

Article

Induction of Convolutional Decision Trees with Success-History-Based Adaptive Differential Evolution for Semantic Segmentation [†]

Adriana-Laura López-Lobato , Héctor-Gabriel Acosta-Mesa *  and Efrén Mezura-Montes 

Artificial Intelligence Research Institute, Universidad Veracruzana, Campus Sur, Calle Paseo Lote II, Sección Segunda No. 112, Nuevo Xalapa, Veracruz 91097, Mexico; adrilau17@gmail.com (A.-L.L.-L.); emezura@uv.mx (E.M.-M.)

* Correspondence: heacosta@uv.mx

[†] This paper is an extended version of our paper published in *New Trends in Computational Intelligence and Applications* 2023.

Abstract: Semantic segmentation is an essential process in computer vision that allows users to differentiate objects of interest from the background of an image by assigning labels to the image pixels. While Convolutional Neural Networks have been widely used to solve the image segmentation problem, simpler approaches have recently been explored, especially in fields where explainability is essential, such as medicine. A Convolutional Decision Tree (CDT) is a machine learning model for image segmentation. Its graphical structure and simplicity make it easy to interpret, as it clearly shows how pixels in an image are classified in an image segmentation task. This paper proposes new approaches for inducing a CDT to solve the image segmentation problem using SHADE. This adaptive differential evolution algorithm uses a historical memory of successful parameters to guide the optimization process. Experiments were performed using the Weizmann Horse dataset and Blood detection in dark-field microscopy images to compare the proposals in this article with previous results obtained through the traditional differential evolution process.

Keywords: semantic segmentation; image segmentation; convolutional decision tree; differential evolution; SHADE



Citation: López-Lobato, A.-L.; Acosta-Mesa, H.-G.; Mezura-Montes, E. Induction of Convolutional Decision Trees with Success-History-Based Adaptive Differential Evolution for Semantic Segmentation. *Math. Comput. Appl.* **2024**, *29*, 48. <https://doi.org/10.3390/mca29040048>

Received: 28 May 2024
Revised: 22 June 2024
Accepted: 24 June 2024
Published: 27 June 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Semantic segmentation is a relevant process in numerous scientific fields for image analysis. It involves assigning labels to the pixels of an image to distinguish objects of interest from the image background; see Figure 1.

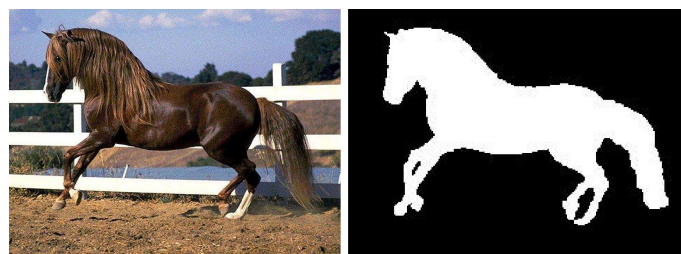


Figure 1. Semantic segmentation technique applied to horse detection.

Several methods have been used to solve the image segmentation problem [1,2], but Convolutional Neural Networks (CNNs) are currently the most popular [3]. CNNs produce powerful results; however, they are often called “black boxes” because the process they follow to produce results can be difficult to understand and explain. Therefore, their use in critical contexts, such as the medical field, needs to be thoroughly studied [4,5].

Convolutional Decision Trees (CDTs) are an alternative to CNNs because they have the graphical structure of a decision tree model that is easy to interpret [6]. Several methods have been proposed to solve optimization problems in the CDT induction process. The original one, proposed in [6], maximizes the information gain function in each tree node through an analytical optimization process. This process is a local search that partitions the data (pixels) to obtain a single CDT.

Since the classical method for CDT induction is a greedy search, another method uses the differential evolution (DE) algorithm to induce a CDT with a global search to improve performance [7]. DE is one of the most popular metaheuristic search strategies for solving optimization problems; it incorporates stochastic elements and parameters that enhance its ability to explore the problem domain, even when the parameters must be adapted to the specific problem [8]. The method in [7] performs a global search to induce a CDT using the DE algorithm to maximize the F1-score. In this proposal, each individual in the DE algorithm represents all the kernels of the CDT, so a set of CDTs is obtained to select the one with the best F1-score.

The latest method, called DE-CDT-BKS [9], performs a local search using the DE algorithm to identify the optimal convolution kernel size and the convolution kernel of each node of the CDT to split the data (pixels) into two sets, using the F1-score as the fitness function. In this method, the user provides a list of kernel sizes. After applying the learning process with these kernel sizes, the kernel with the best F1-score at each partition node is selected. This results in a single CDT with convolutional kernels of different sizes.

The methods proposed in [6,7,9] produce CDTs with more explanatory structure than a CNN and transparency in the image segmentation process. However, the first method requires a post-training process called graph cutting to improve the results [6]. The first method produces better results than the other two but does not allow modifications to the objective function. In contrast, in the other two methods the objective function can be modified due to the nature of the DE algorithm, although they produce lower results [7,9]. Furthermore, in the first two methods, the CDT structure only considers kernels of the same size [6,7].

To overcome this, the present work proposes the use of SHADE [10] instead of the traditional DE algorithm in the second and third methods to achieve superior segmentation results. As an adaptive DE algorithm, SHADE uses a historical memory of successful parameters to guide the optimization process. This article compares these techniques with previous results obtained using the traditional differential evolution process [7,9]. The comparison is based on the segmentation of two sets of images: the “Weizmann Horse dataset” and the “Blood detection in dark-field microscopy images”. The selection of these databases makes it possible to compare the results with those described in [6,7,9].

The remaining paper is structured into three sections. Section 2 describes the DE algorithm and highlights the most relevant characteristics of SHADE. This section also presents the details of the two CDT induction procedures with SHADE. Section 3 is dedicated to the experiments and the results obtained. Finally, Section 4 offers detailed conclusions and future work.

2. Materials and Methods

This section covers the two main subjects of the project: the differential evolution algorithm and SHADE. A description of Convolutional Decision Trees (CDTs) and the methodologies proposed for CDT induction using SHADE are also presented.

2.1. Differential Evolution Algorithm and SHADE

2.1.1. Differential Evolution Algorithm

The Differential Evolution (DE) algorithm is a metaheuristic search strategy for solving optimization problems [11–13]. It works with a population that represents potential solutions to the problem and develops it by recombining solutions within it to generate a new population (offspring). The DE algorithm involves three operators: mutation, crossover,

and selection. It also involves user-defined parameters that guide the search: the scaling factor F , the crossover rate CR , the population size NP , and the number of generations NG .

The standard DE process, known as DE/rand/1/bin, uses the following operators to generate a *trial vector* u_i for each *target vector* x_i in the population:

- **Mutation operator:** To generate the *trial vector* u_i , the mutation operator computes a *noise vector* v_i using Equation (1). The vectors x_{r_0} , x_{r_1} , and x_{r_2} are randomly selected from the population and are different from each other and from x_i . F is a factor that scales the difference between the vectors x_{r_1} and x_{r_2} . This process is equivalent to stepping towards a new point in the search space.

$$v_i = x_{r_0} + F(x_{r_1} - x_{r_2}) \quad (1)$$

- **Crossover operator:** The crossover operator merges the information of the *noise vector* v_i with that of the *target vector* x_i to generate the *trial vector* u_i , component by component, using the function in Equation (2). This equation involves a randomly generated number, $rand_j$, within the range $[0, 1]$ for each vectorial component. To ensure that the *trial vector* takes at least one component from the *noise vector* v_i , a randomly selected position j , denoted as J_{rand} , is used. The process is controlled by the crossover rate CR .

$$u_{ij} = \begin{cases} v_{ij}, & \text{if } (rand_j \leq CR) \text{ or } (j = J_{rand}); j = 1, \dots, |x_i| \\ x_{ij}, & \text{otherwise.} \end{cases} \quad (2)$$

- **Selection operator:** The selection operator adds the vector with the highest fitness value between x_i and u_i to the population for the next generation. This operator guarantees that the best solution is preserved throughout the iterations, thanks to the property of elitism.

As mentioned before, this process is the classic version of differential evolution, called DE/rand/1/bin [14], where “rand” means that the vectors are chosen randomly in the mutation operator, “1” means that only one difference vector is used to form the *noise vector*, and the term “bin” (binomial distribution) means that a uniform crossover is used when creating the *test vector*. However, several variants of the DE algorithm consider different ways of generating the *noise vector* v_i in the mutation operator [15]. Some of these variants use the individual with the best fit x_{best} or use more than three individuals from the population by including more than one scaled difference. Some examples of these variants are:

- DE/rand/2

$$v_i = x_{r_0} + F(x_{r_1} - x_{r_2}) + F(x_{r_3} - x_{r_4}), \quad (3)$$

- DE/best/1

$$v_i = x_{best} + F(x_{r_0} - x_{r_1}), \quad (4)$$

- DE/best/2

$$v_i = x_{best} + F(x_{r_0} - x_{r_1}) + F(x_{r_2} - x_{r_3}). \quad (5)$$

It is known that the values for the parameters CR , F , NP , and NG are problem-dependent and impact the performance of the DE algorithm [15], so it is necessary to tune them to obtain good results when applied to real-world problems. Several self-adaptive mechanisms to adjust these parameters have been studied, such as JADE [16], SHADE [10], and L-SHADE [17].

This research considers the algorithm Success-History-based Adaptive Differential Evolution (SHADE), which uses a historical memory of successful parameters CR and F to readjust their values in the search process. The most relevant characteristics of SHADE are mentioned below.

2.1.2. SHADE

SHADE is a technique that regulates the DE algorithm parameter values CR and F through an adaptive parameter control mechanism. It is well established that the values of these parameters depend on the specific real-world problem considered and directly affect the model’s performance [15]. Therefore, the SHADE technique allows the initial values of the parameters to change and adapt during the search process to achieve the desired results [10].

