Fruit Tree Disease Detection via

Network-Expansive Class Incremental Learning

David Blumenstiel

October 2021

1. **Introduction**
   1. **Crop Pathogens and Pests**

Pathogens and pests have presumably afflicted human agricultural ambitions since their inception. Oerke (2005) estimated that, in modern times, plant pathogens alone reduce global crop yield by round 16%. The Food and Agricultural Organization of the United Nations further estimates that between 20 and 40% of global crop yield is lost to pests (UNFAO, 2019). Combating plant pathogens and pests, hereby collectively referred to as ‘diseases’, poses challenge for growers globally.

Diseases, which are common to every cultivated type of fruit tree, can reduce the quantity and quality of fruit produced. Disease is one of the primary causes of lost yield, and determining which diseases afflict any particular tree is crucial when it comes to prevention and treatment (Tian, Y., Yang, G., Wang, Z., Li, E., & Liang, Z., 2019). Typically, diagnoses are done in person, but this can be error prone and costly (if hiring an arborist). Diagnosis via human observation is also time consuming, and not practical for large-scale, in-depth analysis.

Arable land is a finite resource. Also, developing additional arable land for crop production tends to involve the disruption of or removal of natural environments, which has adverse effects on climate and biodiversity when done on a large scale. Thus, ensuring a high crop yield density is one way to help sustainably meet demand, especially as world population continues to grow, and standards of living improve. Combating plant disease is one task necessary to ensure crop production (both of food and other resources) continues to meet demand. Fortunately, there are many new and promising tools arising to meet this need.

* 1. **Disease Detection via Computer Vision**

Fruit tree diseases tend to manifest themselves visually on leaves, branches, fruits, or combinations thereof, which makes optical diagnoses the go-to. Computer vision, a field of machine learning where information is extracted from images, has been of interest when it comes to detecting plant disease (Liu and Wang, 2021). Computer vision models are able to detect disease accurately and conveniently, often only requiring a digital image. It’s also possible to scale computer aided disease detection systems up to survey entire orchards at once via areal imagery.

Convolutional Neural Networks (CNNs) in particular are used to great effect, both for diagnosing plant diseases and in other classification problems. There are many instances of computer vision techniques used for fruit tree disease detection. For example, Yadav, S., Sengar, N., Singh, A., Singh, A., & Dutta, M. K. (2021) were able to create CNN models that could correctly distinguish between healthy peach leaves and those affected with peach bacteriosis 98.75% of the time. Liu, B., Zhang, Y., He, D., & Li, Y. (2017) were able to create a CNN that could detect several apple diseases, including mosaic virus, cedar apple rust, apple scab, and Alternaria leaf spot with 97.62% accuracy. Many models have also been created that can detect disease amongst more than one type of plant. For example, Khan, M. A., Akram, T., Sharif, M., Awais, M., Javed, K., Ali, H., & Saba, T. (2018) were able to create a model that could distinguish between many apple and banana diseases. There are also methods which do not involve CNNs, although these are less common now due to the high performance of CNNs. For instance, Habib, Md. T., Majumder, A., Jakaria, A. Z. M., Akter, M., Uddin, M. S., & Ahmed, F. (2020) made a system for detecting papaya disease which used K-means clustering in conjunction with a support-vector machine capable of achieving accuracies over 90%.

However, most plant disease detection models tend to only work for one or a few types of plants, and would require re-training to detect more diseases or species. It would be advantageous to have a model which could be progressively constructed to incorporate new diseases and species without complete re-training. However, such machine learning techniques are still quite new, and are still being developed

* 1. **Class Incremental Learning**

In a typical supervised learning setting, labeled data from all tasks are provided at the time of training (Masana, M., Liu, X., Twardowski, B., Menta, M., Bagdanov, A. D., & van de Weijer, J., 2021). However, data is not always available for use at the same time. To accommodate this, there is incremental learning, where machine learning algorithms adjust their knowledge in response to new data (Geng and Smith-Miles, 2005); this has any number of real-world applications and advantages, but also contain some significant limitations.

