Predicting tree failure likelihood for utility risk mitigation via artificial intelligence

Jimi Oke[1], Nasko Apostolov[1], Ryan Suttle[2], Sanjay Arwade[1], and Brian Kane[2]

[1]Department of Civil and Environmental Engineering, University of Massachusetts Amherst, MA 01003, USA
[2]Department of Environmental Conservation, University of Massachusetts Amherst, MA 01003, USA

ABSTRACT

Critical to the resilience of utility power lines, tree failure assessments have historically been performed via manual and visual inspections. In this paper, we develop a convolutional neural network (CNN) to predict tree failure likelihood categories under four classification scenarios. Starting with an original set of 505 expert-labeled images, we perform preprocessing and augmentation tasks to increase the number of samples. We optimize several hyperparameters in our CNN and test its performance for three different image resolutions. The trained classifier has an average validation accuracy of at least 0.94 across all scenario-resolution combinations. Thus, via this novel framework, we demonstrate the potential of artificial intelligence to automate and consequently reduce the costs of tree failure likelihood assessments, thereby promoting sustainable infrastructure.

INTRODUCTION

Despite extensive efforts by utilities to prevent them, contacts between tree parts and power lines cause outages that annually result in tens of billions of dollars in economic costs throughout the United States. Presently, the identification of potential contact between trees and power lines is labor intensive and time-consuming. This paper describes an artificial intelligence and machine learning approach that automatically classifies trees, using only a single photograph and with a high degree of accuracy, into categories used by utility arborists to describe the likelihood of tree failure: probable, possible, and improbable. This preliminary study demonstrates the possible efficacy of AI approaches to tree risk assessment and, following further development of the approach, has the potential to reduce power outages and utility costs by allowing utilities to more effectively target their pruning and mitigation efforts.

Contact between tree parts and power lines can take three forms: tree branches can grow into lines; branches can fail and fall onto lines; whole-tree failure can occur due to uprooting or trunk failure. A study in Connecticut, USA provides some context for the amount of economic disruption, documenting annual disruptions of $8.3 billion between 2005 and 2015 (Graziano et al. 2020). That extremely high cost occurred despite extensive efforts on the part of utilities to mitigate conflicts between trees and power lines through active and aggressive pruning programs that, on their own cost billions of dollars annually (Guggenmoos 2003).

Despite its high cost, pruning trees to maintain clearance from power lines is an effective way to reduce outages due to so-called "preventable" contacts between trees and power lines. For example, in Massachusetts, USA, where tree failure was responsible for 40% of preventable tree-caused outages, pruning was able to improve reliability by 20% to 30% (Simpson and Van Bossuyt 1996); similar results were found in a study conducted in Connecticut (Parent et al. 2019). The efficacy of pruning has also been shown in a study of two states in the Gulf Coast region of the USA that showed wind-induced power outage prediction models becoming less uncertain when pruning was included in the model (Nateghi et al. 2014).

Even effective pruning cannot, however, completely eliminate tree-caused outages. Failure of trees outside the right-of-way can still impact the lines and cause outages (Guggenmoos 2003). The proportion of outages caused by failure of trees outside the right-of-way has not been rigorously quantified. Guggenmoos (2011) estimated that 95% of tree-caused outages in the Pacific Northwest region of the USA, were due to tree failure, and Wismer (2018) reported approximately 25% of interruptions in Illinois, USA, were caused by trees that uprooted or broke in the stem.

Predicting the likelihood of failure is an inexact science, but tree risk assessment best management practices have been developed (Smiley et al. 2017; Goodfellow 2020). Estimating tree risk includes assessing the likelihood of tree failure, the likelihood of impact of the failed tree (or tree part) on a target, and the severity of consequences of the impact. The likelihood of failure depends on the anticipated loads on the tree and its load-bearing capacity. The likelihood of impact depends on proximity to the target (the lines, poles, and other hardware—"infrastructure"—in the case of utility tree risk assessment), the target's occupancy rate (which is constant for utility lines) and whether the target is sheltered, for example by neighboring trees. Severity of consequences depends on the damage done to the infrastructure—which, in turn, is partially related to the size of the tree or tree part that fails, and how much momentum it has when it impacts the infrastructure—and, more importantly in some cases, the economic costs and disruption associated with electrical outages.

Individual tree risk assessment can be costly because of the time it requires. In some situations, a less time-consuming assessment may be justified to reduce costs, i.e. a "Level 1" assessment (Smiley et al. 2017). Studies in Rhode Island, USA (Rooney et al. 2005) and Florida, USA (Koeser et al. 2016) have shown that, compared to more time-consuming risk assessments, Level 1 risk assessments successfully identified trees with a higher degree of risk–precisely the trees that arborists prioritize for risk mitigation. The utility of Level 1 assessments demonstrated in these studies suggests that artificial intelligence (AI) tools may be an effective way to reduce the cost of tree risk assessment while still identifying high risk trees.

The method described in the paper uses convolutional neural networks (CNN) to classify images of trees among three categories of failure likelihood: probable, possible, and improbable. The data used for training, testing and illustration of the method consists of 505 tree images that have been classified by the authors according to best management practices used by utility arborists (Goodfellow 2020).

The remainder of the paper provides a brief history and background of AI and its use in infrastructure risk assessment and tree identification (section 2); describes the methods used to train and validate a novel CNN to categorize likelihood of tree failure (section 3); and presents and discusses the output of the novel CNN (sections 4 and 5). The goal is to further demonstrate an innovative automated approach to tree risk assessment using an AI tool that can be readily deployed for use in various locations and also continually improved through subsequent training on new datasets.

BACKGROUND

AI-based image analysis is relatively widely used, even in engineering applications, such as earthquake risk assessment (Jiao and Alavi 2020; Salehi and Burgueño 2018) and structural health monitoring (Spencer et al. 2019; Wang et al. 2019). Neural networks, which comprise a major category of AI frameworks, have been widely applied in the field of earthquake risk assessment (an excellent review is provided by Xie et al. (2020)), but the authors are not aware of attempts to operate directly on, for example, building images in the absence of technical structural data to predict seismic risk. Neural networks have also been used to interrogate remote sensing data of the landscape to assess landslide risk (Su et al. 2020). A few recent efforts have demonstrated the potential for AI-based tree recognition from drone imagery (dos Santos et al. 2019; Egli and Höpke 2020). Furthermore, an application of a convolutional neural network (CNN) to tree species identification using was recently demonstrated by Fricker et al. (2019). Yet, AI has yet to be applied to the problem of tree-utility line risk assessment—one that is complicated by the very large number of tree species to be considered, seasonal variation in tree appearance and associated risk and local meteorological conditions.

The groundbreaking study of Hubel and Wiesel (1959) showed that visual perception in cats was a result of the activation or inhibition of groups of cells in the visual cortex known as "receptive fields." Further, they attempted to map the cortical architecture in cats and monkeys (Hubel and Wiesel 1962; Hubel and Wiesel 1965; Hubel and Wiesel 1968). Subsequent attempts were then made to model neural networks that could be trained to automatically recognize visual patterns with modest performance (Rosenblatt 1962; Kabrisky 1966; Giebel 1971; Fukushima 1975). However, the breakthrough came with the "neocognitron" (Fukushima 1980), which was a self-learning neural network for pattern recognition that was robust to changes in position and shape distortion, a problem that plagued earlier efforts, including the "cognitron" also proposed by Fukushima (1975).

