Injecting Domain Knowledge into Deep Neural Networks for Tree Crown Delineation

Ira Harmon, Sergio Marconi, Ben Weinstein, Sarah Graves, Daisy Zhe Wang, Alina Zare, Stephanie Bohlman, Aditya Singh, and Ethan White

Abstract

Automated individual tree crown (ITC) delineation plays an important role in forest remote sensing. Accurate ITC delineation benefits biomass estimation, allometry estimation, and species classification among other forest related tasks, all of which are used to monitor forest health and make important decisions in forest management. In this paper, we introduce Neuro-Symbolic DeepForest, a convolutional neural network (CNN) based ITC delineation algorithm that uses a neuro-symbolic framework to inject domain knowledge (represented as rules written in probabilistic soft logic) into a CNN. We create rules that encode concepts for competition, allometry, constrained growth, mean ITC area, and crown color. Our results show that the delineation model learns from the annotated training data as well as the rules and that under some conditions, the injection of rules improves model performance and affects model bias. We then analyze the effects of each rule on its related aspects of model performance.

Index Terms

remote sensing, neuro-symbolics, tree crown delineation, convolutional neural network, forest ecology.

I. Introduction

Remote sensing often uses aerial platforms to efficiently capture data from large geographic regions. Commonly collected data includes hyperspectral imagery covering the visible and near infrared wavelengths, LiDAR data, and synthetic aperture radar imagery [1], [2]. Forest ecologists use remote sensing data to monitor forest health and make predictions of biomass and other forest properties. This is usually done by integrating remote sensing data with allometric relationships [3], [4]. One important metric used in allometric predictions is individual tree crown (ITC) area, which can be estimated from remote sensing based crown delineation [5], [6]. Remote

sensing based crown delineation, particularly for closed canopy forests, remains an open area of research [7]–[9]. The challenges of crown delineation result from the properties of trees as well as limitations in remote sensing technology.

In densely forested areas neighboring tree crowns tend to overlap, may be similar in appearance, and are often multi-storied, making it difficult to tell where one tree ends and another begins. This is further complicated by the unpredictable growth patterns of trees; in short, tree crowns grow in irregular shapes [10]. Adding to this, optical remote sensing technology, which is commonly used for forest remote sensing, can suffer from limited pixel resolution and noise. A typical resolution for visible wavelength optical remotely sensed data is on the order of $1m$ per pixel or less [5]. Therefore, one pixel may contain the edge of one or more tree crowns or may contain the edge of a tree crown as well as foreign objects.

Manual crown delineation is impractical as some forests have stand densities on the order of hundreds of trees per hectare and remote sensing data can cover thousands of hectares thus capturing millions of trees [3]. The problem of ITC delineation can be approached as a two step process: tree detection followed by crown delineation. From a computer vision perspective, both object detection and delineation are well researched, however, the application of each process to forest remote sensing is nuanced. ITC delineation can be considered a specialized case of image segmentation, where the goal is to group image pixels belonging to the same object [11]. Automated approaches to crown delineation can be categorized into three groups by their modality: those based on LiDAR, those based on optical imagery (HSI or RGB), and those based on the fusion of both LiDAR and imagery. These categories can be further differentiated by type of algorithm: non-neural [12], [13] and deep learning based [14], [15]. Non-neural algorithms tend to borrow heavily from image analysis techniques, usually using LiDAR or pixel intensity to find local peaks that are assumed to be crown centers, followed by the application of segmentation algorithms, such as the watershed algorithm, to identify ITCs. Other non-neural algorithms apply simpler statistical machine learning models, such as decision trees to identify crown boundaries. Deep learning based methods, such as DeepForest [12], apply convolutional neural networks (CNNs) to imagery to identify ITCs [8], [16].

In the past, an accurate comparison of algorithm performance was difficult due to lack of benchmark datasets and closed-source algorithms, but recent studies suggest deep learning methods outperform non-neural methods by as much as 54% in precision and 39% in recall [7]. However, deep learning based methods are not without problems. They can require large

amounts of annotated data to train, suffer from dataset bias, and are difficult to tweak in cases where patterns in the dataset are obfuscated by noise or redundancy.

We propose augmenting deep learning crown delineation models with a student-teacher network (STN) capable of incorporating expert knowledge as a set of rules. Using probabilistic soft logic (PSL), ecologists can encode their expert knowledge about a region. Because of the powerful representational abilities of PSL, this knowledge could be anything from average tree heights to leaf color constraints. The only limitation on the encoded rules is that the data associated with those rules must also be represented within the dataset.

The STN is composed of two networks, one that acts as a student and one that acts as a teacher. Using the encoded rules, the teacher-network guides the student-network's training so the student model encodes the rules as well as the information it learns from the dataset. With this technique ITC delineation performance is improved and dataset biases can be induced or removed.

In the remainder of this work we look at related works in section II, describe our data, methodology, and experiments in sections III and IV, and finally discuss our results and conclusions in sections V and VI.

II. Related Work

The availability of aerial imagery along with the impracticality of manual analysis, has spurred forest inventory algorithm research, including algorithms assessing forest structural properties, forest type, and forest biophysical and biochemical properties [17]. Many of these algorithms, and those that perform species classification in particular, require accurate ITC detection and delineation in order to be useful. Non-neural crown delineation algorithms can be classified into 3 categories: valley-following, region-growth, and watershed. One of the earliest approaches to ITC delineation that relied on knowledge formulated as a distinct set of rules within the framework of a larger algorithm was proposed by Gougeon et al. in 1995 [18], [19]. Their valley-following algorithm worked on the premise that there is variation in shading between canopies – in short a gap. A set of rules was followed to trace out the pixel intensity valleys between canopies and thus segment crowns. Gougen's rules primarily use low level visual concepts and do not directly incorporate any high level ecological abstractions. Despite its simplicity, the valley-following algorithm had a reported accuracy of 81% when used with resolutions of 0.3m per pixel or better. As noted by the paper's author, the algorithm's greatest shortcoming was its underlying

assumption that there are bands of shade between crowns. This limits the algorithm to low to moderate density forest stands.

Valley-following algorithms were superseded by region-growing algorithms, which are commonly used in image segmentation [20]. Region-growing algorithms start with a seed pixel whose properties are assumed to be prototypical of the region. Starting from the seed pixel, neighboring pixels are examined in succession and added to the region if they are similar to the seed. The process ends when boundary pixels reach a dissimilarity threshold. Region growing algorithms were first used for image segmentation in the 1990's and algorithms such as Culvenor's TIDA, became popular for ITC delineation in the early 2000's [21]. Erickson created a region-growth algorithm that incorporates fuzzy thresholding and a set of rules to guide region growing, however, like the valley-following algorithm, the rules do not incorporate high level ecological concepts [20]. Erickson's algorithm achieved an overall accuracy of 73%. A common problem with these algorithms comes from their method of choosing seed points. TIDA and similar algorithms use local maxima in pixel intensity to choose seeds. These points are assumed to be crown centers, however this may not always be the case [22].

ITC delineation algorithms based on variations of the watershed algorithm also became increasingly researched in the early 2000's [23]. Starting from a gray-scale image, the watershed algorithm treats pixel intensities as if they were elevations on a topographical map. Following this analogy, if the object were slowly submerged in water, the water would pool in regions of low pixel intensity first. As the object is further submerged the pools of water collected would begin to merge. The boundaries marked by the edge of each pool can be considered a segment. Wang et al. used the watershed algorithm to segment aerial imagery into ITCs [24]. Wang's algorithm had an accuracy of 75.6%. Watershed segmentation ITC algorithms are subject to errors caused by inconsistencies between gray-scale boundaries and tree crown boundaries. Error can also be introduced into the algorithm when treetops deviate from the crown center. This is common when tree growth is not vertical.

More recent approaches to ITC delineation often combine RGB or hyperspectral imaging with point cloud data generated via LiDAR. Many of these algorithms, such as Dalponte's itcSegment, incorporate machine learning models [25]. Sackov et al. developed a purely LiDAR based algorithm for ITC delineation using what they called a point-based approach [26]. Their method uses allometric rules to improve the likelihood of crown detection. Their results showed that their algorithm was reliable only under specific conditions. Kientz et al. give a comparison

of segmentation based ITC delineation algorithms in [27].

Though CNNs were developed in the 1990's they have only recently been applied to crown delineation [28]. Weinstein et al. show CNNs are a viable method for ITC delineation on a range of forest types using a RetinaNet based model in their work [7], [29]. Braga et al. use Mask-RCNN for crown delineation [8]. Though both models are similar, Mask-RCNN segments detected objects allowing for the creation of irregularly shaped boundaries [30]. Weinstein's model, tested on various forest types, averages a F1 of 68%. Braga's model has a global accuracy of 91% but has more limited testing.

Neuro-symbolics is the branch of artificial intelligence concerned with bridging the gap between learning and reasoning [31]. Connectionist machine learning models, such as neural networks, excel at learning from experience but lack the ability to reason (in a way that's understandable to humans) about what they learned. On the other hand, symbolic representations of knowledge, such as those that incorporate propositional or first order logic (FOL) are useful for reasoning and readily understood by humans, but ill fitted for learning. In learning applications, rule based models generally perform poorly compared to connectionist models. The goal of neuro-symbolics is to combine learning with reasoning ability, usually representing domain knowledge as a set of rules, a knowledge base, or a knowledge graph that allows a connectionist model to benefit from the symbolic representation of domain knowledge that is related to the dataset.

Several recent works have applied neuro-symbolics to vision intensive ecological tasks, such as the use of fine grained image classification (FGIC) for species classification in plants and animals [32]–[34]. Xu et al. use a two level object detection model based on R-CNN for FGIC of bird species [32]. The model uses species specific domain knowledge from a knowledge base and relevant text to boost CNN performance by projecting the CNN predictions into the same semantic embedding space as the knowledge base and text. Using this approach their model can reason about which species is most likely in an image by measuring the similarity between the projected CNN prediction and the embedded species descriptions from the knowledge base and text. Xu et al.'s model was able to surpass the state of the art on the Cal-tech UCSD Bird dataset [35].

Sumbul et al. use a similar approach for FGIC of tree species classification from remote sensing imagery [34]. They add to the difficulty by testing their model in a zero shot learning scenario. Their model consists of a combination of CNN and 3 separate sources of domain knowledge:

manually annotated attributes, relevant text corpora, and a hierarchical representation of the species taxonomy. The CNN is used to create visual embeddings of images. The 3 auxiliary data sources are also embedded or encoded before further processing. The model feeds the visual and auxiliary data embeddings into a learned bilinear compatibility function that returns a scalar value representing how closely the image matches a particular class. The results for each class are fed into a softmax function to make a prediction. Using their model, Sumbul et al. achieved an average accuracy of 14%, which is reasonable for zero shot learning.

CNN based models have several drawbacks. They require datasets on the order of thousands of images or more to get reasonable results. Training CNNs is also computationally intensive, requiring thousands of giga-flops to train [36]. The combination of large datasets and computational intensiveness makes working with CNN based models impractical without access to high powered computing hardware such as GPUs or TPUs.

Our approach to neuro-symbolics uses CNNs and high level ecological concepts encoded as a rule to improve ITC delineation performance. To the best of our knowledge we are the first to apply neuro-symbolics to ITC delineation.

III. Data and Methodology

Though our methods can theoretically be applied to any machine learning RGB ITC delineation model, our work is based on DeepForest. DeepForest is built around RetinaNet, a CNN with 31.9M trainable parameters, used to drive crown detection [29]. Like all deep CNNs, DeepForest requires thousands of annotated images to train. DeepForest was trained using data from the National Ecological Observatory Network (NEON), which annually collects remote sensing data from 81 sites across the continental United States, Alaska, and Puerto Rico. LiDAR, RGB, and hyperspectral imagery data are categorized by site and year, and are available for download from the NEON website at no charge [37]. Weinstein et al. used 2018 NEON data products to train DeepForest and create a benchmark dataset for ITC delineation [38]. The dataset consists of annotated images divided into training and evaluation sets and further divided by site. Annotations in the evaluation dataset are verified against field measured crowns when possible. To reduce the burden of manually annotating large numbers of images, DeepForest uses a three step training process consisting of self-supervised and supervised algorithms to generate a training set. The backbone of the ITC delineation model is pre-trained on ImageNet. Then, a large weakly labeled training set from a variety of forest-types is generated based on an unsupervised crown delineation

algorithm to create a set of transferable weights. Using transfer learning [39], these generalized weights are used as starting points to retrain the model on smaller more region specific datasets based on human labeled training data.

A. Knowledge Distillation and Posterior Regularization

A STN is an architecture that uses one or more teacher networks to guide the training of a second network, the student [40]. This technique was proposed by Bucilua et al. as a means of compressing a large or complex model into a smaller model [41]. It was later modified and popularized by Hinton et al. to transfer knowledge from one model to another in a process he named knowledge distillation [42]. In Gou's taxonomy of knowledge distillation strategies, Hinton's method falls into the response-based knowledge distillation category where knowledge is transferred through the response of the output layer of the teacher model [40].

Let the training set, $\mathcal{D}$, consist of a set of points, X, where $X \in \{x_1, x_2, ..., x_n\}$ and a set of labels, Y, where $Y \in \{y_1, y_2, ..., y_n\}$. Each point $x_i \in X$ is paired with its label $y_i \in Y$ to create $\mathcal{D}$, where $\mathcal{D} = \{(x_1, x_2), (x_2, y_2), ..., (x_n, y_n)\}$. In this example we focus on classification, so let y be a 1-hot encoding such that $y \in \{0, 1\}^k$ and k is the number of classes. Using response-based knowledge distillation, the student-network learns from both the training data and the teacher-network through its loss function, which is a function of the teacher output, student output, and ground-truth labels or prior distributions. In Hinton's parlance, after the the logits from the output layer of the teacher and student network are passed through a soft-max layer, they are called soft-targets, z. Fig. 1 shows a generic student teacher network that uses response-based knowledge distillation. z_s and z_t are the soft-targets from the student and teacher networks respectively. The loss function for the entire network is composed of two losses, the student-loss and the distillation-loss. The student-loss measures the student-network's inference skill relative to ground-truth labels. The distillation-loss measures the student network's skill at imitating the predictions of the teacher-network. Basing our example on a classification model acting as the student in a STN, we can use cross-entropy loss to measure the student's inference skill relative to ground-truth as shown in (1).

$$L_S = \mathcal{L}_{CE}(y, z_s) \tag{1}$$

The distillation loss is a function of z_t and z_s as shown in (2). Treating the soft-targets as distributions, a divergence measure such as Kullback-Leibler is commonly used as the distillation

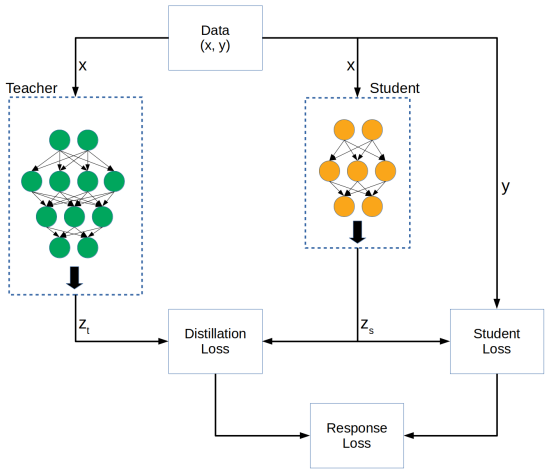


Fig. 1. A generic response-based knowledge distillation student-teacher network and its loss functions.

loss function [40].

$$L_D = \mathcal{L}(z_t, z_s) \tag{2}$$

Thus, the response-loss is a function of L_D and L_S, $\mathcal{L}(L_S, L_D)$, commonly a linear combination of the two. Models trained using knowledge distillation are shown to outperform the same model trained on a dataset alone [42].