Deep neural networks excel at learning individual tasks but tend to forget old tasks when sequentially trained on new tasks, often referred to as catastrophic forgetting (van de Ven and Tolias, 2019). Within the realm of computer vision, there is a desire to be able to incrementally add prediction labels to models as new data arrives without needing to completely re-train the model; this is referred to as Class Incremental Learning (CIL; Rebuffi, S., Kolesnikov, A., Sperl, G., & Lampert, C.H., 2017). However, updating model weights to incorporate new labels tends to either fail to learn new classes, or significantly degrade the performance of the model on the old classes (catastrophic forgetting). Traditionally, if one wants to update the weights of a model to incorporate new labels without significant biasing, they will typically need to keep around the old dataset for re-training. However, it’s not always practical to keep the old data around; it may take up too much storage space, the data may no longer be available, or it may take too long to re-train the model using old data. It would also be convenient to have models that are able to expand their functionality without a complete re-work. Finding a way to add labels models incrementally without significantly decreasing performance on old tasks is a topic of current research. CIL is closer to human methods of learning, in which we can comprehend new tasks (such as classifying objects) without forgetting how to perform old tasks (Masana et al., 2021).

Numerous CIL techniques have emerged recently. For instance, Leo and Katalia, 2021 employed a method where a classification confidence threshold is used to prime the network for incremental learning and reduce forgetting. Another method, Incremental Classifier and Representation Learning (iCaRL) proposed by Rebuffi et al. 2017, allows for class incremental learning by keeping exemplars of past data, which helps prevent catastrophic forgetting, and by learning strong classifiers and data representation together. The use of ‘exemplars’ is controversial however, as one of the main goals of CIL is to eliminate the need to store old datasets. Alternatively, Dai, X., Yin, H., & Jha, N. K. (2020) proposed a grow-and-prune where the neural network first grows new connections to facilitate integration of new data, then prunes connections based on the magnitude of the weights. A different approach taken by Zhang, J., Zhang, J., Ghosh, S., Li, D., Tasci, S., Heck, L., Zhang, H., & Kuo, C.J. (2020) involved model consolidation, wherein new models were trained on new data, and then consolidated via a ‘double distillation’ training objective. Here, the consolidated model is informed by both the new and old models with the help of an auxiliary unlabeled data set.

There are many other examples of CIL, most arising in the past few years. It should be noted that there is some discrepancy concerning the criteria of CIL in the literature. One common theme however is the ability to sequentially learn new tasks, without complete re-training. They all also tackle the issue of catastrophic forgetting either using only a subset of the old data, or none of the old data.

Plant disease detection provides a good use case for class incremental learning. Plants diseases are typically diagnosed visually, thus it’s a reasonable assumption that digital images can contain information pertinent to disease identification. Moreover, there are also publicly available datasets of plants with labeled diseases widely available online. Computer vision models exist to classify plant diseases, but to the author’s knowledge, no class incremental learning techniques have been applied to plant disease detection.

This paper presents a new class incremental learning technique designed to detect and classify diseases in fruit trees given digital images of diseased (or healthy) leaves. This model will utilize convolutional neural network frameworks and be constructed incrementally to include new species of fruit tree and accompanying diseases classes.

1. **Methods**
   1. **Data Sources:**

Models were trained and evaluated on the Plant Village dataset; this dataset contains over 50,000 labeled images of diseased and healthy leaf images from various food crops (Huges and Salathé, 2016). Included in this dataset are several types of fruit tree: apple, peach, cherry, and orange. This dataset was chosen for its simple imagery, ease of access, and size. Limiting the type of objects to solely leaves reduces the complexity, making fruit tree disease modeling more approachable, while remaining practical. Also, because the images are from several different crops, CIL may be approached on a crop-by-crop basis, where each crop may be considered as a new task.