A few notable efforts demonstrated the neural networks for handwritten digit recognition (Fukushima 1988; Denker et al. 1988), but these required significant preprocessing and feature extraction. LeCun et al. (1989) soon afterward introduced a multilayer neural network that mapped a feature in each neuron (representing a "local receptive field") via convolution. This network could also be trained by backpropagation like other existing neural networks and featured pooling operations for better distortion and translation invariance. Further developments from this milestone yielded the LeNet-5 convolutional neural network which attained accuracy levels that rendered it commercially viable.

The big data revolution coupled with technological advancements that have made it possible to capture and store high resolution images have raised challenges that continue to be surmounted with successively high-performing architectures. Over the past decade, some of these efforts resulted in significant breakthroughs in performance. AlexNet (Krizhevsky et al. 2012), with 5 convolutional layers and 3 dense layers—one of the largest CNNs of its time, won the ILSVRC-2012[1] competition with a top-5 error rate of 15.3% and served as a landmark in the Deep Learning subdomain. Zeiler and Fergus (2014) then introduced ZFNet, besting the performance of AlexNet, and pioneered visualization techniques that were foundational for model inference and interpretability. In the same year, GoogLeNet, a 22-layer network, was proposed (Szegedy et al. 2014), featuring the novel "Inception module," which allowed for efficiency and accuracy in a very deep network. Subsequent improvements have been proposed to the original inception framework (Szegedy et al.

[1]ImageNet Large Scale Visual Recognition Challenge; held annually from 2010 through 2017.

2015; Szegedy et al. 2016). VGGNet (Simonyan and Zisserman 2015) also pushed the boundaries of depth with up 19 layers, achieving state-of-the-art performance at ILSVRC-2014. Finally, ResNet (He et al. 2015) addressed the accuracy degradation problem that arises with increasing depth in a network by successively fitting smaller sets of layers to the residual and employing skip connections. With these innovations, an unprecedented level of depth was achieved. Implementations with with 34, 50, 101 and 152 layers were demonstrated. ResNet-152 won first place in ILSVRC-2015.

Along with these developments in their architectures, CNNs have demonstrated viability for applications ranging from image classification, object and text detection to document tracking, labeling, speech, among several other related fields (Gu et al. 2018). In this study, we show that a relatively simple CNN architecture coupled with state-of-the-art approaches for model training and regularization is capable of efficiently and effectively predicting tree failure classes.

DATA AND METHODS

Image data description

The training dataset consisted of 505 images, each having an original size of 4032 × 3024 pixels. Images were captured over a single field season in Massachusetts, USA, between May and September 2020 to limit any potential influence of changes in tree appearance due to seasonal leaf senescence on image processing. ESRI ArcMaps was used to randomly distribute sampling sites across the state. Field assessments of trees to classify likelihood of failure followed the "Level 1" methods outlined in the second edition of the International Society of Arboriculture's (ISA) Tree Risk Assessment Best Management Practices (Smiley et al. 2017) and ISA's Utility Tree Risk Assessment Best Management Practices (Goodfellow 2020). This method is commonly used to assess trees in the United States. A Level 1 assessment was selected for this study because: (1) individual risk assessments may be prohibitively expensive at higher orders, i.e. Level 2 or Level 3 (Smiley et al. 2017), given the hundreds of thousands of trees utilities must manage across territory areas; (2) utility right-of-way (ROW) easements may not allow utility inspectors full access to trees in practical application of higher order risk assessment procedure if the trees are beyond the edge of the ROW (Goodfellow 2020); and (3) studies have shown reasonable efficacy of limited basic visual assessment techniques in identifying more severe tree defects (Rooney et al. 2005; Koeser et al. 2016) leading to greater likelihood of failure ratings. The four categories of likelihood of tree failure, which are always considered in a stated time frame, are defined as follows (Smiley et al. 2017):

- *Improbable*: failure unlikely either during normal or extreme weather conditions;
- *Possible*: failure expected under extreme weather conditions; but unlikely during normal weather conditions;
- *Probable*: failure expected under normal weather conditions within a given time frame;
- *Imminent*: failure has started or is most likely to occur in the near future, even if there is no significant wind or increased load. This is a rare occurrence for a risk assessor to encounter, and may require immediate action to protect targets from impact.

In this study, only images of trees assigned to the likelihood of failure categories of *Improbable*, *Possible* and *Probable* were included in modeling. Images of *Imminent* trees were excluded due to their rarity. Typical examples are shown for each category in Figure 1. In the original set of training images, the class distribution is given in Table 1.

(a) Probable **(b)** Possible **(c)** Improbable

Fig. 1. Examples of training images in each of the three tree risk categories considered in this study. Trees in the right-hand column were categorized as improbable due to their lack of structural defects as well as good physiological health. Trees in the center column were categorized as possible due to weak branch unions and crown dieback. Trees in the left-hand column were categorized as probable because they were dead. Leaves in the bottom image in the left-hand column are from vines attached to the dead tree. Images are antialiased in this figure for greater clarity

Category	**Number of images**
Improbable	322
Possible	80
Probable	56
Total	505

TABLE 1. Category distribution in the set of raw input images

Classification scenarios

In order to investigate the efficacy of an AI classifier to distinguish the failure-likelihood categories, we defined four classification scenarios in Table 2 for our experiments. Each scenario represents a unique grouping of each of the three categories, with a minimum of two derived classes in each case.

Scenario	Description	No. classes
Pr_Im	{*Probable*, *Improbable*}	2
PrPo_Im	{*Probable* + *Possible*, *Improbable*}	2
Pr_PoIm	{*Probable*, *Possible* + *Improbable*}	2
Pr_Po_Im	{*Probable*, *Possible*, *Improbable*}	3

TABLE 2. Classification scenarios

Scenario Pr_Im considered only the highest and lowest likelihood of failure categories used in the study to clearly distinguish between categories. Since previous research has suggested that professionals more often disagree when distinguishing between possible and probable likelihood of failure (Koeser et al. 2020), in scenario PrPo_Im, we pooled trees in the *Probable* and *Possible* categories and compared them to trees in the *Improbable* category. In scenario Pr_PoIm, we pooled trees in the *Possible* and *Improbable* categories and compared them to trees in the *Probable* category. In practice, this scenario is less likely because arborists typically distinguish trees with an *Improbable* likelihood of failure as those with minimal or no structural defects. It requires additional judgment to distinguish trees with probable or possible likelihood of failure because an arborist must assess the severity of structural defects, the presence of response growth, and the expected loads (Smiley et al. 2017). Scenario Pr_Po_Im considered each likelihood of failure category separately, as an arborist would do in practice.

