DeepCS-TRD, a Deep Learning-based Cross-Section Tree Ring Detector

Henry Marichal[1], Verónica Casaravilla[2], Candice Power[3], Karolain Mello[2], Joaquín Mazarino[2], Christine Lucas[2], Ludmila Profumo[2], Diego Passarella[2], and Gregory Randall[1]

[1] Facultad de Ingeniería, Universidad de la República, Uruguay
[2] CENUR, Universidad de la República, Uruguay
[3] Ecoinformatics and Biodiversity, Department of Biology, Aarhus University, Denmark

Abstract. Here, we propose Deep CS-TRD, a new automatic algorithm for detecting tree rings in whole cross-sections. It substitutes the edge detection step of CS-TRD by a deep-learning-based approach (U-Net), which allows the application of the method to different image domains: microscopy, scanner or smartphone acquired, and species (*Pinus taeda*, *Gleditsia triachantos* and *Salix glauca*). Additionally, we introduce two publicly available datasets of annotated images to the community. The proposed method outperforms state-of-the-art approaches in macro images (*Pinus taeda* and *Gleditsia triacanthos*) while showing slightly lower performance in microscopy images of *Salix glauca*. To our knowledge, this is the first paper that studies automatic tree ring detection for such different species and acquisition conditions. The dataset and source code are available in `https://hmarichal93.github.io/deepcstrd/`.

Keywords: Tree rings detection · Dendrochronology · Deep learning · U-Net

1 Introduction

Most existing automatic tree ring measurement methods rely on images from cores (small cylindrical samples that capture a transect of the tree's growth rings) rather than entire cross-sections. However, extracting a core is not feasible for very small trees or shrubs, and analyzing entire cross-sections is essential for individuals with suppressed or irregular growth, where there may be wedging rings. Furthermore, modeling tree growth in forestry often requires 2D information, which can only be obtained from analyzing the entire cross-section.

Automatically delineating rings in images of tree cross-sections presents the challenge of generating a pattern of closed curves that accurately represent the ring boundaries, as seen in Figure 1. These rings may exhibit irregular and asymmetric growth patterns, especially in trees and shrubs growing in extreme environments. Moreover, ring boundaries make up only a small portion of the disc, which may also contain perturbations such as cracks, knots, and fungal growth, further increasing the complexity of the task.

Figure 2 outlines the pipeline proposed in this work: a disc image is provided as input, and tree ring traces are generated as output. Once the tree ring curves are delineated, areas and perimeter can be computed, which is essential for forestry studies [7].

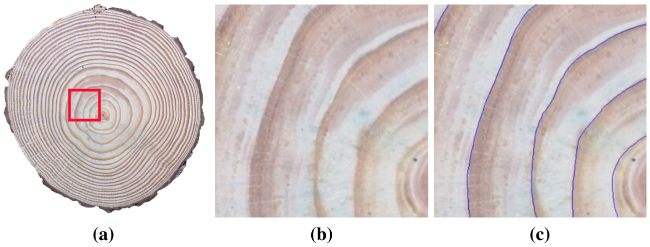


Fig. 1: Tree Ring Detection. Accurate detection of tree ring boundaries in images is critical. (a) Wood cross-section image. (b) Zoomed-in view of the red square in (a). (c) The same view with ring boundaries in blue. The ring thickness is set to 1 pixel.

Marichal et al. [11] proposed an automatic method for tree ring delineation in wood cross-section images called CS-TRD. This method is based on classical edge detection techniques and was applied to two coniferous tree species, *Pinus taeda* and *Abies alba*. Furthermore, they introduced a model for wood cross-section images named *spider web*, see Figure 2c. This model enforces constraints to ensure that tree rings do not intersect and remain roughly concentric.

In this work, the edge detection module from the CS-TRD method is replaced with a U-Net [18] model, a deep learning architecture designed explicitly for semantic segmentation, as illustrated in Figure 2b. U-Net architecture has demonstrated remarkable results in tree ring delineation of cores [2] and in microscopy images of shrubs [4].

