Hindawi
Mathematical Problems in Engineering
Volume 2022, Article ID 4639208, 25 pages
https://doi.org/10.1155/2022/4639208

Research Article

Performance Comparison of Most Recently Proposed Evolutionary, Swarm Intelligence, and Physics-Based Metaheuristic Algorithms for Retinal Vessel Segmentation

Mehmet Bahadır Çetinkaya and Hakan Duran

Mechatronics Engineering Department, University of Erciyes, Kayseri 38039, Turkey

Correspondence should be addressed to Mehmet Bahadır Çetinkaya; cetinkaya@erciyes.edu.tr

Received 24 November 2021; Revised 30 December 2021; Accepted 17 January 2022; Published 22 February 2022

Academic Editor: Thomas Hanne

Biomedical image analysis based on metaheuristic algorithms is one of the most important research areas encountered in recent years. Due to the low contrast differences between the diseased areas and the image background in high-contrast biomedical images, effective methods are required to diagnose diseases with high accuracy. To overcome the difficulties encountered in this field, metaheuristic approaches may offer effective solutions due to their advantages such as the ability of converging to the global optimum, higher convergence rate, and having few control parameters. In this work, Jellyfish Search (JS), Marine Predators (MPA), Tunicate Swarm (TSA), Mayfly Optimization (MA), Chimp Optimization (ChOA), Slime Mould Optimization (SMA), Archimedes Optimization (AOA), and Equilibrium Optimizer (EO) algorithms, which are the most recently proposed metaheuristic algorithms in the literature, have been improved as clustering based in order to achieve vessel segmentation with high precision. Also, a detailed performance comparison of these algorithms has been realized for the rate of convergences, error values reached, CPU time, standard deviation, sensitivity, specificity, accuracy, *F*-score, and Wilcoxon rank sum-test. In order to present the compatibility of the results obtained with the literature, the performances of these novel algorithms have also been compared to that of Particle Swarm Optimization (PSO), Grey Wolf Optimization (GWO), and Differential Evolution (DE) algorithms. The simulation results represent that each algorithm produces similar convergence and error performance. Also, it can be emphasized from the statistical analyses that the stability and robustness of each metaheuristic approach are quite adequate in separating the vessel pixels and the background pixels of a retinal image. In general, this paper proves that although having fewer number of control parameters, the JS, MPA, TSA, MA, ChOA, SMA, AOA, and EO algorithms produce similar but a bit better results in terms of image segmentation when compared to PSO, GWO, and DE algorithms.

1. Introduction

Although the conventional algorithms have frequently been used in literature for retinal image analysis, there are limited number of works based on heuristic and metaheuristic algorithms relevant to this area. In addition, to the best of our knowledge, the most recent algorithms analyzed in this work have not been used yet in biomedical image processing although they have been employed for a limited number of areas such as improvement of metaheuristic methods for reducing energy consumption (for JS algorithm) [1], developing metaheuristic based forecasting algorithms for pandemic diseases (for MPA algorithm) [2], improving artificial intelligence based approaches for optimization of distortions occurring in heat and power systems (for MPA algorithm) [3, 4], comparing the performances of PSO-type optimization algorithms (for MPA and SMA algorithms) [5], development of artificial intelligence based cyber security systems (for TSA algorithm) [6], developing a chaos mechanism based whale optimization algorithm (for TSA algorithm) [7], developing an emperor penguin optimizer for efficient ranking of cloud service providers (for TSA algorithm) [8], developing metaheuristic approaches in order to increase the efficiency of distribution system (for MA algorithm) [9], improving a Chebyshev based mayfly optimizer (for MA algorithm) [10], improving a metaheuristic optimizer modeling hierarchical structures (for ChOA algorithm) [11], developing advanced computational methods for agriculture machinery movement optimization (for ChOA algorithm) [12], development of metaheuristic based approaches in order to enhance the data path structures in data transfer applications (for ChOA algorithm) [13], improvement of optimum modeling approaches for solar panels (for SMA algorithm) [14], improving optimization algorithms for image segmentation problem encountered in chest X-ray images (for SMA algorithm) [15], in a detailed review work for monarch butterfly optimization (for SMA algorithm) [16], improving a golden eagle optimizer (for EO algorithm) [17], improving a multiobjective optimization algorithm using improvement based reference points approach (for EO algorithm) [18], and improving an artificial intelligence based optimizer for parameter extraction of a fuel cell dynamic model [19]. Finally, in the detailed literature review, it has been observed that the AOA algorithm has not been used yet for any application area.