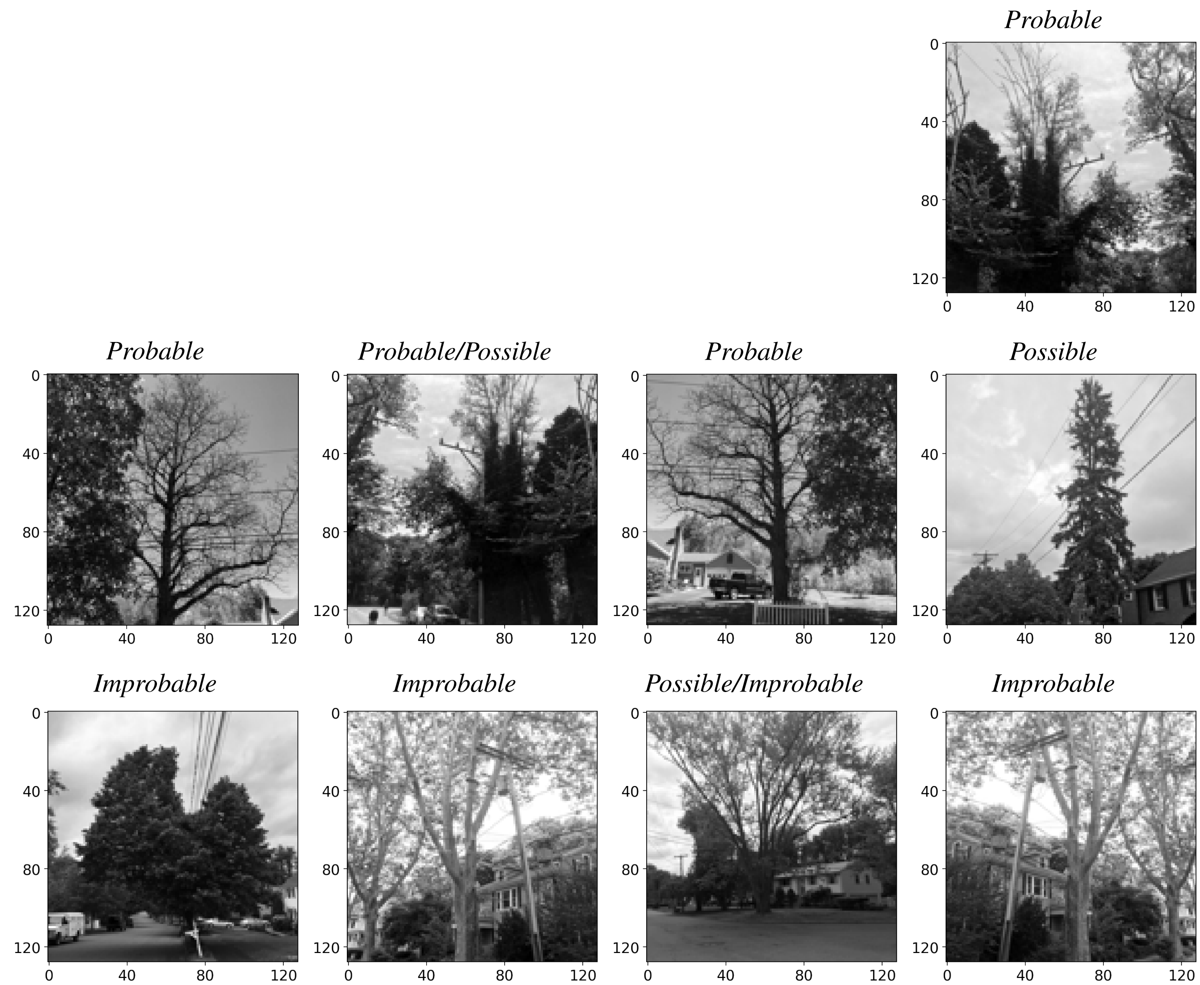
Image pre-processing and augmentation

Data augmentation refers to the variety of methods that are employed for synthetically generating more samples in a training dataset in order to improve model performance (Wong et al. 2016). Augmentation is desired, particularly in situations where the number of original observations is small, and the effectiveness of various relevant techniques in this domain has been amply demonstrated (Shorten and Khoshgoftaar 2019).

In order to achieve robustness in our model, and given the relatively small number of training images, we randomly cropped each image on either axis to 3024×3024 pixels, generating five instances for each one. Thus, we increased the size of our training set from 505 to 2525 images. Further, we performed horizontal flipping with a 50% probability on each of the generated images. For efficiency, we converted the images to grayscale and scaled the pixel values from 0 to 1. Finally, we downsampled the images to the following resolutions (pixels): 64×64, 128×128 and 224×224, creating a training set for each case. Random sets of images from each class across each of the four classification scenarios are shown in Figure 2.

Convolutional neural network

We employ a convolutional neural network (CNN) as the AI framework for tree risk failure likelihood prediction. Like other neural networks, the CNN is an arrangement of neurons within layers, each neuron performing an operation that maps from a pixel in an input image to the final output. The input into each neuron is a weighted sum from the previous layer, while the output from each neuron is modulated by an activation function. The activation function in the final layer of an



(a) Scenario Pr_Im **(b)** Scenario PrPo_Im **(c)** Scenario Pr_PoIm **(d)** Scenario Pr_Po_Im

Fig. 2. Selected processed training images under each classification scenario. The images shown are processed versions of those shown in Figure 1. All were randomly cropped along the vertical axis and some horizontally flipped as well.

CNN is typically the softmax function, which gives the class probabilities of a given input image. A class is assigned to the image based on the one with the corresponding maximum probability.

Unlike other neural networks, however, the CNN performs fundamental pixel-mapping operations in its convolutional layers. Each convolutional layer is defined by a stack of feature maps, which result from a dot product of a filter and correspondinlgy-sized local receptive fields from the input image or preceding layer. The numeric values of the filters correspond to weights whose optimal values are learned during the training of the CNN. The size of the filter in each convolutional layer is given by the *kernel size*.

In this paper, we employ a relatively simple CNN architecture, historically inspired by AlexNet (Krizhevsky et al. 2012). The structure of the CNN is shown in Figure 3 (generated using an automated framework (Bäuerle et al. 2021)). After the input layer (a matrix of pixels from the input image), we use a 64-filter convolutional layer. The output is downsampled using a pooling

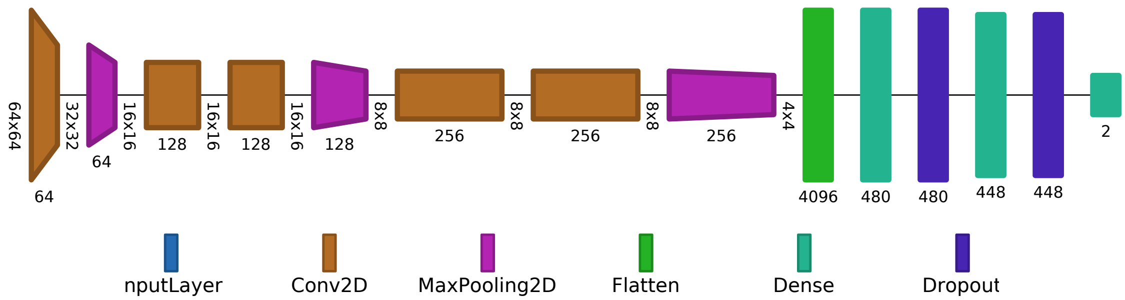


Fig. 3. Diagram of convolutional neural network structure. Hyperparameters that are optimized include the number of units in the penultimate dense layers.

Layer	Layer #	No. Filters	Kernel Size	Strides	Activation	Rate	No. Units
Convolutional	1	64	k^*	2	ReLU		
Max. Pooling	1		2				
Convolutional	2	128	3	1	ReLU		
Convolutional	3	128	3	1	ReLU		
Max. Pooling	2		2				
Convolutional	4	256	3	1	ReLU		
Convolutional	5	256	3	1	ReLU		
Max. Pooling	3		2				
Flatten							
Dense	1				a_1^*		u_1^*
Dropout	1					r_1^*	
Dense	2				a_2^*		u_2^*
Dropout	2					r_2^*	

TABLE 3. Summary of the convolutional neural network hyperparameters. Those indicated by an asterisked symbol are optimized using a guided search.

layer that returns the maximum ouput from a 2×2 subsample from the previous layer. Next, we stack two successive convolutional layers each with a depth of 128 filters. We follow these with a maximum-pooling layer and then two further 256-filter convolutional layers. A final maximum-pooling layer is used before we "flatten" all outputs into a one-dimensional (fully-connected) array of neurons. After flattening the outputs, we use a dense layer to reduce the number of outputs a specified number of units. Batch normalization (Ioffe and Szegedy 2015) is employed after the first dense layer to improve training efficiency. A second dense layer is used prior to the output layer, with the number of outputs corresponding to the number of classes in the dataset. A regularization technique known as "dropout" is used after each dense layer. In each dropout layer, a proportion of the neurons are randomly zeroed during training in order to improve the robustness of the model.