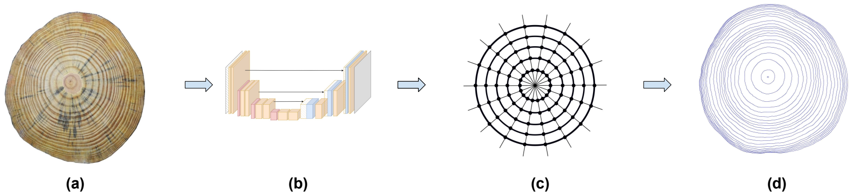
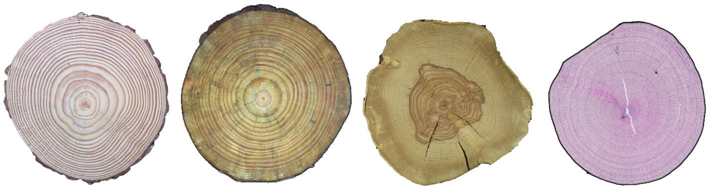


Fig. 2: DeepCS-TRD Tree-Ring detection pipeline. (a) Input disc image. (b) Deep Contour Detector. (c) *Spider web* model. (d) Tree-ring continuous curves.

By replacing the edge detection module in the CS-TRD method with this deep learning-based model, we successfully extended its application to a diverse range of species and growth forms. Notably, this enhanced approach works with angiosperm trees (i.e. *Gleditsia triacanthos*) and shrubs prepared using microscopy (i.e. *Salix glauca*) , samples illustrated in Figure 3. We refer to this new method as DeepCS-TRD.

In addition, we introduce a new image dataset of the *Pinus taeda* species (UruDendro2 [9]), containing 53 samples, and another dataset comprising nine samples of *Gleditsia triacanthos* (UruDendro3a [8]), all with their corresponding expert ring delin-



(a) UruDendro1 **(b)** UruDendro2 **(c)** UruDendro3a **(d)** DiskoIsland

Fig. 3: Examples of the used datasets. Species are (a-b) *Pinus taeda*, (c) *Gleditsia triacanthos*, (d) *Salix glauca*. Acquisition conditions: (a-c) in a laboratory with a smartphone camera, sanded and polished. (d) microsection stained with 1 % safranin and 0.5 % astrablue, permanently fixed to a microscope slide with Eukitt, and scanned at 100 x magnification

.

eations. Generating these types of datasets is a highly time-consuming task that requires the expertise of dendrochronology specialists and is critical for algorithm training.

2 Previous work

Though not yet widely adopted in the tree-ring community, deep learning approaches have become more prevalent in recent years and have naturally been applied to this problem, framing ring detection tasks as a segmentation problem. Two architectures are commonly used: U-Net and Mask R-CNN [17]. Gillert et al. [4] proposed a method for cross-section tree-ring detection called Iterative Next Boundary Detection Network (INBD), based on the U-NET architecture. This method was applied to high-resolution microscopy images of shrub cross-sections, detecting the annual rings individually at each iteration step, from the medulla to the bark. INBD was trained and tested on shrub microscopy images, using standard cross-entropy loss as the main loss for the ring detection network.

Besides the INBD method, most deep-learning-based approaches have been applied to core images. Polek et al. [14] used a Mask R-CNN deep learning architecture to detect rings in cores of coniferous species. During network training, they used mean average precision (mAP) as a loss function. Fabijańska et al. proposed a classic image-processing approach [3] (based on the linking of maximum image gradient pixels) and a convolutional neural network based on a U-Net architecture [2] for detecting tree rings in core images. When comparing both methods, they reported a significant improvement in precision and recall for the deep learning approach over the classic one. The U-Net network was trained to minimize the categorical cross-entropy loss function.

The scarcity of annotated tree ring image datasets remains a significant challenge, as the methods must be tailored to the specific characteristics of each species. Gillert et al. [4] made a dataset available of 213 high-resolution annotated images of cross-sections from three shrub species (*Dryas octopetala*, *Empetrum hermaphroditum*, and *Vaccinium myrtillus*), that were prepared using similar methods to those applied to the *Salix glauca* included in this study (see Figure 3d). Furthermore, the CS-TRD publication included a publicly available dataset of 64 annotated images of *Pinus taeda* (see

Figure 3a). Kennel et al. [5] included a dataset of 7 cross-section images of *Abies alba*, although the ring annotations are not fully accessible. Regarding cores, Polek et al. [14] made available a dataset of 2601 image patches (similar to the red square in Figure 1a) with ring annotations for *Picea abies*, a coniferous tree species.

Among the available methods, only INBD and CS-TRD are adapted for processing full cross-section images and provide accessible code, allowing for comparison with our proposed approach.