This approach uses the mutation strategy called DE/current-to-pbest/1, shown in Equation (6), where $x_{pbest}^{(G)}$ is an individual randomly selected from the top $NP * p$ individuals in the population of generation G, where $p \in (0, 1]$. $x_{r_0}^{(G)}$ and $x_{r_1}^{(G)}$ are vectors randomly selected from the population of generation G, and $F_i^{(G)}$ is the F parameter used by individual $x_i^{(G)}$.

$$v_i^{(G)} = x_i^{(G)} + F_i^{(G)}(x_{pbest}^{(G)} - x_i^{(G)}) + F_i^{(G)}(x_{r_0}^{(G)} - x_{r_1}^{(G)}). \tag{6}$$

The SHADE algorithm employs an archive A that is initially empty. The *target vectors* $x_i^{(G)}$ that lose against its *trial vector* in the selection operator are added to the archive during the evolution process. This process results in the random selection of the vector $x_{r_1}^{(G)}$ from the union of the individuals in the population and the vectors in A. When the size of set A exceeds the maximum size, which is typically equal to NP, randomly selected elements are eliminated from A to maintain the size.

To calculate the values of the parameters $CR_i^{(G)}$ and $F_i^{(G)}$, two memories of size H are used, denoted as M_{CR} and M_F . They store the mean values of the successful parameters of previous generations, understanding as successful parameters those $CR_i^{(G)}$ and $F_i^{(G)}$ values that generate a *trial vector* that defeats its associated *target vector*. Initially, all values in the M_{CR} and M_F memories are set to 0.5. In each generation G, the parameter values $CR_i^{(G)}$ and $F_i^{(G)}$ are calculated for each individual $x_i^{(G)}$ with Equations (7) and (8), where $randn_i(\mu, \sigma^2)$ and $randc_i(\mu, \sigma^2)$ are values randomly obtained from a normal distribution and a Cauchy distribution, respectively, both with mean μ and variance σ^2 , and r_i is a randomly selected position from the memories M_{CR} and M_F .

$$CR_i^{(G)} = randn_i(M_{CR,r_i}, 0.1) \tag{7}$$

$$F_i^{(G)} = randc_i(M_{F,r_i}, 0.1) \tag{8}$$

The successful parameters $CR_i^{(G)}$ and $F_i^{(G)}$ are stored in the variables S_{CR} and S_F . The values at the position k in the memories are updated with Equations (9) and (10).

$$M_{CR,k}^{(G+1)} = \begin{cases} mean_{WA}(S_{CR}) & \text{if } S_{CR} \neq \emptyset \\ M_{CR,k}^{(G)} & \text{otherwise} \end{cases} \tag{9}$$

$$M_{F,k}^{(G+1)} = \begin{cases} mean_{WL}(S_F) & \text{if } S_F \neq \emptyset \\ M_{F,k}^{(G)} & \text{otherwise} \end{cases} \tag{10}$$

G is the current generation of the process. If no successful parameters are identified during this generation, the values stored within the memories remain unchanged. k has the value 1 at the beginning of the process and increases by one unit each time a new element is updated in the memories. When $k > H$, k returns to the value 1. In these equations, $mean_{WA}(\cdot)$ is the weighted mean, calculated with Equations (11) and (13), and $mean_{WL}(\cdot)$ is the weighted Lehman mean, calculated with Equations (12) and (13). The improvement in the fitness function between the *trial vector* that defeats its *target vector* is denoted by $\Delta f_j = |f(u_j^{(G)}) - f(x_j^{(G)})|$.

$$mean_{WA}(S_{CR}) = \sum_{j=1}^{|S_{CR}|} w_j \cdot S_{CR,j}. \tag{11}$$

$$mean_{WL}(S_F) = \frac{\sum_{j=1}^{|S_F|} w_j \cdot S_{F,j}^2}{\sum_{j=1}^{|S_F|} w_j \cdot S_{F,j}}. \tag{12}$$

$$w_j = \frac{\Delta f_j}{\sum_{i=1}^{|S_{CR}|} \Delta f_i}. \tag{13}$$

The value p_i to adjust the mutation strategy (current-to- p best/1) is randomly selected for each individual $x_i^{(G)}$ in the population. See Equation (14), where $p_{min} = 2/NP$. Thus, $x_{pbest}^{(G)}$ is selected from at least 2 individuals up to 20% of the population.

$$p_i = rand[p_{min}, 0.2]. \tag{14}$$

The pseudocode of the SHADE procedure is shown in Algorithm A1 in Appendix A.

2.2. Convolutional Decision Trees (CDTs)

An image is an array of discrete pixels that use different levels of red, green, and blue (RGB) to create the colors and shapes in the image. This work deals with images in grayscale, which are 2D arrays in which each pixel has an integer value between 0 and 255 (the range of values that an 8-bit number can represent).

In computer vision and image processing, convolution kernels are essential for extracting specific features from the image [18]. A convolution kernel is a squared array of discrete numbers, called weights, used to calculate the dot product at each position when a convolution operation is performed. The result of these products, along with the bias associated with the kernel, is the new value for each pixel in the output feature map. Figure 2 illustrates an example of the convolution operation on a 6×6 image with a 3×3 kernel. Full convolution is achieved by repeating the process until the convolution kernel has passed through all pixels of the input image. Therefore, the result of the convolution is a filtered version of the image.

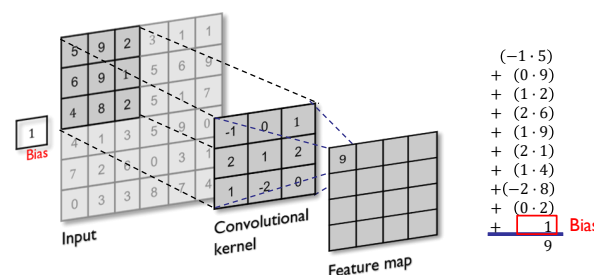


Figure 2. Example of the convolution operation on a 6×6 image with a 3×3 kernel.

Since the convolution kernels extract specific image features, the Convolutional Decision Trees (CDTs) are defined in [6] as algorithms for adaptive feature learning and segmentation, developing the idea of oblique (multivariate) trees [19,20]. The oblique trees successively partition the data into subsets according to a predefined criterion called predicated ϕ , defined in Equation (15), where $x^T \cdot \beta$ represents a linear combination of the attributes, with $x \in \mathbb{R}^d$, and the parameter $\beta \in \mathbb{R}^d$ is used to determine the partition. Thus, for points x in one half-space, $\phi(x) = 1$, while $\phi(x) = 0$ for x in another half-space. The predicate is obtained by maximizing a measure of informativeness, such as the information gain or Gini’s diversity index.

$$\phi(x) = \begin{cases} 1 & \text{if } x^T \cdot \beta > 0, \\ 0 & \text{if } x^T \cdot \beta \leq 0. \end{cases} \tag{15}$$

The CDT is distinguished by its suitability for structural data, such as the spatial structure of image patches in image segmentation. In this context, β represents a convolutional kernel and x denotes the information of a pixel and its neighboring pixels. In consequence, each split of the CDT is represented by a convolutional kernel, which is learned in a supervised manner, maximizing the information gain of the split; see Figure 3.

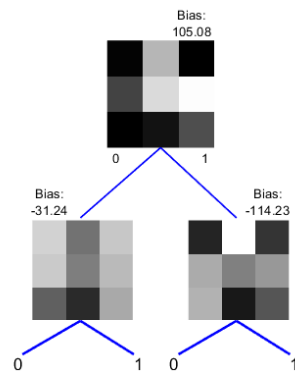


Figure 3. Example of a Convolutional Decision Tree with kernels of size 3×3 .

The classical method for the CDT induction is a greedy search [6]. To improve performance, the methods proposed in [7,9] use the differential evolution (DE) algorithm to induce a CDT. However, these methods have parameters that must be defined by the user and directly affect the model’s performance. Therefore, instead of the traditional DE algorithm, the use of SHADE is proposed in this work.

The following section describes the methodologies proposed for the CDT induction using SHADE.

2.2.1. CDT Induction with SHADE

This paper proposes two search strategies: a global strategy and a local strategy. In the global strategy, SHADE-CDT, the DE algorithm uses the SHADE mechanism to find a complete CDT of given depth d with kernels of fixed size s . In the local strategy, SHADE-CDT-BKS (BKS = Best Kernel Size), the SHADE mechanism is used to find the size of each kernel and the kernels of an optimized CDT of a given depth d .

In both strategies, the images used for the CDT induction are processed, and each pixel (instance) is encoded as the vector with the values of the pixels in the neighborhood of size $s \times s$ surrounding it, adding the value 1 for the bias. Figure 4 shows an example of an instance encoding associated with a pixel, considering a neighborhood of size 3×3 .

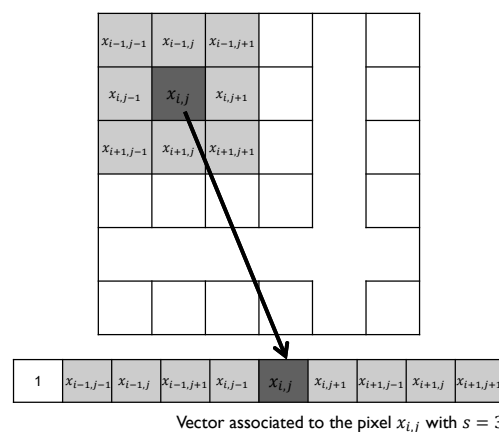


Figure 4. Example of an instance codification associated with a pixel (size $s = 3$).

Algorithm 1 presents the pseudocode of the *buildMatrix* function implemented to obtain a matrix M with the vector representation associated with the pixels of the images and the vector RLV with the real labels of these pixels.

Algorithm 1: Function *buildMatrix*

Input: kernel size (s), set of grayscale images (I) and their corresponding ground-truth images (G) with labels 0 and 1.

