# Pilot Study on Tree Hazard Risk Assessment with Limited Data

Isamu Poy[1], Seiko Goto[2] and Aoi Koga[3]

[1]Graduate Student, University of California, San Diego
(9500 Gilman Dr. La Jolla, CA)
E-mail: ipoy@ucsd.edu
[2]Professor, Dept. of Environmental Science, Nagasaki University
(1-14 Bunkyomachi, Nagasaki, 852-8521, Japan)
E-mail: gotos@nagasaki-u.ac.jp
[3]Member of JSCE, Associate Professor, Center for Information and Communication Technology, Nagasaki University
(1-14 Bunkyomachi, Nagasaki, 852-8521, Japan)
E-mail: amnesia@nagasaki-u.ac.jp (Corresponding Author)

Street trees play an important role in the urban environment and have positive social impacts, such as improving human health. However, they can potentially be hazardous if not properly maintained. Although periodic tree assessments are necessary in urban forest management tasks, such assessments are often neglected because of the lack of trained professionals in local cities. This study aimed to develop a comparison of image classification to identify the problems associated with trees using a limited number of training samples using image classification, object detection, and semantic segmentation.

***Key Words :*** *street trees, risk assessment, transfer learning, deep learning, single-shot multibox detector, CNN, DeepLabV3, image classification, object detection, semantic segmentation*

## 1. INTRODUCTION

Street trees are considered fundamental elements of an urban environment. Nagasaki City is located on the northwest coast of Japan's Kyushu region with a population of approximately 400,000, densely built in a small, reclaimed area surrounded by mountains. It is known as the second A-bombed city during the war in 1945. When the Nagasaki International Culture City Construction Law was passed in 1949, many trees were planted along the main roads.

However, the problem in Nagasaki City is that few roads have enough pedestrian zones in which the required minimum size of the planter for street trees is 50cm x 50cm, whereas the minimum length of one side of the planter in the US is 90cm-150cm. Nagasaki City planted relatively large trees in small planters, such as Chinese tallow, camphor, or Gingko. Furthermore, while mature street trees in San Francisco are pruned once every three to five years, Nagasaki City prunes once or twice every year to keep the tree canopy small. Owing to the small planting strip and frequent excessive pruning, most street trees in Nagasaki city have internal defects and die within 10 years. Currently, approximately 3619 trees are planted throughout the city; however, many of them have decayed roots, which poses a risk of tree falling and damaging powerline wires or neighboring houses when they are blown by strong typhoons during the monsoon season.

**Fig. 1** Heavily pruned street tree in Nagasaki City

The health and risk of street trees are generally identified using tree risk assessments. Therefore, street trees in large cities such as New York, Paris, and Tokyo are managed by specialists; however,

there are no hired experts who can inspect tree conditions in small cities such as Nagasaki [1]. Many applications have been proposed to identify the well-being of plants, such as that of Bhandarkar et al., but very few assess the health of street trees. The problem with street trees is not the general health of the plants, but the hazard caused by deformation and decay. For example, the deformation of plants is not usually related to the degree of plant health, but the deformation of street trees poses a great hazard to roads. It is crucial to have a tool to identify trees at risk for inexperienced city officers, because unhealthy trees in Nagasaki City trees can often pose a significant risk to pedestrians and the environment (**Fig.1**). Demonstrating that this can be done with limited samples would be a significant result, as has been shown in other studies, such as Pan et al. [2), 3)].

## 2. METHODOLOGY

The main purpose of this study was to investigate the feasibility of machine learning for assessing the health of street trees in Nagasaki City. There are various methods for visual tree risk assessment; however, defects in trees are basically assessed by the condition of the root, trunk, and crown, as described by Leers et al. In the tree risk assessment, hazardous conditions are assessed by the following categories: 1) ground heaves; 2) hollows; 3) mold; 4) deteriorating surface/disease; 5) fungi; 6) incline; 7) intrusion; 8) dead branches(**Fig. 2**). The likelihood of tree failure is determined by detections within each category. Among these categories, inspection of dead branches, 8), is not applicable in Nagasaki City because all dead branches were previously removed [4] (**Fig. 1**).