3 Approach

Algorithm 1: Deep Contour Detector

```
Input: $Im_{in}$: RGB disc image;
$tile\_size$: Tile size (0 if no patching is applied);
$total\_rotations$: Number of rotations applied to the image;
$cy$, $cx$: x and y coordinates of the pith position;
Output: $l\_ch_s$: list of continuous curve edges;
$l\_nodes_s$: list of nodes of the curves;
$\Omega \leftarrow$ arange$(0, 360, 360/total\_rotations)$; // Define rotation angles
$P \leftarrow$ Zero matrix of the same size as $Im_{in}$; // Initialize the probability map
// Process each rotation angle
for $\theta \in \Omega$ do
    $I_{rotated} \leftarrow$ rotateImage$(Im_{in}, cy, cx, \theta)$; // Rotate the image around the pith position
    $P_{rotated} \leftarrow$ ringSegmentationModel$(I_{rotated}, tile\_size)$; // Apply the segmentation
      model. See Figure 4.
    $P_{aux} \leftarrow$ rotateImage$(P_{rotated}, cy, cx, -\theta)$; // Undo the rotation of the probability map
    $P \leftarrow P + P_{aux}$; // Accumulate the rotated probability map
$P \leftarrow P/total\_rotations$; // Average the probability map
$M \leftarrow P \geq 0.2$; // Threshold the accumulated probability map
$S \leftarrow$ skeletonize$(M)$; // Skeletonize the binary mask
$m\_ch_e \leftarrow$ findContoursCurves$(S)$; // Extract ring curves from the skeleton
$Nx, Ny \leftarrow$ getNormalComponent$(m\_ch_e)$; // Compute normal directions for each edge
$m\_ch_f \leftarrow$ filterEdges$(m\_ch_e, Nx, Ny)$; // Filter edge curves. See Equation (1)
$l\_ch_s, l\_nodes_s \leftarrow$ samplingCurves$(m\_ch_f)$; // Convert curves to the spider web format
return $l\_ch_s, l\_nodes_s$; // Return the results
```

The proposed approach aims to extract a set of continuous, connected pixel curves from an RGB image of a wood cross-section, a disc. These curves serve as input to the *spider web* model presented in [11], enabling the reconstruction of digital tree-ring curves for the sample. An overview of the whole Tree-Ring detection pipeline is in Figure 2.

The CS-TRD method [11] works as follows: (i) Several rays are traced departing from the center (the pith) outwards. (ii) Canny edge filtering is applied to the image. (iii) Filter out edges not belonging to the latewood to earlywood transitions, which correspond to the annual rings, using the angle between the edge normals and the rays. (iv) All edge chains that belong to the same ring are grouped by imposing the *spider web* structure through an iterative procedure that connects chains that do not cross each other, have strong edge information, and comply with a smoothness condition. The authors claim an F-Score of 89% in *Pinus taeda* and 97% in *Abies alba* species.

3.1 Algorithm

The DeepCS-TRD method maintains the principal steps of the CS-TRD but with a substantial modification: replacing the Canny-based edge detection step with a module based on the U-Net architecture [18] trained to segment ring boundaries (Figure 2b). Algorithm 1 shows the pseudocode of this step. The method's inputs are an RGB image of a disc (Im_{in}) and a parameter to indicate the image is split into tiles of (*tile_size*) size, following the overlap-tile strategy described in [18]. Additionally, the method allows to apply a Test-Time Augmentations (TTA) [6] strategy, which consists of *total_rotation* rotations around the pith location (c_y, c_x). The method produces a list of continuous ring edge curves (l_ch_s) and a list of pixels nodes (l_nodes_s) which are the input to the *spyder web* model. In lines 1 and 2, the method initializes the TTA rotation domain angles Ω and the probability map P. Then, between lines 3 and 7, the inference loop iterates between the elements $\theta \in \Omega$. In line 4, Im_{in} is rotated an angle θ and stored in $I_{rotated}$. In line 5, the ring boundaries are generated with the *ring segmentation model* as illustrated in Figure 4. First, if needed, the image is split into tiles with an overlapping of 10% (to avoid the tile border effects). The tiles are zero-padded when necessary to ensure compatibility with the stride size (*tile_size*). Then, a probability map is computed for each tile with a U-Net network. Finally, the probability map for the whole disc ($P_{rotated}$) is built (with the original image size). The average between the tile's probability map is assigned to the overlapping area.

Once the probability map of the whole image ($P_{rotated}$) has been obtained, the rotation of angle θ needs to be inverted (line 6, angle -θ). Then, in line 7, the current probability map (P_{aux}) is accumulated into P. In line 8, the average probability map is obtained. In line 9, the probability map P is binarized with a 0.2 threshold to generate the mask M. This low threshold is deliberately chosen, considering an additional filtering step in line 13 that further refines the results. In line 10, this mask is skeletonized to improve the tree-ring curve precision. Finally, curves are extracted from the skeleton S using the Suzuki method [19] implemented in the function *findCountour* of the OpenCV library [1]. Figure 5 illustrates the lines 8 to 11 steps. Figure 5b shows the probability map P for a disc region. Brighter pixels indicate a higher probability of belonging to a ring. Figure 5c shows the skeleton S overlaid on the mask M, with colors representing curve labels. The skeleton corresponds to the center of the mask. Finally, Figure 5d presents the extracted curves overlaid on the original image.

