A Brief Analysis of the Iterative Next Boundary Detection Network for Tree Rings Delineation in Images of Pinus taeda

Henry Marichal[1], Gregory Randall[2]

[1], Facultad de Ingeniería, Universidad de la República, Uruguay
[2] IIE, Facultad de Ingeniería, Universidad de la República, Uruguay

PREPRINT August 27, 2024

Abstract

This work presents the INBD network proposed by Gillert et al. [1] and studies its application for delineating tree rings in RGB images of Pinus taeda cross sections captured by a smartphone (UruDendro dataset), which are images with different characteristics from the ones used to train the method. The INBD network operates in two stages: first, it segments the background, pith, and ring boundaries. In the second stage, the image is transformed into polar coordinates and ring boundaries are iteratively segmented from the pith to the bark. Both stages are based on the U-Net architecture. The method achieves an F-Score of 77.5, a mAR of 0.540, and an ARAND of 0.205 on the evaluation set. The code for the experiments is available at https://github.com/hmarichal93/mlbrief_inbd.

Source Code

The reviewed source code and documentation for this algorithm are available from the web page of this article[1]. Usage instructions are included in the README.txt file of the archive. The authors' original method implementation is available here[2]. This is an MLBriefs article. The source code has not been reviewed!

Keywords: tree-rings, U-Net, cross-section, segmentation

Introduction

The study of tree ring structure (Dendrometry) is important for the study of climate (codified by the width of the annual tree rings, which is related to the amount of water the tree accesses during the year) and for other purposes, such as optimizing manipulation in industrial forestry. The precise

---

[1] https://ipolcore.ipol.im/demo/clientApp/demo.html?id=77777000523&key=FF374F2900F98F32912B65BA925739A4
[2] https://github.com/alexander-g/INBD, commit hash:592176e

Figure 1: Samples from **UruDendro** dataset

delineation of the tree rings is tedious and sometimes very time-consuming. As seen in Figure 1, tracing the rings is perturbed by the presence of knots, geometrical irregularities, cracks, and stains. Recently, some approaches have been developed to automatize this process or at least help the practitioners. No method works fine in general, given the particularities of the species, and the common approach is to manually trace the rings.

In this work, we briefly study INBD, a deep neural network-based approach presented in [1] originally developed for automatically delineating annual rings in microscope images of shrubs. Figure 4 illustrates some examples of those images, part of the **EH** dataset [1]. We train the network to automatically delineate tree rings in cross-section images of Pinus taeda. As seen in Figure 1, these images are of very different resolutions, and the texture is very different due to the species' characteristics. This dataset is named **UruDendro**, and its samples are from trees planted in the northern part of Uruguay [2].

Even if the INBD method was originally designed for microscopy images of shrub cross-sections, the global structure is the same: a pith surrounded by concentric annual growth rings in the disk's center. Given this, we use the original model, trained on the **EH** dataset, and refine the model with the **Urudendro** dataset.

We make two small modifications to the author's implementation. The INBD authors originally trained the model by applying a 4x resize factor to the images, reducing the raw resolution (between 2000 and 3500 pixels wide) to a final resolution between 500 and 875 over the **EH** dataset. The **Urudendro** dataset size images vary between 700 and 3500 pixels in width. We decide to have a fixed input size of 1500 pixels in the largest dimension, regardless of the original sizes of the images to be treated. This is the optimal size found in [3] to process the **UruDendro** dataset, considering the width of the rings. In this way, we don't lose resolution on the thicker rings. Secondly, we substitute the bilinear interpolation used by the authors with a Lanczos one, using the use Pillow[3] library. Therefore, we trained the INBD model on the **EH** dataset (provided by the INBD authors) using the same fixed resolution of 1500 pixels in the largest dimension.

We followed a specific procedure to conduct our experiments[4]. First, we resized the **EH** dataset to 1500 pixels (longest dimension) while maintaining the aspect ratio. Next, we trained the INBD method using the resized **EH** dataset. We then resized the UruDendro dataset to 1500 pixels (longest dimension) and fine-tuned the INBD network on this resized dataset using the pre-trained model with the **EH** dataset. Finally, we evaluated the INBD model using the metrics provided in [1] and [2].