A close up of a leaf

Description automatically generated with medium confidenceA green pillow on a purple surface

Description automatically generated with low confidenceEach of the base images contained a singular leaf on a plain (typically grey) background (Figure 1). The images were red-green-blue (RGB), 256 by 256 pixels. Datasets were not balanced, and often contained different proportions of healthy vs diseased leaves (Table 1). The primary species of interest in this dataset were apple (3171), peach (2675), cherry (1732).

Figure 1. Images of a healthy apple leaf (left) and an apple leaf displaying signs of apple scab disease (right). These are the base images used in modeling prior to augmentation.

**Images per Class**

* 1. **Data Preparation**

Prior to modeling, 10% of the dataset was withheld for later evaluation purposes; this data was not used for training or validation. A further 20% of the dataset was used for validation during the training phase. The remaining 70% of the data was used for training. Pixel color values, originally between 0 and 255, were rescaled to 0 through 1. Various augmentations were also implemented on the training and validation images, including: vertical and longitudinal shifting, horizontal flipping, rotating, shearing, dimming and brightening, and zooming. Mirroring was used to fill void spaces induced by image manipulation. The intent behind image augmentation was to make more robust models by artificially creating a more varied dataset for training. Avoiding overfitting is a common challenge in machine learning and using image augmentation is a simple way to make data more complex, without finding larger datasets, so models trained on it generalize better with unseen images. Testing images were only rescaled; they were not augmented. Data preparation and augmentation were done using the ImageDataGenerator function included in the Keras API for Tensorflow.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Plant** | **Healthy** | **Disease 1** | **Disease 2** | **Disease 3** | **Total** |
| Apple | 1645 | 630 (Apple Scab) | 621 (Black Rot) | 275 (Cedar Apple Rust) | 3171 |
| Peach | 360 | 2297 (Bacterial Spot) |  |  | 2675 |
| Cherry | 684 | 1052 |  |  | 1732 |

* 1. **Model Structure**

The models implemented here were based on Convolutional Neural Networks (CNNs). However, what distinguishes the model architectures here are the use of multiple separate CNNs within the same model. The main objective was to create an architecture that could be incrementally expanded to include additional classes by training new smaller CNNs and appending them to prior CNNs in parallel, negating the need to retrain prior components. Two similar model architectures were explored here (Figure 2).

Table 1. Image counts by plant type and disease/healthy. Classes were not well balanced, and there were more apple diseases than peach or cherry.

The first type of model utilized parallel CNNs, each one trained to differentiate between diseased leaves/healthy leaves for a single type of plant and leaves from other plants. This architecture relies on each sub-network’s ability to distinguish between leaves from the plant it was meant to detect, and other types of leaves. Each sub-network was trained by on a dataset consisting of the plant in question and its various diseased states, along with a generic ‘other’ class which consisted of a mixed set of leaves from plants exclusive of the target. Each CNN was introduced incrementally, expanding number of classifications the model could make.

The second type of model was similar to the first in that it utilized parallel CNNs and relied on the ability of sub-models to distinguish between a target and an ‘other’ class, but consisted of two separate layers. The first layer consisted of binary classifiers which were capable of distinguishing between one target plant, and a generic ‘other’ class similar to those in model type one. However, this layer does not distinguish between diseased and healthy leaves, and both were used in training. The second layer consisted of multi-class classifiers that were capable of distinguishing between healthy and diseased leaves or a single type of plant. The Diagram

Description automatically generatedpurpose of the first layer was to determine which of the models in the second layer was appropriate for any given test sample, while the second one would distinguish between types of diseased and healthy leaves. New classes could be introduced incrementally by adding additional sub-networks for each type of new plant (one to classify target plant or other, and the other to diagnose disease).