Prior to Hinton's work Ganchev et al. produced a similar result using posterior regularization [43]. Posterior regularization (PR) is similar to the STN concept. The loss function of a model trained with PR includes a regularization term that essentially functions as the distillation loss. To align Ganchev's work with Gou's terminology, let p_θ be the student model with parameters θ. The divergence between an optimal distribution, q^*, and the model's posterior distribution, $p_\theta(y|x)$, are used as arguments in what is equivalent to a distillation-loss term. Distribution q^* is then equivalent to z_t in that knowledge is passed from the teacher to the student through q^*. As originally formulated in [43], distributions Q are constrained based on prior knowledge:

$$Q = \{q(Y) : E_q[\phi(X, Y)] \leq b\}. \tag{3}$$

In (3) Q is the set of distributions that satisfy a constraint. In this case, Q contains distributions where the expected value of the constraint feature, ϕ, is bounded by b. In [43] the loss function for model p_θ is given as

$$\mathcal{L}(\theta) = log(p_\theta(Y|X)) + log(p(\theta)), \tag{4}$$

where the first term is the log-loss of the model's posterior and the second term is a prior on the model's parameters. Using Gou's terminology, $\mathcal{L}(\theta)$ is the student-loss. Finally, Ganchev creates the posterior regularized likelihood as

$$J_Q(\theta) = \mathcal{L}(\theta) - KL(Q \| p_\theta(Y|X)) \tag{5}$$

where

$$KL(Q \| p_\theta(Y|X)) = \min_{q \in Q} KL(q(Y) \| p_\theta(Y|X). \tag{6}$$

The divergence term penalizes p_θ for failing to match distribution q, making the divergence term equivalent to the distillation loss and J_Q the response loss. To train p_θ, J_Q is optimized with respect to the bounded expected value of the constraint features. Ganchev shows that the primal has closed form solution

$$q^* = \frac{p_\theta(Y|X) \cdot exp\{-\lambda^* \cdot \phi(X,Y)\}}{Z(\lambda^*)} \tag{7}$$

where q^* is the optimal distribution from set Q, λ^* is a real number ≥ 0, and

$$Z(\lambda^*) = \sum_Y p_\theta(Y|X) \cdot exp\{-\lambda^* \cdot \phi(X,Y)\} \tag{8}$$

serves to normalize q^*. See [43] for proof. Equation (7) can be geometrically interpreted as the projection of $p_\theta(Y|X)$ into a subspace constrained by $\phi(X,Y)$ [43].

We have shown that elements of a response-based knowledge distillation STN are present in PR. The next step is to build a framework to formalize the symbolic expression of domain knowledge as a set of constraints and a means to inject that knowledge into neural models. These concepts were incorporated into a single machine learning paradigm by Hu et al. [44]. Hu's work builds on Ganchev's by encoding constraints, $\phi(X,Y)$, in PSL and augmenting a student-loss function with a distillation-loss term to transfer knowledge from a teacher-network.

PSL is framework for probabilistic reasoning built on FOL and fuzzy logic [45]. See section III-B for more information. As in PR, [44] uses the projection of the student-network into a rule regularized subspace to obtain z_t. Because Hu's framework can be applied to any neural model and the domain knowledge is encoded in PSL, Hu uses the term logic harness rather than student-teacher network. Using Hu's framework, we modified DeepForest to have the architecture shown in Fig 2. DeepForest is built around RetinaNet [29]. As shown in Fig. 5 RetinaNet has classification and regression heads. The classification heads are used to predict object class while

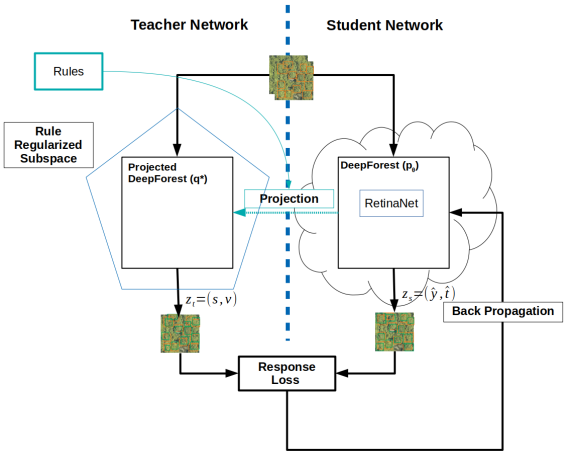


Fig. 2. Network architecture. The student-network (right side of the blue dotted line) is trained using annotated RGB images. The annotations are shown in orange and predictions in dark green. The student-network is projected into a rule-regularized subspace (blue arrow) to create the teacher-network (left side of the dashed blue line). The student-network is penalized for failing to imitate the predictions of the teacher-network as well as incorrect inferences relative to ground-truth. The combined loss is used to update the student-network through backpropagation.

regression heads are used to predict object location. As such, its loss function is composed of two parts, one representing the classification loss and a second representing bounding box regression loss. We call the loss from the classification heads L_{cln} and the loss from the regression heads L_{reg}. We use the unmodified loss from RetinaNet as the student-loss and implement PR by adding a distillation-loss term [44]. Because the student network generates two predictions, a class and location, we modify our notation. We introduce location vectors. Location vectors are 4-vectors, with elements representing the upper-left corner and the lower-right corner of a bounding box:

$$\vec{t} = \langle x_{min}, y_{min}, x_{max}, y_{max} \rangle. \tag{9}$$

We now represent the soft-targets, z_s (10) and z_t (11), as tuples composed of a class prediction and a location vector. The class prediction is as a k-dimensional probability simplex and the location vector is formatted as previously described.

$$z_s = (\hat{y}, \hat{t}) \tag{10}$$

$$z_t = (s, v) \tag{11}$$

The response-loss is shown in (12) where N is the number of instances in a batch and π is a function of step number with range $[0, 1]$. The parameter π is used to linearly blend the student-

loss with the distillation-loss and set how strongly the student-network imitates the teacher. Pi is termed the imitation parameter in [44].

$$L_R = \frac{1}{N}\sum_{i=1}^{N}(1-\pi)\cdot\Big(L_{cln}(y_i,\hat{y_i}) + L_{reg}(t_i,\hat{t_i})\Big) + \pi\cdot L_D(z_t,z_s) \tag{12}$$

$$L_D = L_{cln}(s,\hat{y}) + L_{reg}(v,\hat{t}) \tag{13}$$

For classification, L_{cln} is cross-entropy loss, in the case of DeepForest, which only has two classes, binary cross entropy loss. For bounding box regression, L1 loss is used. As in [43], the student-network is projected to create the teacher-network. For constraints that only affect classification the teacher-network uses the same location predictions as the student-network, $v = \hat{t}$. Similarly, for constraints that only affect bounding box predictions, the teacher-network uses the same classification predictions as the student-network, $s = \hat{y}$. Hu modifies (7) to work with constraints written in PSL. The framework allows for the incorporation of multiple rules. In (14) constraint function $\phi(X,Y)$ is replaced with a set of rules written in PSL, but the projection paradigm is the same. See [44] for a proof.

$$q^* \propto p_\theta(Y|X)\cdot exp\Big\{-\sum_{l\in L, g_l\in G_l} C\cdot\lambda_l\cdot(1-r_{l,gl}(X,Y))\Big\} \tag{14}$$

In (14) q^* is projection of $p_\theta(Y|X)$ into a subspace constrained by the rules, where L is the set of rules and G_l is the set of groundings for each rule. Each rule is paired with a constant, λ, where

$$R = \{(R_l,\lambda_l)\}\ where\ \lambda_l \in [0,\infty). \tag{15}$$

λ represents the subjective confidence of the rule. C is a regularization parameter. In Hu's framework, q^* is passed through a soft-max function prior to use as the prediction of the teacher-network.

The resulting network is trained with standard optimization algorithms such as stochastic gradient descent [46]. We develop the objective function, (16), from (12). In (16) τ is the iteration number and θ the student-network parameters. At the start of a training iteration, the network weights, rules, π, and C are initialized. One batch of the training data is sampled. The training data is forward-propagated through the student-network. Using (14) the teacher predictions are

computed. Using Equation (16) the error is back-propagated through the student-network and θ is updated. The process is then repeated. The algorithm is summarized below.

$$\theta^{(\tau+1)} = arg\min_{\theta\in\Theta} \frac{1}{N}\sum_{i=1}^{N}(1-\pi)\cdot\Big(L_{cln}(y_i,\hat{y_i}) + L_{reg}(t_i,\hat{t_i})\Big) + \pi\cdot L_D(z_t^{\tau}, z_s) \quad (16)$$

Algorithm 1 Logic Harness Training

INPUT:

D, dataset of images ($\mathbf{X}$) and annotations ($\mathbf{Y}$)

Rule set $R = \{(R_1,\lambda_1),(R_2,\lambda_2),...,(R_L,\lambda_L)\}$

Parameters: π, C, numEpochs

OUTPUT:

Trained network, p_θ

METHOD:

Initialize weights, θ

for 1 to numEpochs:

for batch $(\mathbf{X},\mathbf{Y}) \subset \mathbf{D}$

$z_s \leftarrow p_\theta(X)$

$z_t \leftarrow Eqn.(14)$

$\theta^{(\tau+1)} \leftarrow Eqn.(16)$

return p_θ

B. Probabilistic Soft Logic

Rules are written using a relaxed version of probabilistic soft logic (PSL), a combination of FOL and Lukasiewicz logic. FOL is a powerful and highly expressive framework where logical expressions are composed of constants, variables, predicates, and functions linked by logical operators “and”, “or”, “if”, and “not” [47], [48]. Predicates take a set of arguments as input and return a value of true or false. When soft-logical operations are used they return a value between 0 and 1 [49]. Functions accept arguments and return a constant of any kind. When used with soft-logic function range is also limited to [0,1]. An atom is a predicate and its arguments. A literal is an atom or a negated atom. A PSL rule can be weighted or unweighted. In our

framework the weight of a rule is represented by the rule's λ. Each rule has a precondition and post condition, referred to as the body and head, respectively. In strict PSL a rule's body consists of a conjunction of literals and the head consists of a single literal or a disjunction of literals. Being continuous, these expressions can be evaluated algebraically. Examples are shown in equations (17) - (20).

$$A \ \& \ B = \max(A + B - 1, 0) \tag{17}$$

$$A_1 \wedge A_2 \wedge ... \wedge A_N = \sum \frac{A_i}{N} \tag{18}$$

$$A \vee B = \min(A + B, 1) \tag{19}$$

$$\neg A = 1 - A \tag{20}$$

PSL also allows for rules written in terms of arithmetic, using arithmetic operators. Arithmetic rules constrain linear combinations of atoms with an equality or an inequality. Kimmig et al. provide a good introduction to PSL [45]. We motivate an example of a simple set of rules written in PSL with a potential use case of logic-harness augmented DeepForest. Assume that we have a large image of trees in profile and we want to write rules to detect trees, by examining rectangular subsections of the image. Given the two rules,

$$0.3: \ hasLeaves(X) \wedge isWoody(X) \implies isTree(X) \tag{21}$$

$$0.8: \ hasLeaves(X) \wedge isWoody(X) \wedge isTall(X) \implies isTree(X), \tag{22}$$

let X be a variable representing a subsection of the image. The rule predicates are $hasLeaves$, $isWoody$, $isTall$, and $isTree$. The atoms to the left of the arrows represent the body of the rule and the atom to the right, the head of the rule. Expressed in natural language rule 1 reads, "if x has leaves and is woody then there is a chance x is a tree". Rule 2 reads, "if x has leaves and is woody and tall then there is a chance that x is a tree". The weights, 0.3 and 0.8, signify that the second rule is more likely to be true than the first. A rule is evaluated by mapping each atom to a soft truth value. This is called an interpretation, I. A rule, r, is satisfied if $I(r_{body}) \leq I(r_{head})$. Each rule can be expressed algebraically by evaluating the atoms and substituting the equations for the operators listed in (17) - (20). Using (24), the first rule is expressed algebraically as

$$\min\{1 - \frac{hasLeaves(X) + isWoody(X)}{2} + isTree(X), 1\}. \tag{23}$$

$$A \implies B \equiv \neg A \vee B \tag{24}$$

Encoded rules can be evaluated by simplifying expressions and transforming them into their algebraic equivalents.

C. Data

We used data from 4 NEON sites: Niwot Ridge (NIWO), Teakettle Experimental Forest (TEAK), San Joaquin Experimental Range (SJER), and Mountain Lake Biological Station (MLBS). NIWO is an alpine forest in the Rocky Mountains of Niwot, Colorado, located at 40°03'N latitude and 105°36'W longitude [50], [51]. Its elevation is between 3,000-3,500 m. The mean temperature is 1.5°C and its annual precipitation is 800 mm per year. The primary tree species are lodgepole pine, subalpine fir, and Englemann spruce. The mean canopy height is 0.2 m [51] (all canopy height values listed here include bare ground and the prevalence of bare ground at NIWO is why this value is so small). The tree density is 1,726 stems per hectare. The NIWO dataset, the largest of all datasets, contained 10,757 training annotations, 1,655 validation annotations, and 1,624 testing annotations. The average ITC area for the NIWO training set was 4.12 m^2.

TEAK is a coniferous forest located in California's Sierra National Forest at 36°58' N latitude and 119°1' W longitude [52], [53]. Its elevation is between 1,900 and 2,300 m and the mean annual temperature is 8°C. It receives an average of 1,222.5 mm precipitation per year. The primary vegetation consists of Jeffrey pine, red fir, white fir, and lodgepole pine. The mean canopy height is 35 m [52]. The TEAK dataset contained 4,514 training annotations, 885 validation annotations, and 734 testing annotations. The average training set ITC area was 24.8 m^2.

SJER is an oak savanna located at the foot of the Sierra Nevada Mountains in California at 37°5'45" N latitude and 119°43'45" W longitude [47], [48] [54], [55]. Its elevation is between 213 - 518 m and the mean annual temperature is 16.4°C. The mean annual precipitation is 539.6 mm. The predominant vegetation is blue oak, interior live oak, and foothill pine. The mean canopy height is 21 m [54]. The SJER dataset contained 2,824 training annotations, 223 validation annotations, and 255 testing annotations. The average training set ITC area was 59.9 m^2.

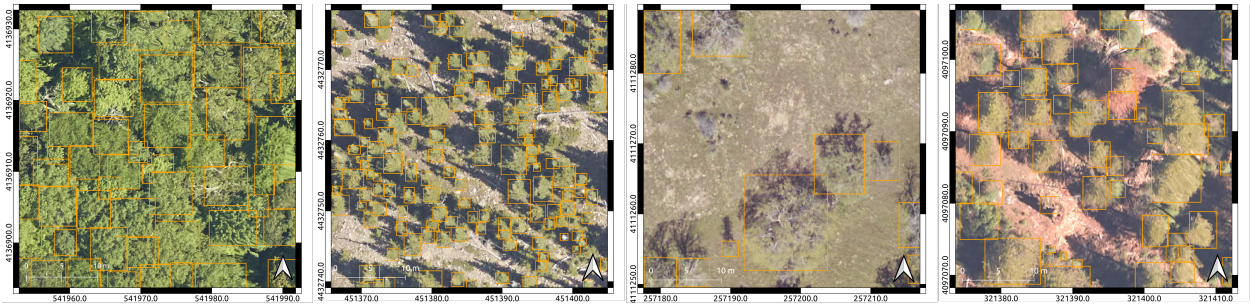


Fig. 3. Example training images of sites from left to right: MLBS , NIWO, SJER, TEAK. Ground truth annotations are shown in orange.

MLBS is a deciduous forest in the Appalachia Mountains of Virginia located at 37°22' 42" N latitude and 80°31' 29.4492" W longitude [56]. Its elevation is 1,170 - 1,320 m and the mean annual temperature is 8.8°C. The mean annual precipitation is 1,227mm. The primary tree species are red maple and white oak. The mean canopy height is 18 m [56]. The MLBS dataset contained 2,349 training annotations, 372 validation annotations, and 481 testing annotations. The average training set ITC area was 24.0 m^2.