---

[3]Version 10.4.0

[4]The exact commands used are provided in the repository's README
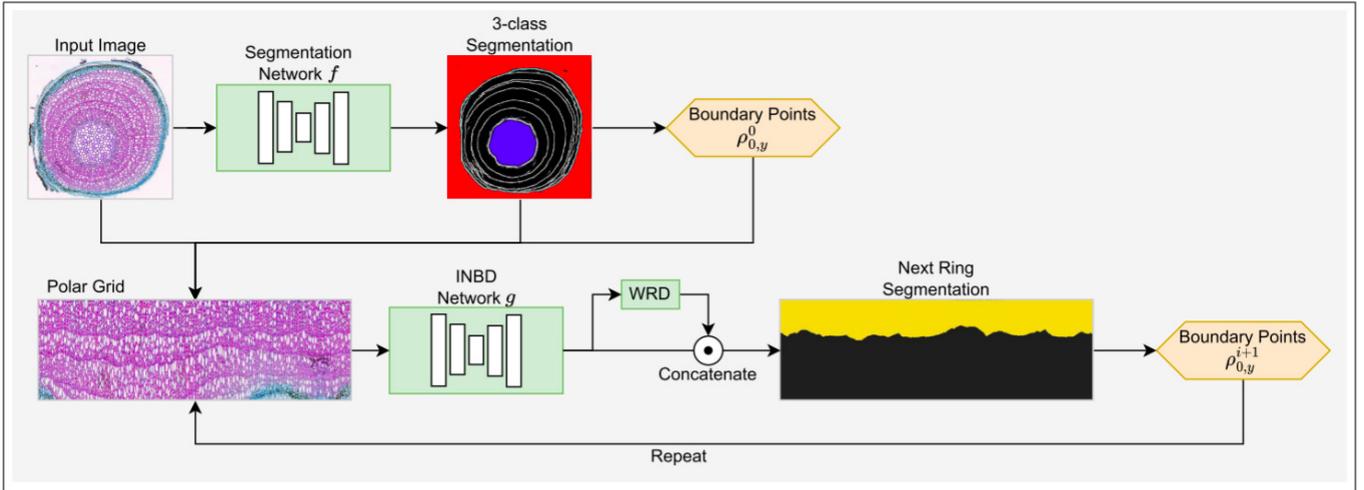
The INBD method

Figure 2: Overview of the INBD pipeline. An input image is first passed through a generic semantic segmentation network that detects 3 classes: background, ring boundaries, and the pith region. A polar grid is sampled starting from the border of the detected pith region (the first ring) and passed to the main INBD network that detects the next ring. This process is repeated until the background is encountered. Image taken from [1].

Figure 2 illustrate the INBD architecture. The input of the network is a cross-section of a tree image. The input image is first segmented into the background, ring boundaries, and pith region. The cross-section image is then transformed into polar coordinates, with the pith's center as the origin. Iteratively rectangular image patches are extracted, allowing individual ring segmentation from the inner one (closer to the pith) to the outer rings (closer to the bark). In the second step of the INBD method, the segmented regions obtained in the first step are used as follows: the pith's center is used as the origin of the polar coordinates transformation, the ring boundaries are used to determine the width of the rectangular image patches and the background is used as a stop criterion in the iterative process. The second step can be interpreted as a refinement of the pixels assigned to the ring boundary category in the first step, plus a transformation of these pixels to the mathematical object curve. Both stages utilize a U-NET network.

Training

The training of this method is made in two steps (Figure 2): first, the **Segmentation Network f** is trained, and then the **INBD Network g**. The training must be done sequentially because the training procedure for the network **g** needs the network **f** (more details can be found in [1]).

To train the INBD model with the UruDendro dataset, we randomly divided it into train and test sets with 40 and 24 images, respectively. The **EH** dataset consists of 82 images with 949 rings. **UruDendro** dataset comprises 64 images and 1123 rings. For training the INBD model, each image is divided into patches determined by successive rings in polar coordinates, ensuring that each patch includes an entire ring. Even though the EH dataset has more images, the UruDendro dataset has more rings per image. Considering the total number of tree rings, both datasets are comparable.

UruDendro image's annotations must be transformed to the format required by the INBD method, identifying four categories in a wood cross-section image: background, pith, rings, and boundary rings.