The various hyperparameters in the model are summarized in Table 3.

Hyperparameter optimization

We optimized eight of the CNN hyperparameters using Hyperband (Li et al. 2018), an efficient guided grid-search algorithm. Twelve searches were performed for each classification scenario and image resolution combination. Each search was conducted using 90 trials of unique hyperparameter combinations. The specified range of each parameter along with the search results are shown in Table 4. For the kernel size in the first convolutional layer, we allowed for a choice between a 5×5 and a 7×7 kernel. The activation function in both dense layers was specified as a choice between the rectified linear unit (ReLU) function and the hyperbolic tangent (tanh). The ReLU was introduced to address the so-called "vanishing gradient" problem and has been shown to improve performance in CNNs (Glorot et al. 2011). Nevertheless, the tanh function remains a viable option, as well. The dropout rates were allowed to range from 0 to 5 in steps of 0.05, while the number of neurons or units in each dense layer varied from 32 to 512 in steps of 32. Finally, we uniformly sampled learning rates for the optimizer in the $\log_{10}$ space of $[10^{-4}, 10^{-2}]$.

Model training and assessment

In this subsection, we provide an overview of the learning procedure for the convolutional neural network. As a reference, all the symbols used here are summarized in Table 5

The softmax activation function $f(\cdot)$ in the output layer returns the class prediction probabilities for a given observation i. It is defined in terms of the class-specific score s_c as:

$$f(s_c) = \frac{e^{s_c}}{\sum\limits_{c' \in C} e^{s_{c'}}} \tag{1}$$

where C is the set of classes and c, c' are indices for a given class. Thus for the ith observation, the softmax activation returns the predicted probability $\hat{p}_{i,c}$ that the ith observation belongs to class c. The CNN is trained using a variant of the stochastic gradient algorithm, Adam (Kingma and Ba 2017). The goal of the training procedure is to learn the optimal weights and bias terms for the CNN by minimizing a loss function. In this case, we use the categorical cross-entropy loss function, which for a single observation can be simply defined as:

$$L_i^{CE} = -\log(\hat{p}_{i,c}) = -\log(f(s_c^i)) \tag{2}$$

Training is iteratively performed, with gradient of the loss function computed and averaged over a batch of input images. Here, we use a batch size of 32. The learning rate of the optimization algorithm is an important hyperparameter that affects training performance. We optimized for this in the hyperparameter search as discussed. Furthermore, the CNN is trained over multiple passes through the entire training set. Each such pass is referred to as an epoch.

In real terms, we measured the performance of the trained CNN by how accurately it predicts the classes in a validation set excluded from the training set. For this paper, we used a randomly sampled validation that was 20% of the size of the input dataset of 2525 images in each training instance. Thus, we define the accuracy as the overall proportion of correct predictions across all classes. This metric was computed both for the training and validation sets in each epoch. In addition to overall accuracy of making correct classifications, we assessed the models trained under these scenarios based on the macro-averages of the precision, recall and F_1 score metrics computed over the validation set in each epoch. The precision score Pr_c captures the proportion of correct predictions for a certain class relative to all the predictions for that class, and is an

Scenario	Hyperparameter	Range	Resolution		
			64	**128**	**224**
Pr_Im	1st conv. kernel size, k	$\{5,7\}$	7	7	7
	1st dense activation, a_1	{ReLU, tanh}	ReLU	ReLU	ReLU
	2nd dense activation, a_2	{ReLU, tanh}	ReLU	ReLU	ReLU
	1st dropout rate, r_1	$\{0, .05, \dots, 5\}$	0.1	0.1	0.1
	2nd dropout rate, r_2	$\{0, .05, \dots, 5\}$	0.3	0.3	0.3
	1st dense layer units, u_1	$\{32, 64, \dots, 512\}$	480	480	480
	2nd dense layer units, u_2	$\{32, 64, \dots, 512\}$	448	448	448
	learning rate, λ	$[10^{-4}, 10^{-2}]$	$1.03 \cdot 10^{-4}$	$1.03 \cdot 10^{-4}$	$1.03 \cdot 10^{-4}$
PrPo_Im	1st conv. kernel size, k	$\{5,7\}$	7	5	7
	1st dense activation, a_1	{ReLU, tanh}	ReLU	tanh	ReLU
	2nd dense activation, a_2	{ReLU, tanh}	ReLU	ReLU	ReLU
	1st dropout rate, r_1	$\{0, .05, \dots, 5\}$	0.1	0.25	0.1
	2nd dropout rate, r_2	$\{0, .05, \dots, 5\}$	0.3	0.35	0.3
	1st dense layer units, u_1	$\{32, 64, \dots, 512\}$	480	384	480
	2nd dense layer units, u_2	$\{32, 64, \dots, 512\}$	448	256	448
	learning rate, λ	$[10^{-4}, 10^{-2}]$	$1.03 \cdot 10^{-4}$	$1.09 \cdot 10^{-4}$	$1.03 \cdot 10^{-4}$
Pr_PoIm	1st conv. kernel size, k	$\{5,7\}$	7	7	7
	1st dense activation, a_1	{ReLU, tanh}	ReLU	ReLU	ReLU
	2nd dense activation, a_2	{ReLU, tanh}	tanh	ReLU	tanh
	1st dropout rate, r_1	$\{0, .05, \dots, 5\}$	0.25	0.1	0.2
	2nd dropout rate, r_2	$\{0, .05, \dots, 5\}$	0.3	0.3	0.15
	1st dense layer units, u_1	$\{32, 64, \dots, 512\}$	128	480	416
	2nd dense layer units, u_2	$\{32, 64, \dots, 512\}$	320	448	416
	learning rate, λ	$[10^{-4}, 10^{-2}]$	$1.76 \cdot 10^{-4}$	$1.03 \cdot 10^{-4}$	$1.15 \cdot 10^{-4}$
Pr_Po_Im	1st conv. kernel size, k	$\{5,7\}$	7	5	7
	1st dense activation, a_1	{ReLU, tanh}	ReLU	tanh	ReLU
	2nd dense activation, a_2	{ReLU, tanh}	tanh	ReLU	ReLU
	1st dropout rate, r_1	$\{0, .05, \dots, 5\}$	0.25	0.25	0.1
	2nd dropout rate, r_2	$\{0, .05, \dots, 5\}$	0.3	0.35	0.3
	1st dense layer units, u_1	$\{32, 64, \dots, 512\}$	128	384	480
	2nd dense layer units, u_2	$\{32, 64, \dots, 512\}$	320	256	448
	learning rate, λ	$[10^{-4}, 10^{-2}]$	$1.76 \cdot 10^{-4}$	$1.09 \cdot 10^{-4}$	$1.03 \cdot 10^{-4}$

TABLE 4. Optimal hyperparameters found using the Hyperband search algorithm for 12 classification scenario and input image resolution combinations.