Output: matrix M with the vector representation associated with the pixels of I and vector RLV with the real labels of the pixels

```

1  $aux \leftarrow$  smaller nearest integer to  $s/2$ ;
  // Construction of matrix  $M$  with the information of the pixels
2 Initialize matrix  $M \leftarrow []$ ;
3  $np \leftarrow 1$ ; //  $np$  for number of pixels
4 for image in  $I$  do
5    $[row, cols] \leftarrow$  image dimension;
6   for  $i \leftarrow (aux + 1)$  to  $(rows - aux)$  do
7     for  $j \leftarrow (aux + 1)$  to  $(cols - aux)$  do
8        $M(np, :) :=$  Vector of size  $s^2 + 1$  associated to the pixel in position  $(i, j)$ ;
9       // Each vector is a row of  $M$ 
10       $np \leftarrow np + 1$ 
11    end
12  end
  // Construction of vector  $RLV$  with the real labels of the pixels
13 Initialize vector  $RLV \leftarrow ()$ ;
14  $np \leftarrow 1$ ; //  $np$  for number of pixels
15 for ground-truth image in  $G$  do
16    $[row, cols] \leftarrow$  ground-truth image dimension;
17   for  $i \leftarrow (aux + 1)$  to  $(rows - aux)$  do
18     for  $j \leftarrow (aux + 1)$  to  $(cols - aux)$  do
19        $RLV(np) \leftarrow$  ground-truth label in position  $(i, j)$ ;
20       // Each label is an element of  $RLV$ 
21        $np \leftarrow np + 1$ 
22     end
23  end
24 Return:  $M, RLV$ 

```

To manipulate the pixel information in the CDT, a real-valued vector of size $s^2 + 1$ is used to represent a convolutional kernel associated with an internal CDT node. These values represent the weights of the convolutional kernel and a value for the bias. Figure 5 shows an example of a convolutional kernel coding for a kernel of size 3.

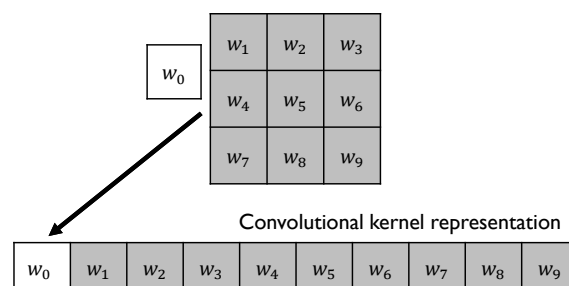


Figure 5. Example of a convolutional kernel codification with size $s = 3$.

A perceptron-like structure is used to determine which branch of the tree to take when classifying an instance, where the dot product of the instance and the weights of the corresponding kernel pass through an activation function that returns a label 0 or 1; see Figure 6. This label indicates the node where the instance goes next, the kernel on the left branch for label 0, or the kernel on the right branch for label 1.

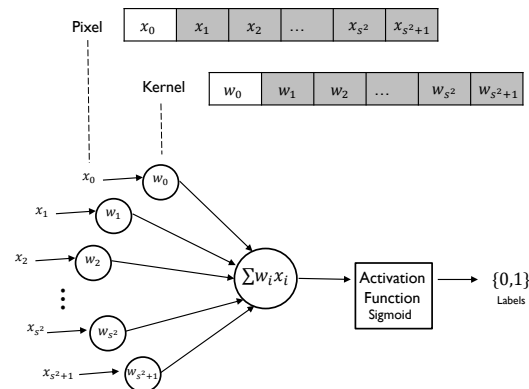


Figure 6. Processing an instance associated with a pixel.

The strategies implemented in this paper use this process to assign a label to each instance with a CDT, passing it through the corresponding kernels until it reaches a leaf node. The label assigned to the instance is obtained with the corresponding kernel in the leaf node. The pseudocode for this process is presented in Algorithm 2 with the *TreePerform* function, where the aptitude of a CDT is calculated. Moreover, in both strategies, the F1-score metric is used to determine the fitness value of each individual in the process. The F1-score compares the labels assigned by a model with the actual labels of the instances to obtain the resulting fitness value between 0 and 1. This project aims to maximize this fitness value, either globally or locally, using the DE algorithm with SHADE.

Global Strategy (SHADE-CDT)

The SHADE-CDT method is a global search strategy based on the methodology proposed in [7]. In this method, the SHADE algorithm is used to induce a CDT of a given depth d with kernels of size s . The population size (NP), the number of generations (NG), and the memory size (H) for the SHADE algorithm are user-defined parameters. The individuals in the population are vectors representing all the convolutional kernels in the CDT, so the encoding of the potential solutions depends on two factors: the depth d of the tree and the size of the convolutional kernel s . If the number of weights that are needed for each kernel is $s^2 + 1$, and the number of kernels in the CDT is $2^d - 1$, then the individuals of the population are vectors of size $(s^2 + 1)(2^d - 1)$ with random values chosen from -255 to 255 . This range of values was chosen because grayscale images have an integer value between 0 and 255 in each pixel, and the same range was given on the negative side to allow the convolution kernel to take negative values if needed. Figure 7 shows an example of encoding for a CDT of depth 3 and kernel size of 3, and Algorithm 3 shows the pseudocode of this process with the *GlobalCDT* function.

Algorithm 2: Function TreePerform

Input: Matrix of image information (M), real label vector (RLV), vector with the convolutional kernels of the CDT ($kernels$), kernel size (s), and depth of the CDT (d)

Output: Aptitude of a CDT

```

// Obtain a list of kernels from the vector kernels
1 Initialize kernelsList  $\leftarrow \{ \}$ ;
2 for index  $\leftarrow 1$  to  $(2^d - 1)$  do
  // Kernel is formed by the first  $s^2 + 1$  elements in kernels vector
3   kernelsList{index}  $\leftarrow kernels(1:s^2 + 1)$ ;
  // Remove the first  $s^2 + 1$  elements of kernels vector
4   kernels  $\leftarrow kernels(s^2 + 2:end)$ 
5 end
6  $m \leftarrow$  number of rows in  $M$ ;
  // Perform the classification of each pixel
7 Initialize PLV  $\leftarrow ( )$ ; //Predicted label vector
8 for pixel  $\leftarrow 1$  to  $m$  do
9    $u \leftarrow 1$ ; //Current depth value
10  index  $\leftarrow 1$ ;
11  while  $u \leq d$  or kernelsList{index}  $\neq \emptyset$  do
12    kernel  $\leftarrow kernelsList\{index\}$ ;
13     $r \leftarrow dot(M(pixel,:), kernel)$ ; //Dot product of an instance and the
      kernel
14     $\phi \leftarrow sigmoid(r)$ ; //Activation function (sigmoid)
15    if  $\phi > 0.5$  then
16      | index  $\leftarrow 2 * index + 1$ ; //The instance goes to the right branch
17    else
18      | index  $\leftarrow 2 * index$ ; //The instance goes to the left branch
19    end
20     $u \leftarrow u + 1$ ; //Add one level to the current depth
21  end
22  if index is odd then
23    // If the pixel ends in a right leaf, assign it class 1
24    PLV(pixel)  $\leftarrow 1$ 
25  else
26    // If the pixel ends in a left leaf, assign it class 0
27    PLV(pixel)  $\leftarrow 0$ 
28  end
29 aptitude  $\leftarrow F1score(RLV, PLV)$ ;
30 Return: aptitude

```

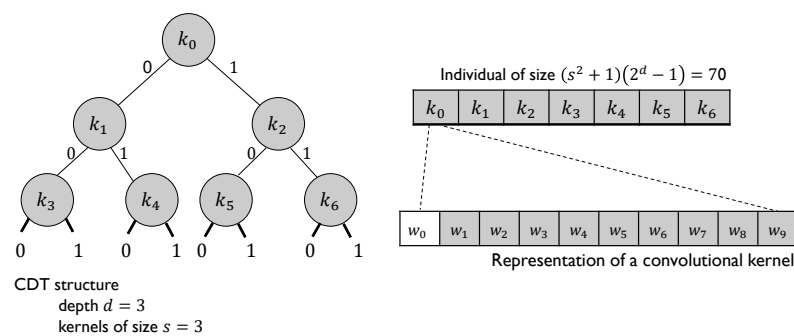


Figure 7. Codification of the convolutional kernels in a CDT for the SHADE-CDT method.

Algorithm 3: Function *GlobalCDT*

Input: Set of grayscale images (I) and their corresponding ground-truth images (G), kernel size (s), depth of the CDT (d), population size (NP), number of generations (NG) and memory size (H)

Output: Kernels of a CDT induced with the SHADE-CDT strategy and its aptitude

- 1 $[M, PRL] \leftarrow buildMatrix(s, I, G);$
- 2 Initialize population $P = \{x_1, x_2, \dots, x_{NP}\}$ of vectors of size $(s^2 + 1) * (2^d - 1)$ with random values chosen from -255 to 255 ;
- 3 $x_{best} \leftarrow SHADE(P_0 = P, N = NP, H, f = TreePerform(M, PRL, x_i, s, d));$
- 4 $aptitude(x_{best}) \leftarrow TreePerform(M, PRL, x_{best}, s, d);$
- 5 **Return:** $x_{best}, aptitude(x_{best})$

Local Strategy (SHADE-CDT-BKS)

The SHADE-CDT-BKS method is a local search strategy based on the proposal in [9]. The main difference between the SHADE-CDT approach and the SHADE-CDT-BKS method is that the SHADE algorithm is used to find the kernel sizes and the kernels of a CDT of a given depth d . This is achieved by a systematic approach that ensures that each kernel partitions the instances in the most effective way. The population size (NP), the number of generations (NG), and the memory size (H) for the SHADE algorithm are user-given parameters.

In this proposal, the user provides a list S of possible sizes for the kernels, with odd values s , and the depth d of the tree. The SHADE algorithm is applied to each value in S , where the individuals of the population are real-valued vectors of size $s^2 + 1$ with random values chosen from -255 to 255 ; see Figure 5. For each value s , the individual with the best fitness is found, and among them, the one with the best F1-score is selected as the best solution. In this way, the CDT is induced by finding each kernel until the depth specified by the user is reached.

All instances are considered in the SHADE process for the root node kernel, and for the other kernels, only the instances tagged with the class corresponding to the branch in which the kernel is located are considered. With the SHADE-CDT-BKS method, a new kernel is found only if more than 20 instances are considered for the SHADE process of the kernel. These instances must belong to different classes, at least 5% of one of them. Otherwise, the method does not find a new kernel. The pseudocode for this process is described in the *buildLocalCDT* function of Algorithm 4. In this way, the SHADE-CDT-BKS method produces an optimized CDT with different kernel sizes without using a pruning process; see Figure 8.