The INBD method relies on an important hyperparameter: the number of iterations at each epoch (n). We seek the best-performing INBD model by exploring a grid with $n \in \{1, 2, 3, 4\}$. The
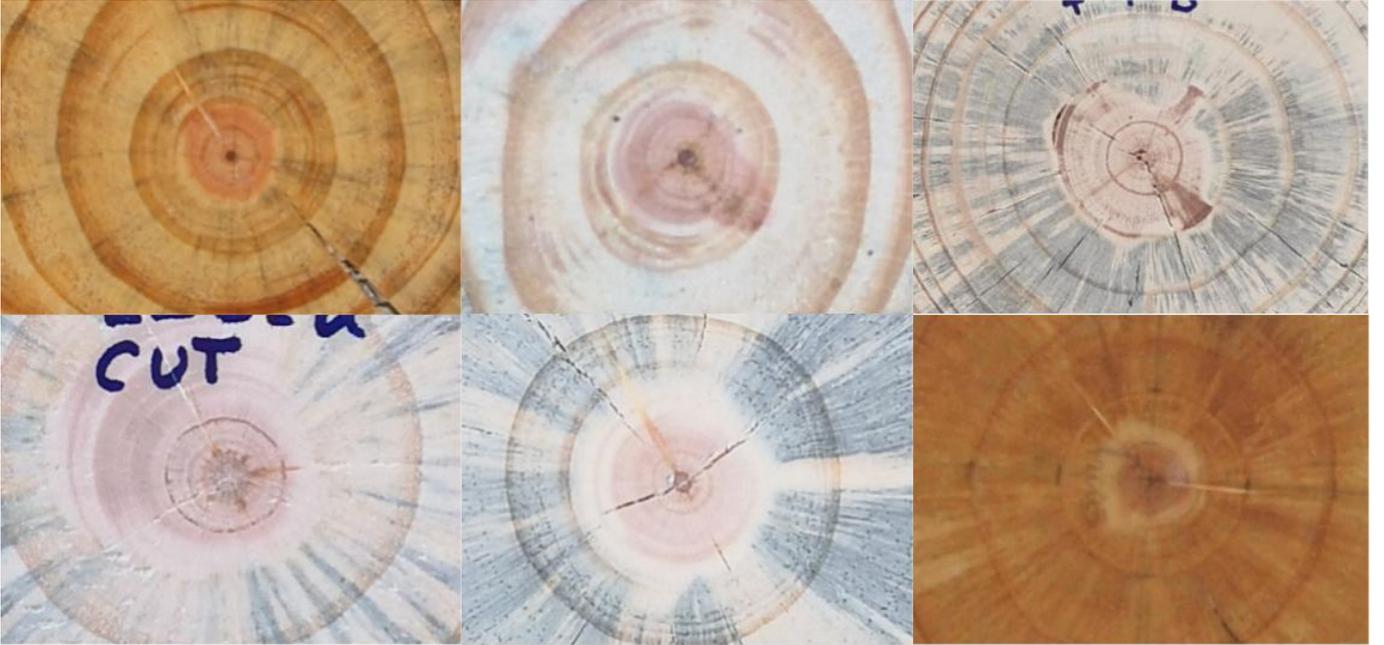
Figure 3: Pith samples from the **UruDendro** dataset. In the Pinus taeda species, the characteristics of the pith present differences within the species.

model that best performs on the **UruDendro** training set was chosen. For training, we utilized the ClusterUy infrastructure, as described in ClusterUy [4], equipped with an Nvidia Tesla P100 GPU with 12GB of RAM. The hyperparameter $n = 3$ yields the best performance on the training set using an input image resolution of 1500 pixels in the longest dimension.

The Segmentation Network f (see Figure 2) didn't learn how to segment the pith for the Pinus taeda species. As shown in Figure 3 the pith characteristics vary greatly within the species. Additionally, the pith size, relative to the sample size, is much smaller in Pinus taeda than in Shrub (**EH** dataset). As seen in Figure 4, the pith size for this species is much bigger than for the samples in the **UruDendro** dataset.

INBD uses a loss that combines the three classes it wants to detect (background, boundaries, and center region). This is done by using a combination of cross-entropy loss and Dice loss, as shown below:

$$L_f = \lambda_1 L_{CE}^{background} + \lambda_2 L_{Dice}^{boundaries} + \lambda_3 L_{CE}^{center} \tag{1}$$

Where $(\lambda_1, \lambda_2, \lambda_3) = (0.01, 1.0, 0.1)$ as given by the authors in [1].

