rate schedule, which can reduce learning rate as the training proceeds, and also to prevent overfitting.

3. Combine image classification with object detection

By inspecting the mis-classified images, we can see that the current model is worst at classifying multiple diseases. This can be resolved if the algorithm can detect multiple regions of disease from the image before classification.

VIII. CONCLUSION

We attempted SVM and four deep learning methods, including VGG, DenseNet, ResNet and GoogLeNet, to find the best model to classify apple tree diseases. The models are evaluated on the Kaggle test set. SVM performed poorly on this task with accuracy merely over 50%. Among the deep learning models, GoogLeNet achieved the best result with 94% accuracy. However, no model is significantly better than other models and they all have lower accuracies on the "Multiple diseases" class. We proposed some possible future works to improve the performance.

REFERENCES

[1] *Project github page*, 2020 (accessed June 13, 2020). https://github.com/deltawater/ECE228_Group15_Foliar-diseases-classification.

[2] S. Kaur, S. Pandey, and S. Goel, "Plants disease identification and classification through leaf images: A survey," *Archives of Computational Methods in Engineering*, vol. 26, pp. 507–530, 2019.

[3] A. Camargo and J. S. Smith, "Image pattern classification for the identification of disease causing agents in plants," *Comput. Electron. Agric.*, vol. 66, p. 121–125, May 2009.

[4] S. Wallelign, M. Polceanu, and C. Buche, "Soybean plant disease identification using convolutional neural network," in *The Thirty-First International Flairs Conference*, 2018.

[5] M. Sardogan, A. Tuncer, and Y. Ozen, "Plant leaf disease detection and classification based on cnn with lvq algorithm," in *2018 3rd International Conference on Computer Science and Engineering (UBMK)*, pp. 382–385, 2018.

[6] S. P. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," *Frontiers in plant science*, vol. 7, p. 1419, 2016.

[7] K. P. Ferentinos, "Deep learning models for plant disease detection and diagnosis," *Computers and Electronics in Agriculture*, vol. 145, pp. 311 – 318, 2018.

[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014.

[9] S. Zhang, S. Zhang, C. Zhang, X. Wang, and Y. Shi, "Cucumber leaf disease identification with global pooling dilated convolutional neural network," *Computers and Electronics in Agriculture*, vol. 162, pp. 422–430, 2019.

[10] S. Zhang, W. Huang, and C. Zhang, "Three-channel convolutional neural networks for vegetable leaf disease recognition," *Cognitive Systems Research*, vol. 53, pp. 31–41, 2019.

[11] Y. Yuan, S. Fang, and L. Chen, "Crop disease image classification based on transfer learning with dcnns," in *Chinese Conference on Pattern Recognition and Computer Vision (PRCV)*, pp. 457–468, Springer, 2018.

[12] A. Picon, A. Alvarez-Gila, M. Seitz, A. Ortiz-Barredo, J. Echazarra, and A. Johannes, "Deep convolutional neural networks for mobile capture device-based crop disease classification in the wild," *Computers and Electronics in Agriculture*, vol. 161, pp. 280–290, 2019.

[13] Q. Liang, S. Xiang, Y. Hu, G. Coppola, D. Zhang, and W. Sun, "Pd2se-net: Computer-assisted plant disease diagnosis and severity estimation network," *Computers and electronics in agriculture*, vol. 157, pp. 518–529, 2019.

[14] *Kaggle Plant Pathology Competition*, 2020 (accessed June 10, 2020). https://www.kaggle.com/c/plant-pathology-2020-fgvc7.

[15] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, vol. 1, pp. 886–893, IEEE, 2005.

[16] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, (New York, NY, USA), p. 144–152, Association for Computing Machinery, 1992.

[17] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[18] G. Huang, Z. Liu, and K. Q. Weinberger, "Densely connected convolutional networks," *CoRR*, vol. abs/1608.06993, 2016.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[20] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *CoRR*, vol. abs/1409.4842, 2014.