The normal component of each pixel belonging to a curve in m_ch_e is computed using a straightforward procedure (line 12). The pixels of each curve are ordered sequentially from one end to the other. The tangent direction of pixel p_0 is $\boldsymbol{T_{p_0}} = p_1 - p_{-1}$, where p_{-1} and p_1 are the preceding and succeeding pixels along the curve, respectively. The normal direction $\boldsymbol{N_{p_0}}$ is then computed as a vector perpendicular to $\boldsymbol{T_{p_0}}$, given by $\boldsymbol{N_{p_0}} = \left(-T_{p_0,y}, T_{p_0,x}\right)$, where $\boldsymbol{T_{p_0}} = (T_{p_0,x}, T_{p_0,y})$. This ensures that $\boldsymbol{N_{p_0}}$ is orthogonal to

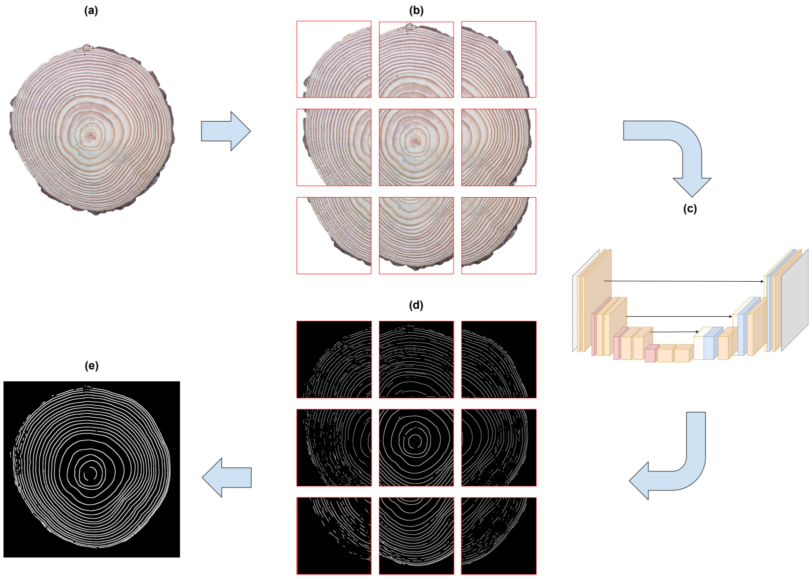


Fig. 4: Ring Segmentation Model Inference. The input to the model is a complete cross-section disc (a). (b) The disc is divided into square tiles. (c) and (d) Each tile is processed individually using the U-Net network to generate its corresponding probability map. Finally, the tile predictions are combined to reconstruct the probability map for the entire sample (e). This procedure corresponds to line 5 of Algorithm 1.

the curve at p_0. The point c is the pith location (c_x, c_y). In line 13, the angle δ between $\boldsymbol{cp_0}$ and $\boldsymbol{N_{p_0}}$ is computed as $\delta\left(\boldsymbol{cp_0}, \boldsymbol{N_{p_0}}\right) = \arccos\left(\frac{\boldsymbol{cp_0} \cdot \boldsymbol{N_{p_0}}}{\|\boldsymbol{cp_0}\| \|\boldsymbol{N_{p_0}}\|}\right)$

We filter out all *pixels* p_0 with normals not collinear and outbound oriented concerning the ray's direction at that point:

$$\alpha \leq \delta(\boldsymbol{cp_0}, \boldsymbol{N_{p_0}}) \leq 180 - \alpha \tag{1}$$

We set $\alpha = 45$ degrees. When a pixel p_0 is removed, the curve to which p_0 belongs is split at that position. This step is straightforward because m_ch_e is a matrix of dimensions $N \times 2$. Each curve is separated from the others by a row with values $(-1, -1)$. Therefore, if a pixel does not satisfy the condition in Equation (1), we insert the vector $(-1, -1)$ at its position.

Finally, in line 14, the curves in m_ch_f are converted into the format required by the *spider web* model, structured around a central *pith*, the origin of the *rays*. Each *curve* is represented as a set of points, with *nodes* in the intersections of the curve with the *rays*. The resulting sampled *curve* is called a *chain*. These *chains* have the notable property of being non-intersecting, inspired by the biological characteristics of tree rings observed in conifer species. More details can be found in [11].