important measure of how good a classifier is. The recall Re_c captures the ability of a classifier to correctly predict observations for a certain class relative to all the true observations in that class. The F_{1c} metric is given as the class-specific harmonic mean of the precision and recall, and is thus more sensitive than the overall accuracy score. These three metrics are macro-averaged. Thus, each category is given equal weight, ensuring that misclassifications within the smaller classes

Symbol	Definition
c	Index of given class
$f(s_c)$	Softmax activation function
F_{1c}	Class-specific F_1 score
F_1^m	Macro-average F_1 score
i	Index of given image observation
L_i^{CE}	Categorical cross entropy loss function of a single observation
$\hat{p}_{i,c}$	Predicted probability that the i^{th} observation belongs to class c
Pr_c	Class-specific precision
Pr^m	Macro-average precision
Re_c	Class-specific recall
Re^m	Macro-average recall
s_c	Class-specific score
y_i	Observed (true) class of a given observation
$\hat{y}_i$	Predicted class of a given observation

TABLE 5. Summary of symbols related to the training and assessment of the convolutional neural network

(*Probable* and *Possible*) are adequately represented in the aggregation. These metrics are formally defined as follows:

$$\text{Macro-average precision: } Pr^m = \frac{1}{|C|}\sum_{c\in C}\left(\frac{\mathrm{TP}_c}{\mathrm{TP}_c + \mathrm{FP}_c}\right) \tag{3}$$

$$\text{Macro-average recall: } Re^m = \frac{1}{|C|}\sum_{c\in C}\left(\frac{\mathrm{TP}_c}{\mathrm{TP}_c + \mathrm{FN}_c}\right) \tag{4}$$

$$\text{Macro-average } F_1 \text{ score: } F_1^m = \frac{1}{|C|}\sum_{c\in C}\left(\frac{2\mathrm{Pr}_c\mathrm{Re}_c}{\mathrm{Pr}_c + \mathrm{Re}_c}\right) \tag{5}$$

where c is the index of a class in the set C and $|C|$ the number of classes in the dataset. The class-specific prediction metrics are given by:

$$\text{True positives for class } c\text{: } \mathrm{TP}_c = \sum_{i\in c} I(\hat{y}_i = y_i) \tag{6}$$

$$\text{False positives for class } c\text{: } \mathrm{FP}_c = \sum_{i\in c} I(\hat{y}_i \in c | y_i \notin c) \tag{7}$$

$$\text{False negatives for class } c\text{: } \mathrm{FN}_c = \sum_{i\in c} I(\hat{y}_i \notin c | y_i \in c) \tag{8}$$

where $\hat{y}_i$ is the predicted class and y_i the observed (true) class for a given image i in class c. The indicator function $I(\cdot)$ returns 1 if the corresponding condition is true, and 0 otherwise.

RESULTS

We trained the CNN for each of the 4 classification scenarios. In each scenario, we also trained on three image resolution sets (64, 108 and 224). The goal was to determine the best performing resolution, given the tradeoff between performance and computational expenditure. Thus, we generated twelve learning cases. In each case, we performed a 5-fold cross-validation (resulting in a training-validation ratio of 80:20 in each fold). We trained the CNN over 20 epochs in all the cases. The trajectories of the loss and accuracy are shown in Figure 4 for both the training and validation sets.

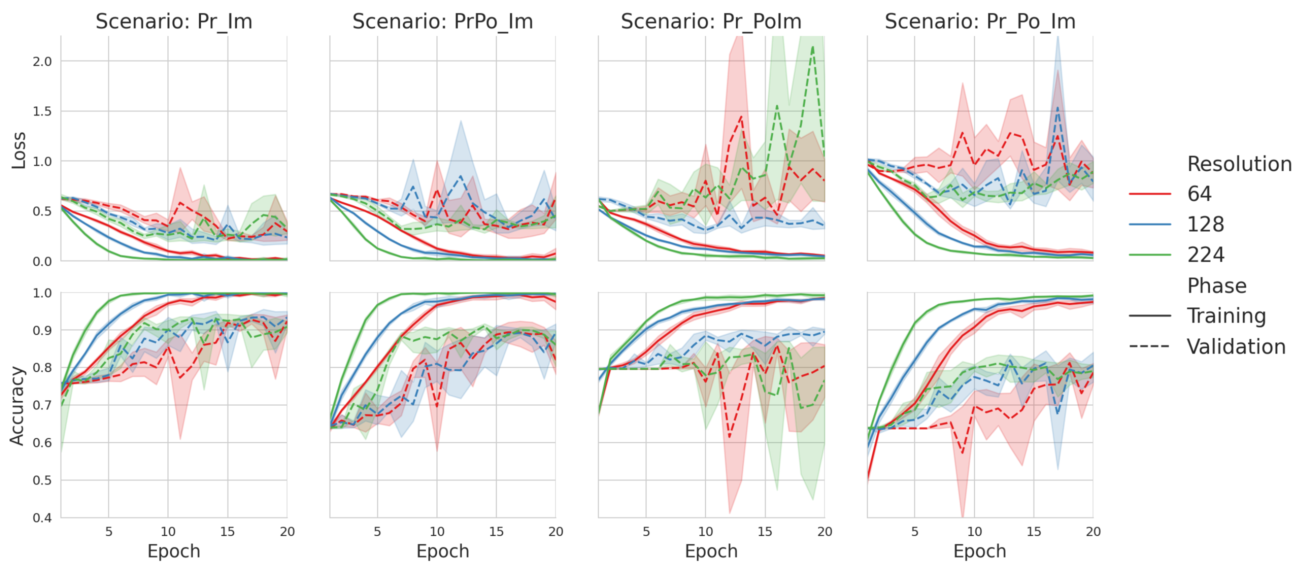


Fig. 4. Performance metrics for each scenario and input resolution instance. Average metrics and 95% confidence intervals (trials, $n = 5$) are shown for 20 epochs in each case.

Sensitivity to training resolution

Figure 5 shows the boxplots of the validation performance metrics for all twelve cases considered. The metrics are: accuracy and the macro-averages of precision, recall and the F_1 score. Taking all metrics into consideration, Welch pairwise tests showed that there are no significant differences in performance among the 3 training resolutions, with one exception. In the scenario Pr_PoIm, the accuracy, F_1^m, Re^m for the 128-pixel case are greater than those for the 224-pixel case ($p < .001$). This difference in performance is not as stark between the 128-pixel and 64-pixel cases in the same scenario. This outcome implies that we can achieve efficiency by training at lower resolutions without significant losses in performance.