The data was collected via NEON's airborne observation platform (AOP), an array of remote sensing instruments installed on a light aircraft. The instruments include a discrete and full-waveform LiDAR, a digital camera, an imaging spectrometer, a GPS antenna, and an inertial measurement unit (IMU) [57]. Data is collected from an above ground altitude of 1000 m. The flights occur annually over all NEON sites during each forests' peak greenness and occur in conjunction with field surveys.

The RGB imagery has a resolution of 0.1 m per pixel. The resolution of the LiDAR point cloud data varies, but averages 2 - 8 points per square meter [58]. The lidar point clouds are used to generate canopy height model (CHM) rasters with a resolution of 1 m^2 per pixel. For our experiments we used NEON's L3 RGB and CHM data products which are 1 km^2 mosaiced tiles stored separately in a co-registered geotiff format.

For training, model rasters larger than 500 pixels in either dimension were decomposed into multiple rasters of 400 x 400 square pixels or less with 5% overlap. The train-test-split used followed that used by [38], however for rule 3, we only used RGB rasters that also had a corresponding canopy height model raster available. The validation datasets were created to be similar in size to the test set. More information on the creation of the dataset can be found

in [38]. The annotation count and ITC area distributions for each dataset are summarized in Table I and Fig. 4 respectively. Examples of images from each site are shown in Fig. 3. The ground-truth tree crown annotations are shown in orange.

TABLE I

THE NUMBER OF ANNOTATED TREES COMPOSING THE TRAINING, VALIDATION, AND TEST SETS BY SITE.

Site	Training Annotations	Validation Annotations	Testing Annotations
NIWO	10,757	1,655	1,624
TEAK	4,514	885	734
SJER	2,824	223	254
MLBS	2,349	372	481

To support the incorporation of the ecology of competition, for each annotated tree in the training set we measure the number of other annotated trees that touch or intersect its bounding box. We use this metric as a measure of competition at each site. Tree functional traits are highly dependent on species, but also influenced by environment. During growth, there is a trade-off between vertical growth and horizontal growth. Dense stands limit crown expansion due to mechanical interaction with neighboring crowns, while encouraging vertical growth. On the other hand, lateral space promotes horizontal expansion: open-grown trees are shorter with larger crowns throughout their life [59], [60]. Table II shows the number of trees in competition at each site. MLBS has the most trees in competition with almost 100% of its population, followed by NIWO, TEAK, and SJER. Table II also summarizes the percentage of trees that support the assertion that competing trees have smaller than average crowns while non-competing have larger than average crowns.

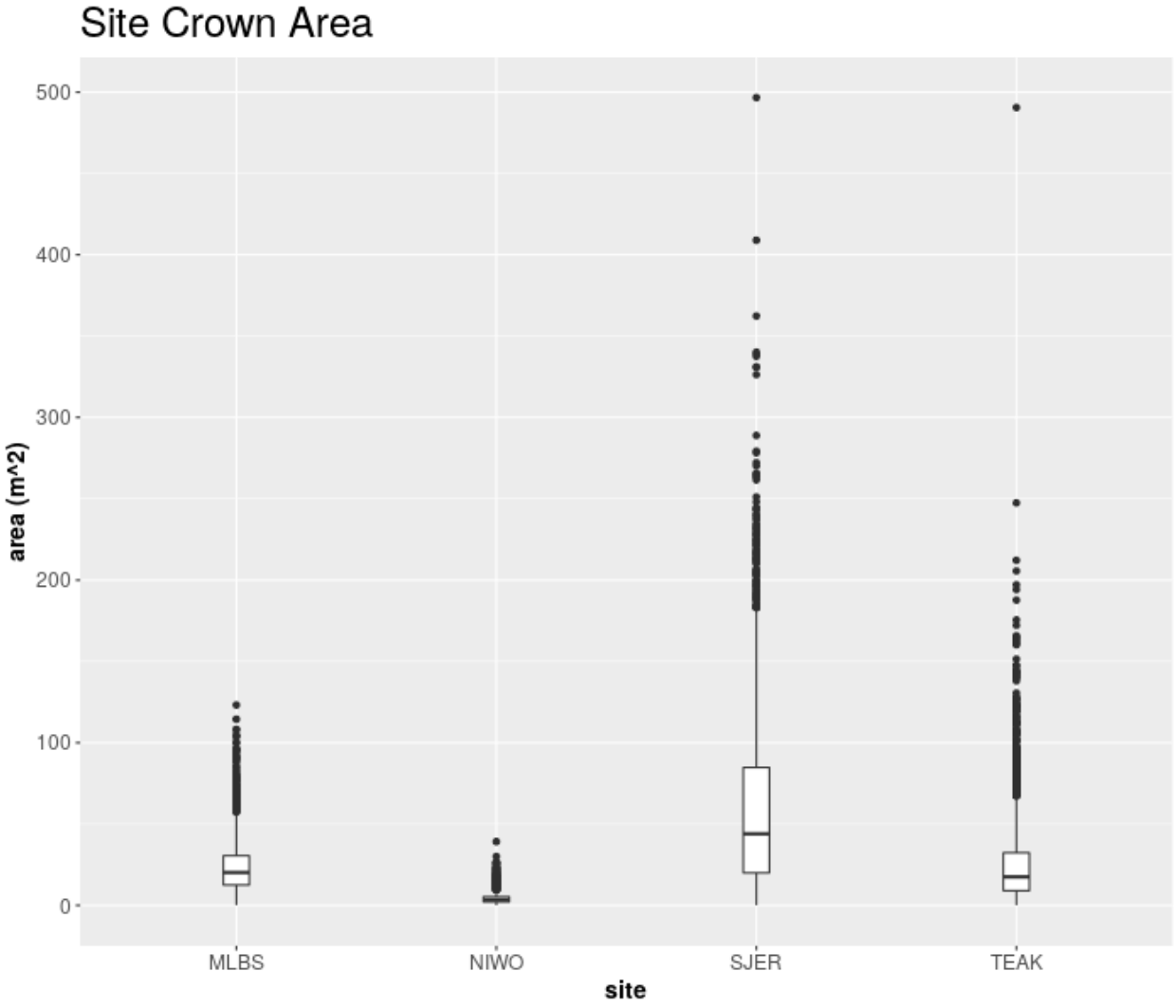


Fig. 4. Box plot of the crown area distribution for each site.

D. Model

Deep Forest is built around Retinanet [29]. Retinet is a deep convolutional neural network (CNN) for object detection. The network architecture is shown in Fig 5. The network consists of a backbone, feature pyramid network (FPN) [61], and classification and regression heads. The backbone is built from ResNet-50 using ImageNet pre-trained weights [62]. The backbone provides the primary feature extraction. The FPN provides scale invariant feature detection. The classification heads classify candidate objects. The classification heads use a version of cross entropy loss. The regression heads localize candidate objects by generating bounding box coordinates. The regression heads use F1 loss.

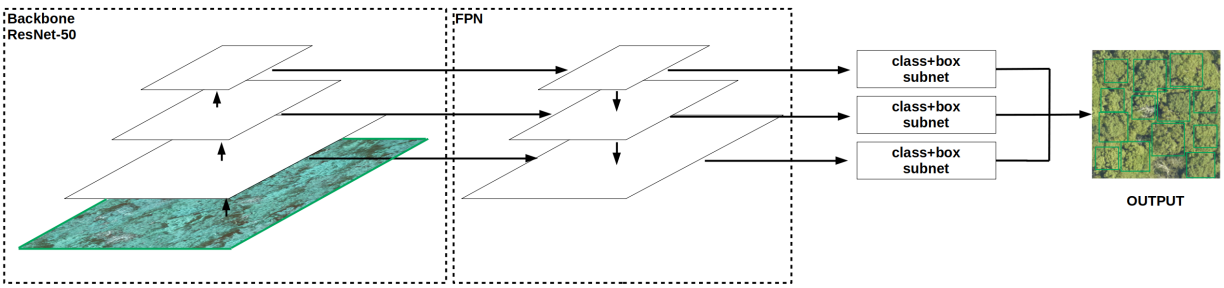


Fig. 5. RetinaNet architecture. The ResNet backbone generates feature maps that are combined at different scales in the feature pyramid network. Classification and regression heads perform detection and localization respectively.

Object detection networks come in two flavors, single stage and two-stage [29], [63]. Both use the concept of anchor boxes [61]. Anchor boxes are a predefined set of bounding boxes of a particular size and aspect ratio. Two stage networks use a region proposal network (RPN) to reduce the number of candidate objects sent to classification and regression heads and are thus bounding box sparse. Single stage networks lack a RPN and thus send thousands of times more candidate objects to the classification and regression heads and are anchor box dense. RetinaNet can produce 100K anchor boxes per feature map or more.

Single stage networks tend to have faster inference, while 2 stage networks have more accurate classification and localization [29]. RetinaNet's innovation was its ability to outperform two stage models by employing focal loss. The poor performance of single stage object detection models stemmed from the class imbalance between foreground and background candidate objects with background candidate objects being in the majority. Focal loss reduces the loss contribution from background objects by reducing the loss contribution from high confidence classifications like background and increasing the loss contribution from low confidence foreground objects that the model is intended to detect. With Focal-loss, RetinaNet outperforms two-stage models. Its fast inference and high performance make it a popular choice for remote sensing applications.

The prediction process for RetinaNet occurs as follows: The convolutions are performed by the ResNet backbone to produce feature maps at different scales. These are passed to the FPN. The anchor boxes associated with each candidate object are in 3 aspect ratios, {1:2, 1:1, 2:1} and each of those aspect ratios are at 3 scales $\{2^0, 2^{\frac{1}{3}}, 2^{\frac{2}{3}}\}$.Only the top 1K predictions per FPN level are passed to classification and regression heads. These top predictions are merged and non-maximal suppression (NMS) is performed, removing any candidate objects with a class

probability of less than 0.5. Each classification is assigned to 1 ground-truth object if it has an intersection over union (IoU) > 0.5. If the IoU is between 0 and 0.4 the candidate object is assigned to the background class.

E. Metrics

The IoU, also known as the Jaccard Index, is used to measure the overlap between predicted and ground-truth bounding boxes. Given two bounding boxes A and B their IoU is given by

$$IoU = \frac{A_{area} \cap B_{area}}{A_{area} \cup B_{area}}, \tag{25}$$

where A_{area} and B_{area} are the areas of box A and B respectively.

The primary metrics used to measure model performance are precision, recall, and macro-F1. When a predicted bounding box has an IoU ≥ 0.5 with a ground-truth bounding box it is considered a true positive. If the IoU is less than 0.5 or it does not intersect with a ground-truth bounding box it is considered a false positive. A ground-truth bounding box without a matching prediction is a false negative.

Precision is defined as

$$Precision = \frac{TP}{TP + FP}, \tag{26}$$

where TP is the number of true positives, and FP is the number of false positives. Recall is

$$Recall = \frac{TP}{TP + FN} \tag{27}$$

where TP is the same as previously defined and FN is the number of false negatives. The equation used for macro-F1 is

$$F_1 = \frac{2 \cdot Pr \cdot Rec}{Pr + Rec} \tag{28}$$

where Pr is the precision, as defined in (26) and Rec is recall as defined in (27).

The root mean square error (RMSE) is used to measure the cumulative error between predicted values and the ground-truth values. The RMSE is given by

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^{N} \left(y_i - \hat{y}_i\right)^2}, \tag{29}$$

where y is the ground-truth value and $\hat{y}$ is the predicted value. The mean absolute error measures the same concept, but is less sensitive to outliers than the RMSE. The MAE is given by the equation

$$\frac{1}{N} \sum_{i=1}^{N} |y_i - \hat{y}_i|. \tag{30}$$

Both RMSE and MAE are used to measure the cumulative difference between the predicted and ground-truth ITC areas.

The Kullback-Leibler Divergence (KL-Divergence) is used to measure the difference between two probability distributions. The output is a scalar value given by the equation

$$D_{KL}(P\|Q) = \sum_{x \in X} P(x) log(\frac{P(x)}{Q(x)}) \tag{31}$$

where $P(X)$ and $Q(X)$ are probability distributions. KL-divergence is not a true metric as it is not commutative. We use it to measure the difference between predicted and actual ITC area distributions.

To measure rule enforcement we use the verification ratio as in [64]. Given a set of predictions, $\hat{Y}$, let $\hat{Y}_R$ be the subset of $\hat{Y}$ that satisfies rule R. Then the verification ratio is defined as

$$\text{ver. ratio} = \frac{|\hat{Y_R}|}{|\hat{Y}|}. \tag{32}$$

IV. Experiments

We develop 4 rules to demonstrate a variety of approaches to the injection of domain knowledge into a crown delineation CNN. Rules 1 - 3 are designed to improve performance, while rule 4 is designed to alter model behavior without changing the dataset at the expense of performance. This allows the model to be trained on the same dataset, but accomplish two different tasks. Rules can be further subdivided based on the strength of their grounding in ecological domain knowledge. Rules 1 and 4 have the weakest ecological basis and are more oriented towards correcting or inducing bias in the model, whereas rules 2 and 3 have a stronger ecological foundation. Rule 2 is designed to only operate in some contexts. Furthermore, rules 1, 3, and 4 feed into the model through the classification heads. Rule 2 feeds into the model using the regression heads.

Much of ecology domain knowledge is site and species specific. Concepts that apply to one site may need modification to be applied to another site or may not be applicable at all. There is a great body of research on site and species specific allometry [59], [60], [65], [66]. Of particular interest to this work is crown allometry. Stem diameter breast height (DBH) and stem height are frequently studied in relation to crown area. These studies include modeling the crown area distribution as well as fitting log-linear models for height crown area allometry. For the 4 sites in this work, despite the canopies being composed of multiple species, the data does not include species identities, therefore the rules applied are not species specific.

A. Rule 1

For rule 1 we measure the mean area of the annotated crowns in the training set and use this knowledge to improve test set performance. As shown in Fig. 4 crown area varies significantly among sites. For example, the average crown area for trees at NIWO is an order of magnitude less than the average crown area for trees at SJER. Since tree size distributions are highly right-skewed most trees at a site will be smaller than the mean. In addition, in some cases DeepForest produces size measurements that are somewhat larger than field measurements [67]. Therefore, we use the following rule to incorporate mean ITC area into our model while acknowledging the shape of the size distribution and potential overestimates of crown size by the model:

For all predictions, if the ITC area of a predicted tree is less than the site's average ITC area, it is probably a tree.

To encode this rule into PSL we create a function of ITC area whose range is (0, 1). We use the sigmoid function as a soft unit step function flipped about the y-axis. We set the site average crown area at the inflection point thus $p_\theta(y|x)$ for crowns that are greater than average are reduced. We call this function $bboxSize$. The equation for the function is

$$bboxSize(A) = \frac{1}{1 + e^{-k_{sig} \cdot (\mu_{area} - A)}} \tag{33}$$

where k_{sig} is a constant, μ_{area} is the mean ITC area for the site, and "A" is the area of the bounding box. We formulate Rule 1 in relaxed PSL as follows

$$\forall \hat{y} \Big(\mathbf{1}(y = +) \implies bboxSize(A_{bbox}(\hat{t}))_+ \\ \wedge bboxSize(A_{bbox}(\hat{t}))_+ \implies \mathbf{1}(y = +) \Big). \tag{34}$$

Using the notation from [44], $\mathbf{1}$ is an indicator function that returns 1 when its argument is true and 0 otherwise. The "+" represents the positive class,

$$bboxSize(a)_+ = bboxSize(a), \tag{35}$$

and

$$bboxSize(a)_- = 1 - bboxSize(a), \tag{36}$$

$A_{bbox}()$ is a function that takes a predicted bounding box vector and returns its area in pixels. Replacing "$\wedge$" with the equation for "&" given in 17, the PSL for rule 1 simplifies to $bboxSize(A_{bbox}(\hat{t}))$ when $y = +$ and $1 - bboxSize(A_{bbox}(\hat{t}))$ otherwise.