In order to diagnose and treat the important symptoms of retinal diseases, the retinal vessel segmentation is one of the most important areas of biomedical image processing [20]. Due to the difficulties and insufficiencies encountered in conventional methods, computer-aided diagnosis systems for ophthalmic disorders are developed [21]. Computer aided segmentation of retinal images provide higher accuracy in retinal image analysis, and so, more effective treatment methods are able to be developed for the abnormalities emerging as a result of obesity [22], hypertension [23], glaucoma [24], and diabetic retinopathy [25–27]. Computer based approaches also enable highly accurate diagnostics for applications such as prematurity retinopathy [28], determination of vessel sizes [29], regional analysis on the retinal image [30], arteriolar stenosis [31], surgical procedures [32], improving treatment methods for retinal diseases [33–35] and optic disk detection [36], and retinal image analyses by using the well-known swarm and population based approaches [37].

In addition to the literature analyses given above, there are also several works in the literature about the application of metaheuristic algorithms to image segmentation. In these works, it is emphasized that image segmentation based on metaheuristic algorithms has the potential of obtaining near optimum solutions. In [38], a novel algorithm called self-adaptive moth-flame optimization TH (SAMFO-TH) has been improved for the aim of multilevel thresholding in color image segmentation, and its performance has been compared with other well-known eight metaheuristic algorithms in the literature. Xu and friends have introduced a novel algorithm that is called improved dragonfly algorithm (IDA) in [39] for color image segmentation and then compared its performance with that of differential evolution (DE) and dragonfly algorithms. In [40], a multilevel thresholding method for color image segmentation exploiting 1D OTSU cuckoo search (CS), 1D OTSU lightning search (LSA), and cuttlefish (CFA) algorithms has been proposed by Bhandari and friends. The work presented in [41] describes a novel bee foraging algorithm (BFA) based multilevel thresholding method for image segmentation. In [42], the task of designing an efficient methodology based on Harris Hawks Optimization (HHO) algorithm for multilevel image segmentation has been investigated. In [43], a detailed performance analysis between widely used evolutionary and swarm based optimization algorithms has been realized for multilevel color image thresholding. In work [44], a reliable retinal blood vessel segmentation approach consisting of a cascade connection of edge detection and shape analysis methods has been improved by the authors. In another work [45], a benchmark of thirteen metaheuristic optimization algorithms for image thresholding have been performed in high-dimensional search spaces. The authors have proposed a new hybrid algorithm called *K*-means Firefly algorithm (KFA) and then analyzed its performance in image segmentation in [46]. In [47], a water cycle algorithm (WCA) based efficient method has been proposed by Kandhway and Bhandari for multilevel thresholding in color image segmentation. Finally, an effective image clustering method based on human mental search (HMS) algorithm has been improved by Mousavirad et al. [48].

The segmentation performances of the Jellyfish Search (JS), Marine Predators (MPA), Tunicate Swarm (TSA), Mayfly Optimization (MA), Chimp Optimization (ChOA), Slime Mould Optimization (SMA), Archimedes Optimization (AOA), and Equilibrium Optimizer (EO) algorithms improved in this work have been analyzed in detail for both healthy and diseased retinal images taken from the DRIVE and STARE databases. In Figures 1 and 2, the images for each of the databases have been given, respectively.

In order to distinguish the vessel and the background pixels with high accuracy, the retinal image should be enhanced before clustering by applying the band selection, bottom-hat transformation, and brightness correction preprocessing.

When the red (R), green (G), and blue (B) layers of RGB retinal images have been analyzed separately in terms of illuminance, contrast, and brightness levels, it has been seen that the G layer provides the highest clustering performance due to its higher illuminance, optimal contrast, and optimal brightness levels.