To conduct the experiments, a Google Colaboratory notebook was used in a remote setup using an NVIDIA RTX A6000 GPU, and a prediction model was built based on RGB images taken with an iPhone. The photos were formatted to place the trees on a vertical centerline, and the ground was placed below the horizontal centerline of the photograph, as shown in **Fig. 3**. For roadside trees in Nagasaki City, there was a total of daytime images of 3619 trees, consisting of 22 different species and different symptoms for each tree. The collected images were sorted based on the severity of the following three categories: 1) root heaves; 2) holes; 3) surface defects. The level of health was assessed using only three categories with this research because there were few photographs capturing the incline properly, and there were few cases of mushrooms and intrusion among the collected images. For better performance, the symptoms of the trees were detected using transfer learning and fine tuning. This enabled the classification of images

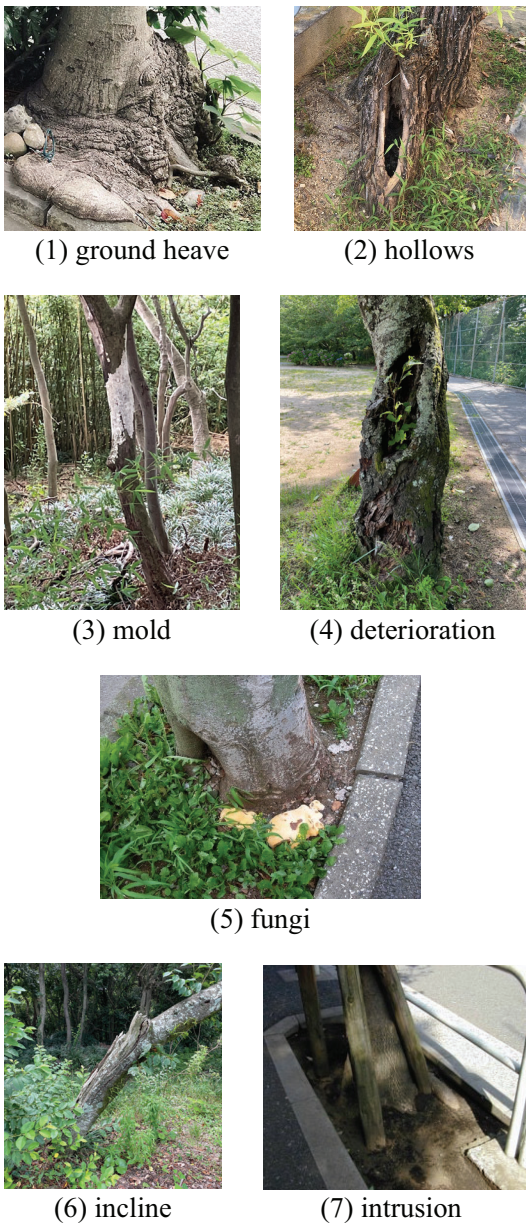



**Fig. 2** Categories of tree risk assessment, hazardous conditions

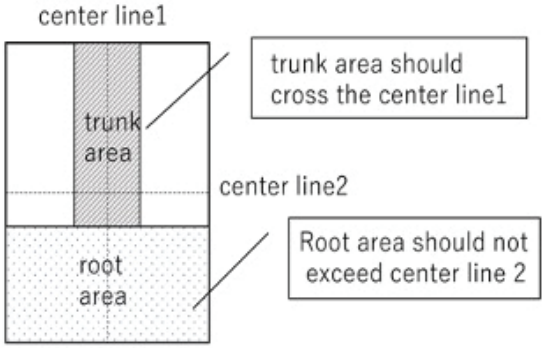



**Fig. 3** Tree photo format

with high accuracy regardless of the small sample size. This classification model is sufficient for categorizing three symptoms with limited samples.

## 3. EXPERIMENTS

Various image classification and object detection frameworks currently exist. Among them, convolutional neural networks (CNNs) and single-shot detectors (SSD) are the most common techniques for applied agriculture. Gandhi et al. and Ghoury et al. have demonstrated applications in agriculture with CNNs and SSDs. Therefore, the symptoms of the three diseases were diagnosed in three ways: 1) CNNs, 2) SSDs, and 3) DeepLabV3 [5), 6)].