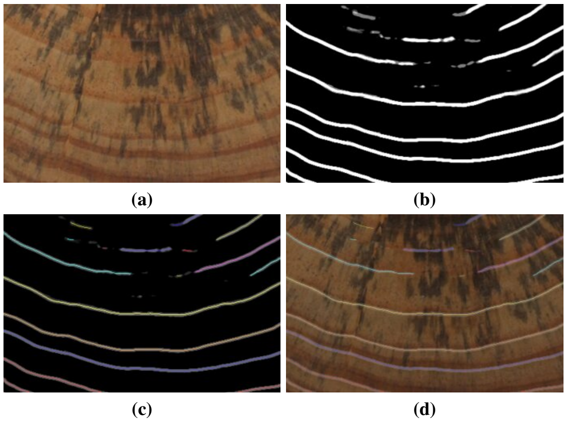


Fig. 5: Stages for extracting the ring edge curves. (a) Zoomed-in view of the RGB image. (b) The probability map of the image section is shown in (a). (c) Probability map with overlaid edge curves. (d) Extracted edge curves overlaid on the image.

Preprocessing The background was removed from all images using the U^2-Net network [16] and manually corrected when necessary. Furthermore, portions of the image background were cropped to focus on the cross-section so that the minimum distance between the disc and the image edges is 50 pixels. Images from all datasets were resized to 1504x1504 pixels using Lanczos interpolation. To complete the image size, if the smallest dimension of the resized image does not reach the new dimension, a 255-value padding was applied, maintaining the aspect ratio.

3.2 Network Training

Non-overlapping square tiles were used for training the U-Net network. Only the patches with ring presence of each image sample were used. The tile size is a hyperparameter of the training step (see Section 5.2) where 60% of the samples were used for training, 20% for validation and 20% for assessing the full tree-ring detection pipeline (Figure 2). Training and validation subsets were split into patches to train and select the network's hyperparameters. We also trained the network to predict the ring boundaries on the complete disc sample.

The network was trained over 100 epochs on an NVIDIA P100 GPU with 12 GB of VRAM and 40 GB of system RAM. The Adam optimizer was used with an initial learning rate of 1e-3, while a cosine annealing schedule was applied to adjust the learning rate dynamically during training. The Dice Loss function was employed to optimize the model's performance. The Resnet18 backbone, pre-trained on ImageNet, was also used to enhance feature extraction. The network's weights that result in the lowest validation loss during training are selected as the optimal model.

Regarding data augmentation, in 50% of the dataset images, randomly selected, we generate three augmented images: one with a random rotation around the pith, another with occlusions simulating cracks or fungi, and a third with elastic deformations.

4 Datasets

The datasets used in this study encompass a variety of species as shown in Figure 3. Some species, like the conifer *Pinus taeda*, have clear ring contrasts, while *Gleditsia triacanthos* and *Salix glauca* are more challenging. Due to the differences in growth forms, the sample preparation methods also varied. While the cross sections from the trees in datasets UruDendro 1, 2, and 3, were sanded and polished as typical for tree samples, the cross-sections in the DiskoIsland dataset were prepared using standard protocols for shrubs. Specifically, 12-15 μm thick cross-sections were cut, stained with 1% safranin and 0.5% astrablue, and permanently fixed to microscope slides using Eukitt (BiOptica, Milan). The image acquisition methods also varied. The images from the UroDendro 1, 2, and 3 datasets were captured with smartphone cameras (iPhone 6s and Huawei P20 Pro), while the images of the microsections from the DiskoIsland dataset were captured at 100x magnification using a Zeiss slide scanner (Axio Scan Z1, Zeiss, Germany (2.26 pixel/μm)).

Ring boundaries in the selected acquired images were annotated using the Labelme Tool [20], which allows users to mark polylines over the image and export the annotations in JSON format. Ring annotations were performed by users with different levels of expertise, from undergraduate students to expert professors, and in all cases, they were reviewed by at least one other expert. Finally, a binary image mask was generated from the ring boundary annotations, with a thickness of 3 pixels. Table 1 depicts the datasets, the number of samples and rings in each one, the species, and the acquisition method.

Table 1: Dataset description.