Figure 2. Model architectures: there are two types. Type 1 (left) uses individual CNN classifiers for each plant. Type 2 (right) uses both binary classifiers (to determine if the image contains the appropriate plant), and a classifier to determine disease. Type 1 returns the best, ‘non-other’ class fit as the class. Type 2 will first run images through the first binary classification level. This will determine what type of plant the image is, after which the image will then be passed to the sub-model in the second level specific to that plant, which will determine disease.

Both types of models required were dependent on a dataset of generic plant leaves (these were not all fruit tree leaves) in order for sub-models to distinguish between their tree’s leaves and leaves from trees. The hypothesis was that leaves of a specific plant would be more similar to each-other than a generic mix of leaves from other plants, and that leaves any other plant would be more similar to the generic mix of leaves than the specific leaf the sub-network was designed to detect.

For simplicity, ease of training, and comparative purposes, the architecture of the sub-models within each of the models were kept similar. Transfer learning was used extensively: most of the sub-model architecture was comprised of the MobileNetV2 architecture and weights, without the top layer included. MobileNetV2, designed for object detection on mobile devices, was chosen as it has a simple architecture that boasts high accuracy and is computationally inexpensive compared to other state-of-the-art architectures (Sandler et al., 2018). Keeping the pre-trained weights of the base layers reduced model training time and improved model performance. The large dataset (ImageNet) the pre-trained model was trained on gave those base layers a comprehensive understanding of basic features (lines, colors, etc.) Retaining these weights also cut down on training time, allowing more resources to be allocated to training the top layers. The top layers consisted of two fully connected layers, the final one containing the same number of nodes as classes that sub-network was trained on (always two for the first set of sub-models in model type 2). The Keras API for Tensorflow was used to construct the models

* 1. **Model Training**

Each sub-network was trained individually, using its own dataset subset. The data for each sub-network within model type 1 consisted of one type of plant (e.g. apple) with one class for each disease/healthy, and an additional ‘other’ class as (discussed previously). The datasets the first layer of sub-networks in model type 2 were contained of all images of one type of plant (diseased and healthy together) grouped into one class, and another ‘other’ class. The datasets for the second layer in model type 2 were trained on data from one type of plant, with diseases and healthy separated into their own classes. Training was done locally. Tensorflow GPU support was utilized to facilitate training. Training was performed using a GeForce GTX 1050, 2GB graphics card (NVIDIA Corporation, 2016) and an Intel Core i5-7300HQ processor (Intel Corporation, 2017).

* 1. **Model Evaluation**

Models will be evaluated on holdout (test) sets and compared to one-another using categorical accuracy. These holdout sets will only include data not used in training. Due to the class-expansive nature of CIL, different testing sets will be employed depending on what the scope of the model in question. As new tasks are introduced to models, they will be re-evaluated on datasets which contain data pertinent to the new tasks. This will allow for checkpoint comparisons for models as they learn incrementally, which will importantly allow for evaluating performance changes among older tasks. Final models will also be compared to traditional, non CIL networks; these are exposed to all training data during the same session.

1. **Conclusions**
   1. **Current Limitations and Future Implications**

Some of what was written may be subject to future tweaking. Modeling is still in the preliminary phase, and changes may be made. I’ve avoided mentioning some of the more specific parts of the models, such as which loss and activation functions are used, how long the models took to train, and memory usage. In the future I intend to discuss these matters further

It’s also possible that, time allowing, the scope may grow beyond fruit trees. The Plant Village dataset contains samples of many food crops, and utilizing more of it may make for a more robust modeling/evaluation process. Also, to my knowledge, no one has yet to use this dataset in a continuous learning setting, and I suspect the nature of this dataset is well suited to exploring it; it may prove beneficial to use this dataset to its entire extent.

Future sections to be added: Results and Discussion, Conclusions, Abstract. Section 2.5 will be changed to past tense.