Experiments and Results

This section discusses the inference results over some samples from the test set. The following modifications were made to the code: First, the inference code was updated to accept the pith boundary as an input argument. As shown in Figure 4, the pith size in the **EH** dataset is relatively bigger than in the **UruDendro** dataset. The code was also modified to accept the disk mask as an input, using it as a stopping criterion. As shown in Figure 4, the automatic background detection is not working properly, and the mask solves this problem for the performance evaluation. Finally, a condition was added regarding the patch width in polar coordinates: if the automatically determined width is too small regarding the image dimensions, it is set to one-fourth of the image's dimensions. The next sections discuss the INBD results with the test samples.
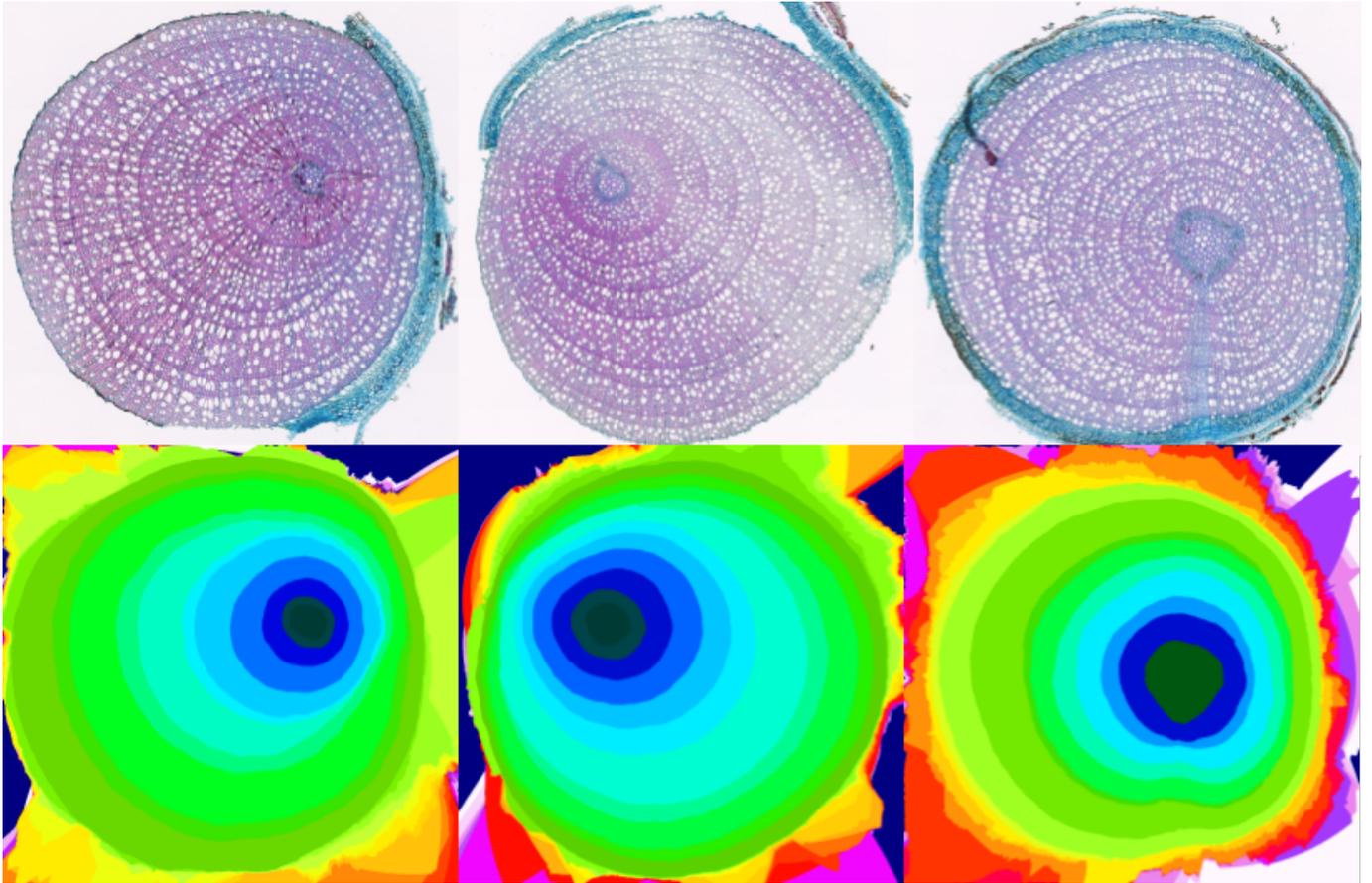
Figure 4: Samples from the INBD **EH** train set. In the first row are the original samples, and in the second are the inference results of the corresponding samples.

Qualitative results

Figure 5 presents three sample disks where the INBD method performs very well. The common aspect of these disks is that the texture characteristics of the center (pith) do not differ from the rest of the disk, allowing the method to propagate smoothly. In the southern direction of the F07d disk, a significant error only begins to appear in the last ring (first row, third column), indicated by a deep blue in the absolute error graph. On the other hand, in the F08b disk, the method also successfully delineates the rings automatically and reasonably well, despite the disk showing a significant presence of fungi (black spots) with a radial pattern. Finally, the method performs very well on the F03c disk, even with a blue stain in the southern region.