Collection	Samples	Tree-rings	Species	Acquisition
UruDendro1 [10]	64	1221	*Pinus taeda*	Smartphone
UruDendro2 [9]	53	1151	*Pinus taeda*	Smartphone
UruDendro3a [8]	9	216	*Gleditsia triacanthos*	Smartphone
DiskoIsland [15]	50	654	*Salix glauca*	Scanner

5 Results and discussion

5.1 Metrics

We adopt the same evaluation metrics used by the authors of the INBD method: the mean Average Recall (mAR) and the Adapted Rand error (ARAND). False positives are preferred to false negatives because an incorrect ring can be deleted with just one click, while tracing a new one is more time-consuming. In this sense, the mAR metric is particularly suitable for this application.

5.2 Experiments

The DeepCS-TRD method involved two key parameters: *tile_size* and *total_rotations* (see Algorithm 1). We tested tile sizes of 64, 128, 256, 512, and 1504 (full resolution, no tiles) pixels for all the datasets. Figure 1a illustrates a tile size of 256px. For the *total_rotations* parameter, combinations of 0, 3, and 5 rotations were evaluated. The selection of this parameter was made by optimizing the validation subset. After experiments, parameter values were fixed in $tile\_size = 256$ and $total\_rotations = 5$ for the four datasets.

We noted that the thickness of the annotated ring boundary masks during training impacts the method's performance. The best value was 3-pixel in all datasets. In consequence, DeepCS-TRD and INBD methods are trained using ring boundary thicknesses of three pixels. The two networks in the INBD method were trained following the author's procedure during 100 epochs, setting the downsampling parameter to 1, which gave the best results.

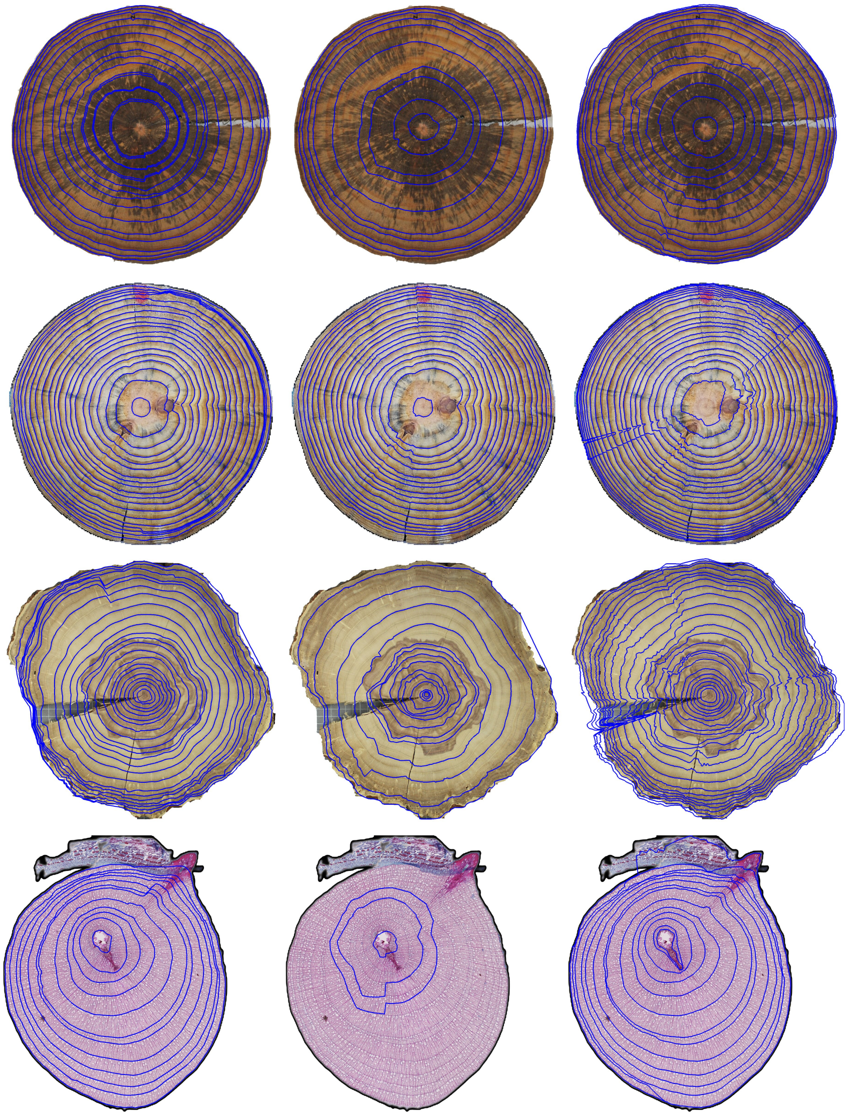
Table 2: Results on the test set for the four used datasets and the available SOTA methods: INBD [4] and CS-TRD [11] as well as the proposed DeepCS-TRD. The best performance is in boldface. Uru1 stands for UruDendro1, and the same is done for UruDendro2 and UruDendro3a datasets. Disko stands for DiskIsland dataset.