Scenario sensitivity

We conducted pairwise tests between each scenario combination for all resolutions. The results indicated that considering all metrics, the scenarios are statistically significantly different from each other ($p < .005$) except Pr_Im compared to PrPo_Im and Pr_PoIm compared to Pr_Po_Im, with respect to certain metrics. If the accuracy is not taken into account, then there is no strong evidence that Pr_Im differs in performance when compared with PrPo_Im. In the case of Pr_PoIm versus Pr_Po_Im, there is greater evidence to support their similarity in performance with respect to F_1^m and Re^m. Otherwise, the performance between these two scenarios is significantly different. From Figure 5, we observe the best performances in Pr_Im and PrPo_Im, with all metrics greater than or

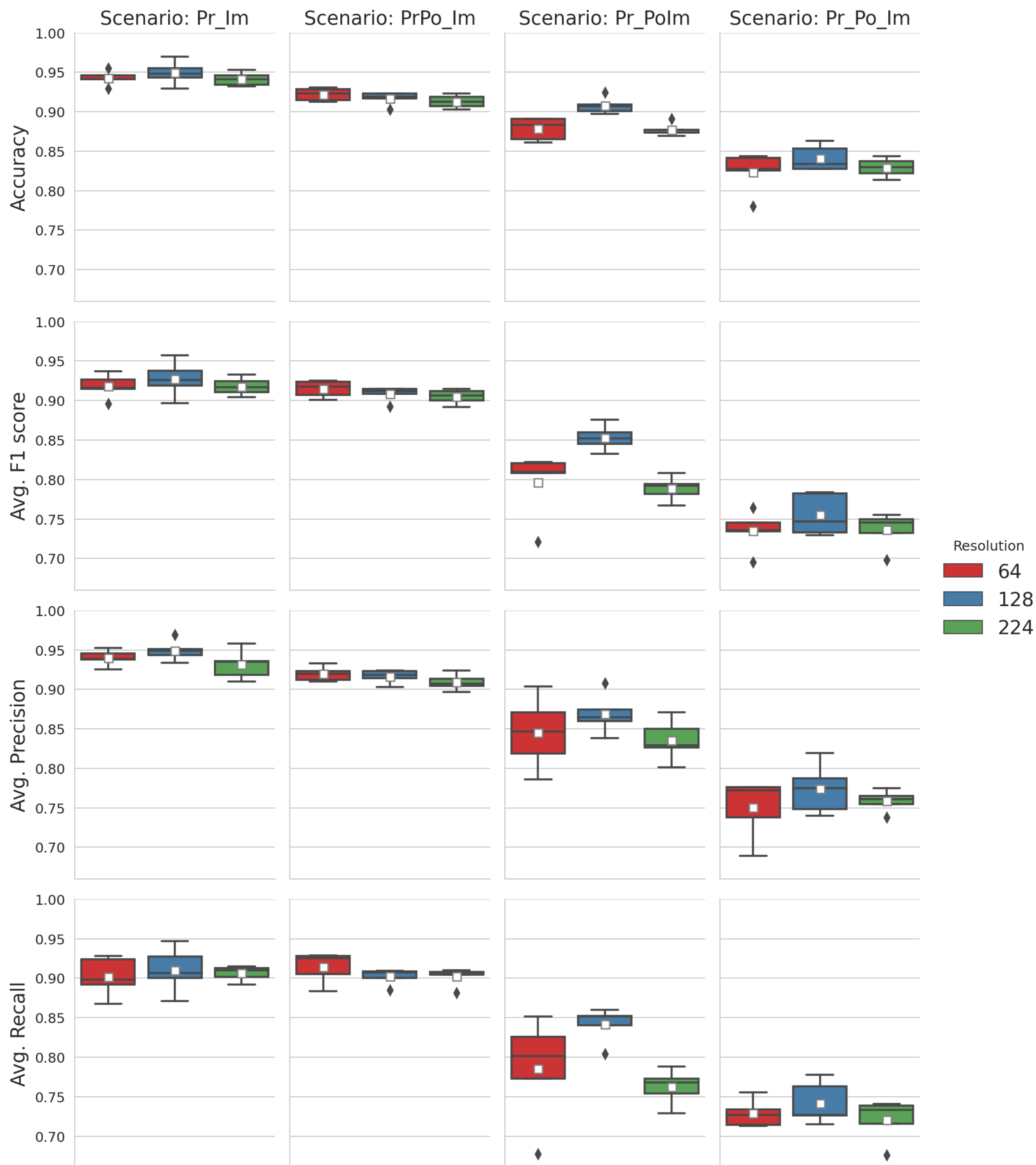


Fig. 5. Boxplots of validation performance metrics for each scenario and resolution combination. The upper and lower bounds of each box are the first and third quartiles; the whiskers are three standard deviations apart; and the diamonds are outliers. Mean values are depicted as white squares in each box.

equal to 0.90. The performance in Pr_PoIm is lower, and Pr_Po_Im has the lowest performance of all four.

Analysis of classification strategies

Given that there was no significant difference in performance among the three resolutions tested, we focus our attention on the 128-px case, as a tradeoff between efficiency and performance. We re-trained the model on all the four scenarios and then evaluated performance on the following validation metrics: average precision, average recall and accuracy. The goal of this analysis was to explore the efficiencies of various classification strategies for tree failure risk. Using a randomly sampled validation set that was 20% of the augmented dataset as before, we trained the CNN with the respective optimal hyperparameters for a single instance in order to compare the performance across the scenarios. The metrics are shown in Figure 6. In each scenario, we trained the model over 20 epochs, but we employed early stopping using validation loss as the criterion with a patience of 3. That is, the training routine would set the weights of the final model at the epoch from which there was no further improvement in validation loss after three subsequent iterations. From this figure, we see that Pr_Po_Im performed significantly worse than the other three. This indicates the uncertainty surrounding the subjective risk classification of the trees to begin with. Given the results from the sensitivity tests, however, it is possible that a different training instance may have provided better results, or perhaps, further iterations were required to achieve convergence.

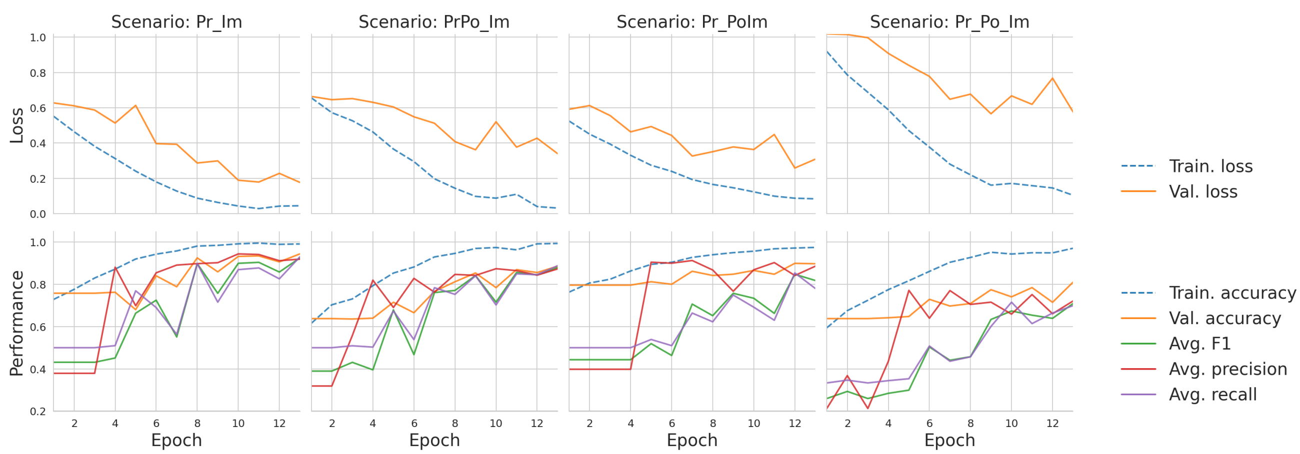
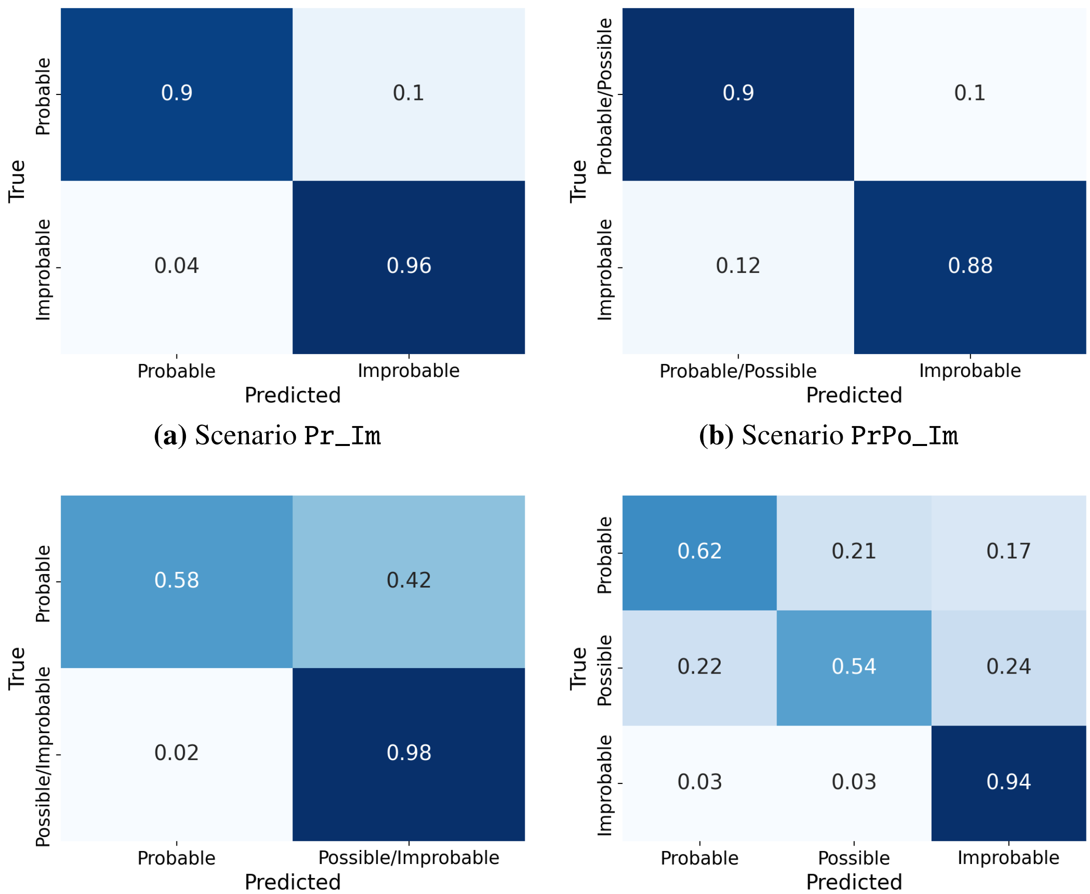