As explained in [3], a significant proportion of the detected ring must be close enough to the GT one to assign a detected ring to a given ground truth ring. As the detections are grossly perturbed, this condition doesn't arise, and the detected rings are not assigned to the ground truth ones. In the graphs showing the absolute errors in Figures 5 to 7, each circular band corresponds to a ground true ring, and if left uncolored, it indicates that no detection has been assigned.

Figure 6 presents three additional samples with knots near the pith (F04c and L03c) and cracks (F03e). Despite the presence of knots and cracks, the rest of the disk areas show a high contrast in the annual ring transitions, making these disks not particularly challenging for ring detection. By construction, the INBD method iteratively propagates errors from the inner rings to the outer ones without recovery. In the examples shown in Figure 6, knots near the center disturb the detection and propagate outward errors. This is indicated by the yellow intensity in the absolute error graphs in polar coordinates. Additionally, in disks L03c and F03e, it can be observed that the detections
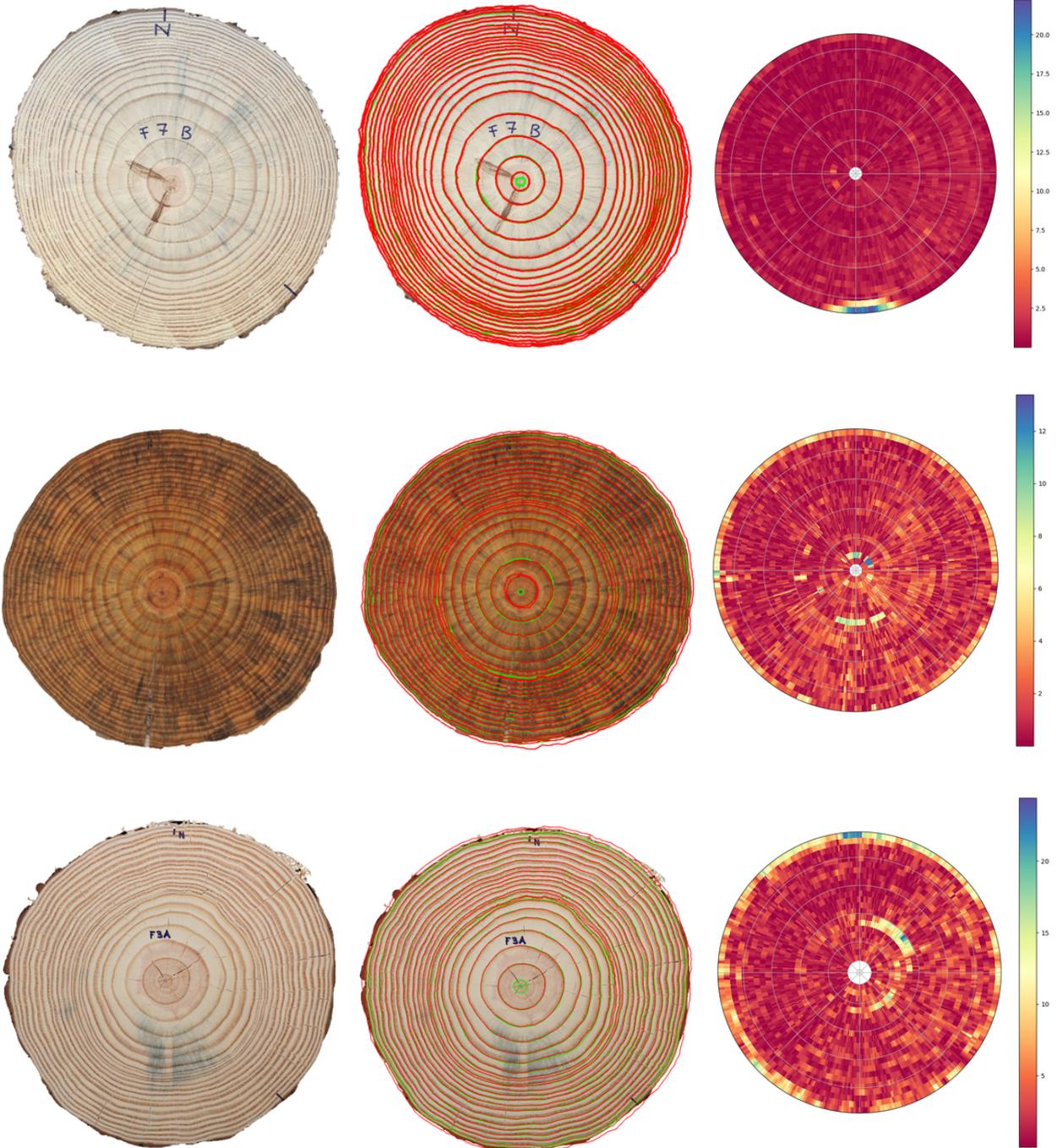
Figure 5: Disks F07d (First row), F08b (Second row) and F03c (Third row) from the **Urudendro** dataset. The INBD method performs exceptionally well on these samples. The first column displays the raw disk. The second column shows the ground truth rings in green and the INBD detections in red. In the third column, the absolute error between the detections and the ground truth rings is visualized, with colors ranging from red (low error) to blue (high error).