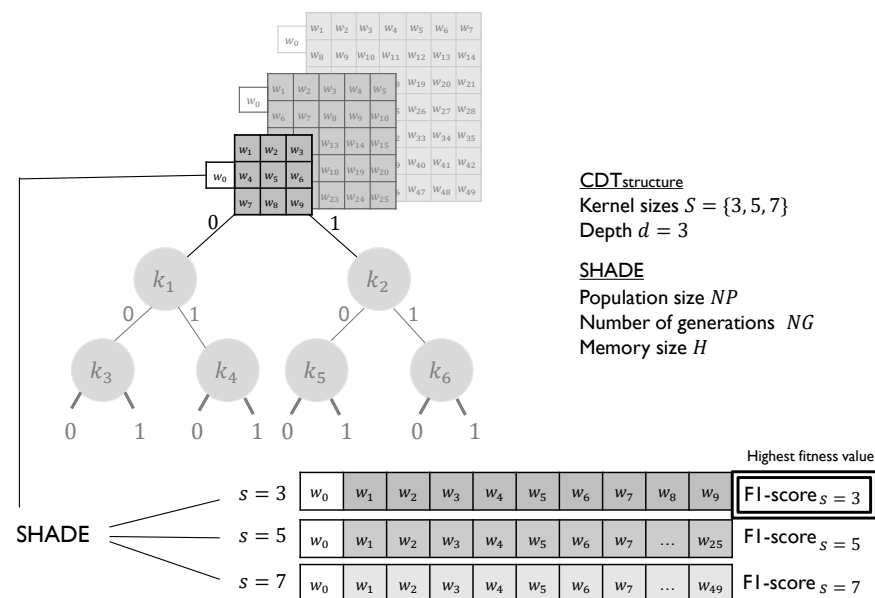


Figure 8. Kernel selection for each internal node of the CDT with the SHADE-CDT-BKS method.

Algorithm 4: Function *buildLocalCDT*

Input: Matrix of image information (M) and their corresponding vector of real labels (PLR), kernel sizes (S), depth of the CDT (d), population size (NP), number of generations (NG), and memory size (H)

Output: Structure of a CDT induced with the SHADE-CDT-BKS strategy

```

1  $m \leftarrow$  number of rows in  $M$ ; //  $m$  is the number of pixels in the images
2  $Indexes = \{1, 2, \dots, m\}$ ;
3  $LocalCount \leftarrow |PLR|$ ;
4  $LocalLabelPercentage \leftarrow 0.95 * LocalCount$ ;
5  $C_0 \leftarrow |\{pixel \in Indexes : PLR(pixel) = 0\}|$ ; //Number of class 0 pixels
6  $C_1 \leftarrow |\{pixel \in Indexes : PLR(pixel) = 1\}|$ ; //Number of class 1 pixels
7 if ( $LocalCount > 20$ ) and ( $C_1 < LocalLabelPercentage$ ) and
   ( $C_2 < LocalLabelPercentage$ ) then
8   for  $s$  in  $S$  do
9     Initialize population  $P = \{x_1, x_2, \dots, x_{NP}\}$  of vectors of size
       ( $s^2 + 1$ ) * ( $2^d - 1$ ) with random values chosen from  $-255$  to  $255$ ;
10     $x_{best}(s) \leftarrow SHADE(P_0 = P, N = NP, H, f =$ 
        $TreePerform(M, PRL, x_i, s, 1))$ ;
11     $aptitude(x_{best}^{(s)}) \leftarrow TreePerform(M, PRL, x_{best}(s), s, 1)$ 
12  end
13   $x_{best} \leftarrow \max_{s \in S} \{aptitude(x_{best}^{(s)})\}$ ;
14  for  $pixel$  in  $Indexes$  do
15     $r \leftarrow \text{dot}(M(pixel, :), x_{best})$ ;
16     $\phi(pixel) \leftarrow \text{sigmoid}(r)$ ;
17  end
18  // Indexes of instances going to the left branch
    $Indexes_{left} = \{pixel \in Indexes : \phi(pixel) \leq 0\}$ ;
19  // Indexes of instances going to the right branch
    $Indexes_{right} = \{pixel \in Indexes : \phi(pixel) > 0\}$ ;
20  // Real labels of instances going to the left branch
    $PLR_{left} = \{PLR(pixel) \in PLR : \phi(pixel) \leq 0\}$ ;
21  // Real labels of instances going to the right branch
    $PLR_{right} = \{PLR(pixel) \in PLR : \phi(pixel) > 0\}$ ;
22 end
23 if ( $x_{best} = null$ ) or ( $Indexes_{left} = \emptyset$ ) or ( $Indexes_{right} = \emptyset$ ) or
   ( $|PLR_{left}| = LocalCount$ ) or ( $|PLR_{right}| = LocalCount$ ) then
24    $CDTStructure.left \leftarrow null$ ;
25    $CDTStructure.right \leftarrow null$ ;
26 else
27    $M_{left} \leftarrow M(Indexes_{left}, :)$ ; //Instances going to the left branch
28    $CDTStructure.left = buildLocalCDT(M_{left}, PLR_{left}, S, d, NP, NG, H)$ ;
29    $M_{right} \leftarrow M(Indexes_{right}, :)$ ; //Instances going to the right branch
30    $CDTStructure.right = buildLocalCDT(M_{right}, PLR_{right}, S, d, NP, NG, H)$ 
31 end
32 Return:  $CDTStructure$ 

```

3. Experiments and Results

This section presents the results obtained by inducing a CDT using the SHADE-CDT and SHADE-CDT-BKS methods on images from the *Weizmann Horse Dataset* [21] and *Blood detection in dark-field microscopy images* obtained from Kaggle (<https://github.com/PerceptiLabs/bacteria/tree/main?tab=readme-ov-file> accessed on 28 May 2024). The se-

lection of these databases made it possible to compare the results with those described in [6,7,9]. Three subsections are presented for each dataset: SHADE-CDT experiments, SHADE-CDT-BKS experiments, and a comparison between methods. The first two subsections show the experiments and describe the results. The last subsection compares the two approaches based on the differences in the segmentation task per image. A review of the explicability of a CDT is also included at the end of this section.

The proposed strategies were implemented in Matlab R2023b software. Table 1 provides the specifications of the computer that was used to perform the experiments.

Table 1. Specifications of the computer that was used to perform the experiments.

Operating system	Windows 11 Pro 23H2
RAM	64 GB
Processor	AMD Ryzen 5 5600G with Radeon Graphics
Processor speed	3.90 GHz

3.1. Weizmann Horse Dataset

The *Weizmann Horse Dataset* [21] consists of 327 manually segmented images of horses of different colors and in a broad spectrum of landscapes. To perform experiments with the methods proposed in this work, an image resizing procedure was applied to reduce the number of pixel-associated instances in the CDT induction process to 40% of their original size. Controlled experiments were performed to calibrate the parameter values of the SHADE algorithm (population size NP , number of generations NG , and size of memories H), using fixed training and test sets of 33 and 295 images, respectively. These proportions were chosen based on the best result of the study in [7]; see Table 2 for details. For the values of NP , NG , and H for each method, the different combinations of the following values were considered: 50, 100, and 200 for NP and NG , and 15, 30, 50, and 100 for H . After performing several experiments with these features, the highest F1-scores were obtained with $NP = 100$, $NG = 200$, and $H = 100$ for the SHADE-CDT method, and with $NP = 50$, $NG = 200$, and $H = 100$ for the SHADE-CDT-BKS method, so these six values were maintained in the following experiments.

Table 2. Best result obtained with the CDT induction method proposed in [7] with the variant DE/best/1/bin, and the parameters CR and F set to 0.9. A training dataset with a proportion of 1/10 (33 images) was used in the learning process.

Exp.	Popsiz	Generations	Depth	F1-Score	Accuracy	Time
8	80	200	3	0.4882	0.6798	23.17 h

3.1.1. SHADE-CDT Experiments

To induce a CDT with the SHADE-CDT method, the following parameter values for the structure of the CDT were considered: depths from one to five and kernel sizes three, five, seven, and nine. Table 3 shows the results of 20 experiments performed with this method under the aforementioned conditions. The best results for kernel sizes three, five, and seven were obtained with a CDT of depth four in experiments 4, 9, and 14, respectively. For kernel size nine, the best result was obtained in experiment 17 with a CDT of depth two.

Based on the results of experiment 17, where the best overall result was obtained with a kernel of the maximum size considered, nine, and the best results by depth were obtained with kernels of size nine, it is clear that the larger kernel size leads to a higher F1-score for the Weizmann Horse Dataset. Figure 9 shows the CDT induced in this experiment, along with the best and worst individual results obtained for the test dataset, the original images, the ground truth, and the predictions generated using the SHADE-CDT method. The F1-scores and accuracies obtained for these images are also displayed.

Table 3. Experiments with the Weizmann Horse Dataset for CDT induction using the SHADE-CDT method with $NP = 100$, $NG = 200$, and $H = 100$. The best result by kernel size is shown in bold.

Experiment	Kernel Size	Depth	Time	F1-Score	Accuracy
1	3	1	28.41 min	0.45442	0.76264
2	3	2	35.86 min	0.47463	0.75427
3	3	3	42.62 min	0.45648	0.7625
4	3	4	48.87 min	0.47894	0.72317
5	3	5	56.43 min	0.47538	0.70138
<hr/>					
6	5	1	34.94 min	0.45667	0.75605
7	5	2	43.31 min	0.49249	0.72448
8	5	3	49.02 min	0.49795	0.72471
9	5	4	56.95 min	0.49805	0.73129
10	5	5	1.09 h	0.49601	0.71724
<hr/>					
11	7	1	41.36 min	0.45423	0.74869
12	7	2	51.27 min	0.50945	0.72868
13	7	3	58.68 min	0.50874	0.72947
14	7	4	1.11 h	0.5151	0.73056
15	7	5	1.27 h	0.50853	0.72809
<hr/>					
16	9	1	51.3 min	0.47535	0.74975
17	9	2	1.44 h	0.52204	0.74759
18	9	3	1.71 h	0.51292	0.74353
19	9	4	2.05 h	0.51652	0.73279
20	9	5	1.57 h	0.51517	0.72943

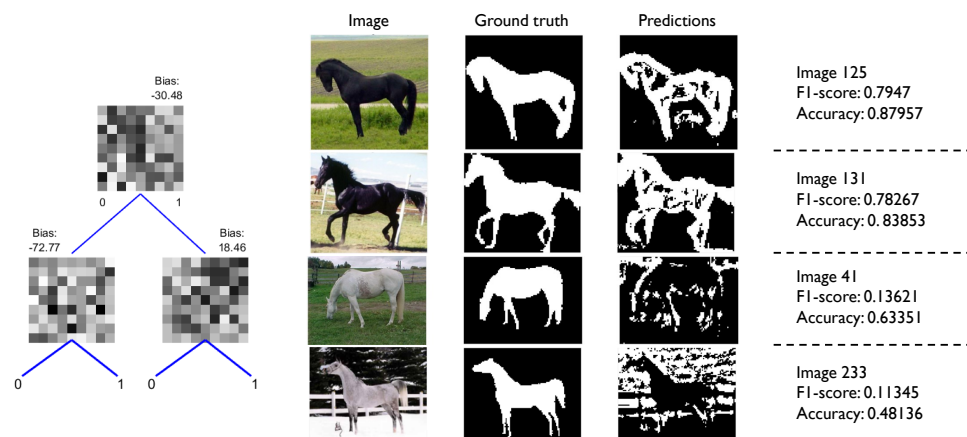


Figure 9. CDT induced by the SHADE-CDT method in experiment 17. The two best and two worst segmentation results obtained, with the original images, the ground truth (real segmented masks), and the predictions are shown along with the corresponding image number, F1-score, and accuracy.