B. Rule 2

Rule 2 addresses the challenge of excessively large trees more directly by decreasing the dimensions of the predicted bounding box for ITCs that are in the extreme tail of the empirical size distribution. It does this while accounting for an additional piece of ecological knowledge, which is that trees with very large canopy areas may occur naturally when there is no competition from surrounding trees, and inversely, trees are likely to have a smaller canopy areas when there is competition from surrounding trees. We do this by applying the domain knowledge as a PSL statement, but we wanted to see if we could inject this knowledge through the regression heads rather than the classification heads like rules 1, 3, and 4. We implement the output of the rule as a vector and deviate from strict PSL by changing the range of the predicate to the reals. Changing the output range to the reals allowed for finer control. The predicate used can be easily returned to strict PSL by passing the real output through a sigmoid function. We use clipping to avoid underflow and overflow from exponentiation. We assume that for a value of $\vec{0}$ the expression is satisfied and the expression's distance from $\vec{0}$ is used to measure distance from satisfaction.

First, for each site, we create a distribution of the bounding box widths and lengths from the training set data. Like the area, this distribution is modeled as a Weibull distribution. We want to reduce the size of bounding boxes with a width or length that falls outside of 98% of the distribution on the right tail. By constructing separate tests of each dimension, we can control for bounding box aspect ratio. Let $\hat{t}$ be a predicted bounding box of class tree for prediction $\hat{y}$. As described in section III-A, $\hat{t}$ has the form given in (9).

Let's assume the optimum aspect ratio for bounding boxes at this site is 1:1. Thus the optimum width and height, w^* and h^* respectively, are the same. We define width as

$$w = t_{xmax} - t_{xmin} \tag{37}$$

and height as

$$h = t_{ymax} - t_{ymin}. \tag{38}$$

Then let function d_x be defined as

$$d_x = \begin{cases} 0, (w^* - w) \in [l_{lw}, l_{rw}] \\ w^* - w, otherwise \end{cases} \tag{39}$$

and d_y be defined as

$$d_y = \begin{cases} 0, (h^* - h) \in [l_{lh}, l_{rh}] \\ h^* - h, otherwise \end{cases} \tag{40}$$

where l_{lw}, l_{rw}, l_{lh}, and l_{rh} are the right and left limits representing the cutoffs for the width and height bounding box distributions respectively. We define function $constrained$ as

$$constrained : \hat{t} \rightarrow \langle d_x(w), d_y(h) \rangle \tag{41}$$

To avoid applying this rule in contexts where trees with very large crown areas are expected to occur, we incorporate the ecological concept of competition as defined in section III-C. The rule can be interpreted in natural language as

If a prediction is in competition then there is a chance its corresponding bounding box should be constrained.

We write this in PSL as

$$\forall(\hat{y}, \hat{t})\ isComp((\hat{y}, \hat{t})) \implies constrained(\hat{t})) \tag{42}$$

where $isComp$ is true if $\hat{y}$ intersects another prediction's bounding box and false otherwise. The coordinates of $\hat{t}$ are converted to homogeneous coordinates. We use the components q^* from (14) as the scaling coefficients of an affine transformation that scales prediction $\hat{t}$ while keeping its original center [68].

C. Rule 3

Rule 3 imposes the well established ecological pattern of an allometry between tree height and crown area on the model [69], [70]. For each site the co-registered RGB images are combined with the LIDAR-based CHM to provide data on tree height for training and testing. The CHM is used to create a log-linear model of height to crown area. Following the method given in [51] ordinary least squares on log-transformed data is employed for fitting the regression model. Crown area as a function of height is given by the power function

$$A_{itc} = b \cdot h^a \tag{43}$$

where a and b are constants that can vary by forest type, A_{itc} is the ITC area, and h is maximum height detected within the ITC bounding box. The function $bboxSize$ is modified, replacing the constant, μ_{area}, with the fitted function (43):

$$bboxHCA(A) = \frac{1}{1 + e^{-k_{sig} \cdot (b \cdot h^a - A)}} \tag{44}$$

Because we have access to the CHM in this context it is also used to encode ecological knowledge about minimum tree height. This is done by creating a mask that has the same dimensions as the CHM and RGB image. The value of the mask is set to zero where the CHM height is less than 2 m and 1 everywhere else. Heights of 2-3 m are typically considered reasonable approximations for the minimum height of trees that can be detected in remote sensing data. Morphological dilation is applied to the mask with a 7x7 kernel [58] [68]. The mask is then multiplied by the RGB image, effectively removing areas that are unlikely to have trees. Rule 3 is then applied using the same PSL in Rule 1, but with the modified $bboxSize$ function, $bboxHCA$, describing the ecological allometry. This combination of mask and rule provides integrated information of two sources of ecological knowledge about how tree crowns are related to tree height.

Because DeepForest is designed purely for RGB imagery, we incorporate the CHM into the RGB rasters as a 4th channel and modify the dataset object to apply the mask and remove the 4th channel prior to passing the RGB raster to RetinaNet. Then RetinaNet receives the masked raster and the CHM as separate tensors. In this way, RetinaNet is able to incorporate the CHM model without major changes to its architecture allowing for better comparisons to the original model. We also alter the baseline for rule 3. The rule 3 augmented version of DeepForest is compared to the predictions of plain DeepForest on the masked RGB rasters.

D. Rule 4

Rule 4 is used to demonstrate that in addition to improving model performance neuro-symbolic approaches can be used to modify what models can do. This is different from the other rules in that it changes the general behavior of the model. This rule modifies the basic DeepForest model to ignore dead trees. While identification of dead trees from remote sensing is challenging, it is generally associated with a lack of green coloration during the growing seasons [71]–[73]. Traditional approaches to this would use different training data with alive and dead labels. Instead we add an additional rule without relabeling the data.

For this experiment we concentrate on TEAK, as it is the dataset with the largest number of easily identifiable dead trees. The TEAK test set is composed of 734 annotated trees. To determine if we could qualitatively adjust the behavior of the model we used a rule based on the appearance of alive vs. dead trees, with alive trees having a reasonable amount of green foliage (NEON imagery is collected during the growing season) and dead trees having little to no green and lots of brown. For analysis, we placed each tree in 1 of 3 categories: trees composed only of $> 20\%$ green pixels, trees composed only of $> 20\%$ brown pixels. And trees composed of both $> 20\%$ green and $> 20\%$ brown pixels. We consider the trees with $> 20\%$ green to be living regardless of their brown composition. Twenty percent was empirically determined to give observably differentiable results between a tree that appears living and dead. Table IV gives the count for each category of tree.

After converting training images to the hue, saturation, and value (HSV) color space, values for the three components were chosen that include the majority of the green trees in the training set. Unlike RGB, HSV is a non-linear color space that better matches human intuition regarding colors [68]. In the HSV space it is easier to select for color due to the arrangement of hue on the color cylinder [74]. We chose the values of HSV between (18, 47, 0) and (44, 161, 227), representing the green color found in living trees at the site. For brown we use HSV values between (2, 68, 224) and (20, 132, 255). The brown color limits were only used to categorize the trees for analysis purposes.

We create a function named $isGreen$ to represent the probability a prediction is a tree based on the percentage of green pixels it contains. We measure the percentage of green pixels in a predicted bounding box by first converting the sub-image within the bounding box to the HSV color space and then counting the number of green pixels. Let A_b be the set of pixels in the bounding box defined by $\hat{t}$. Then the cardinality of set A_b is given by

$$|A_b| = A_{bbox}(\hat{t}). \tag{45}$$

Then for pixels in set A_b let A_g be the pixels that are within the HSV green color boundaries defined above such that $A_g \subseteq A_b$. Let function $inRange$ count the number of pixels within $\hat{t}$ on image I that are within the HSV green color limits such that

$$|A_g| = inRange(I, \hat{t}). \tag{46}$$

Then let

$$p_g = \frac{|A_g|}{|A_b|}. \tag{47}$$

Again using the sigmoid, we create the function $isGreen$ as

$$isGreen(p_g) = \frac{1}{1 + e^{(100 \cdot p_g - t_p)}}, \tag{48}$$

where, t_p is between $[0, 100]$ representing the threshold between the color composition for living and dead trees.

Rule 4 uses a PSL statement similar to rule 1 (see III), but replaces the function $bboxSize$ with the function $isGreen$. Unlike rule 1, during training, rule 4 is only applied to predictions that match ground-truth trees using the $isTree$ function. This was found to boost performance. Ground-truth is not needed to make predictions once the model is trained. The PSL for rule 4 can be interpreted in natural language as

For all predictions that match ground truth trees, if the ITC color composition is more than 20% green pixels, it is probably a tree.

Table III summarizes each rule, its PSL, and the domain knowledge it incorporates.

E. Hyperparameters

After initializing the model with pre-trained weights, rules 1 - 3 were trained for a fixed number of epochs depending on the training site. The number of epochs for each site was determined empirically. Because the performance of the teacher-network is dependent on the student-network's skill, performance is usually best when a rule is applied in the latter epochs of the training process. Therefore, there is an inherent trade-off between not over-fitting the model and training for enough epochs to allow the rule to take effect, but not so many epochs that the rule causes an imbalance between what is learned from the training set and what is learned from the rules. Continued application of the rule beyond some point degrades performance. Adding to this, the lengthy hyperparameter tuning process, we erred on the side of fewer training epochs. Fig. 6 shows the training and validation loss for the baseline model at each site. Models trained on NIWO were trained for 7 epochs and the remaining sites were trained for 5 epochs. The use of transfer learning causes the initial inversion of the validation and training curves. Rule 4 was trained for 6 epochs. As shown in Fig. 6 none of the models suffered from over-fitting. We use

Fig. 6. The training and validation loss at each site for plain DeepForest using pre-trained weights.

a version of the function used in [44] as the imitation parameter, π. In (49) t is the training step number, α is a constant ≤ 1, and π_0 is a constant < 1 that limits the growth of the function.

$$\pi(t) = \begin{cases} 1.0 - max\{\pi_0, \alpha^t\}, t > \pi_s \\ 0, otherwise \end{cases} \tag{49}$$

The baseline and experiment models shared the same non-rule related hyperparameters in each experiment. The only difference between the baseline and experimental models for rules 1, 2, and 4 were within the logic harness and the modification of the loss function. For rule 3, the DeepForest data pipeline was modified to accommodate the addition of the CHM.

Hyperparameters related to the rules were optimized using a combination of automated and manual tuning. The recommended manual tuning process for rules is discussed in section IV-F. For rules 1 - 3 site specific hyperparameters were chosen to optimize performance as measured by F1. Rule 4 hyperparameters were selected to minimize the percentage of dead trees detected. The hyperparameters for each experiment are summarized in table V - VIII.

DeepForest shows a large model variance. We found that the variance varies by site. The sites in order of decreasing variance are SJER, MLBS, TEAK, and NIWO. After removing all other sources of variation, it was found that weight initialization, and thus random number generator (RNG) initialization, is a factor in the final performance of the trained model. Therefore, to make accurate comparisons between baselines and experiments, we chose to train and test the baseline model and experiments across a randomly selected set of RNG initial values. To calculate the change in performance between the baseline and experiment we make a 1-to-1 comparison for each RNG seed and take the average. We report the mean and 95% confidence intervals for F1, bounding box precision, and bounding box recall.

A λ_r of 100 was used for each rule. Each rule was trained with a batch size of 1 and a learning rate of 0.0018 using stochastic gradient descent. The experiments were performed on a high performance computing cluster using a NVIDIA A100 GPU with 80 GB of GPU memory, 20GB of RAM, and a single processor.

F. Creating Rules

We show that rules can be created from a variety of domain knowledge and used to constrain the neural network training process. There are no limitations on the domain knowledge that can be used, but [44] suggests logic harnessing works best when the domain knowledge is also well represented in the training set data. Once this requirement is met, the next hurdle is the PSL formulation of rules. The most straightforward formulations come from the creation of a set of predicates that map an attribute of interest to a real number in $[0, 1]$, where the range of the output can be interpreted as the probability that attribute is in agreement with the metric it is intended to measure.

Hyperparameter tuning is one of the most time consuming parts of the training process. We recommend using an automated tuner when possible. When tuning by hand these are the steps we followed.

1) Determine the number of epochs that give the best model performance without the harness, let this be e_b. Let the number of training epochs where π begins to be incremented be π_s.
2) Set the parameters of π for a low rate of growth, we recommend starting with $\alpha = 0.95$ and $\pi_0 = 0$.
3) With the harnessed model and modified loss function, set $\pi_s = e_b - 1$ and train the model.

4) If the model's performance improves versus the baseline, repeat step 3 decrementing π_s by 1. If the model's performance does not improve, the rule may not be useful for the dataset. Otherwise, repeat step 3 until the model's performance decreases relative to the previous value of π_s.
5) Set π_s to the value that gave the best performance. Decrement α in increments of 0.05 - 0.10 until a value is found that worsens performance. Note the final value of π.
6) Set α to 0.5 to 0.10 less than the value that gave the best performance in step 5. Try setting π_0 to the value that allows the maximum value of π found in step 5. Retrain the network. If this step improves performance, then try decreasing π_s by 1. Keep the parameters that give the best performance.

V. Results

A. Ablation Study of π

Before delving into a site by site analysis of the results, we explore the effect of variation in π on F1 and the verification ratio of each rule. For each sample point, the value of π is fixed and the model is trained for the number of epochs associated with each rule. The top row of Fig. 7 shows the average F1 score versus π across all sites. The bottom row shows the average verification ratio versus π across all sites. The results suggest the model pays a high cost in F1 for enforcing rules 1 and 3, and a much lower cost for rules 2 and 4. For rule 2, the marginal change in F1 is likely the result of the limited number of crowns the rule is designed to affect. Similarly, rule 4 only affects a fraction of trees. Rules 1 and 3 affect all predictions, a significant proportion of which are larger than the mean, as shown in Fig. 4. Therefore, increasing the verification ratio for rules 1 and 3 has ramifications for globally affecting what features are recognized as trees in addition to reducing IoU scores. This negatively impacts F1.

On average rule 2 and 4 are least likely to produce an increase in F1 versus the baseline. Because rule 4 is designed to induce a bias that causes DeepForest to ignore certain trees, the drop in F1 with increasing rule verification is a logical outcome. For all rules, π is positively correlated with the verification ratio. We leave the ablation study needed to determine how the dataset features effect curve structure to future works.