The green layers obtained after band selection preprocessing of the retinal images taken from the DRIVE database have been given in Figures 3(a) and 3(c). The retinal images enhanced in terms of contrast difference by applying the bottom-hat transformation and brightness correction have also been shown in Figures 3(b) and 3(d).

The green layers of the retinal images taken from the STARE database have been obtained as shown in Figures 4(a) and 4(c). Also, the retinal images enhanced in terms of contrast difference obtained after the two subsequent preprocessing operations have been given in Figures 4(b) and 4(d).

The main contributions of this paper to literature can be summarized as follows:

(i) The most recently proposed JS, MPA, TSA, MA, ChOA, SMA, AOA, and EO algorithms have been improved as clustering based and then applied to retinal vessel segmentation for the first time in literature.

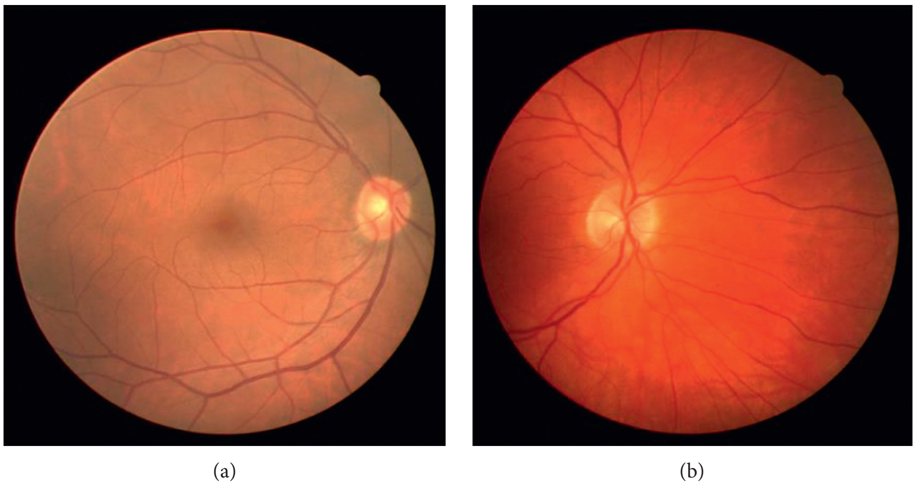


Figure 1: Healthy and diseased images taken from DRIVE database. (a) Healthy image. (b) Diseased image with bleeding.

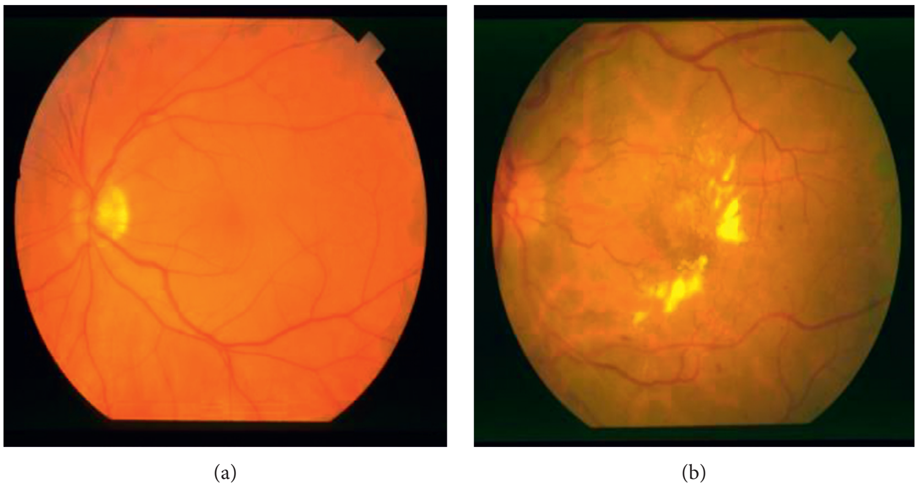


Figure 2: Healthy and diseased images taken from STARE database. (a) Healthy image. (b) Diseased image with bleeding.