	mAR ↑				**ARAND ↓**			
Method	**Uru1**	**Uru2**	**Uru3a**	**Disko**	**Uru1**	**Uru2**	**Uru3a**	**Disko**
CS-TRD	.787	.710	.007	.026	.093	.144	.466	.634
INBD	.846	.742	.200	**.735**	.081	.132	.494	**.099**
DeepCS-TRD	**.884**	**.809**	**.620**	.628	**.053**	**.105**	**.207**	.107

5.3 Results

This section evaluates the performance of INBD, CS-TRD, and DeepCS-TRD on the test set across all datasets. The INBD method was modified to accept the pith boundary ground truth as input to ensure a fair comparison, as both DeepCS-TRD and CS-TRD take the pith locations as input but modeled as a pixel [12].

Table 2 presents each method's mAR and ARAND values across all datasets. The proposed DeepCS-TRD method has the best results on all the UruDendro datasets. It achieves outstanding results with near-perfect ring detection from the pith to the bark in UruDendro1 and UruDendro2, as illustrated in Figure 6. Despite the INBD method giving very good performance in most of the samples, in some situations, it generates notorious propagation errors in the presence of significant fungal growth, cracks, or knots because it begins delineating ring boundaries from the pith center (see rows 1 and 2). In row 1's sample, DeepCS-TRD detects some false rings near the center, but its remaining detections are highly accurate. In contrast, CS-TRD gives good results, but rings are inaccurate with this strong fungus presence. In the sample in row 2, both DeepCS-TRD and CS-TRD performed well.



(a) DeepCS-TRD **(b)** CS-TRD **(c)** INBD

Fig. 6: Tree-Ring Delineation Results. Each column displays the ring boundaries produced by each method, shown in blue. Each row corresponds to a different disc sample. Column (a) DeepCS-TRD; Column (b) CS-TRD; Column (c) INBD. Note the presence of knots, cracks, fungus, and the differences between species. Note that the darker region in the third row does not correspond to the tree rings of that species.

In the UruDendro3a dataset, DeepCS-TRD performs well; in the sample shown in Figure 6, row 3, it correctly delineates 15 out of 19 rings. In contrast, the INBD method detects only six rings. CS-TRD detects five rings but interestingly identifies the three innermost rings missed by the other methods (see the region around the pith in row 3 for all methods).

In the DiskoIsland dataset, DeepCS-TRD performs slightly worse than the INBD method on average. In the sample in row 4, the INBD method correctly detects all the rings, while DeepCS-TRD missed the second ring. CS-TRD performs poorly in this dataset.

There are differences regarding the training time for the DeepCS-TRD and INBD methods. Both methods' maximum required training time was in the UruDendro1 dataset, with 25 and 4 hours for the INBD and DeepCS-TRD, respectively, using the same HW.

6 Conclusions

A new method for automatic Tree-Ring delineation has been proposed. It adapts the CS-TRD method by replacing the edge detector step with a U-Net deep convolutional network. This modification allows us to apply the *spider web* model to other species, such as *Gleditsia triachantos* or *Salix glauca*, with different, and often more difficult, annual ring patterns than for the coniferous species. Additionally, the proposed method also increases the performance of CS-TRD for *Pinus taeda* as can be seen in Table 2; in this case, the accuracy (mAR) has improved from 0.787 to 0.884 in UruDendro1 and from 0.710 to 0.809 in UruDendro2.

By applying data augmentation techniques and the overlap-tile strategy, a high ring detection rate has been achieved in UruDendro3a despite using a limited number of images. Additionally, satisfactory results have been obtained on the *Salix glauca* dataset. One advantage of this method over INBD is that INBD requires two training steps, whereas DeepCS-TRD requires one. Moreover, DeepCS-TRD significantly reduces training time, requiring just 4 hours compared to 25 hours for INBD on the UruDendro1 dataset.

7 Acknowledgments

The experiments presented in this paper used ClusterUY [13] (site: https://cluster.uy). This work was supported by project ANIIFMV176061. CCP was supported by IRFD (702700133B) and H2020 CHARTER (869471).