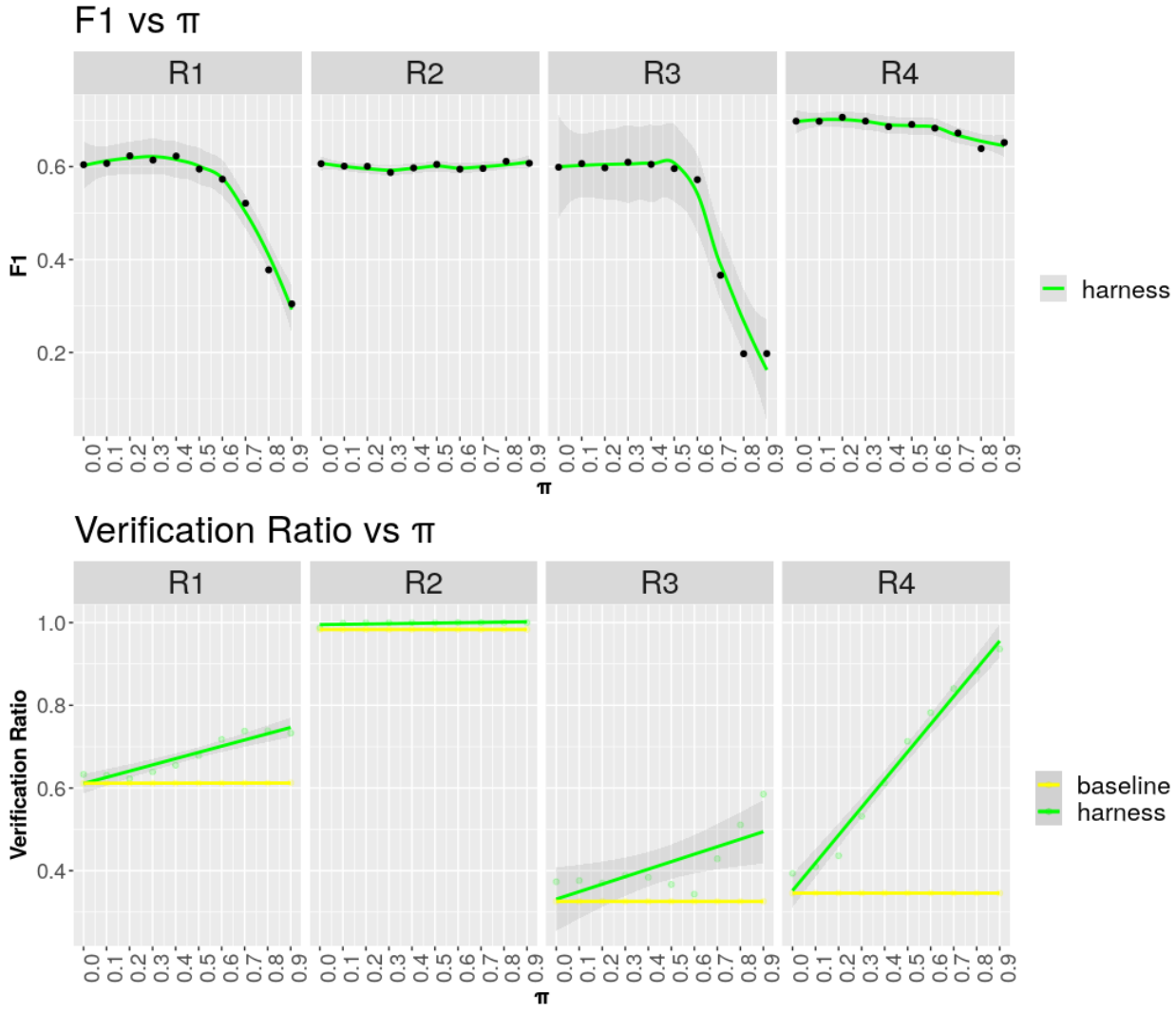


Fig. 7. A plot of F1 and verification ratio versus π averaged over 4 samples. For each sample point, the value of π is fixed and the model is trained for the number of epochs associated with its rule.

B. Rules 1 - 3

Next we look at the effect of rules 1 - 3 on each site. Rule 1 was the most effective. It improved the F1 score for every site. It was most impactful on SJER, increasing its F1 by 4.01 F1 points. It was least effective on MLBS, increasing its F1 by 0.708 F1 points.

Of the rules, rule 2 had the poorest showing, definitively improving F1 only for TEAK. Rule 2 appears to have no effect on NIWO and a negative effect on MLBS and SJER. It decreased the F1 score at MLBS by 2.09 F1 points and at SJER by 1.31 F1 points.

Rule 3 had a net positive effect at 3 of the 4 sites: MLBS, NIWO, and TEAK. It was most

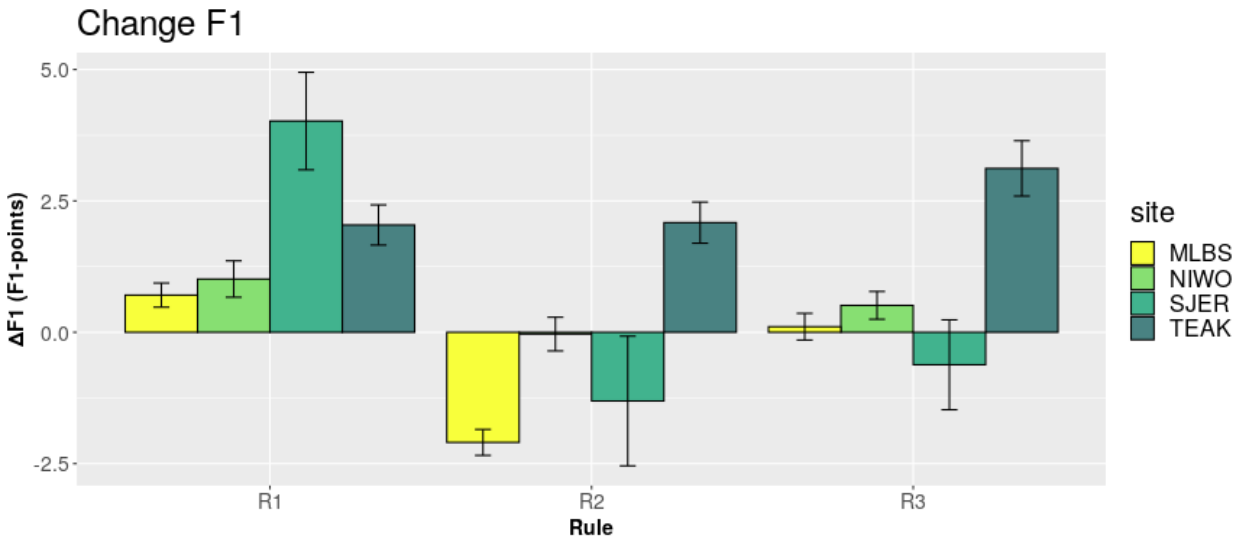


Fig. 8. The average change in F1 for each site-rule combination with 95% confidence interval error bars.

effective at TEAK, increasing its F1 by 3.12 F1 points. Its effect on SJER was not definitively negative, but on average its net result was a decrease in F1 by 0.61 F1 points. Fig.8 gives a per-site breakdown of rule performance.

The effect on F1 for each rule came as a result of changes to both precision and recall. Rule 1 improved precision for all sites, but had a net negative effect on recall for MLBS and SJER. Rule 2 increased precision for NIWO and TEAK and decreased precision for MLBS and SJER. It had a negative effect on recall for all sites except TEAK. Rule 3 had a positive effect on precision for all sites except NIWO. It had a negative effect on recall for every site except NIWO. Table IX summarizes the effect of each rule on precision, recall, and F1.

Averaged across all sites, rule 1 decreased the number of predictions by 6.34%. Rule 2 reduced the number of predictions by 3.88% and rule 3 by 7.75%. On a per-site basis, each rule caused a reduction in the number of predictions except for rule 1 and rule 3 on NIWO. Rule 1 increased the number of predictions on NIWO by 1.72% and rule 3 increased the number of predictions on NIWO by 3.08%.

To try to understand how application of the rules produces its effect on models we examined IoU for predictions that matched a ground-truth bounding box. Fig. 9 shows the distribution of IoUs for the baseline in yellow and the experiments in green. The vertical line represents the mean of the respective distribution. Every rule, even in cases where the rule did not improve

F1, shifted the distribution of IoU's to the left. The number of predictions with IoU's between 50 - 60% increases while predictions with IoU's $> 70\%$ shrank. Furthermore, for all sites except MLBS, the baseline model predictions show a positive correlation between the size of a predicted bounding box and IoU. Application of the harness preserved this trend when it existed, but reduced the slope of the regression line for IoU versus predicted bounding box area (See Appendix Fig. 14).

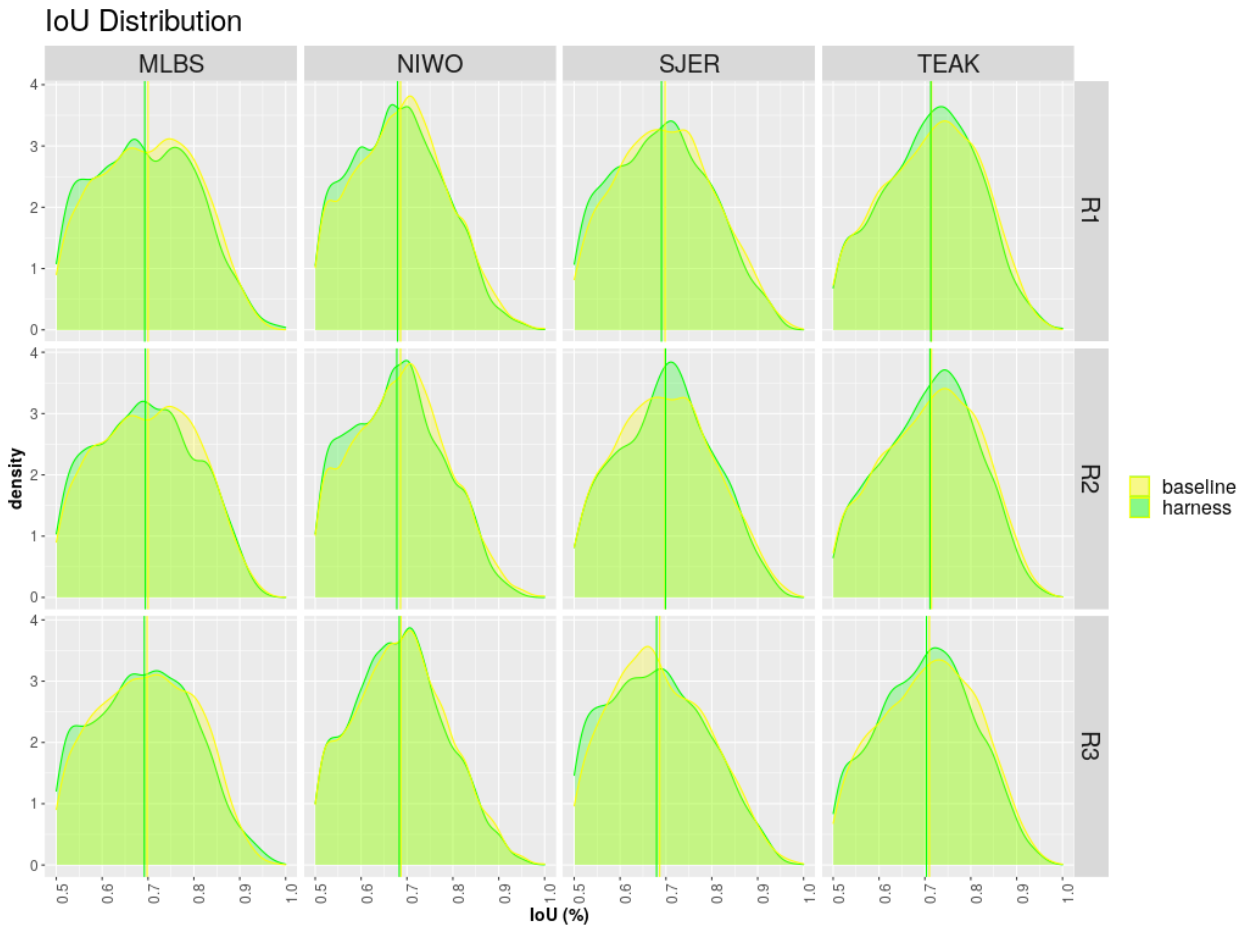


Fig. 9. The distribution of IoU values for predictions that were matched with ground-truth bounding boxes. The vertical lines represent the mean of their respective distribution.

Next we looked at how well predicted bounding box area matched with ground-truth area. Fig. 10 shows a scatter plot of predicted area to ground truth area for predictions that matched with a ground truth bounding box. The baseline predictions are in yellow and experimental predictions in green. The change in RMSE and MAE for each site is listed in the upper right hand corner. On average, the rules had the following effect on RMSE: rule 1 reduced the RMSE by 83.6, rule

2 reduced it by 99.3, and rule 3 increased RMSE by 41.7. A similar effect was seen with MAE. Rule 1 decreased MAE by 36.2, rule 2 decreased MAE by 43.7, and rule 3 increased MAE by 33.6. Change in RMSE and MAE was most pronounced for SJER for both positive and negative change. NIWO generated the smallest changes in RMSE and MAE. Reduction in RMSE and MAE were consistently negatively correlated with improvement in F1. Some sites where the F1 worsened with application of a rule, such as rule 2 on MLBS still showed an improvement in RMSE. Inversely, there were sites such as rule 1 on NIWO, where F1 improved, but RMSE was slightly worse.

It can also be seen from Fig. 10 that the application of the logic harness tends to reduce the size of the area of the predictions. In the majority of the cases, the regression line for the harness falls below the baseline regression line. And in most cases the maximum predicted area for the harness is less than the maximum predicted area for the baseline.

We also examined how well baseline and experiment models matched the distribution of the ground-truth bounding box areas for each test set. In Fig. 11, the baseline distribution is shown in yellow, the experiment distribution is shown in green, and the ground-truth distribution for the test set is shown in orange. We measured the KL-divergence between the baseline and the test set as well as between the experiment and the test set. The difference between the baseline and experiment KL-divergence is in the upper right corner of each graph. The color coded vertical lines represent the mean of their respective distributions. On average, each rule decreased the KL-divergence. Rule 1 decreased KL-divergence by 0.028, rule 2 by 0.005, and rule 3 by 0.0428. On a per site basis the KL-divergence was reduced most on NIWO by rule 3. There was an increase in KL-divergence for 3 out of the 12 rule-site combinations: rule 1 on TEAK, rule 2 on NIWO, and rule 3 on TEAK. For 7 out of the 8 rule-site combinations that resulted in an increase in F1 there was a decrease in KL-divergence.