for the inner rings are not assigned to any ground true ring, as reflected by the white coloring in the absolute error graph, indicating a lack of assignment. The authors claim that the INBD network can learn to avoid propagation errors during the training stage thanks to its *iterative training* logic. However, this error is not avoided in the **UruDendro** dataset.

Finally, we present three examples where the method performs poorly (Figure 7). These disks exhibit large black stains (L02a and L02b) or a strong presence of knots (L04e) near the pith. While the method manages to detect the initial rings —despite having significant errors compared to the
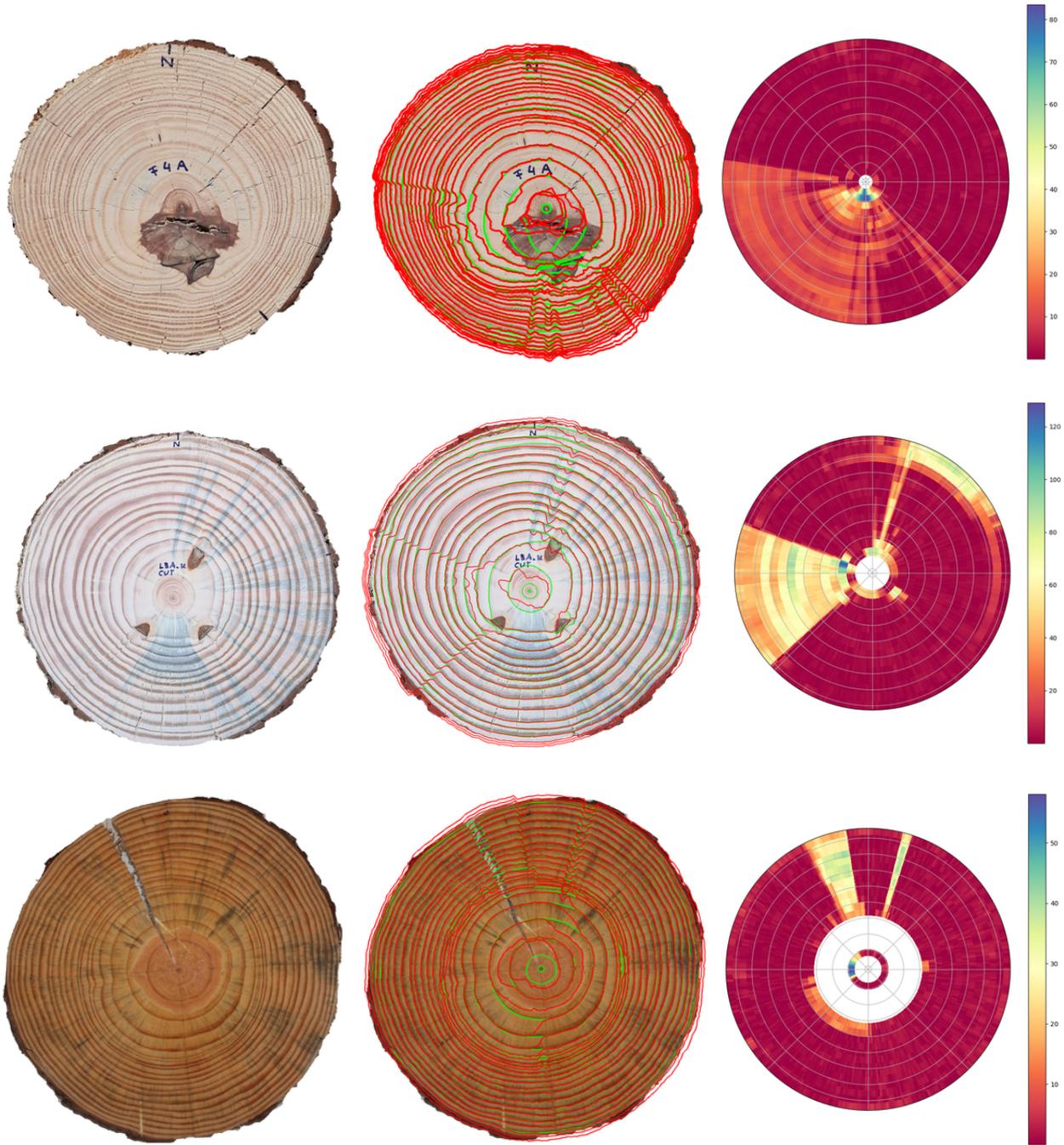
Figure 6: Disks F04c (First row), L03c (Second row), and F03e (Third row). The INBD method struggles to recover from propagation errors originating in the inner rings along certain radial directions. The first column displays the raw disk. The second column shows the ground truth rings in green and the INBD detections in red. In the third column, the absolute error between the detections and the ground truth rings is visualized, with colors ranging from red (low error) to blue (high error).