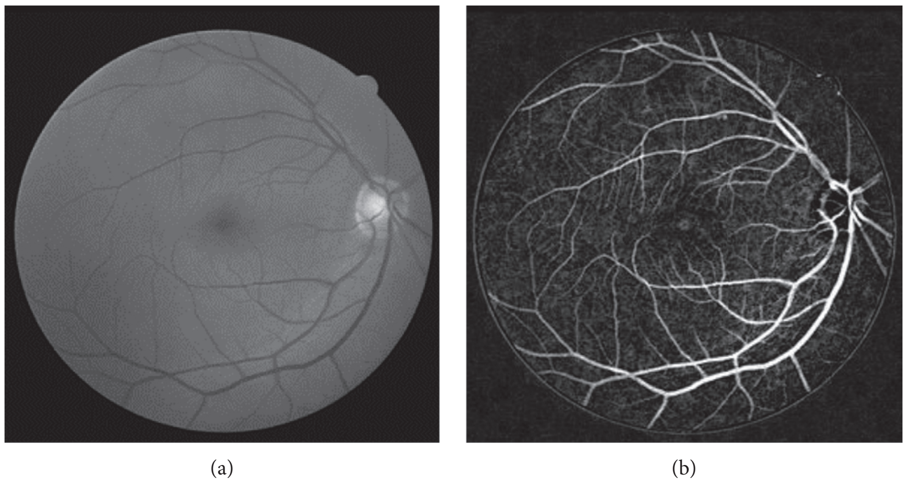


Figure 3: Continued.

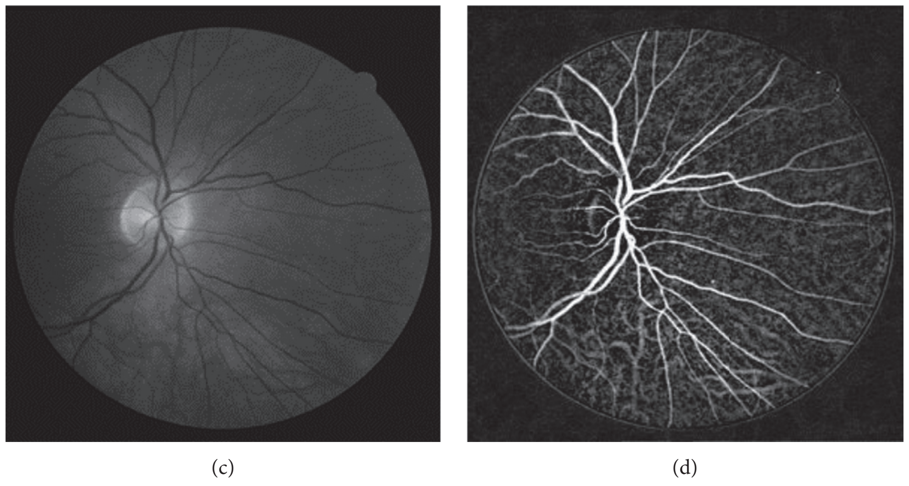


Figure 3: The resulting DRIVE database images after each preprocessing phase. (a, c) G layers obtained after band selection; (b, d) The resulting images after the bottom-hat filtering and brightness correction.