Fig. 6. Model performance across four classification scenarios for a single training instance in each case. Training resolution was 128px. All weighted average metrics were computed on the validation set.

We also plotted the confusion matrix for each scenario in Figure 7. Each row of the confusion matrix indicates the proportion of observations in a given class that are predicted to be in the classes across the columns. Thus, the diagonal entry in each matrix represents the class-specific recall score, Re_c. The matrices show that generally, the "Possible" category is difficult to predict accurately as an individual class. The classifier performed best when it only had to distinguish between *Probable* and *Improbable* (Figure 7a). Performance only suffered slightly when *Possible* was grouped with *Probable* in the PrPo_Im scenario (Figure 7b).



(a) Scenario `Pr_Im` **(b)** Scenario `PrPo_Im`

(c) Scenario `Pr_PoIm` **(d)** Scenario `Pr_Po_Im`

Fig. 7. Scenario confusion matrices for a single training instance using the 128-pixel images. Each row of a confusion matrix indicates the proportional distribution of class predictions for the true members of each class. Thus, the diagonals indicate the recall for each class Re_c. The average of the diagonal values gives the macro-average Re^m for each scenario.

CONCLUSION

We have demonstrated the efficacy of an artificial intelligence framework for predicting tree failure likelihood (*Probable*, *Possible* and *Improbable*) with respect to utility infrastructure. Specifically, we developed a convolutional neural network with state-of-the-art configurations. We applied data augmentation and pre-processing strategies to increase the size of our dataset by a factor of 5, thus generating 2525 images. From an initial resolution of 4032×3024 pixels, we created three sets of image resolutions: 64×64, 128×128 and 224×224. We then optimized eight of our CNN hyperparameters for 12 classification scenario and image resolution combinations. We conducted ten model training trials in each of the 12 cases, using a 20% validation set to measure model performance. In these trials, the average accuracy of classification was ≥ 0.94, with a maximum standard deviation of 0.7. There was no statistically significant difference between the performance across image resolutions.

We further conducted more detailed classification performance analyses over single training instances for the 128-pixel case. Our results indicate that the CNN performed best at recalling *Probable* vs. *Improbable* cases. The *Possible* category appeared to be a confounding case. This was not unexpected, given the uncertainty and subjectivity in assessing these trees to begin with. The CNN, however, performed better at distinguishing between *Possible*/*Improbable* and *Probable*, than between *Probable*/*Possible* and *Improbable*. This might be an indicator that trees in the *Possible* category are more likely to be identified as *Improbable*. When all three failure-likelihood categories were predicted individually, the *Possible* category had the lowest recall score.

Nevertheless, given the relatively small input dataset of original images, these preliminary results are extremely promising for future improvements. First, we can train better models with more data. We also plan to rigorously quantify the uncertainty in ground truth category assignments by incorporating predictions from multiple experts on the same images. In order to better understand how the CNN is classifying each image, we will conduct extensive visual inference in further work, for instance, using the gradient-weighted class activation mapping approach (Zeiler and Fergus 2014).

DATA AVAILABILITY STATEMENT

All data, models, or code generated or used during the study are available in a GitHub repository online at https://github.com/narslab/tree-risk-ai.

ACKNOWLEDGMENTS

The authors acknowledge the support of Eversource.

REFERENCES

Bäuerle, A., van Onzenoodt, C., and Ropinski, T. (2021). "Net2Vis – A Visual Grammar for Automatically Generating Publication-Tailored CNN Architecture Visualizations." *IEEE Transactions on Visualization and Computer Graphics*, 1–1.

Denker, J., Gardner, W., Graf, H., Henderson, D., Howard, R., Hubbard, W., Jackel, L. D., Baird, H., and Guyon, I. (1988). "Neural Network Recognizer for Hand-Written Zip Code Digits." *Advances in Neural Information Processing Systems*, 1, 323–331.

dos Santos, A. A., Marcato Junior, J., Araújo, M. S., Di Martini, D. R., Tetila, E. C., Siqueira, H. L., Aoki, C., Eltner, A., Matsubara, E. T., Pistori, H., Feitosa, R. Q., Liesenberg, V., and Gonçalves, W. N. (2019). "Assessment of CNN-Based Methods for Individual Tree Detection on Images Captured by RGB Cameras Attached to UAVs." *Sensors*, 19(16), 3595.

Egli, S. and Höpke, M. (2020). "CNN-Based Tree Species Classification Using High Resolution RGB Image Data from Automated UAV Observations." *Remote Sensing*, 12(23), 3892.

Fricker, G. A., Ventura, J. D., Wolf, J. A., North, M. P., Davis, F. W., and Franklin, J. (2019). "A Convolutional Neural Network Classifier Identifies Tree Species in Mixed-Conifer Forest from Hyperspectral Imagery." *Remote Sensing*, 11(19), 2326.

Fukushima, K. (1975). "Cognitron: A self-organizing multilayered neural network." *Biological Cybernetics*, 20(3), 121–136.

Fukushima, K. (1980). "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position." *Biological Cybernetics*, 36(4), 193–202.