We examined the ratio of the two terms of the loss function to understand how the interplay of the loss terms may affect model performance. In Fig. 12, we show the average loss versus epochs during training. The distillation-loss is in green, the student-loss is in orange, and π is in black. The graphs show that as π increases the loss contributed by the distillation term tends to decrease. The enforcement of the rule causes predictions to be more in line with the teacher reducing the distillation-loss. This trend is most easily observed in the TEAK-R3 and TEAK-R4 graphs. For the majority of the graphs, the reduction in the student-loss is not proportional to the reduction in distillation-loss as π increases. However, in some cases there is an increase in

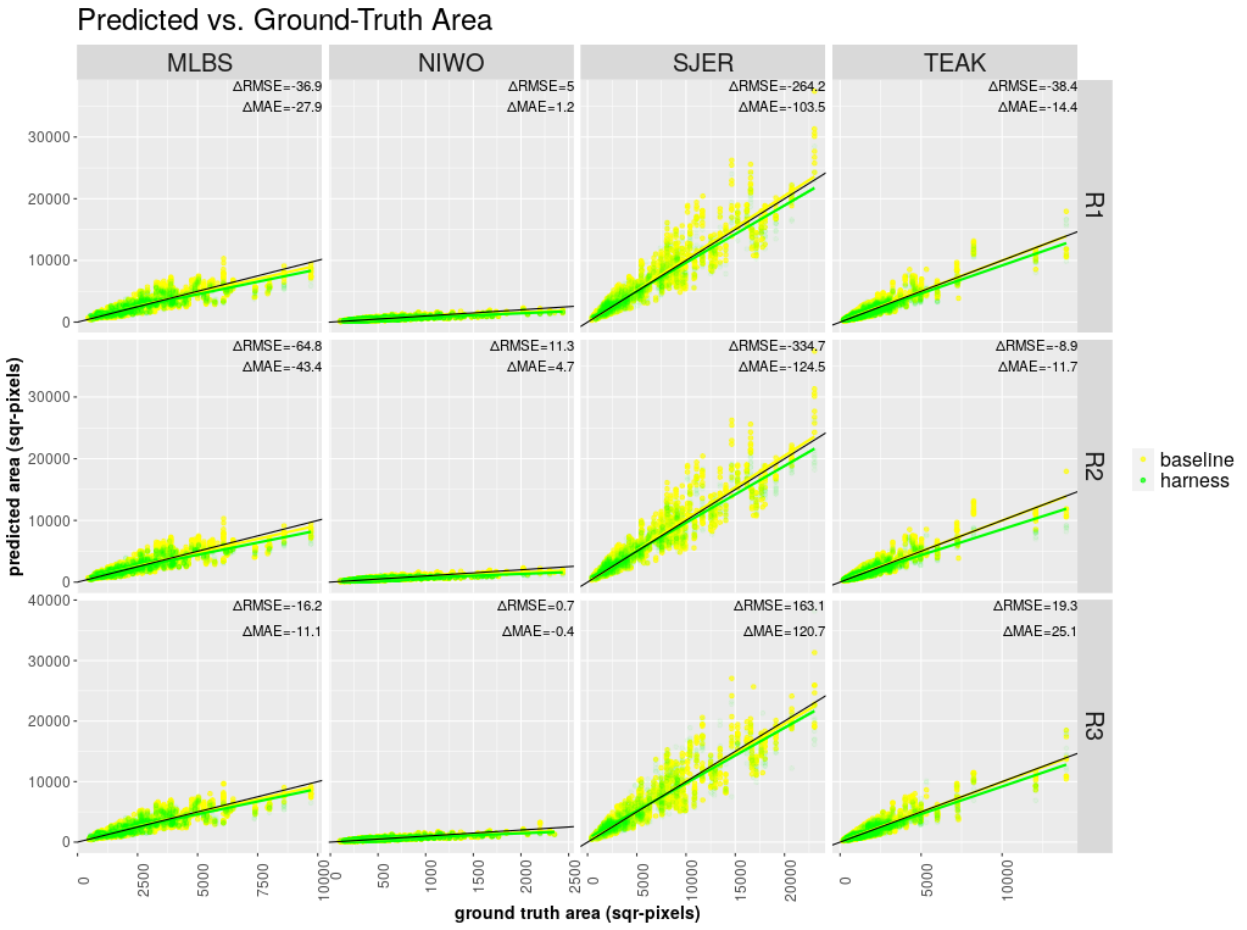


Fig. 10. The scatter plot of the predicted bounding box area versus the ground-truth bounding box area for predictions that were matched with ground-truth boxes. The change in RMSE and MAE for each site-rule combination is shown in the upper right corner.

the student-loss as the distillation-loss decreases. This suggests there may be a conflict between the rules and the dataset annotations. This is most visible in TEAK-R3 and SJER-R3 at the end of their last epoch.

C. Rule 4

Of the rules, the results from rule 4 are the easiest to observe. Rule 4 was designed to bias the model to ignore dead trees, where we define a dead tree to be a tree whose pixel composition is less than 20% green. We only applied this rule to TEAK. In the images of Fig. 13 the orange boxes are ground-truth annotated trees. The green boxes are predictions. In the first image, the ground-truth annotations are shown with the percentage of green pixels that compose each

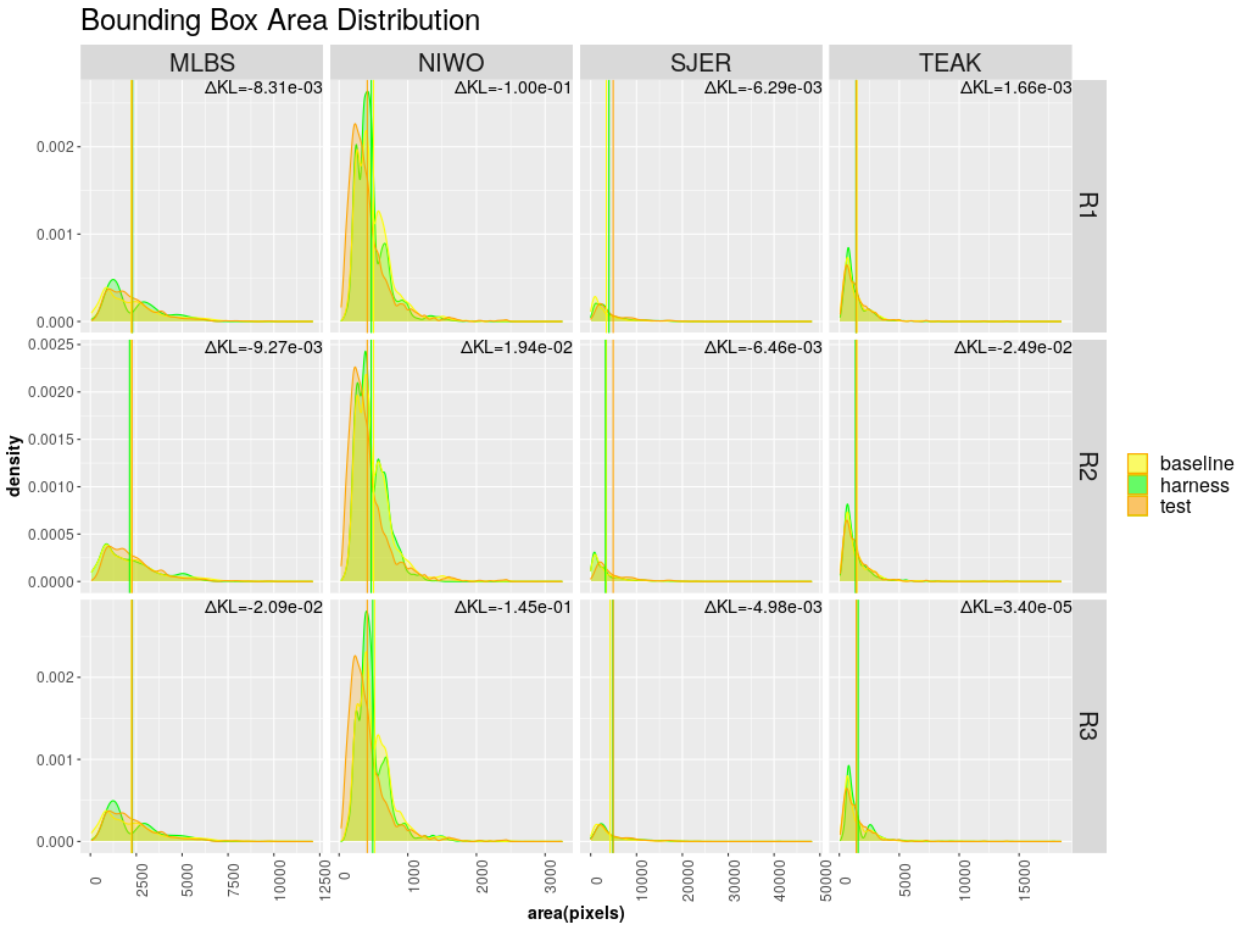


Fig. 11. The distribution of prediction bounding box areas for each site-rule combination. The predictions for plain DeepForest are shown in yellow, DeepForest+Rule in green, and the distribution from the test set in orange. The change in the KL-divergence between the baseline and the experiment for each site-rule combination is given in the upper right corner.

bounding box. Many of the brown trees have a pixel composition of less than 20% and are thus considered dead. The second image shows the predictions from the baseline model after being trained on the dataset. Both living and dead trees are detected. The third image shows the predictions from the model after being trained on the same dataset with rule 4. The majority of the dead trees are ignored.

On average, application of rule 4 decreased the percentage of dead trees detected by 52 percentage points compared to the baseline. The detection of trees that had a composition of $>$ 20% green pixels was reduced by 2.5 percentage points compared to the baseline. The detection of trees composed of $>$ 20% green pixels and $>$ 20% brown pixels were decreased by 6.2 percentage points compared to the baseline model. The application of rule 4 increased the bounding box

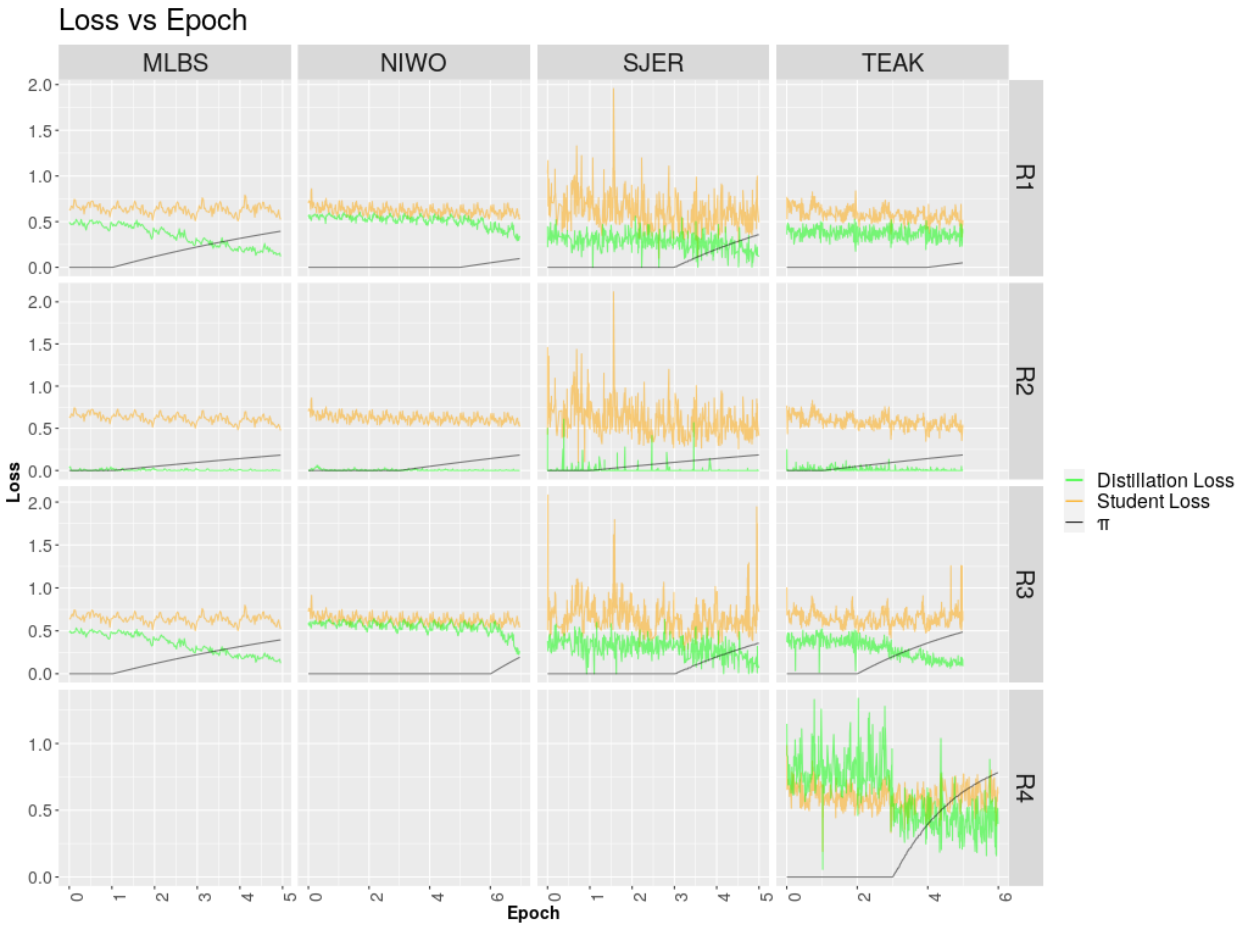


Fig. 12. The average of the terms of the loss function for each site-rule combination. The distillation-loss in green represents the loss from the logic harness, the student-loss in orange represents the sum of the loss terms from the original model. The value of π is shown in black.

precision by 8.85 percentage points and reduced the recall by 9.82 percentage points. This resulted in a drop in F1 of 1.34 F1 points. The detailed results of rule 4 are listed in Table X.

As indicated by the reduction in green trees detected and the slight increase in the student-loss with an increase in π (see Fig. 12), the enforcement of rule 4 causes contradictions with the annotated training data. The reason there is not a larger drop in F1 is likely the small number of dead trees within the test set.

VI. Conclusion

Neuro-symbolics is a means of incorporating domain knowledge into neural networks. To the best of our knowledge, this work is the first to attempt applying neuro-symbolics to CNN

Fig. 13. Each image is based on the same RGB raster from the TEAK test set. The left image shows the ground-truth annotations with the percentage of green pixels found in each bounding box. The middle image shows the predictions from plain DeepForest in green with the ground-truth annotations in orange. The right image shows the predictions from DeepForest with rule 4 applied.

based ITC delineation. Current ITC delineation models, both non-neural and machine learning based, have no standardized way of ensuring domain knowledge is absorbed into their models. Nor is there a standardized method of formulating the knowledge. Using our method, ecologists can ensure that high level ecological concepts like competition, allometries, and even specific visual features are baked into their models and that the domain knowledge can be formulaically represented as a rule. Though it can be argued that similar results can be obtained using the traditional method of creating datasets specifically for a task, a neuro-symbolic approach has several benefits by comparison. Foremost, neuro-symbolics guarantee some level of explainability. Being able to formulate a rule and constrain the model to obey the rule during inference creates a more trustworthy model and provides some guarantee as to what features the model will use to make its inference. Secondly, neuro-symbolics can ensure that a model will learn the desired concept. If there is a contradiction between the concept captured in the training data and the concept represented as a rule, it is usually discernible from the model's reduced performance. Finally, neuro-symbolics can provide a means of creating multi-use datasets. By changing a rule, a model can be trained to perform different tasks using the same dataset.

In this study we implemented a neuro-symbolic CNN ITC delineation model that uses domain knowledge encoded as rules written in PSL. We tested 4 rules that were based on different aspects of forest ecology expert knowledge, including competition, constrained growth, allometry, and average ITC areas.

We found that ecologically sound and seemingly applicable rules do not always boost model performance. The rules can have unpredictable effects on other aspects of model inference, but with careful tuning useful parameters can sometimes be found. The degree to which a rule enhances or degrades the target metric is highly dependent on parameter tuning.

While at the time of writing, the authors know of no other study that applies neuro-symbolics to ITC delineation, some of our more generalizable findings are in line with other neuro-symbolic research that has examined the effects the degree of rule enforcement has on model skill, particularly, the work of Seo et al, where it is shown that for an applicable rule, increasing the imitation parameter improves the target metric rapidly in the low to mid range values, but have decreasing or negative returns for increasingly higher settings [64].

A. Limitations

Our study had three notable limitations. 1. We only used NEON data. 2. The annotations were all generated by our group using rectangular bounding boxes. 3. We only tested the harness framework on a single ITC delineation model.

DeepForest alone has been shown to work on other remote sensing datasets, so theoretically, the framework should perform comparably, but this has not been shown empirically. The annotations in our dataset were created all in our group using only airborne imagery for annotation. As such, it is not known how mixing datasets annotated using different methods would affect the performance and enforcement of rules. Similarly, while the logic harness is model agnostic, and should be applicable to other ITC delineation models, we do not know how the use of a different model affects the framework overall.

Of minor note, the rules we implemented all focused on shrinking the bounding boxes, but the framework is not limited to rules that shrink area. We found rules that shrink the bounding box area were most likely to improve F1 and focused on these for our work. However, rules can be applied to perform any number of operations to predictions including expansion, rotation, and altering aspect ratio.

B. Future Work

We believe we have only scratched the surface of what can be done with neuro-symbolics in this area. Neuro-symbolics is known to be capable of reducing the size of a dataset required to train models, and is therefore sometimes used in few shot and zero shot learning [34], [44].

In future works we would like to examine the applicability of this property to ITC delineation. We would also like to perform a study to determine how features in remote sensing datasets influence verification ratio curves and other inference parameters.