The analysis of individual images in the test set underscored the significant influence of color and background structure on the model’s performance. The wide range of colors of the horses strongly influenced the negative results of the model. In addition, they were set against a variety of background structures—fences, trees, mountains, and plants. When the images were converted to grayscale, the tones of these structures interfered with the tones of the pixels corresponding to the horses, resulting in classification errors.

When applied to images of white horses, the proposed method produced a reverse labeling result, highlighting the pixels corresponding to the background structures and leaving out the horse’s shape, as illustrated in Figure 9.

Table 3 shows that the SHADE-CDT method gives more varied and better results compared to those in [7]. None of the experiments achieved results like the F1-score of 80.4% obtained in [6], but their method for CDT induction required 12 h for training, and the maximum time in the experiments presented in Table 3 was 2.05 h. These results show

that explainable trees with shallow depth were effective since the induction of deeper trees did not necessarily result in a better F1-score. As shown in Table 3, the best results were obtained with a depth not greater than four.

3.1.2. SHADE-CDT-BKS Experiments

The following parameter values were considered to induce a CDT with the SHADE-CDT-BKS method: depths from one to five, and two sets of kernel sizes, $S = \{3, 5, 7\}$ and $S = \{3, 5, 7, 9\}$.

Table 4 shows the results of 10 experiments performed with this method under the conditions described above. The best result for the set of kernel sizes $\{3, 5, 7\}$ was obtained in experiment 4 with a CDT of depth four. For the other set of kernel sizes, the best result was obtained in experiment 9 with a CDT of depth four.

Table 4. Experiments with the Weizmann Horse Dataset for CDT induction using the SHADE-CDT-BKS method with $NP = 50$, $NG = 200$, and $H = 100$. The best result by set of kernel sizes is shown in bold.

Experiment	Kernel Sizes	Depth	Time	F1-Score	Accuracy
1	{3, 5, 7}	1	1.4 h	0.4755	0.75228
2	{3, 5, 7}	2	2.63 h	0.51567	0.70908
3	{3, 5, 7}	3	4.09 h	0.51902	0.67215
4	{3, 5, 7}	4	5.34 h	0.52558	0.66385
5	{3, 5, 7}	5	5.50 h	0.50859	0.59686
6	{3, 5, 7, 9}	1	1.67 h	0.48134	0.74936
7	{3, 5, 7, 9}	2	3.31 h	0.51361	0.67674
8	{3, 5, 7, 9}	3	4.88 h	0.50145	0.60426
9	{3, 5, 7, 9}	4	6.48 h	0.53651	0.67394
10	{3, 5, 7, 9}	5	7.44 h	0.53062	0.62843

The best overall result was obtained using the CDT of depth four induced in experiment 9, shown in Figure A1 of Appendix B. Figure 10 shows the best and worst individual results obtained for the test dataset, the original images, the ground truth, the generated predictions, and the F1-score and accuracy obtained for each image. Again, the negative results of the model were influenced by the color of the horses, as the white horses had lower F1-scores, and the dark horses had higher scores.

Tables 3 and 4 show that the SHADE-CDT-BKS method produced better results than the SHADE-CDT method and consequently, better results than those in [7].

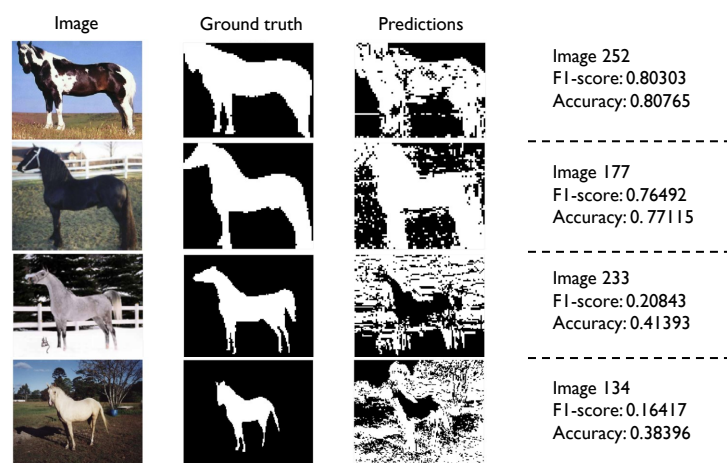


Figure 10. The two best and two worst segmentation results from experiment 9 using the SHADE-CDT-BKS method, with its corresponding image number, F1-score, and accuracy.

3.1.3. Comparison between Methods

In Figure 11, images 125, 131, 41, and 233 are shown to compare the change in the segmentation task between experiment 1 with the SHADE-CDT method and the results with the CDT induced in experiments 4 and 9 by the SHADE-CDT-BKS method. Experiment 4 with the SHADE-CDT-BKS method was chosen because it gave the best F1-score for the kernel sizes set $S = \{3, 5, 7\}$.

There are some important notes to make:

- The SHADE-CDT-BKS method outperformed the SHADE-CDT method with the highest F1-score of 0.53651. Although it required more computational time due to the use of the DE algorithm, $|S| = 4$ times in each node to induce the CDT, as suggested by corresponding Tables 3 and 4, and the array (F1-score, accuracy) in blue in Figure 11.
- Analyzing images 41 and 233, it is clear that more pixels associated with the horse were classified as class 1, as they should be; see the rows corresponding to these images in Figure 11. The F1-scores of these images also supported this. For example, the F1-score of image 41 increased significantly from 0.13621 in experiment 17 using the SHADE-CDT method to 0.48607 using the SHADE-CDT-BKS method. It is important to note that this increase indicates a shift in the distribution of the pixels in the CDT, resulting in a more significant number of pixels with clear tones being assigned to class 1.
- On the other hand, in the predictions for images 125 and 131, the profile of the horses was extended by adding to class 1 the surrounding pixels that were previously classified as pixels of label 0. This confirms a shift in the distribution of the pixels. See the corresponding rows for these images in Figure 11.













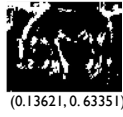







Image	Ground truth	SHADE-CDT Experiment 17 Predictions (0.52204, 0.74759)	SHADE-CDT-BKS Experiment 4 Predictions (0.52558, 0.66385)	SHADE-CDT-BKS Experiment 9 Predictions (0.53651, 0.67394)
 125		 (0.7947, 0.87957)	 (0.72951, 0.81648)	 (0.75591, 0.81983)
 131		 (0.78267, 0.83853)	 (0.73704, 0.77939)	 (0.7367, 0.76761)
 41		 (0.13621, 0.63351)	 (0.43156, 0.64716)	 (0.48607, 0.69726)
 233		 (0.11345, 0.48136)	 (0.22071, 0.42026)	 (0.20843, 0.41393)

Figure 11. Comparison of the segmentation results for images 125, 131, 41, and 233 of the Weizmann Horse Dataset for experiment 1 with the SHADE-CDT method and for experiments 4 and 9 with the SHADE-CDT-BKS method. The original images, the ground truth, and the predictions for these experiments are shown, along with their corresponding image number (under the original image) and the array (F1-score, accuracy) under the corresponding image result. The overall F1-Score and accuracy of each experiment are shown in blue under the corresponding column.

- Figure 11 shows that the increase in the overall F1-score between these experiments was because the images with white horses received higher F1-scores; however, the images with dark horses decreased their F1-score by adding more class 0 pixels to the pixels corresponding to a horse (class 1).

- Thus, even if the F1-score increased between these experiments, the comparison with Figure 11 suggests that the model's behavior on images as diverse as those in this dataset could lead to classification errors because it is difficult for the learning process to make the appropriate divisions in the dataset.

3.2. Blood Detection in Dark-Field Microscopy Images

The *Blood detection in dark-field microscopy images* is a dataset containing 366 dark-field microscopy images for observing and segmenting erythrocytes in blood tissue. Each image in the database was resized to a uniform size of 200×200 . This was performed to streamline the CDT induction process by reducing the number of instances associated with the pixels. It should be noted that certain images in this dataset were excluded from the training and test sets because their ground truth contained only labels for class 0, indicating the absence of erythrocytes for identification. See Appendix C for more details.

The experiments used predetermined training and test sets with a split of 70% and 30%, respectively. After removing 21 images, the training set consisted of 241 images, and the test set had 104 images. The experiments with this dataset were intentionally limited to CDTs of depth one and two. This decision was not arbitrary but was based on careful consideration of the amount of information in the training set, the computational time required, and the results obtained.

3.2.1. SHADE-CDT Experiments

For the SHADE-CDT method, the values $NP = 100$ and $NG = 200$ were considered based on the results in [9]. Also, $H = 100$ was considered for the size of the memories in the SHADE algorithm since this value seemed to be the best option in previous experiments. Table 5 shows the results obtained with these parameter values, considering kernel sizes of three, five, and seven, and CDTs of depth one and two.

Table 5. Experiments with the Blood Detection in Dark-Field Microscopy images for CDT induction using the SHADE-CDT method with $NP = 100$, $NG = 200$, and $H = 100$. The best result by kernel size is shown in bold.