### (1) Convolutional Neural Networks

VGG16 is a multilayer neural network containing convolutional and max pooling layers. ReLU and softmax activations were also used [7)].

To detect defects in the tree, the diagnosis was divided into three classes: heaves, holes, and fungi. Separate datasets were created for each class and were trained using VGG16. Because CNNs only classify one class per image, the test accuracy rate was only 65%. Even with fine-tuning, CNNs cannot produce dependable results. CNNs could be useful for classifying images with small variations, such as the satellite imagery classification performed by Kussul et al.; however, object detection with SSD was considered to be more appropriate for detecting multiple symptoms in an image [8)].

### (2) Single Shot MultiBox Detector

SSD is a framework for boundary-box object detection using deep learning developed by Liu et al. SSD300, which is capable of achieving high performance, was used for this paper. It converts the output space into a discrete space of a generated bounding box with different aspect ratios for each feature map position, combines predictions from multiple feature maps at different resolutions, and processes objects of different sizes. This framework is easier to train and superior to other frameworks, such as YOLO by Redmon et al. [9), 10)]

To train the SSD model, images were labelled from level 2 to level 5, corresponding to an increasing severity of the symptoms of the trees (**Table 1**). **Fig. 4** shows SSD classification and data flow. During Step 1, the image was resized to 300 x 300. In Step 2, 8,732 default boxes were generated, and images were then input into the SSD network, which generated a default box according to the offset in step 3. Only reliable boxes were retained during Step 4, and Step 5 provided the corresponding predicted probabilities in Step 6 after duplicate boxes were filtered based on offset information.

**Fig. 5** shows the training performances of the three classes. The loss decreased exponentially, starting at 500 iterations, and continued to decay to a loss of 1.0. Here, an inference algorithm was applied to the SSD to avoid overfitting, and this achieved low loss and high training accuracy.

To contrast the accuracy of SSDs for "heave," "hole," and "fungi," IoU (a popular evaluation metric used in object detection) was used as an indicator for the overlap between the two boundary boxes. Accuracy and recall were computed using the following formulae and those in **Fig. 6**.

**Table 1** Number of boxes per class

| | Labels | | | | | |
|---|---|---|---|---|---|---|
| Class | train | test (level2) | test (level2) | test (level2) | test (level2) | test (overall) |
| heave | 107 | 16 | 14 | 13 | 12 | 55 |
| hole | 143 | 4 | 9 | 8 | 10 | 31 |
| fungi | 213 | 18 | 13 | 29 | 27 | 87 |