C. Conclusion

Neuro-symbolics can be successfully applied to ITC delineation and build upon a wide range of ecology domain knowledge, but results are highly rule and site dependent. Even when a rule and annotated data seem to be compatible there may be issues that limit the effectiveness of the pairing. Applying rules can have unintended effects, if the rule and training data are not in agreement; the rule is enforced at the expense of general model performance.

VII. Acknowledgments

The authors would like to thank the National Ecological Observatory Network. This work was made possible by NSF grant 1926542.

VIII. Appendix

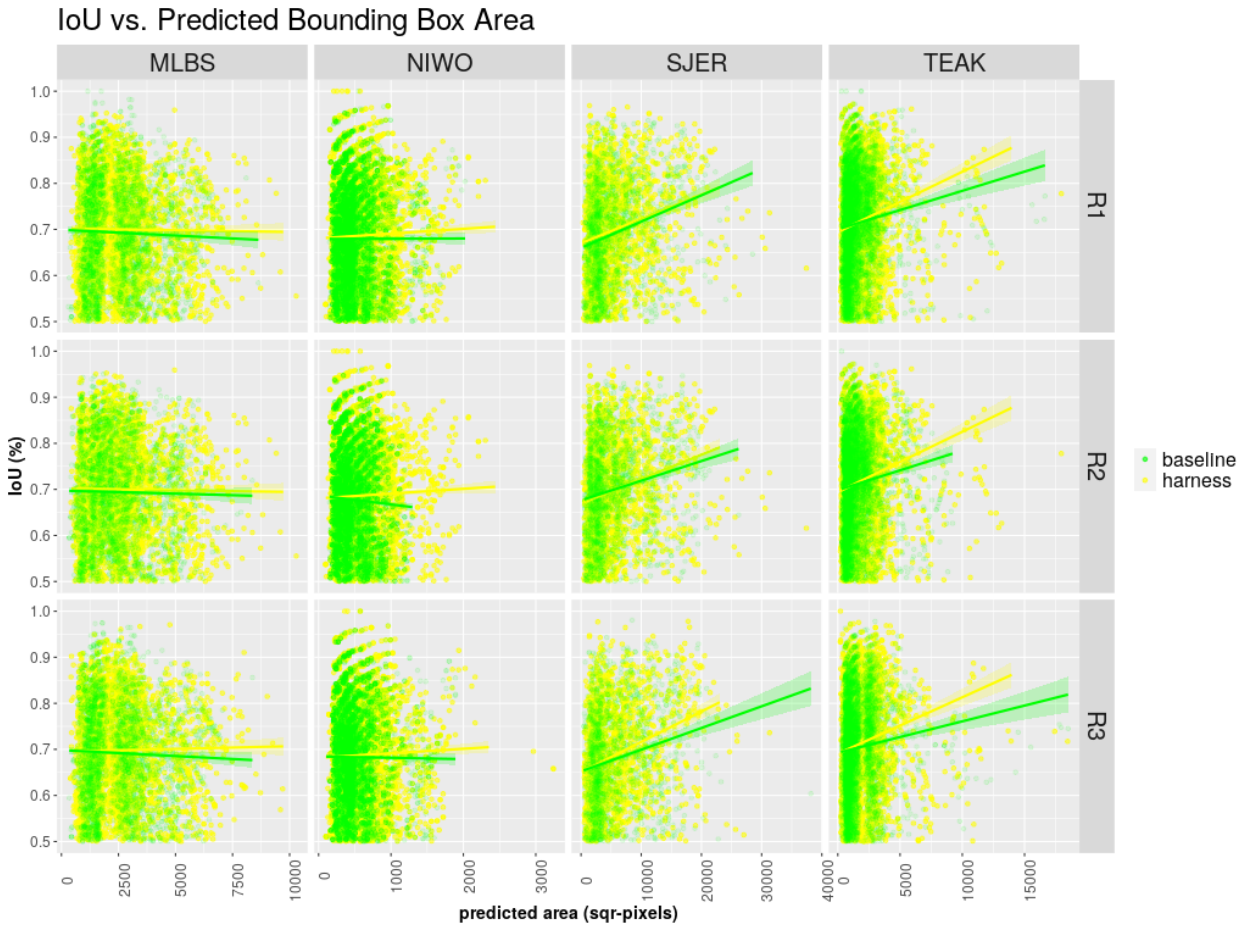


Fig. 14. A scatter plot of IoU vs predicted area for each site-rule combination. Predictions for plain DeepForest are in yellow. Predictions using the logic harness are in green.

REFERENCES

[1] M. Dalla Mura, S. Prasad, F. Pacifici, P. Gamba, J. Chanussot, and J. A. Benediktsson, "Challenges and opportunities of multimodality and data fusion in remote sensing," *Proceedings of the IEEE*, vol. 103, no. 9, pp. 1585–1601, 2015, publisher: IEEE.

[2] E. Chuvieco, *Fundamentals of satellite remote sensing: An environmental approach*. CRC press, 2020.

[3] A. M. Lechner, G. M. Foody, and D. S. Boyd, "Applications in remote sensing to forest ecology and management," *One Earth*, vol. 2, no. 5, pp. 405–412, 2020, publisher: Elsevier.

[4] S. E. Franklin, *Remote sensing for sustainable forest management*. CRC press, 2001.

[5] Y. Ke and L. J. Quackenbush, "A review of methods for automatic individual tree-crown detection and delineation from passive remote sensing," *International Journal of Remote Sensing*, vol. 32, no. 17, pp. 4725–4747, 2011, publisher: Taylor & Francis.

[6] T. Brandtberg, "Automatic individual tree based analysis of high spatial resolution aerial images on naturally regenerated boreal forests," *Canadian Journal of Forest Research*, vol. 29, no. 10, pp. 1464–1478, 1999.

[7] B. G. Weinstein, S. Marconi, S. Bohlman, A. Zare, and E. White, "Individual Tree-Crown Detection in RGB Imagery Using Semi-Supervised Deep Learning Neural Networks," *Remote Sensing*, vol. 11, no. 11, p. 1309, Jan. 2019, number: 11 Publisher: Multidisciplinary Digital Publishing Institute. [Online]. Available: https://www.mdpi.com/2072-4292/11/11/1309

[8] J. R. G Braga, V. Peripato, R. Dalagnol, M. P Ferreira, Y. Tarabalka, L. E. OC Aragão, H. F de Campos Velho, E. H. Shiguemori, and F. H. Wagner, "Tree crown delineation algorithm based on a convolutional neural network," *Remote Sensing*, vol. 12, no. 8, p. 1288, 2020, publisher: Multidisciplinary Digital Publishing Institute.

[9] F. H. Wagner, M. P. Ferreira, A. Sanchez, M. C. Hirye, M. Zortea, E. Gloor, O. L. Phillips, C. R. de Souza Filho, Y. E. Shimabukuro, and L. E. Aragão, "Individual tree crown delineation in a highly diverse tropical forest using very high resolution satellite images," *ISPRS journal of photogrammetry and remote sensing*, vol. 145, pp. 362–377, 2018, publisher: Elsevier.

[10] S. Marconi, S. J. Graves, D. Gong, M. S. Nia, M. Le Bras, B. J. Dorr, P. Fontana, J. Gearhart, C. Greenberg, D. J. Harris, and others, "A data science challenge for converting airborne remote sensing data into ecological information," *PeerJ*, vol. 6, p. e5843, 2019, publisher: PeerJ Inc.

[11] R. M. Haralick and L. G. Shapiro, "Image segmentation techniques," *Computer vision, graphics, and image processing*, vol. 29, no. 1, pp. 100–132, 1985, publisher: Elsevier.

[12] B. Koch, U. Heyder, and H. Weinacker, "Detection of Individual Tree Crowns in Airborne Lidar Data," *Photogrammetric Engineering & Remote Sensing*, vol. 72, no. 4, pp. 357–363, Apr. 2006. [Online]. Available: http://openurl.ingenta.com/content/xref?genre=article&issn=0099-1112&volume=72&issue=4&spage=357

[13] W. S. Wan Mohd Jaafar, I. H. Woodhouse, C. A. Silva, H. Omar, K. N. Abdul Maulud, A. T. Hudak, C. Klauberg, A. Cardil, and M. Mohan, "Improving individual tree crown delineation and attributes estimation of tropical forests using airborne LiDAR data," *Forests*, vol. 9, no. 12, p. 759, 2018, publisher: Multidisciplinary Digital Publishing Institute.

[14] A. S. Alon, E. D. Festijo, and C. D. Casuat, "Tree extraction of airborne lidar data based on coordinates of deep learning object detection from orthophoto over complex mangrove forest," *International Journal*, vol. 8, no. 5, 2020.

[15] X. Chen, K. Jiang, Y. Zhu, X. Wang, and T. Yun, "Individual tree crown segmentation directly from UAV-borne LiDAR data using the PointNet of deep learning," *Forests*, vol. 12, no. 2, p. 131, 2021, publisher: Multidisciplinary Digital Publishing Institute.

[16] M. Aubry-Kientz, A. Laybros, B. Weinstein, J. G. C. Ball, T. Jackson, D. Coomes, and G. Vincent, "Multisensor Data Fusion for Improved Segmentation of Individual Tree Crowns in Dense Tropical Forests," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 14, pp. 3927–3936, 2021, conference Name: IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing.

[17] D. Boyd and F. Danson, "Satellite remote sensing of forest resources: three decades of research development," *Progress in Physical Geography*, vol. 29, no. 1, pp. 1–26, 2005, publisher: Sage Publications Sage CA: Thousand Oaks, CA.

[18] F. A. Gougeon, "AUTOMATIC INDIVIDUAL TREE CROWN DELINEATION USING A VALLEY-FOLLOWING ALGORITHM AND A RULE-BASED SYSTEM," *Proc. International Forum on Automated Interpretation of High Spatial Resolution Digital Imagery for Forestry, Victoria, British Columbia, Canada*, pp. 11–23, 1998.

[19] ——, "A crown-following approach to the automatic delineation of individual tree crowns in high spatial resolution aerial images," *Canadian journal of remote sensing*, vol. 21, no. 3, pp. 274–284, 1995, publisher: Taylor & Francis.

[20] R. Adams and L. Bischof, "Seeded region growing," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 16, no. 6, pp. 641–647, 1994, publisher: IEEE.

[21] D. S. Culvenor, "TIDA: an algorithm for the delineation of tree crowns in high spatial resolution remotely sensed imagery," *Computers & Geosciences*, vol. 28, no. 1, pp. 33–44, 2002, publisher: Elsevier.

[22] P. Bunting and R. Lucas, "The delineation of tree crowns in Australian mixed species forests using hyperspectral Compact

Airborne Spectrographic Imager (CASI) data," *Remote Sensing of Environment*, vol. 101, no. 2, pp. 230–248, 2006, publisher: Elsevier.

[23] S. Beucher and F. Meyer, "The morphological approach to segmentation: the watershed transformation," in *Mathematical morphology in image processing*. CRC Press, 2018, pp. 433–481.

[24] L. Wang, P. Gong, and G. S. Biging, "Individual tree-crown delineation and treetop detection in high-spatial-resolution aerial imagery," *Photogrammetric Engineering & Remote Sensing*, vol. 70, no. 3, pp. 351–357, 2004, publisher: American Society for Photogrammetry and Remote Sensing.

[25] M. Dalponte and D. A. Coomes, "Tree-centric mapping of forest carbon density from airborne laser scanning and hyperspectral data," *Methods in ecology and evolution*, vol. 7, no. 10, pp. 1236–1245, 2016, publisher: Wiley Online Library.

[26] I. Sačkov, T. Hlásny, T. Bucha, and M. Juriš, "Integration of tree allometry rules to treetops detection and tree crowns delineation using airborne lidar data," *iForest-Biogeosciences and Forestry*, vol. 10, no. 2, p. 459, 2017.

[27] M. Aubry-Kientz, R. Dutrieux, A. Ferraz, S. Saatchi, H. Hamraz, J. Williams, D. Coomes, A. Piboule, and G. Vincent, "A comparative assessment of the performance of individual tree crowns delineation algorithms from ALS data in tropical forests," *Remote Sensing*, vol. 11, no. 9, p. 1086, 2019, publisher: Multidisciplinary Digital Publishing Institute.

[28] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998, publisher: Ieee.

[29] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[30] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[31] A. S. Garcez, L. C. Lamb, and D. M. Gabbay, *Neural-symbolic cognitive reasoning*. Springer Science & Business Media, 2008.

[32] H. Xu, G. Qi, J. Li, M. Wang, K. Xu, and H. Gao, "Fine-grained Image Classification by Visual-Semantic Embedding." in *IJCAI*, 2018, pp. 1043–1049.

[33] J. Zhou, J. Li, C. Wang, H. Wu, C. Zhao, and G. Teng, "Crop disease identification and interpretation method based on multimodal deep learning," *Computers and Electronics in Agriculture*, vol. 189, p. 106408, 2021, publisher: Elsevier.

[34] G. Sumbul, R. G. Cinbis, and S. Aksoy, "Fine-grained object recognition and zero-shot learning in remote sensing imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 2, pp. 770–779, 2017, publisher: IEEE.

[35] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The caltech-ucsd birds-200-2011 dataset," 2011, publisher: California Institute of Technology.

[36] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting unreasonable effectiveness of data in deep learning era," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 843–852.

[37] T. U. Kampe, B. R. Johnson, M. A. Kuester, and M. Keller, "NEON: the first continental-scale ecological observatory with airborne remote sensing of vegetation canopy biochemistry and structure," *Journal of Applied Remote Sensing*, vol. 4, no. 1, p. 043510, 2010, publisher: SPIE.

[38] B. G. Weinstein, S. J. Graves, S. Marconi, A. Singh, A. Zare, D. Stewart, S. A. Bohlman, and E. P. White, "A benchmark dataset for canopy crown detection and delineation in co-registered airborne RGB, LiDAR and hyperspectral imagery from the National Ecological Observation Network," *PLOS Computational Biology*, vol. 17, no. 7, p. e1009180, Jul. 2021, publisher: Public Library of Science. [Online]. Available: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1009180

[39] M. Hussain, J. J. Bird, and D. R. Faria, "A study on cnn transfer learning for image classification," in *UK Workshop on computational Intelligence*. Springer, 2018, pp. 191–202.

[40] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," *International Journal of Computer Vision*, vol. 129, no. 6, pp. 1789–1819, 2021.

[41] C. Buciluǎ, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2006, pp. 535–541.

[42] G. Hinton, O. Vinyals, and J. Dean, "Distilling the Knowledge in a Neural Network," *arXiv:1503.02531 [cs, stat]*, Mar. 2015, arXiv: 1503.02531. [Online]. Available: http://arxiv.org/abs/1503.02531

[43] K. Ganchev, J. Graça, J. Gillenwater, and B. Taskar, "Posterior Regularization for Structured Latent Variable Models," *The Journal of Machine Learning Research*, vol. 11, p. 49, 2010.

[44] Z. Hu, X. Ma, Z. Liu, E. Hovy, and E. Xing, "Harnessing Deep Neural Networks with Logic Rules," *arXiv:1603.06318 [cs, stat]*, Aug. 2020, arXiv: 1603.06318. [Online]. Available: http://arxiv.org/abs/1603.06318

[45] A. Kimmig, S. Bach, M. Broecheler, B. Huang, and L. Getoor, "A short introduction to probabilistic soft logic," in *Proceedings of the NIPS Workshop on Probabilistic Programming: Foundations and Applications*, 2012, pp. 1–4.

[46] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[47] R. M. Smullyan, *First-order Logic*. Courier Corporation, 1995.

[48] S. H. Bach, M. Broecheler, B. Huang, and L. Getoor, "Hinge-Loss Markov Random Fields and Probabilistic Soft Logic," *arXiv preprint arXiv:1505.04406*, p. 67, 2015.