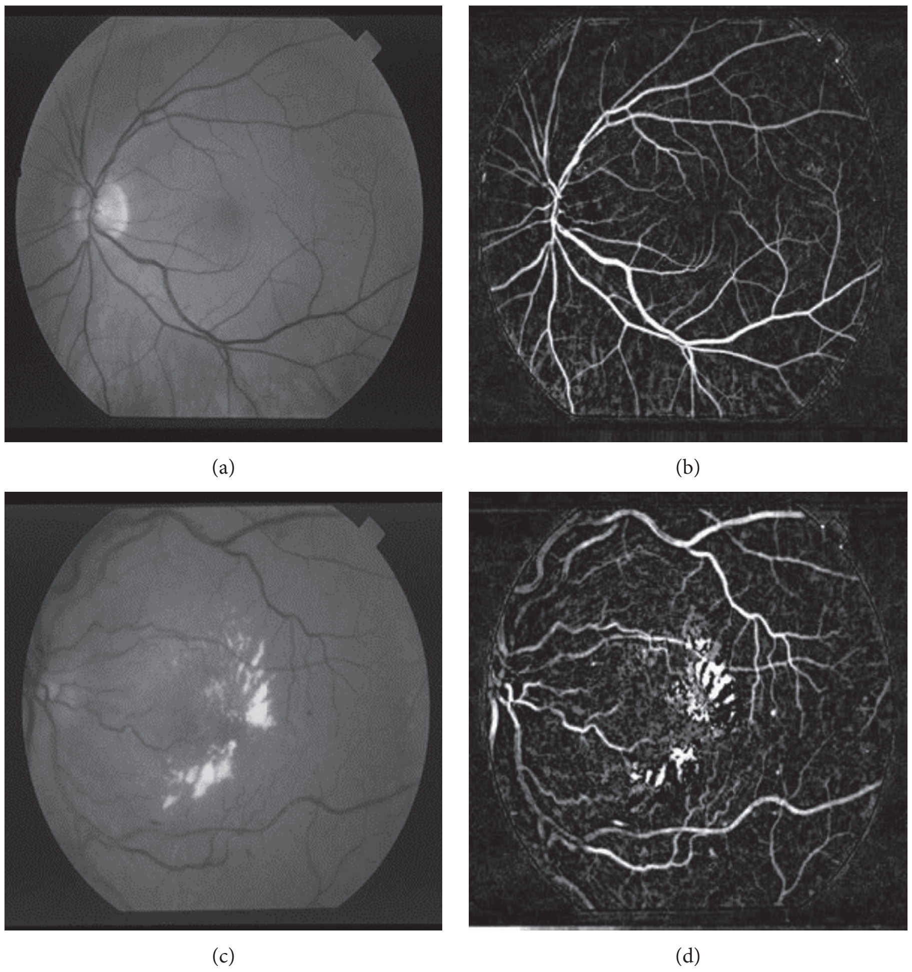


Figure 4: The resulting STARE database images after each preprocessing phase. (a, c) G layers obtained after band selection; (b, d) The resulting images after the bottom-hat filtering and brightness correction.

(ii) The performances of the improved algorithms have been compared to those of the well-known PSO, GWO, and DE algorithms in the literature.

(iii) The segmentation results obtained demonstrate that these algorithms produce similar or a bit better results than PSO, GWO, and DE, and they can successfully be used in segmentation of retinal images with high accuracy.

(iv) The statistical analyses demonstrate that these algorithms produce stable and reliable results despite their noncomplex algorithm structures.

The rest of this paper is organized as follows. Section 2 presents brief reviews about both the most recently proposed metaheuristic algorithms used in this work and the properties of the retinal images used. In Section 3, the simulation and the statistical analysis results obtained for DRIVE and STARE databases are given in detail and then compared with each other and literature. The relevant discussions are presented in Section 4. Finally, conclusions and future directions are given in Sections 5.

2. Material and Methods

Nature-inspired metaheuristic algorithms, which simulate biological or physical phenomena to solve optimization problems, can be classified into 4 major groups as evolutionary based, swarm intelligence based, physics based, and human based.

In this work, the most novel evolutionary based, swarm intelligence based, and physics based metaheuristic algorithms in the literature have been applied to biomedical image analysis, and then their performances have been compared in detail.

2.1. Jellyfish Search Algorithm.

2.1. Jellyfish Search Algorithm. Jellyfish search algorithm, which is proposed by Chou and Truong in 2020 [49], is a swarm based optimizer, which simulates the search behavior of jellyfishes. One of the most important advantages of JS algorithm is that it contains only two control parameters, namely, population size and cycle number. It is inspired from 3 main rules; (i) a jellyfish performs two types of movement, which can be defined as "moving through the ocean current" or "moving along with a swarm," and the transition process between these movements is coordinated by a time control mechanism, (ii) while searching for food a jellyfish moves towards the locations having more amount of food, and (iii) the quality of a food source is determined by its location and its corresponding objective function.