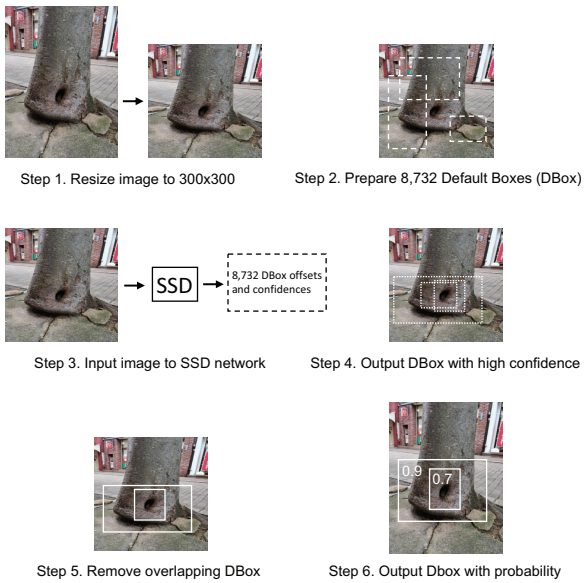



**Fig. 4** Numerical prediction tags

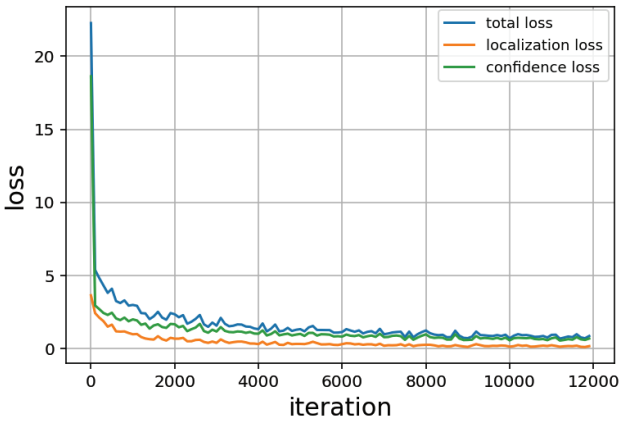



**Fig. 5** Training performance of "heave", "hole", and "fungi"

| | |
|---|---|
| True Positive (TP) | IoU ≥ threshold |
| False Positive (FP) | IoU < threshold |
| False Negative (FN) | ground truth not detected |

**Fig. 6** IoU Equations

The area under the curve (AUC) of the accuracy plot was computed, and a recall plot was used to compare specific performances. Padilla et al. explained that localizers are considered good if they remain highly accurate with increasing recalls. After interpolating all points, the average precision (AP) was recognized as an approximate AUC of the accuracy vs. recall curve shown in **Fig. 7**. In addition, the mean average precision (mAP) and AP were used as metrics to compare the object positioning accuracy. mAP is the average AP of all classes, where AP is obtained from the following equation, p is the precision, and r is the recall value:

$$\int_0^1 p(r)2dr$$

**Table 2** Performance accuracy per class

| Average Precision (AP) | | | | |
|---|---|---|---|---|
| Class | stage5 | stage4 | stage3 | stage2 |
| heave | 91.67% | 79.24% | 88.52% | 83.33% |
| hole | 83.74% | 65.83% | 40.00% | 25.00% |
| fungi | 18.89% | 3.45% | 13.85% | 16.67% |
| mAP | 64.77% | 49.51% | 47.46% | 41.67% |

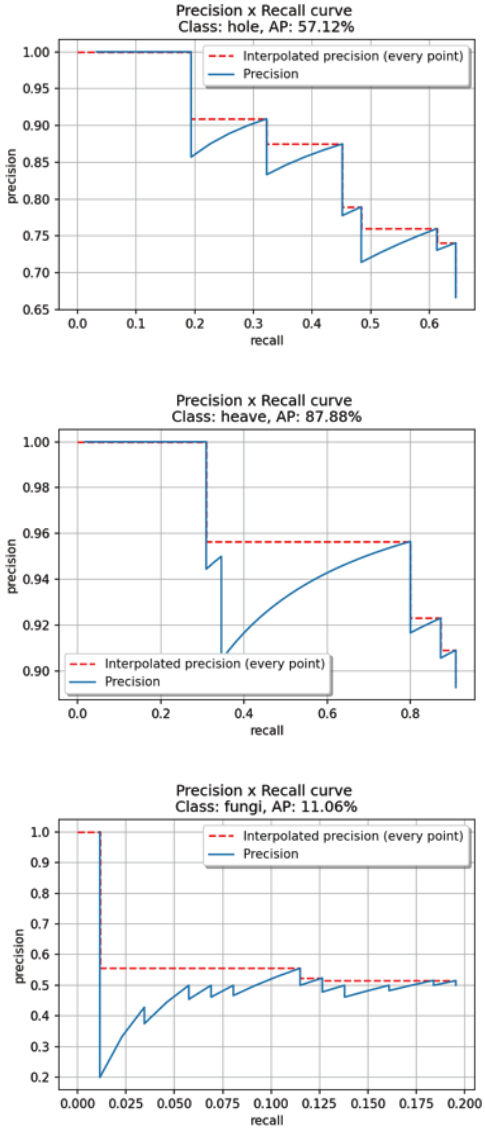



**Fig. 7** Precision vs. recall curves for hole, heave, and fungi

**Table 2** summarizes the results obtained using the SSD model. With the exception of "fungi," overall mAP exceeded 60%, which showed that SSD was effective, considering that the sample size was relatively small. In particular, mAP exceeded 90% at all stages for "heave" and was able to classify tree species from a random test set. SSDs have limited detection for small objects, and it is therefore likely that they cannot detect very small holes in a tree. Furthermore, the accuracy of "fungi" was 20%, although fungi were the most common symptoms observed. This was likely because fungi had varying colors, patterns, and sizes, which are difficult to distinguish, and were scattered around the tree trunk. This is likely unrecognizable using SSD methods [11), 12)].