**4.0. Appendix**

The Food and Agriculture Organization of the United Nations. (2019, April 3). *New standards to curb the global spread of plant pests and diseases*. Food and Agriculture Organization of the United Nations. [https://www.fao.org/news/story/en/item/1187738/icode/](about:blank)

Geng X., Smith-Miles K. (2015). Incremental Learning. In: Li S.Z., Jain A.K. (eds) *Encyclopedia of Biometrics*. Springer, Boston, MA. [https://doi.org/10.1007/978-1-4899-7488-4\_304](about:blank)

Habib, Md. T., Majumder, A., Jakaria, A. Z. M., Akter, M., Uddin, M. S., & Ahmed, F. (2020). Machine vision based papaya disease recognition. *Journal of King Saud University - Computer and Information Sciences*, *3*, 300–309. [https://doi.org/10.1016/j.jksuci.2018.06.006](about:blank)

Hughes, D. P., & Salathe, M. (2016). An open access repository of images on plant health to enable the development of mobile disease diagnostics. *arXiv [cs.CY]*. Opgehaal van [http://arxiv.org/abs/1511.08060](about:blank)

Intel Corporation. (2017). Intel Core i5-7300HQ @ 2.50 GHz CPU. [Apparatus and software]. [https://ark.intel.com/content/www/us/en/ark/products/97456/intel-core-i5-7300hq-processor-6m-cache-up-to-3-50-ghz.html](about:blank)

Khan, M. A., Akram, T., Sharif, M., Awais, M., Javed, K., Ali, H., & Saba, T. (2018). CCDF: Automatic system for segmentation and recognition of fruit crops diseases based on correlation coefficient and deep CNN features. *Computers and Electronics in Agriculture*, 220–236. [https://doi.org/10.1016/j.compag.2018.10.013](about:blank)

Leo, J., & Kalita, J. (2021). Incremental Deep Neural Network Learning Using Classification Confidence Thresholding. *IEEE Transactions on Neural Networks and Learning Systems*, 1–11. [https://doi.org/10.1109/tnnls.2021.3087104](about:blank)

Liu, B., Zhang, Y., He, D., & Li, Y. (2017). Identification of Apple Leaf Diseases Based on Deep Convolutional Neural Networks. *Symmetry*, *1*, 11. [https://doi.org/10.3390/sym10010011](about:blank)

Liu, J., & Wang, X. (2021). Plant diseases and pests detection based on deep learning: a review. *Plant Methods*, *1*. <https://doi.org/10.1186/s13007-021-00722-9>

Masana, M., Liu, X., Twardowski, B., Menta, M., Bagdanov, A. D., & van de Weijer, J. (2021). Class-incremental learning: survey and performance evaluation on image classification. *arXiv [cs.LG]*. Opgehaal van http://arxiv.org/abs/2010.15277

NVIDIA Corporation. (2016). GeForce GTX 1050 2GB GPU. [Apparatus and software]. [https://www.nvidia.com/en-in/geforce/products/10series/geforce-gtx-1050/](about:blank)

OERKE, E.-C. (2005). Crop losses to pests. *The Journal of Agricultural Science*, *1*, 31–43. <https://doi.org/10.1017/s0021859605005708>

Rebuffi, S., Kolesnikov, A., Sperl, G., & Lampert, C.H. (2017). iCaRL: Incremental Classifier and Representation Learning. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 5533-5542.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2019). MobileNetV2: Inverted Residuals and Linear Bottlenecks. *arXiv [cs.CV]*. Opgehaal van [http://arxiv.org/abs/1801.04381](about:blank)

Yadav, S., Sengar, N., Singh, A., Singh, A., & Dutta, M. K. (2021). Identification of disease using deep learning and evaluation of bacteriosis in peach leaf. *Ecological Informatics*, 101247. [https://doi.org/10.1016/j.ecoinf.202 1.101247](about:blank)

Zhang, J., Zhang, J., Ghosh, S., Li, D., Tasci, S., Heck, L., Zhang, H., & Kuo, C.J. (2020). Class-incremental Learning via Deep Model Consolidation. 2020 IEEE Winter Conference on Applications of Computer Vision (WACV), 1120-1129.