Fukushima, K. (1988). "Neocognitron: A hierarchical neural network capable of visual pattern recognition." *Neural Networks*, 1(2), 119–130.

Giebel, H. (1971). "Feature Extraction and Recognition of Handwritten Characters by Homogeneous Layers." *Zeichenerkennung Durch Biologische Und Technische Systeme / Pattern Recognition in Biological and Technical Systems*, O.-J. Grüsser and R. Klinke, eds., Berlin, Heidelberg, Springer, 162–169.

Glorot, X., Bordes, A., and Bengio, Y. (2011). "Deep Sparse Rectifier Neural Networks." *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, JMLR Workshop and Conference Proceedings, 315–323 (June).

Goodfellow, J. W. (2020). "Best Management Practices - Utility Tree Risk Assessment." *Report No. P1321*, International Society of Arboriculture.

Graziano, M., Gunther, P., Gallaher, A., Carstensen, F. V., and Becker, B. (2020). "The wider regional benefits of power grids improved resilience through tree-trimming operations evidences from Connecticut, USA." *Energy Policy*, 138, 111293.

Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T., Wang, X., Wang, G., Cai, J., and Chen, T. (2018). "Recent advances in convolutional neural networks." *Pattern Recognition*, 77, 354–377.

Guggenmoos, S. (2003). "EFFECTS OF TREE MORTALITY ON POWER LINE SECURITY." *Journal of Arboriculture*, 29(4), 181–196.

Guggenmoos, S. (2011). "Tree-related Electric Outages Due To Wind Loading." *Arboriculture and Urban Forestry*, 37(4), 147–151.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). "Deep Residual Learning for Image Recognition." *arXiv:1512.03385 [cs]*.

Hubel, D. H. and Wiesel, T. N. (1959). "Receptive fields of single neurones in the cat's striate cortex." *The Journal of Physiology*, 148(3), 574–591.

Hubel, D. H. and Wiesel, T. N. (1962). "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex." *The Journal of Physiology*, 160(1), 106–154.2.

Hubel, D. H. and Wiesel, T. N. (1965). "Receptive fields and functional architecture in two nonstriate visual areas (18 and 19) of the cat." *Journal of Neurophysiology*, 28(2), 229–289.

Hubel, D. H. and Wiesel, T. N. (1968). "Receptive fields and functional architecture of monkey striate cortex." *The Journal of Physiology*, 195(1), 215–243.

Ioffe, S. and Szegedy, C. (2015). "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift." *arXiv:1502.03167 [cs]*.

Jiao, P. and Alavi, A. H. (2020). "Artificial intelligence in seismology: Advent, performance and future trends." *Geoscience Frontiers*, 11(3), 739–744.

Kabrisky, M. (1966). *A Proposed Model for Visual Information Processing in the Human Brain*. University of Illinois Press.

Kingma, D. P. and Ba, J. (2017). "Adam: A Method for Stochastic Optimization." *arXiv:1412.6980 [cs]*.

Koeser, A. K., McLean, D. C., Hasing, G., and Allison, R. B. (2016). "Frequency, severity, and detectability of internal trunk decay of street tree Quercus spp. in Tampa, Florida, U.S.." *Arboriculture & Urban Forestry*, 42(4), 217–226.

Koeser, A. K., Thomas Smiley, E., Hauer, R. J., Kane, B., Klein, R. W., Landry, S. M., and Sherwood, M. (2020). "Can professionals gauge likelihood of failure? – Insights from tropical storm Matthew." *Urban Forestry & Urban Greening*, 52, 126701.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). "ImageNet Classification with Deep Convolutional Neural Networks." *Advances in Neural Information Processing Systems*, 25, 1097–1105.

LeCun, Y., Boser, B. E., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. E., and Jackel, L. D. (1989). "Handwritten Digit Recognition with a Back-Propagation Network." *Advances in Neural Information Processing Systems 2*, 9.

Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., and Talwalkar, A. (2018). "Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization." *Journal of Machine Learning Research*, 18(185), 1–52.

Nateghi, R., Guikema, S., and Quiring, S. M. (2014). "Power Outage Estimation for Tropical Cyclones: Improved Accuracy with Simpler Models." *Risk Analysis*, 34(6), 1069–1078.

Parent, J. R., Meyer, T. H., Volin, J. C., Fahey, R. T., and Witharana, C. (2019). "An analysis of enhanced tree trimming effectiveness on reducing power outages." *Journal of Environmental Management*, 241, 397–406.

Rooney, C. J., Ryan, H. D., Bloniarz, D. V., and Kane, B. (2005). "THE RELIABILITY OF A WINDSHIELD SURVEY TO LOCATE HAZARDS IN ROADSIDE TREES." *Journal of Arboriculture*, 31(2).

Rosenblatt, F. (1962). *Principles of Neurodynamics; Perceptrons and the Theory of Brain Mechanisms*. Spartan Books, Washington.

Salehi, H. and Burgueño, R. (2018). "Emerging artificial intelligence methods in structural engineering." *Engineering Structures*, 171, 170–189.

Shorten, C. and Khoshgoftaar, T. M. (2019). "A survey on Image Data Augmentation for Deep Learning." *Journal of Big Data*, 6(1), 60.

Simonyan, K. and Zisserman, A. (2015). "Very Deep Convolutional Networks for Large-Scale

Image Recognition.” *arXiv:1409.1556 [cs]*.
Simpson, P. and Van Bossuyt, R. (1996). “TREE-CAUSED ELECTRIC OUTAGES.” *Journal of Arboriculture*, 22, 117–121.
Smiley, E. T., Matheny, N., and Lilly, S. (2017). “Best Management Practices - Tree Risk Assessment, Second Edition.” *Report No. P1542*, International Society of Arboriculture.
Spencer, B. F., Hoskere, V., and Narazaki, Y. (2019). “Advances in Computer Vision-Based Civil Infrastructure Inspection and Monitoring.” *Engineering*, 5(2), 199–222.
Su, Z., Chow, J. K., Tan, P. S., Wu, J., Ho, Y. K., and Wang, Y.-H. (2020). “Deep convolutional neural network–based pixel-wise landslide inventory mapping.” *Landslides*.
Szegedy, C., Ioffe, S., Vanhoucke, V., and Alemi, A. (2016). “Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning.” *arXiv:1602.07261 [cs]*.
Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2014). “Going Deeper with Convolutions.” *arXiv:1409.4842 [cs]*.
Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2015). “Rethinking the Inception Architecture for Computer Vision.” *arXiv:1512.00567 [cs]*.
Wang, N., Zhao, X., Wang, L., and Zou, Z. (2019). “Novel System for Rapid Investigation and Damage Detection in Cultural Heritage Conservation Based on Deep Learning.” *Journal of Infrastructure Systems*, 25(3), 04019020.
Wismer, S. (2018). “Targeted Tree Trimming Offers Reliability Benefits.” *T&D World* (May).
Wong, S. C., Gatt, A., Stamatescu, V., and McDonnell, M. D. (2016). “Understanding data augmentation for classification: When to warp?.” *arXiv:1609.08764 [cs]*.
Xie, Y., Ebad Sichani, M., Padgett, J. E., and DesRoches, R. (2020). “The promise of implementing machine learning in earthquake engineering: A state-of-the-art review.” *Earthquake Spectra*, 36(4), 1769–1801.
Zeiler, M. D. and Fergus, R. (2014). “Visualizing and Understanding Convolutional Networks.” *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, eds., Lecture Notes in Computer Science, Cham, Springer International Publishing, 818–833.