Experiment	Kernel Size	Depth	Time	F1-Score	Accuracy
1	3	1	11.36 h	0.74946	0.89667
2	3	2	14.68 h	0.60856	0.79846
3	5	1	12.64 h	0.62068	0.85601
4	5	2	16.45 h	0.47262	0.82742
5	7	1	15.68 h	0.52155	0.83746
6	7	2	20.33 h	0.49722	0.82554

The experiments showed that the induction of a CDT with the SHADE-CDT method gave better results with shallow convolutional trees. The results of experiments 1, 3, and 5 in Table 5 with trees of depth one outperformed those with depth two at each kernel size.

Experiment 1 achieved the best result for this dataset by inducing a CDT of depth one with a kernel of size three. Figure 12 shows the CDT induced in this experiment, along with the best and worst individual results obtained for the test dataset, the original images, the ground truth, and the predictions generated using the SHADE-CDT method. The F1-scores and accuracies obtained for these images are also presented. This experiment yielded F1-scores higher than 0.9 for some images. However, it is important to note that due to several structures with low intensity labeled as class 1 in the ground truth, image 331 received an F1-score of less than 0.03, significantly reducing the average F1-score. The second image with the lowest score, image 85, had an F1-score of 0.35623. While this score may not be exceptionally high, the corresponding ground-truth image and the predicted labels demonstrated the precision of the CDT in identifying the brighter parts of the structure in the original image.

With the results for experiments 1, 3, and 6 shown in Table 5 and those shown in [9], we can conclude that this dataset had a better segmentation with shallow CDTs of depth one, since the best results by kernel size were obtained with CDTs of this depth.

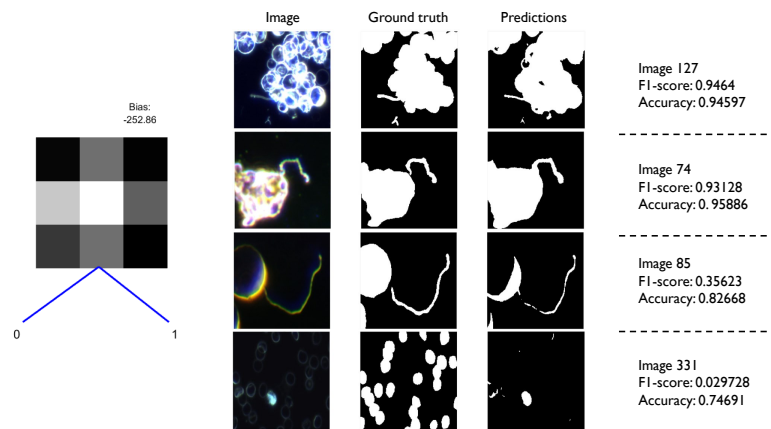


Figure 12. CDT induced by the SHADE-CDT method in experiment 1. The two best and two worst segmentation results obtained with the original images, the ground truth, and the predictions are shown, along with the corresponding image number, F1-score, and accuracy.

3.2.2. SHADE-CDT-BKS Experiments

After analyzing the results and the computational time used for the experiments with the SHADE-CDT method, it was decided to reduce the population size NP to 50 for the SHADE-CDT-BKS method, keeping the number of generations $NG = 200$ and the size of the memories $H = 100$. Table 6 shows the results obtained under these conditions. As expected, the computational time increased with this method since the DE algorithm was applied $|S| = 3$ times to find each CDT node.

As with the SHADE-CDT method, the best result was obtained with a CDT of depth one and kernel size three, i.e., in experiment 1; see Figure 13. In that experiment, as in the SHADE-CDT method, image 331 had the worst F1-score. Image 303 was the second-worst segmented image with an F1-score higher than 0.3. In that case, the original image had several structures with brighter parts that were not considered as label 1 in the ground truth, so the CDT had classification errors.

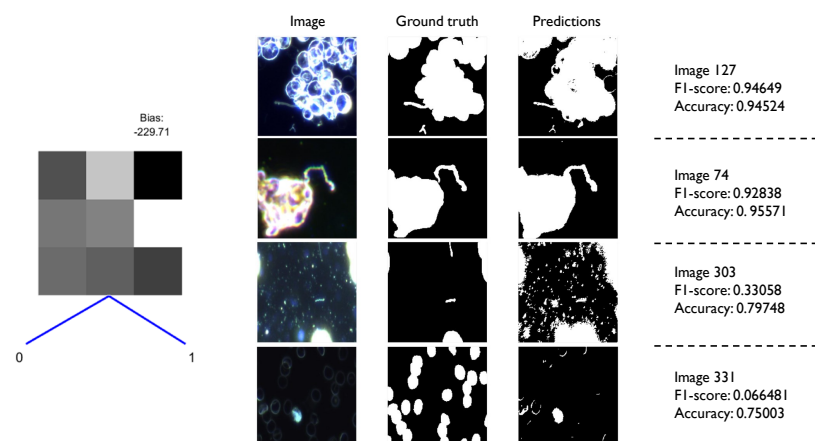


Figure 13. CDT induced by the SHADE-CDT-BKS method in experiment 1. The two best and two worst segmentation results obtained with the original images, the ground truth, and the predictions are shown on the right, along with the corresponding image number, F1-score, and accuracy.

Table 6. Experiments with the Blood Detection in Dark-Field Microscopy Images for CDT induction using the SHADE-CDT-BKS method with $NP = 50$, $NG = 200$, and $H = 100$. The best result by kernel size is shown in bold.

Experiment	Kernel Sizes	Depth	Time	F1-Score	Accuracy
1	{3, 5, 7}	1	1.61 days	0.72588	0.88132
2	{3, 5, 7}	2	3.15 days	0.66765	0.84057

3.2.3. Comparison between Methods

In Figure 14, images 127, 74, 85, and 331 are shown to compare the change in the segmentation task between experiment 1 with the SHADE-CDT method and the results with the CDT induced in experiments 1 and 2 by the SHADE-CDT-BKS method. Experiment 2 with the SHADE-CDT-BKS method was chosen because it gave the third best F1-score for the dataset.

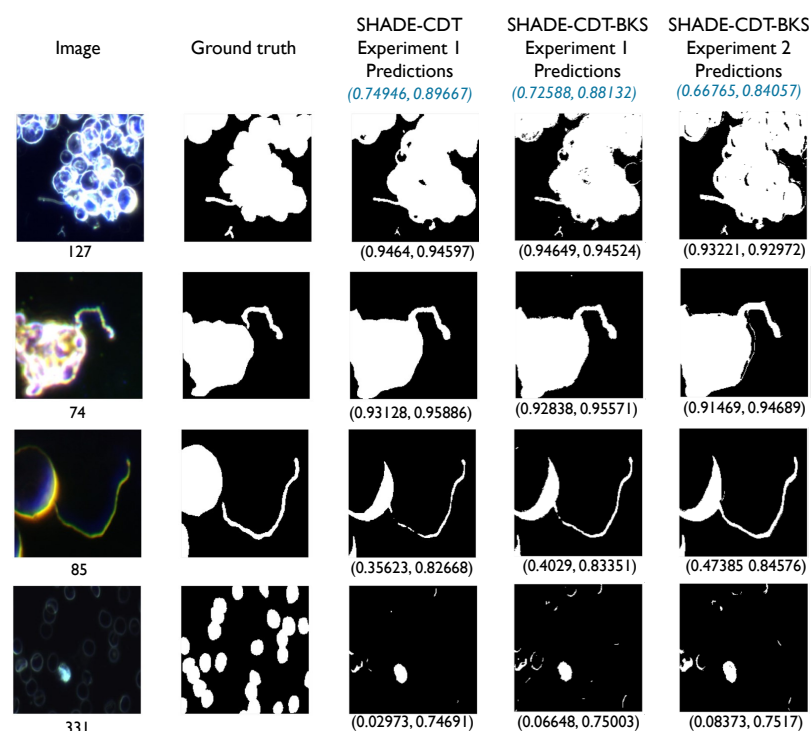


Figure 14. Comparison of the segmentation results for images 127, 74, 85, and 331 of the Blood detection in dark-field microscopy images for experiment 1 with the SHADE-CDT method and for experiments 1 and 2 with the SHADE-CDT-BKS method. The original images, the ground truth, and the predictions for these experiments are shown, along with their corresponding image number (under the original image) and the array (F1-score, accuracy) under the corresponding image result. The F1-score and accuracy of each experiment are presented in blue under their respective column.

The following observations can be made about the results of this dataset:

- The SHADE-CDT method outperformed the SHADE-CDT-BKS method with the highest F1-score of 0.74946, using a CDT of depth one and a kernel size of three. Thus, a normal convolution process was the best way to segment the images in this dataset. It is also important to note that in the SHADE-CDT-BKS experiments, the population size was decreased to reduce the computation time. For this reason, the result was slightly worse.
- The predictions for images 85 and 331 showed that the CDT induced for any of the methods proposed in this work failed to obtain a high F1-score in images where some structures with low tones were labeled as class 1 in the corresponding ground truth.

For example, image 331 consistently exhibited an F1-score below 0.1. This is because in the ground truth, certain structures that were difficult to see in the original image, such as those with tones similar to the background tones, were classified as class 1. Something similar happened in image 85, where the brighter pixels were identified with label 1, and some pixels with a tone like the background tone were also labeled as class 1.

- The predictions for images 127 and 74 showed that the CDT induced with these methods gave good results when the original images contained bright structures labeled as class 1 in their corresponding ground truth.
- Figure 14 shows that the decrease in the overall F1-score between these experiments was because the images with brighter structures received higher F1-scores when these structures were labeled as class 1 in the ground truth, but the images with structures labeled as class 1 with tones similar to the tones of the background pixels had lower F1-scores.
- For these experiments, it is clear that the depth of the induced CDT affected the F1-score, probably leading to overfitting in the training data set. However, it is important to note a difference between experiment 2 with the SHADE-CDT method and experiment 2 with the SHADE-CDT-BKS method; see Tables 5 and 6. In experiment 2 with the SHADE-CDT method, a CDT of depth two was induced with kernels of size three, yielding an F1-score of 0.60856 against the CDT induced in experiment 2 with the SHADE-CDT-BKS method, with an F1-score of 0.66765, also of depth two, but with two kernels of size three and one kernel of size seven; see Figure 15. In these experiments, the same depth was considered for the CDT, but better results were obtained with the SHADE-CDT-BKS method since, in that case, kernels of different sizes were used. This observation leads to the question of whether considering deeper trees with this method allows for a better partitioning of the dataset. These experiments were not initially considered due to the high computational time required for this type of experiment. However, they could be considered by reducing the kernel sizes, i.e., by using $S = \{3, 7\}$.
- Thus, the comparison with Figure 14 suggests that the behavior of the model on the images of this dataset was influenced by those images where structures with tones similar to the tones of the pixels in the background were labeled as class 1, or conversely, where pixels with lighter tones were labeled as class 0.