## 4. SEMANTIC SEGMENTATION

DeepLabV3 was used for semantic segmentation because it is capable of generating finer and more accurate segmented images for various classifications. DeepLabV3, an extension of DeepLab, can divide the area of a tree into its heaves, holes, and fungi. The shape, heaves, holes, and fungi were detected separately, and the outputs were determined using four different AIs, which were integrated together to determine the disease level. The pathological condition of the tree and its severity were analyzed by extracting the location and ratio of defects to the total area of the tree. **Fig. 8** shows the results of DeepLabV3 for several different classifications. Only pixels that matched the shapes were extracted. Matches with the shape were computed using an AND operation, and the heave+hole+fungi classification was computed using an OR operation on the pixel data [13), 14)].

Although the results outputted by DeepLabV3 appeared promising, the segmentation figures did not necessarily help to make a decision as to whether to cut down a tree or to monitor it for hazard risks.

Thus, a weighted calculation was formulated to utilize the segmentation results to create a metric for diagnosing supposedly ill trees. The weight value was associated with each defect; weights corresponding to holes had the highest value because holes indicated an area of the tree that had already begun to rot. Weights corresponding to heaves had the second-highest weight because they are a sign that the tree will begin to lose balance and potentially collapse.

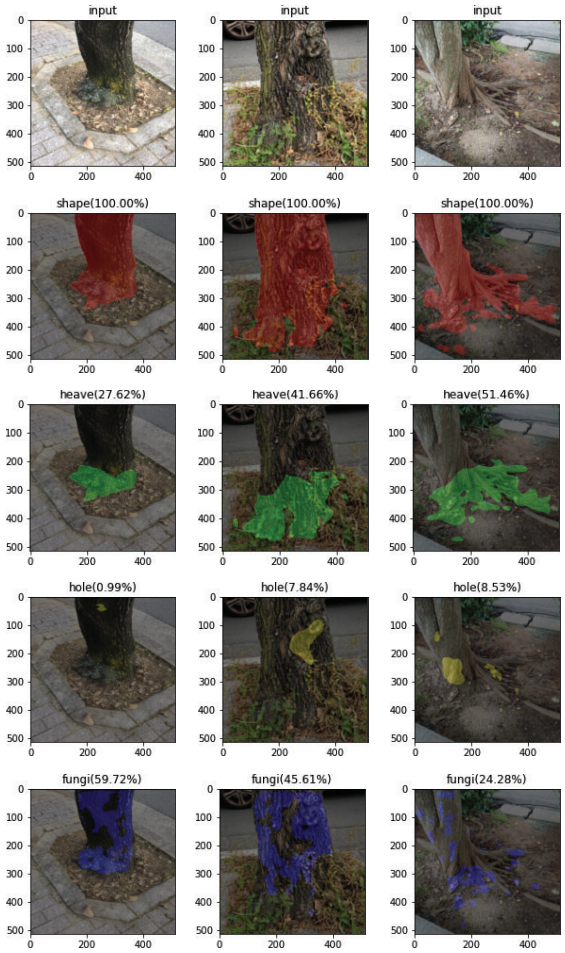
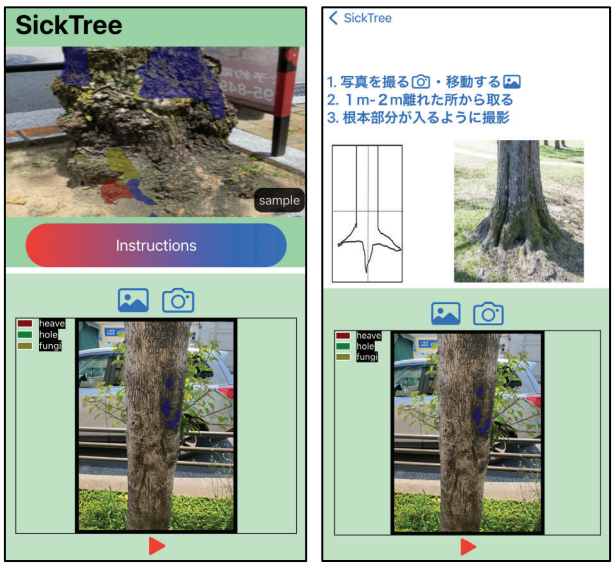