ground true rings— the subsequent detections for the outer rings are not assigned to any ground true rings, which explains the white coloration in the absolute error graph, indicating no assignment.
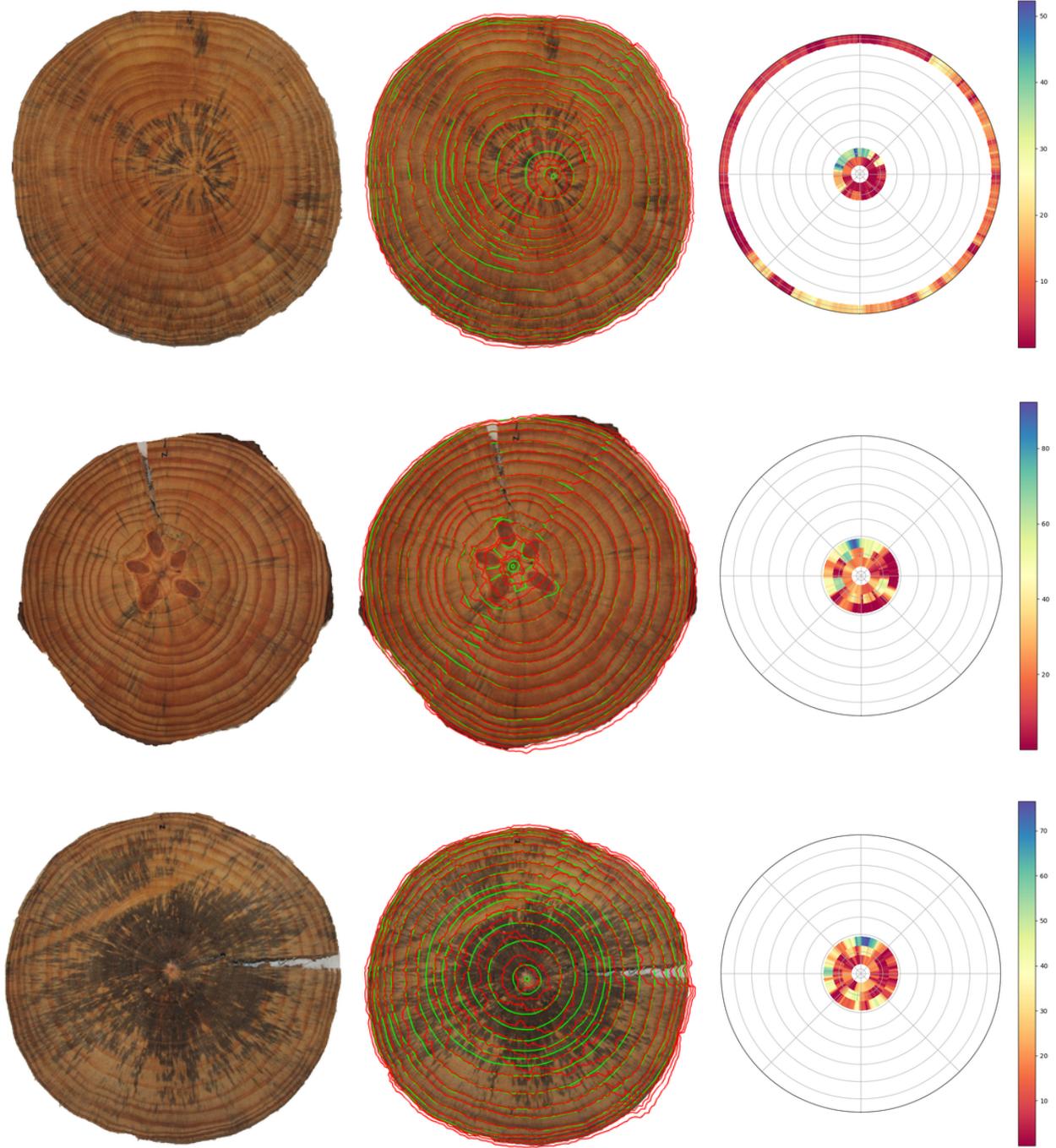
Figure 7: L02a (1), L04e (2) and L02b (3). INBD method cannot recover from an error in the first rings. The first column displays the raw disk. The second column shows the ground truth rings in green and the INBD detections in red. In the third column, the absolute error between the detections and the ground truth rings is visualized, with colors ranging from red (low error) to blue (high error).