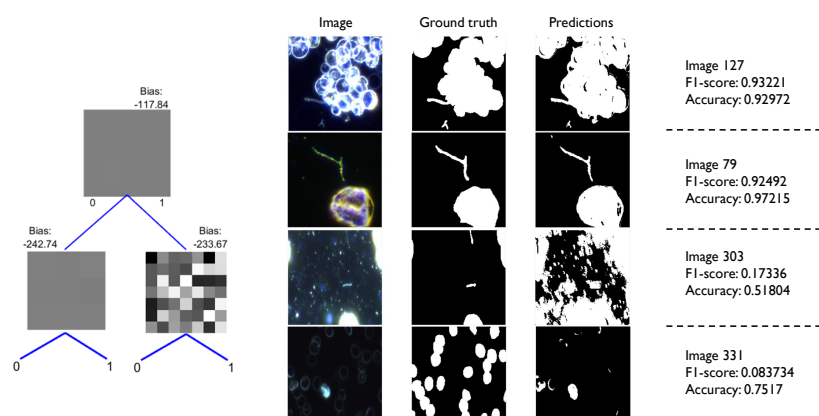


Figure 15. CDT induced by the SHADE-CDT-BKS method in experiment 2. The two best and two worst segmentation results obtained with the original images, the ground truth, and the predictions are shown on the right, along with the corresponding image number, F1-score, and accuracy.

3.3. Review of Explainability in a CDT

The structure of a CDT allows an analysis of how each internal kernel classifies the image pixels, since the user can follow the CDT structure through convolutional operations on an image to analyze the results for each branch and node. The kernels obtained by the

proposed methods are expected to classify image pixels by patterns and shapes, as shown in Figure 16a.

For example, in the Weizmann Horse dataset, the segmentation performance of each kernel can be analyzed on a CDT of depth two in image 125; see Figure 16b. The first kernel classified the pixels well, while the lower kernels focused exclusively on the horse's profile. Similarly, in the case of Blood detection in dark-field microscopy images, a CDT of depth one, i.e., one kernel of size three, performed an exemplary classification of the pixels in image 74; see Figure 16c. This analysis can be used to identify the reasons why the model made mistakes in images with low F1-scores.

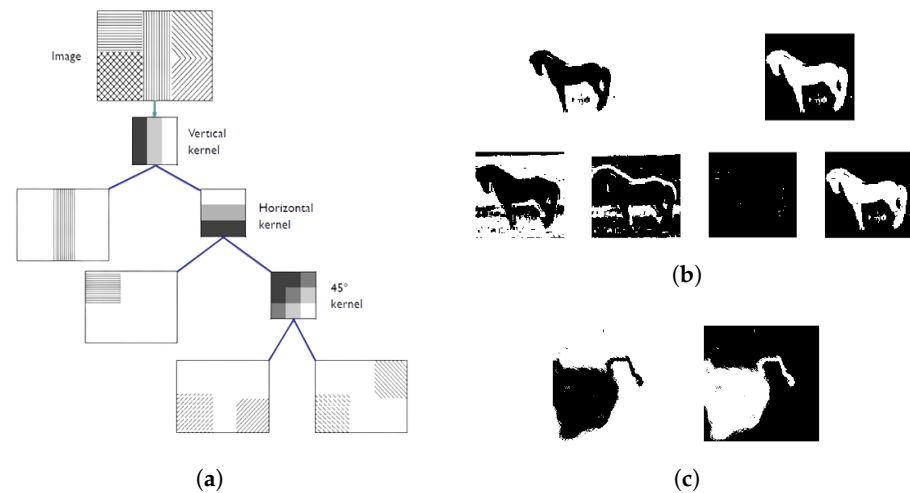


Figure 16. (a) Example of the expected explainability in a CDT. Here, the kernels performed the classification of image pixels by patterns. (b) Segmentation of image 125 of the Weizmann Horse dataset by a CDT of depth 2. (c) Segmentation of image 74 of the Blood detection in dark field microscopy images by a CDT of depth 1.

4. Conclusions and Future Work

This paper presented various experiments analyzing the performance of the globally induced CDT and locally induced CDT using the SHADE-CDT and SHADE-CDT-BKS methods, respectively. These methods are new variants of the proposals made in [7,9], but they use the SHADE algorithm to guide the differential evolution process.

It is crucial to consider the time required for the computation of the fitness function during the DE process to analyze the performance of the methods proposed in this paper. In the global approach, the SHADE-CDT method, each pixel in the training set must pass through the corresponding kernels of a CDT. Since each individual in the population represents a CDT during the DE process, the F1-score and computation time for each CDT increase with the number of training instances. In the local approach, the SHADE-CDT-BKS method, since the training set is partitioned kernel by kernel, the F1-score of each individual in the population is computed using a classical convolution process. However, this approach is complex because for each kernel, multiple sizes in the set S are analyzed to select the optimal kernel size, i.e., the kernel size with the best F1-score. Thus, the DE process must be performed $|S|$ times for each kernel, which increases the computation time.

The best results in this paper were obtained with the local search strategy, but the computational time used for this method was high compared to the other one. The SHADE algorithm outperformed the results obtained in [7] with the *Weizmann Horse dataset*. However, the result of the F1-score obtained in [6], equal to 0.804, was better than the results in this work. Although it is important to mention that in [6], the proportions in the train and test sets employed were two-thirds and one-third, respectively, and the CDT obtained was reported with a depth of 18 with kernels of size 31, so the results presented here are explicable models compared to the other.

Regarding the *Blood detection in dark-field microscopy images*, F1-scores greater than 0.92 were reported in [9]. However, in these experiments, the training and test sets with 8 and 4 images, respectively, were generated from a pre-selection of 12 images with similar characteristics in the dataset. This pre-selection allowed the authors to analyze the behavior of the model, but in the experiments performed here, only some of the images were excluded since in their ground truths, all the pixels were labeled as class 0. Thus, the F1-scores obtained were lower than those in [9] but with more interesting features to analyze.

Thanks to the experiments carried out in this work, the following conclusions can be highlighted:

- The DE algorithm can induce explicable CDTs in conjunction with the SHADE algorithm, and both methods, SHADE-CDT and SHADE-CDT-BKS, can be trained with small data sets.
- It is important to note that the model has certain limitations. The two datasets analyzed revealed that the model struggled with textures and specific background components. This is particularly relevant since the project was limited to grayscale images. The use of such images can potentially lead to confusion regarding the tones of the different structures within them, thereby increasing the risk of classification errors.
- After performing the multiple experiments shown in this paper, both methods, SHADE-CDT and SHADE-CDT-BKS, seem to have a linear complexity time with respect to the tree depth.
- Explainable CDTs with shallow depth are effective since inducing deeper trees does not necessarily result in a better F1-score.
- This work highlights the importance of generating explainable models where the segmentation process is clear to the user.

For future work, some techniques to reduce individuals' evaluation time will be implemented. Furthermore, the performance of the proposed models with other image segmentation approaches, including Convolutional Neural Networks, will be compared. Finally, the possibility of using alternative fitness functions will be explored.

Author Contributions: Conceptualization, A.-L.L.-L., H.-G.A.-M., and E.M.-M.; methodology, A.-L.L.-L., H.-G.A.-M., and E.M.-M.; software, A.-L.L.-L.; validation, H.-G.A.-M. and E.M.-M.; formal analysis, A.-L.L.-L.; investigation, A.-L.L.-L.; resources, A.-L.L.-L., H.-G.A.-M., and E.M.-M.; data curation, A.-L.L.-L.; writing—original draft preparation, A.-L.L.-L.; writing—review and editing, A.-L.L.-L., H.-G.A.-M., and E.M.-M.; visualization, A.-L.L.-L.; supervision, H.-G.A.-M. and E.M.-M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The original data presented in the study are openly available. The *Weizmann Horse dataset* on Kaggle at <https://www.kaggle.com/datasets/ztaihong/weizmann-horse-database/data> and the *Blood detection in dark-field microscopy images* on GitHub at <https://github.com/PerceptiLabs/bacteria/tree/main?tab=readme-ov-file> (accessed on 28 May 2024).

Acknowledgments: The first author would like to thank the Consejo Nacional de Humanidades, Ciencia y Tecnología (CONAHCyT), an institution of the Government of Mexico, for the financial support provided through the “beca de estancia posdoctoral” with CVU 712182 as part of the program Estancias Posdoctorales por México.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CDT	Convolutional Decision Tree
DE	Differential evolution
BKS	Best Kernel Size
SHADE	Success-History-based Adaptive Differential Evolution
CNN	Convolutional Neural Network

Appendix A. SHADE Algorithm

The pseudocode shown in Algorithm A1 corresponds to the SHADE algorithm proposed in [10], where the differential evolution process is performed using a historical memory of successful parameters for the crossing rate (CR) and scale factor (F) in the mutation and crossover operators, respectively.