**Fig. 8 DeepLabV3 results on test data**




(a) (b)

**Fig. 9 a)** Home page of the iPhone app, and **b)** page showing image requirements (for Japanese audience)

Fungi had the lowest weight since they were the most abundant and did not pose much risk to the tree's health compared to holes.

During the calculation, the pixel areas of each defect were taken from the segmented image and used to compute a weighted sum (weight associated with each defect multiplied by its area cover). These were subsequently divided by another weighted sum (the weighted sum associated with the total area of the tree multiplied by each weight). Using this result, three thresholds were created for increasing severity (levels 1, 2, and 3) and the trees were classified based on which threshold the tree's metric fell under.

With the Swift programming language, coreML's DeepLabV3 model was used to produce overlays onto a tree image to show any signs of fungal growth, heaves, or holes (**Fig. 9**). An overlay named "shape" was also created to identify the tree silhouette; this was used to compute the denominator of the weighted sum. To test the model in a real-life setting, a minimal user interface was designed (**Fig. 9a**), such that it would be straightforward for anyone from any technical background. Upon opening the app, there is a sample image showing a sample segmented result on the top-half plane (**Fig. 9a**). One can tap on "instructions" to obtain guidance regarding how to use the app (**Fig. 9b**). In the bottom-half plane, the user has the option to either import or take a photo. By tapping the red play button at the bottom of the screen, a segmented result will appear with a legend referencing the color scheme. Each classification had a dedicated color to allow the user to distinguish between different symptoms, and translucent colors were used to allow the user to retain the original image.

## 5. RESULTS

### (1) Annotation Generation

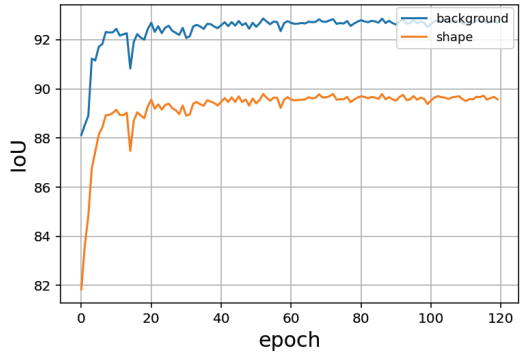
To annotate the obtained images of trees taken on the iPhone, an open-source labeling software called LabelMe was utilized. Using this tool, images were manually annotated using polygon annotations. Polygons were drawn over the section of the image which contained a "hole," "heave," or "fungi" with different colors respectively. Because fungi not only have different textures and colors, but also grow under the same conditions as moss, they require close attention and detailed annotation [15].

### (2) Training

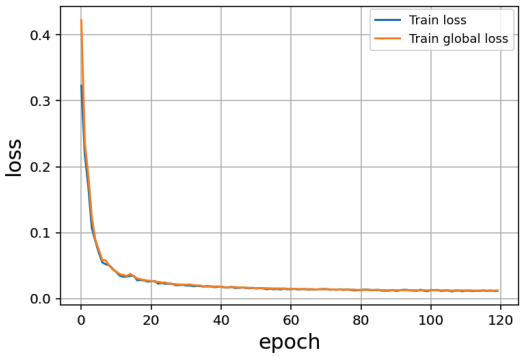
To train DeepLabV3, PyTorch and NumPy libraries were initialized with random seeds. During training, the weights were updated using stochastic gradient descent (SGD). The training parameters were initialized to 120 epochs, a batch size of 4, a weight decay of 0.0001, and a momentum of 0.9. For better performance, pretrained weights were borrowed from ImageNet, and images were cropped to 513 × 513. **Fig. 10a**, **b**, **c**, and d shows the results after training shape class on the DeepLabV3 model for