Quantitative results

This section presents quantitative metrics to evaluate the performance of the INBD model. Precision (P), Recall (R), and F-score (F) metrics are used. We follow the methodology defined in [2] to determine the assignment of detections to each ground true ring. The mAR and ARAND metrics defined in the INBD article are also utilized. Table 1 lists the disks in the order they appear in the figures. Notably, the F-score decreases in the order of the disks' appearance, which aligns with

the qualitative impression that the disks are ordered from a higher to a lower number of correct detections. Regarding the mAR and ARAND metrics, the low numerical value for the F04c disk stands out, even though, upon visual inspection, the detections do not seem to perform as poorly as the metric suggests.

Disk	P ↑	R ↑	F ↑	mAR ↑	ARAND ↓
F07d	91.3	95.5	93.3	0.805	0.069
F08b	91.7	95.7	93.6	0.817	0.080
F03c	95.7	92.7	93.6	0.713	0.107
F04c	76.9	95.2	85.1	0.105	0.427
L03c	82.4	87.5	84.9	0.257	0.354
F03e	60.9	66.7	63.6	0.362	0.264
L02a	16.7	18.8	17.7	0.044	0.521
L04e	17.7	20.0	18.8	0.0	0.499
L02b	18.8	20.0	19.4	0.020	0.617

Table 1: Metrics computed over the samples shown in this paper.

Table 2 presents the average results for the training and the evaluation set of the **Urudendro** dataset. The different performances between train and test sets are normal and comparable to the observed behavior in the **EH** dataset.

Set	P ↑	R ↑	F ↑	mAR ↑	ARAND ↓
Train	85.9	90.9	88.1	0.639	0.143
Test	75.5	79.7	77.5	0.540	0.205

Table 2: Average metrics computed over the train set (40 samples) and test set (24 samples) of the **UruDendro** dataset.

Table 3 shows the metrics results over the **EH** dataset resized to 1500 pixels in the longer dimension. We trained the INBD network using the resized EH dataset to validate the resized procedure. As can be seen, there appears to be a minor improvement for both metrics (test set) regarding the results reported in the original article, which can be explained by the fact we added the ground truth center boundary as input and the modification in input image resolution.

Set	Our training		Reported by authors	
	mAR ↑	ARAND ↓	mAR ↑	ARAND ↓
Train	0.912	0.041		
Test	0.760	0.107	0.738	0.113

Table 3: Average metrics computed over the EH train set (24 samples) and test set (58 samples) resized to 1500 pixels in the longest dimension. The last two columns show the results reported by the authors for the test set.

Conclusions

We successfully train the INBD network for tree ring delineation in the smartphone's cross-section images of Pinus taeda. Although the results are not perfect, in situations where the disk was properly treated (no fungus presence or cracks) like F03c disk (Figure 5), the method gives a good estimation of the boundaries of the rings once the pith boundary is provided. However, in situations where

the disk presents a knot or stains near the pith, the errors in the initial boundary predictions are propagated to the outer rings as was illustrated in Figure 6 and Figure 7.

Additionally, for the Pinus taeda species, the method fails to detect the pith location automatically.

Image Credits

 Images from [1]

 Images from [2]

References

[1] Alexander Gillert, Giulia Resente, Alba Anadon-Rosell, Martin Wilmking, and Uwe Freiherr von Lukas, *Iterative next boundary detection for instance segmentation of tree rings in microscopy images of shrub cross sections*, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2023, pp. 14540–14548.

[2] Henry Marichal, Diego Passarella, Christine Lucas, Ludmila Profumo, Verónica Casaravilla, María Noel Rocha Galli, Serrana Ambite, and Gregory Randall, *Urudendro, a public dataset of cross-section images of pinus taeda*, 2024.

[3] Henry Marichal, Diego Passarella, and Gregory Randall, *Cs-trd: a cross sections tree ring detection method*, 2023.

[4] Sergio Nesmachnow and Santiago Iturriaga, *Cluster-uy: Collaborative scientific high performance computing in uruguay*, in Supercomputing, Moisés Torres and Jaime Klapp, eds., Cham, 2019, Springer International Publishing, pp. 188–202.