Algorithm A1: SHADE algorithm

```

// Initialization phase
1  $G \leftarrow 0$ ;
2 Initialize population  $P_0 = \{x_1^{(1)}, \dots, x_N^{(0)}\}$  randomly;
3 Set all values in  $M_{CR}, M_F$  to 0.5;
4 Archive  $A = \emptyset$ ;
5 Index Counter  $k \leftarrow 1$ ;
// Main loop
6 while Termination criteria are not met do
7    $S_{CR} = \emptyset$ ;
8    $S_F = \emptyset$ ;
9   for  $i = 1$  to  $N$  do
10     $r_i \leftarrow$  Select from  $[1, H]$  randomly;
11     $CR_i^{(G)} \leftarrow randn_i(M_{CR, r_i}, 0.1)$ ;
12     $F_i^{(G)} \leftarrow randc_i(M_{F, r_i}, 0.1)$ ;
13     $p_i^{(G)} \leftarrow rand[p_{min}, 0.2]$ ;
14    Generate trial vector  $u_i^{(G)}$  by current-to- $p$ best/1/bin;
15    for  $i = 1$  to  $N$  do
16     if  $f(u_i^{(G)}) \leq f(x_i^{(G)})$  then
17       $x_i^{(G+1)} \leftarrow u_i^{(G)}$ ;
18     else
19       $x_i^{(G+1)} \leftarrow x_i^{(G)}$ ;
20     end
21     if  $f(u_i^{(G)}) < f(x_i^{(G)})$  then
22       $x_i^{(G)} \rightarrow A$ ;
23       $CR_i^{(G)} \rightarrow S_{CR}$ ;
24       $F_i^{(G)} \rightarrow S_F$ ;
25     end
26    end
27    Whenever the size of the archive exceeds  $|A|$ , randomly selected
    individuals are deleted so that  $|A| \leq |P|$ ;
28    if  $S_{CR} \neq \emptyset$  and  $S_F \neq \emptyset$  then
29     Update  $M_{CR, k}, M_{F, k}$  based on  $S_{CR}, S_F$ ;
30      $k++$ ;
31     if  $k > H$  then
32       $k \leftarrow 1$ ;
33     end
34    end
35  end
36 end

```

Appendix B. CDT for the Weizmann Horse Dataset

In Figure A1, the CDT induced for the Weizmann Horse dataset with the SHADE-CDT-BKS method in experiment 9 is shown. This CDT consists of 15 convolution kernels with the following dimensions: kernel 1–7, kernel 2–3, kernel 3–5, kernel 4–9, kernel 5–5, kernel 6–9, kernel 7–3, kernel 8–9, kernel 9–7, kernel 10–9, kernel 11–3, kernel 12–7, kernel 13–7, kernel 14–7, and kernel 15–3.

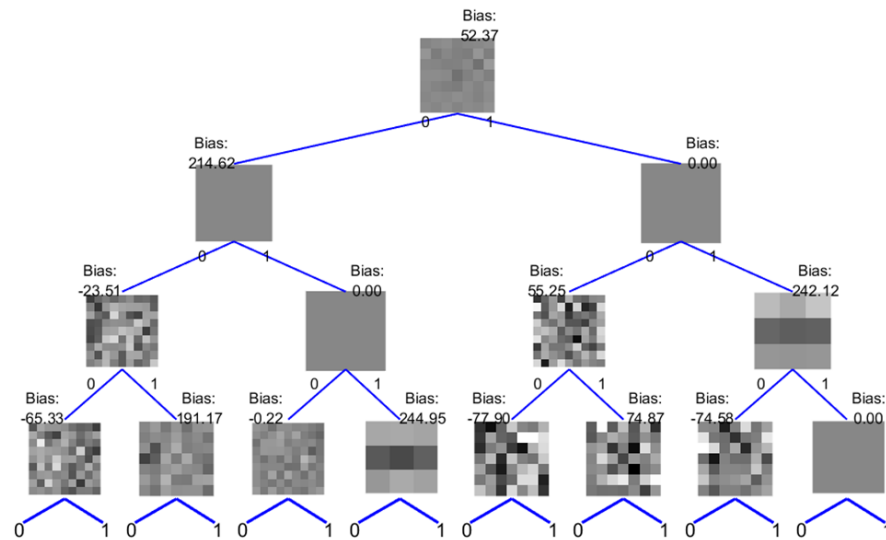


Figure A1. CDT induced with the SHADE-CDT-BKS method in experiment 9 for the Weizmann Horse dataset.

Appendix C. Analysis of Blood Detection in Dark-Field Microscopy Images

The images displayed in this section were not used for training or testing during the CDT induction with the *Blood Detection in Dark-Field Microscopy images* dataset.

The analysis excluded the following 21 images: 13, 45, 61, 62, 63, 77, 86, 87, 88, 156, 161, 162, 163, 269, 288, 305, 308, 320, 324, 328, and 337. All their ground-truth labels were of class 0, representing the background of the images. However, it is important to note that some structures were visible in the original images, as shown in Figure A2.

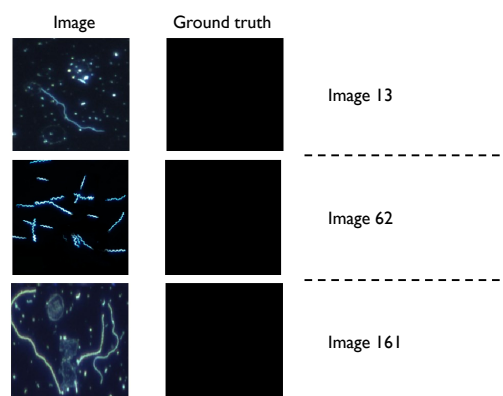


Figure A2. Examples of images excluded in the training and testing process for the CDT induction with the Blood detection in dark-field microscopy images.

This characteristic consistently caused errors in the segmentation process of the induced CDTs when using any of the methods proposed in this work with all the images in the dataset. The structures observed in these 21 images significantly shifted the threshold for the segmentation task, resulting in a notable decrease in F1-score values.

References

1. Lateef, F.; Ruichek, Y. Survey on semantic segmentation using deep learning techniques. *Neurocomputing* **2019**, *338*, 321–348. [CrossRef]
2. Patil, D.D.; Deore, S.G. Medical image segmentation: A review. *Int. J. Comput. Sci. Mob. Comput.* **2013**, *2*, 22–27.
3. Minaee, S.; Boykov, Y.; Porikli, F.; Plaza, A.; Kehtarnavaz, N.; Terzopoulos, D. Image segmentation using deep learning: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 3523–3542. [CrossRef] [PubMed]
4. Diakogiannis, F.I.; Waldner, F.; Caccetta, P.; Wu, C. ResUNet-a: A deep learning framework for semantic segmentation of remotely sensed data. *ISPRS J. Photogramm. Remote Sens.* **2020**, *162*, 94–114. [CrossRef]
5. Molnar, C. *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*, 2nd ed. 2024. Available online: <https://christophm.github.io/interpretable-ml-book> (accessed on 28 May 2024).
6. Laptev, D.; Buhmann, J.M. Convolutional decision trees for feature learning and segmentation. In Proceedings of the German Conference on Pattern Recognition, Münster, Germany, 2–5 September 2014; Springer: Cham, Switzerland, 2014; pp. 95–106.
7. Barradas Palmeros, J.A.; Mezura Montes, E.; Acosta Mesa, H.G.; Márquez Grajales, A.; Rivera López, R. Induction of Convolutional Decision Trees with Differential Evolution for Image Segmentation. In Proceedings of the Congreso Mexicano de Inteligencia Artificial, Guadalajara, Mexico, 30 May–3 June 2023.
8. Eiben, A.E.; Smith, J.E. *Introduction to Evolutionary Computing*; Springer: Berlin/Heidelberg, Germany, 2015.
9. López-Lobato, A.L.; Acosta-Mesa, H.G.; Mezura-Montes, E. Blood Cell Image Segmentation Using Convolutional Decision Trees and Differential Evolution. In Proceedings of the Advances in Computational Intelligence, MICAI 2023 International Workshops, Yucatán, Mexico, 13–18 November 2023; Springer Nature: Cham, Switzerland, 2024; pp. 315–325.
10. Tanabe, R.; Fukunaga, A. Success-history based parameter adaptation for differential evolution. In Proceedings of the 2013 IEEE Congress on Evolutionary Computation, Cancun, Mexico, 20–23 June 2013; IEEE: Piscataway, NJ, USA, 2013; pp. 71–78.
11. Storn, R.; Price, K. Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [CrossRef]
12. Rivera-Lopez, R.; Canul-Reich, J. Construction of near-optimal axis-parallel decision trees using a differential-evolution-based approach. *IEEE Access* **2018**, *6*, 5548–5563. [CrossRef]
13. Rivera-Lopez, R.; Canul-Reich, J.; Mezura-Montes, E.; Cruz-Chávez, M.A. Induction of decision trees as classification models through metaheuristics. *Swarm Evol. Comput.* **2022**, *69*, 101006. [CrossRef]
14. Price, K.; Storn, R.M.; Lampinen, J.A. *Differential Evolution: A Practical Approach to Global Optimization*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2006.
15. Ahmad, M.F.; Isa, N.A.M.; Lim, W.H.; Ang, K.M. Differential evolution: A recent review based on state-of-the-art works. *Alex. Eng. J.* **2022**, *61*, 3831–3872. [CrossRef]
16. Zhang, J.; Sanderson, A.C. JADE: Adaptive differential evolution with optional external archive. *IEEE Trans. Evol. Comput.* **2009**, *13*, 945–958. [CrossRef]
17. Tanabe, R.; Fukunaga, A.S. Improving the search performance of SHADE using linear population size reduction. In Proceedings of the 2014 IEEE Congress on Evolutionary Computation (CEC), Beijing, China, 6–11 July 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 1658–1665.
18. Kim, S.; Casper, R. *Applications of Convolution in Image Processing with MATLAB*; University of Washington: Seattle, WA, USA, 2013; pp. 1–20.
19. Dolotov, E.; Zolotykh, N. Evolutionary algorithms for constructing an ensemble of decision trees. In Proceedings of the Analysis of Images, Social Networks and Texts: 8th International Conference, AIST 2019, Kazan, Russia, 17–19 July 2019; Revised Selected Papers 8; Springer: Cham, Switzerland, 2020; pp. 9–15.
20. Rivera-Lopez, R.; Canul-Reich, J. Differential evolution algorithm in the construction of interpretable classification models. In *Artificial Intelligence-Emerging Trends and Applications*; IntechOpen: London, UK, 2018; pp. 49–73.
21. Borenstein, E.; Sharon, E.; Ullman, S. Combining top-down and bottom-up segmentation. In Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop, Washington, DC, USA, 27 June–2 July 2004; IEEE: Piscataway, NJ, USA, 2004; p. 46.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.