**Table 3** Confusion matrix of 32 test images

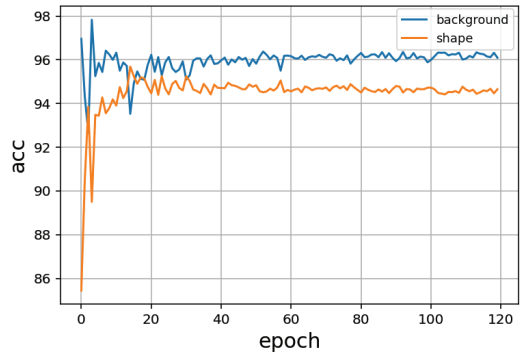
| Actual/Pred. | Level 1 | Level 2 | Level 3 |
|---|---|---|---|
| **Level 1** | 6 | 4 | 0 |
| **Level 2** | 2 | 7 | 2 |
| **Level 3** | 0 | 0 | 11 |



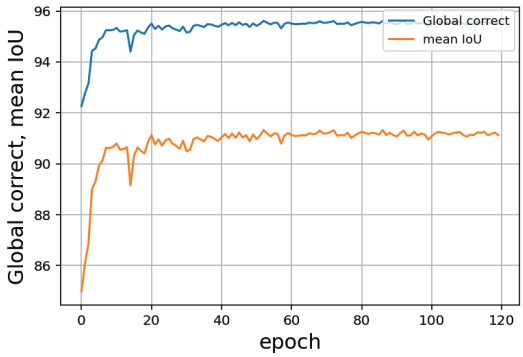

(a) IoU




(b) loss




(c) accuracy




(d) mean IoU

**Fig. 10** For shape a) IoU vs. epoch, b) loss vs. epoch, c) accuracy vs. epoch, and d) mean IoU vs. epoch

heave, hole, fungi, and shape detection. For example, based on **Fig. 10c**, high accuracy was achieved with little loss over 120 epochs. **Fig. 10a** shows that the model has high IoU metrics that translate to high-quality object detection [16].

### (3) Testing

For testing, a small new dataset was constructed containing sick trees diagnosed by professional arborists and healthy trees. The dataset included images classified into three categories: Level 1 (a healthy state), Level 2 (requires monitoring for potential hazard risks), and Level 3 (diagnosed with Witch's Broom Disease and designated to be cut down). Using the threshold values obtained from trial and error, a confusion matrix was created to observe the number of false positives and true positives generated by our model. The thresholds were set such that higher averages fell into categories of higher hazard risk severity.

Based on the results in **Table 3**, the classifier performed well in predicting the correct class. However, if a tree image matched the criteria specified in **Fig. 3**, it resulted in false predictions. For example, if an image did not include the tree root, the classifier misrepresented part of the tree trunk as a heave, which resulted in a more pessimistic result. Thus, in the UI in **Fig. 9a**, an "instructions" tab was included to provide a guide on the type of tree images that the user should import (**Fig. 9b**).

### (4) Comparison

There is no clear distinction between DeepLabV3, CNNs, and SSDs since the three types do not have an overlapping performance metric. However, each model provides a different qualitative insight which helps in tree diagnosis.

DeepLabV3 has taken advantage of both CNNs and SSDs, which ultimately makes it ideal for identifying the locations of illnesses on a tree. Similar to the nature of CNNs, the results of segmented objects were used to construct weighted averages to calculate the final predicted class. DeepLabV3 was able to visualize defects in the tree better than the SSDs because the objects were labeled with colored polygons.

### (5) Limitations

Among the three problem features (holes, heaves, and fungi), holes were easily identified because of their dark color, and heaves were identified because of their outline silhouette. However, fungi were difficult to detect because of variations in color, shape, and texture. When the image resolution was low, the classifier had difficulty distinguishing textures between healthy bark and fungi. Moreover, when the image was shaded, the darker portions of

the image were sometimes classified as ”fungi” even if there was no growth. To identify fungi accurately, resolution and lighting also had to be considered when choosing the test images. Furthermore, although our model did not consider any tilt in the angle between the tree and ground due to the lack of test images, the incline of the tree should additionally be considered in future studies.