References

1. G. Bradski, *The OpenCV Library*. Dr. Dobb's Journal of Software Tools (2000).
2. A. Fabijańska and M. Danek, *DeepDendro – A tree rings detector based on a deep convolutional neural network*, Computers and Electronics in Agriculture, 150 (2018), p. 353–363.
3. A. Fabijańska, M. Danek, J. Barniak, and A. Piórkowski, *Towards automatic tree rings detection in images of scanned wood samples*, Computers and Electronics in Agriculture, 140 (2017), pp. 279–289.

4. A. Gillert, G. Resente, A. Anadon-Rosell, M. Wilmking, and U. F. Von Lukas, *Iterative next boundary detection for instance segmentation of tree rings in microscopy images of shrub cross sections*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023, pp. 14540–14548.
5. P. Kennel, P. Borianne, and G. Subsol, *An automated method for tree-ring delineation based on active contours guided by dt-cwt complex coefficients in photographic images: Application to abies alba wood slice images*, Computers and Electronics in Agriculture, 118 (2015), pp. 204–214.
6. M. Kimura, *Understanding Test-Time Augmentation*, Springer International Publishing, 2021, pp. 558–569.
7. J.-P. Labrecque-Foy, S. Angers-Blondin, P. Ropars, M. Simard, and S. Boudreau, *The Use of Basal Area Increment to Preserve the Multi-Decadal Climatic Signal in Shrub Growth Ring Chronologies: A Case Study of Betula glandulosa in a Rapidly Warming Environment. Atmosphere*, 14 (2013).
8. H. Marichal, D. Passarella, C. Lucas, V. Casaravilla, K. Mello, L. Profumo, and G. Randall, *UruDendro3a, a public dataset of 9 cross-section images and manual annual ring delineations of Gleditsia triachantos L.*., 2025. V1. [Dataset]. https://doi.org/10.5281/zenodo.15652667.
9. H. Marichal, D. Passarella, C. Lucas, V. Casaravilla, L. Profumo, and G. Randall, *UruDendro2, a public dataset of 53 cross-section images and manual annual ring delineations of Pinus taeda L.*., 2025. V1. [Dataset]. https://doi.org/10.5281/zenodo.15652451.
10. H. Marichal, D. Passarella, C. Lucas, L. Profumo, V. Casaravilla, M. N. R. Galli, S. Ambite, and G. Randall, *Urudendro, a public dataset of 64 cross-section images and manual annual ring delineations of Pinus taeda L.*, Annals of Forest Science, 82 (2025), p. 25.
11. H. Marichal, D. Passarella, and G. Randall, *CS-TRD: a Cross Sections Tree Ring Detection method*. Preprint on arXiv: 2305.10809 (2023).
12. H. Marichal, D. Passarella, and G. Randall, *Automatic wood pith detector: Local orientation estimation and robust accumulation*, in International Conference on Pattern Recognition (ICPR), 2025, pp. 1–15.
13. S. Nesmachnow and S. Iturriaga, *Cluster-uy: Collaborative scientific high performance computing in uruguay*, in Supercomputing, M. Torres and J. Klapp, eds., Cham, 2019, Springer International Publishing, pp. 188–202.
14. M. Poláček, A. Arizpe, P. Hüther, L. Weidlich, S. Steindl, and K. Swarts, *Automation of tree-ring detection and measurements using deep learning*, Methods in Ecology and Evolution, 14 (2023), p. 2233–2242.
15. C. C. Power, H. Marichal, U. A. Treier, G. Resente, S. Normand, and G. Randall, *Dataset of 50 cross-section images and manual annual ring delineations of salix glauca from western greenland*, 2025. V1. [Dataset]. https://doi.org/10.5281/zenodo.15311335.
16. X. Qin, Z. Zhang, C. Huang, M. Dehghan, O. R. Zaiane, and M. Jagersand, *U2-Net: Going deeper with nested U-structure for salient object detection*, Pattern Recognition, 106 (2020), p. 107404.
17. L. Rettenberger, F. R. Münke, R. Bruch, and M. Reischl, *Mask R-CNN Outperforms U-Net in Instance Segmentation for Overlapping Cells*, Current Directions in Biomedical Engineering, 9 (2023), pp. 335–338.
18. O. Ronneberger, P. Fischer, and T. Brox, *U-Net: Convolutional Networks for Biomedical Image Segmentation*, in Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, Cham, 2015, Springer International Publishing, pp. 234–241.
19. S. Suzuki and K. be, *Topological structural analysis of digitized binary images by border following*, Computer Vision, Graphics, and Image Processing, 30 (1985), pp. 32–46.
20. K. Wada, *Labelme: Image polygonal annotation with python*. Zenodo, 2024.