[49] G. J. Klir and B. Yuan, "Fuzzy sets and fuzzy logic: theory and applications," *Possibility Theory versus Probab. Theory*, vol. 32, no. 2, pp. 207–208, 1996.

[50] "Database for landscape-scale carbon monitoring sites," publication Title: Database for Landscape-scale Carbon Monitoring Sites - NACP - Northern Research Station - USDA Forest Service. [Online]. Available: https://www.nrs.fs.fed.us/data/lcms/niwot/

[51] "Niwot Ridge NEON." [Online]. Available: https://www.neonscience.org/field-sites/niwo

[52] "Lower Teakettle NEON/TEAK," 2019. [Online]. Available: https://www.neonscience.org/field-sites/teak

[53] "Teakettle Experimental Forest." [Online]. Available: https://www.fs.fed.us/psw/ef/teakettle/

[54] "San Joaquin Experimental Range NEON / SJER." [Online]. Available: https://www.neonscience.org/field-sites/sjer

[55] "San Joaquin Experimental Range." [Online]. Available: https://www.fs.fed.us/psw/ef/san_joaquin/

[56] "Mountain Lake Biological Station NEON / MLBS." [Online]. Available: https://www.neonscience.org/field-sites/mlbs

[57] "Airborne Remote Sensing." [Online]. Available: https://www.neonscience.org/data-collection/airborne-remote-sensing

[58] "Lidar." [Online]. Available: https://www.neonscience.org/data-collection/lidar

[59] E. Blanchard, P. Birnbaum, T. Ibanez, T. Boutreux, C. Antin, P. Ploton, G. Vincent, R. Pouteau, H. Vandrot, V. Hequet, and others, "Contrasted allometries between stem diameter, crown area, and tree height in five tropical biogeographic areas," *Trees*, vol. 30, no. 6, pp. 1953–1968, 2016, publisher: Springer.

[60] S. Bohlman and S. O'Brien, "Allometry, adult stature and regeneration requirement of 65 tree species on Barro Colorado Island, Panama," *Journal of Tropical Ecology*, vol. 22, no. 2, pp. 123–136, 2006, publisher: Cambridge University Press.

[61] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[62] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2009, pp. 248–255, iSSN: 1063-6919.

[63] Z. Zou, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," *arXiv preprint arXiv:1905.05055*, 2019.

[64] S. Seo, S. Arik, J. Yoon, X. Zhang, K. Sohn, and T. Pfister, "Controlling Neural Networks with Rule Representations," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[65] F. Ogana and J. Dau, "Deriving Tree Crown Distributions from Diameter at Breast Height," *Journal of Tropical Forestry and Environment*, vol. 9, no. 1, 2019.

[66] M. Mirzaei, J. Aziz, A. Mahdavi, and A. M. Rad, "Modeling frequency distributions of tree height, diameter and crown area by six probability functions for open forests of Quercus persica in Iran," *Journal of Forestry Research*, vol. 27, no. 4, pp. 901–906, Aug. 2016. [Online]. Available: https://doi.org/10.1007/s11676-015-0194-x

[67] B. G. Weinstein, S. Marconi, S. A. Bohlman, A. Zare, A. Singh, S. J. Graves, and E. P. White, "A remote sensing derived data set of 100 million individual tree crowns for the national ecological observatory netwo@articlezielewska2020detection, title=Detection of standing deadwood from aerial imagery products: two methods for addressing the bare ground misclassification issue, author=Zielewska-Büttner, Katarzyna and Adler, Petra and Kolbe, Sven and Beck, Ruben and Ganter, Lisa Maria and Koch, Barbara and Braunisch, Veronika, journal=Forests, volume=11, number=8, pages=801, year=2020, publisher=MDPI rk," *Elife*, vol. 10, 2021.

[68] D. Forsyth and J. Ponce, *Computer vision: A modern approach.* Prentice hall, 2011.

[69] T. Jucker, J. Caspersen, J. Chave, C. Antin, N. Barbier, F. Bongers, M. Dalponte, K. Y. van Ewijk, D. I. Forrester, M. Haeni, and others, "Allometric equations for integrating remote sensing imagery into forest monitoring programmes," *Global change biology*, vol. 23, no. 1, pp. 177–190, 2017, publisher: Wiley Online Library.

[70] C. M. Hulshof, N. G. Swenson, and M. D. Weiser, "Tree height–diameter allometry across the united states," *Ecology and evolution*, vol. 5, no. 6, pp. 1193–1204, 2015.

[71] X. Deng, Z. Tong, Y. Lan, and Z. Huang, "Detection and location of dead trees with pine wilt disease based on deep learning and uav remote sensing," *AgriEngineering*, vol. 2, no. 2, pp. 294–307, 2020.

[72] K. Zielewska-Büttner, P. Adler, S. Kolbe, R. Beck, L. M. Ganter, B. Koch, and V. Braunisch, "Detection of standing deadwood from aerial imagery products: two methods for addressing the bare ground misclassification issue," *Forests*, vol. 11, no. 8, p. 801, 2020.

[73] J. W. Thomas, *Wildlife habitats in managed forests: the Blue Mountains of Oregon and Washington*. Wildlife Management Institute, 1979, no. 553.

[74] S. Sural, G. Qian, and S. Pramanik, "Segmentation and histogram generation using the HSV color space for image retrieval," in *Proceedings. International Conference on Image Processing*, vol. 2. IEEE, 2002, pp. II–II.

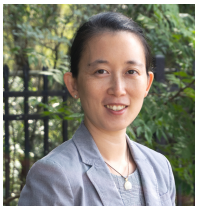
IX. Biography Section

Ira Harmon is a computer science Ph.D. student at the University of Florida. His research interests include computer vision, remote sensing, machine learning, and information security.

Sergio Marconi received a Ph.D. degree in Natural Resources and Environment from the University of Florida, Florida, FL, USA, in 2020. He is currently a post-doctoral researcher with the School of Forest, Fisheries and Geomatics Sciences at the University of Florida. His research focuses on the intersection between forests ecology, remote sensing, eco-informatics and machine learning.

Ben Weinstein Photograph and biography not available at the time of publication.

Sarah Graves Photograph and biography not available at the time of publication.



Daisy Zhe Wang is an Associate Professor in the CISE department at the University of Florida. She is the Director of the Data Science Research Lab at UF. Dr. Wang is particularly interested in bridging scalable data management and processing systems with probabilistic models and statistical methods. She currently pursues research topics such as probabilistic databases, probabilistic knowledge graphs, large-scale inference engines, query-driven interactive machine learning, neuro-symbolic AI, and crowd assisted machine learning.

Alina Zare teaches and conducts research in the area of machine learning and artificial intelligence as a Professor in the Electrical and Computer Engineering Department at the University of Florida. Dr. Zare's research has focused primarily on developing new machine learning algorithms to automatically understand and process data and imagery. Her research work has included automated plant root phenotyping, hyperspectral image analysis, target detection and underwater scene understanding using synthetic aperture sonar, LIDAR data analysis, and Ground Penetrating Radar analysis.

Stephanie Bohlman received a Ph.D. degree in Forest Resources from the University of Washington, Seattle in 2004. She is currently an Associate Professor with the School of Forest, Fisheries and Geomatics Sciences at the University of Florida. Her research interests include understanding large scale patterns of forest composition, function and dynamics using a combination of remote sensing, models and field data.

Aditya Singh Photograph and biography not available at the time of publication.

Ethan White received his Ph.D. degree in biology from the University of New Mexico, Albuquerque, NM, USA, in 2005. He is currently an Associate Professor with the Department of Wildlife Ecology and Conservation and the co-director of Weecology at the University of Florida. His research interests include the use of large ecological datasets, remote sensing, and machine learning to understand and forecast the dynamics of ecosystems.

TABLE II

THE COUNT OF COMPETING TREES (CMP) AND NON-COMPETING TREES (NCMP) FOR EACH SITE'S TRAINING SET. FREQUENCY RELATIVE TO THE SITE SAMPLE SIZE IS IN PARENTHESES.

	NIWO	**TEAK**	**MLBS**	**SJER**
cmp $\leq$ mean	5,590 (0.520)	2,243 (0.497)	1,408 (0.599)	720 (0.255)
cmp $>$ mean	4,108 (0.382)	1,272 (0.282)	934 (0.398)	451 (0.160)
tot. cmp	9,698 (0.902)	3,515 (0.779)	2,342 (0.997)	1,171 (0.415)
ncmp $\leq$ mean	840 (0.078)	688 (0.152)	7 (0.003)	999 (0.354)
ncmp $>$mean	219 (0.020)	311 (0.069)	0 (0.000)	654 (0.232)
tot. ncmp	1,059 (0.098)	999 (0.221)	7 (0.003)	1,653 (0.585)
tot. trees	10,757 (1.000)	4,514 (1.000)	2,349 (1.000)	2,824 (1.000)

TABLE III

A SUMMARY OF EACH RULE'S FUNCTION WRITTEN IN NATURAL LANGUAGE AND PSL ALONG WITH THE DOMAIN KNOWLEDGE IT INCORPORATES. THE RULES ARE GROUPED BY THE INTEGRATION INTO RETINANET. RULES 1, 3, AND 4 HOOK INTO THE NETWORK USING THE CLASSIFICATION HEAD, WHILE RULE 2 USES THE REGRESSION HEAD.

Rule	Natural Language	PSL	Domain Knowledge Incorporated
1	*For all predictions, if the ITC area of a predicted tree is less than the site's average ITC area, it is probably a tree.*	$\forall \hat{y} \left(\mathbf{1}(y = +) \implies bboxSize(A_{bbox}(\hat{t}))_+ \wedge bboxSize(A_{bbox}(\hat{t}))_+ \implies \mathbf{1}(y = +) \right)$	Site mean ITC area
3	*For all predictions, if the ITC area is less than or equal to the area predicted from the H-CA allometry, then it's probably a tree.*	$\forall \hat{y} \left(\mathbf{1}(y = +) \implies bboxHCA(A_{bbox}(\hat{t}))_+ \wedge bboxHCA(A_{bbox}(\hat{t}))_+ \implies \mathbf{1}(y = +) \right)$	CHM, height-crown allometry
4	*For all predictions that match ground truth trees, if the ITC color composition is more than 20% green pixels, it is probably a tree.*	$isTree(\hat{y}) \implies \left(\mathbf{1}(y = +) \implies isGreen(p_g)_+ \wedge isGreen(p_g)_+ \implies \mathbf{1}(y = +) \right)$	Crown color
2	*If the predicted ITC is in competition then its area is probably constrained.*	$\forall (\hat{y}, \hat{t})\ isComp(\hat{y}, \hat{t}) \implies constrained(\hat{t})$	Crown area distribution

TABLE IV

THE NUMBER OF TREES IN THE TEAK TEST SET BROKEN DOWN BY HSV COLOR COMPOSITION. TREES COMPOSED OF MORE THAN 20% GREEN PIXELS ARE CONSIDERED LIVING. FOR RULE 4, TREES WITH LESS THAN 20% GREEN PIXELS ARE CONSIDERED DEAD.

	only green >20%	**only brown >20%**	**both green & brown >20%**	**Total**
Count	589	47	634	734

TABLE V

RULE 1 HYPERPARAMETERS.

Site	**Epochs**	**C**	π_s	μ_{area}	k_{sig}	π_0	α
MLBS	5	1.25e-4	1	2304	0.5	0.88	0.4
NIWO	7	1.25e-4	5	400	0.5	0.95	0.5
SJER	5	1.25e-4	3	5184	0.5	0.80	0.5
TEAK	5	0.01	4	2304	1e-6	0.95	0.5

TABLE VI

RULE 2 HYPERPARAMETERS.

Site	**Epochs**	**C**	π_s	w^*	h^*	l_{lw}	l_{rw}	l_{lh}	l_{rh}	π_0	α
MLBS	5	1	3	48	48	0	82.6	0	85.5	0.95	0.5
NIWO	7	1	3	20	20	0	34.6	0	36.0	0.95	0.5
SJER	5	1	1	72	72	0	146.3	0	149.5	0.95	0.5
TEAK	5	1	1	46	46	0	96.1	0	97.7	0.95	0.5

TABLE VII

RULE 3 HYPERPARAMETERS.

Site	**Epochs**	**C**	π_s	**a**	**b**	k_{sig}	π_0	α
MLBS	5	1.25e-4	1	1.40645	0.33549	0.5	0.88	0.4
NIWO	7	1.24e-4	6	0.87992	0.32658	0.5	0.95	0.9
SJER	5	1.25e-4	3	1.22315	3.32067	0.5	0.80	0.5
TEAK	5	0.01	2	0.83606	1.28339	1e-6	0.80	0.5

TABLE VIII

RULE 4 HYPERPARAMETERS

Site	**Epochs**	**C**	π_s	$\mathbf{t_p}$	**green lower limit**	**green upper limit**	$\mathbf{k_{sig}}$	π_0	α
TEAK	6	1	3	20	(18, 47, 0)	(44, 161, 227)	1	0.6	0.1

TABLE IX

THE AVERAGE CHANGE IN BOUNDING BOX PRECISION, BOUNDING BOX RECALL, AND F1 FOR EACH SITE-RULE COMBINATION WITH 95% CONFIDENCE INTERVALS ($\mu \pm CI$). n VARIED BY SITE AND RULE.

Site	Metric	DF+Harness Rule 1	DF+Harness Rule 2	DF+Harness Rule 3
MLBS	$\boldsymbol{\Delta Bboxprec.}$	3.55 ± 0.23	-0.17 ± 0.23	2.71 ± 0.25
	$\boldsymbol{\Delta Bboxrec.}$	-2.01 ± 0.33	-4.00 ± 0.34	-2.43 ± 0.35
	$\boldsymbol{\Delta F1}$	0.708 ± 0.23	-2.09 ± 0.25	0.11 ± 0.25
	n	300	300	300
NIWO	$\boldsymbol{\Delta Bboxprec.}$	0.34 ± 0.21	0.72 ± 0.22	-0.12 ± 0.27
	$\boldsymbol{\Delta Bboxrec.}$	1.37 ± 0.52	-0.51 ± 0.48	0.79 ± 0.35
	$\boldsymbol{\Delta F1}$	1.01 ± 0.35	-0.03 ± 0.32	0.51 ± 0.26
	n	101	101	101
SJER	$\boldsymbol{\Delta Bboxprec.}$	8.87 ± 1.02	-1.64 ± 1.56	1.96 ± 1.00
	$\boldsymbol{\Delta Bboxrec.}$	-6.32 ± 0.90	-1.17 ± 1.02	-4.42 ± 0.87
	$\boldsymbol{\Delta F1}$	4.01 ± 0.93	-1.31 ± 1.23	-0.61 ± 0.86
	n	301	76	222
TEAK	$\boldsymbol{\Delta Bboxprec.}$	3.37 ± 0.47	2.73 ± 0.44	11.81 ± 0.72
	$\boldsymbol{\Delta Bboxrec.}$	0.61 ± 0.61	1.34 ± 0.61	-4.28 ± 0.93
	$\boldsymbol{\Delta F1}$	2.04 ± 0.38	2.09 ± 0.39	3.12 ± 0.52
	n	101	101	90

TABLE X

THE AVERAGE CHANGE IN DETECTION RATE FOR TREES OF EACH COLOR COMPOSITION WITH 95% CONFIDENCE INTERVALS ($\mu \pm CI$; $n = 101$). TREES WITH >20% GREEN PIXELS AND >20% BROWN PIXELS SHOWED THE LARGEST DECREASE. SEE SECTION IV-D FOR A DESCRIPTION OF EACH CATEGORY.

	Δ % only grn detected	**Δ % only brn detected**	**Δ % brn & grn detected**	**Δ Bbox Prec.**	**Δ Bbox Rec.**	**Δ F1**
DF+R4	-2.5±0.76	-52.1±2.17	-6.2±0.78	8.85±0.57	-9.82±0.90	-1.34±0.52