## 6. CONCLUSION

In this pilot study, several different techniques were used to make fast, educated decisions on whether a tree would cause potential risk. First, CNNs run into inaccuracies because both trees and their defects appear in various shapes and sizes. Second, although SSDs performed better in localizing the tree’s defects with bounding labels, they were still inaccurate because the localization output was limited to bounding boxes. Third, DeepLabV3 produced the best results for the classification of tree hazard risks. It was flexible in classifying various shapes and allowed a new method of illness classification. Output color-coded overlays made it possible to differentiate defects. This led to a new method using segmented areas and computing weighted averages to classify trees based on a set threshold.

Although street tree diseases could cause hazards in urban environments, few studies have been conducted to automate tree hazard risk assessments. With the advent of applications to remote sensing and agriculture, this study applied state-of-the-art classification models to solve this problem. In future work, tuning the weights of each defect and threshold values will lead to a new learning problem. This pilot study tuned the values through trial and error; however, this process could be further optimized using a neural network.

**ACKNOWLEDGMENT:** This study was supported by the 30th Botanical Research Grant from the Ichimura Foundation for New Technology.

**REFERENCES**

1) G. W. Watson, A. M. Hewitt, M. Custic, and M. Lo, “The management of tree root systems in urban and suburban settings ii: A review of strategies to mitigate human impacts.” Arboriculture & Urban Forestry, vol. 40, no. 5, 2014.
2) S. Bhandarkar, R. Prasad, V. Agarwal, R. Hebbar, D. Uma, Y. Venkata Reddy, Y. Raghuramulu, and K. Ganesha Raj, “Deep learning and statistical models for detection of white stem borer disease in arabica coffee.” International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences, 2019.
3) B. Pan, Z. Shi, and X. Xu, “Mugnet: Deep learning for hyperspectral image classification using limited samples,” ISPRS Journal of Photogrammetry and Remote Sensing, vol. 145, pp. 108–119, 2018.
4) M. Leers, G. Moore, and P. May, “Assessment of six indicators of street tree establishment in melbourne, australia.” Arboriculture & Urban Forestry, vol. 44, no. 1, 2018.
5) R. Gandhi, S. Nimbalkar, N. Yelamanchili, and S. Ponkshe, “Plant disease detection using cnns and gans as an augmentative approach,” in 2018 IEEE International Conference on Innovative Research and Development (ICIRD). IEEE, 2018, pp. 1–5.
6) S. Ghoury, C. Sungur, and A. Durdu, “Real-time diseases detection of grape and grape leaves using faster r-cnn and ssd mobilenet architectures,” in International conference on advanced technologies, computer engineering and science (ICATCES 2019), 2019, pp. 39–44.
7) K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” arXiv preprint arXiv:1409.1556, 2014.
8) N. Kussul, M. Lavreniuk, S. Skakun, and A. Shelestov, “Deep learning classification of land cover and crop types using remote sensing data,” IEEE Geoscience and Remote Sensing Letters, vol. 14, no. 5, pp. 778– 782, 2017.
9) J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779–788.
10) W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “Ssd: Single shot multibox detector,” in European conference on computer vision. Springer, 2016, pp. 21–37.
11) R. Padilla, S. L. Netto, and E. A. B. da Silva, “A survey on performance metrics for object-detection algorithms,” in 2020 International Conference on Systems, Signals and Image Processing (IWSSIP), 2020, pp. 237–242.
12) A. P. Bradley, “The use of the area under the roc curve in the evaluation of machine learning algorithms,” Pattern recognition, vol. 30, no. 7, pp. 1145–1159, 1997.
13) L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, “Rethinking atrous convolution for semantic image segmentation,” arXiv preprint arXiv:1706.05587, 2017.
14) L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 40, no. 4, pp. 834–848, 2018.
15) B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, “Labelme: a database and web-based tool for image annotation,” International journal of computer vision, vol. 77, no. 1, pp. 157–173, 2008.
16) H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, “Generalized intersection over union: A metric and a loss for bounding box regression,” in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 658–666.

(Received September 30, 2022)
(Accepted January 6, 2023)