The detailed pseudocode of a bioinspired swarm intelligence based JS algorithm has been given as follows:

```
Randomly create an initial population, $x_i \in \{1, 2, \ldots, N_{\mathrm{POP}}\}$
Calculate the fitness value of each $x_i$ solution in the population.
  Cycle = 1
  REPEAT
    Calculate the value of time control parameter by using the following equation:
```

$$c(t) = \left| \left(1 - \frac{1}{\text{Max Cycle Number}} \right) (2.r\text{and}\ (0,1) - 1) \right|. \tag{1}$$

```
    IF $c(t) \geq 0,5$: follow ocean current
      (i) Determine ocean current by
```

$$\overrightarrow{\text{trend}} = X^{*} - \beta.r\text{and}\ (0,1).\mu, \tag{2}$$

```
          where $X^*$ represents the jellyfish currently with the best location in the swarm, $\mu$ is the mean location of all jellyfish and $\beta > 0$ is determined as the distribution coefficient which is related to the length of $\overrightarrow{\text{trend}}$.
      (ii) Determine a new location by
```

$$X_i(t+1) = X_i(t) + r\text{and}\ (0,1).\overrightarrow{\text{trend}}. \tag{3}$$

```
    ELSE: jelly fish moves inside a swarm
      If $r\text{and}\ (0,1) > (1 - c(t))$ define a new position,
```

$$X_i(t+1) = X_i(t) + \gamma.r\text{and}\ (0,1).(U_b - L_b), \tag{4}$$

```
      where; $U_b$ and $L_b$ represent upper and lower bounds of search space, respectively and $\gamma > 0$ is the motion coefficient.
      else: determine the direction ($\vec{d}$) and new location of $i^{th}$ and $j^{th}$ jellyfishes by,
```

$$\vec{d} = \begin{cases} X_j(t) - X_i(t) \text{ if } f(X_i) \geq f(X_j), \\ X_i(t) - X_j(t) \text{ if } f(X_i) < f(X_j), \end{cases} \tag{5}$$

$$X_i(t+1) = X_i(t) + r\text{and}\ (0,1).\vec{d}.$$

```
      end if
    END IF
    Check boundary conditions and calculate the fitness value of food source at new location
    Update the location of $X_i$ and location of $X^*$
  Cycle = Cycle + 1
  UNTİL (termination criteria are met)
```

2.2. Marine Predators Algorithm. Marine Predators algorithm, which is proposed by Faramarzi et al. in 2020 [50], is a nature-inspired swarm intelligence based metaheuristic algorithm, which simulates foraging behavior of ocean predators by taking into account the relationship between the prey and predator. The most important metric of this algorithm is to optimize the search strategies used by the predators which is also called random walk strategy. The algorithm focuses on two main random walk strategies, namely, Brownian motion and Lévy motion.

A detailed pseudocode of a nature-inspired swarm intelligence based MPA algorithm can be given as follows:

Randomly create a uniformly distributed prey population, $i = 1, 2, \ldots, n$

Cycle = 1

WHILE Cycle ≤ Maximum Cycle

Calculate the fitness values and construct the Elite matrix which includes the best solutions (best positions of preys)

$$\text{Elite} = \begin{pmatrix} X^I{}_{1,1}, & \ldots, & X^I{}_{1,d}, \\ \vdots, & \ddots, & \vdots \\ X^I{}_{n,1}, & \cdots, & X^I{}_{n,d}. \end{pmatrix}. \tag{6}$$

IF Cycle < ((Maximum Cycle Number)/3): high velocity ratio

Update prey by $\overrightarrow{\text{prey}_i} = \overrightarrow{\text{prey}_i} + P.\vec{P} \otimes \overrightarrow{\text{stepsize}_i}$

where, $\overrightarrow{\text{stepsize}_i} = \overrightarrow{R_B} \otimes (\overrightarrow{\text{Elite}_i} - \overrightarrow{R_B} \otimes \overrightarrow{\text{stepsize}_i})$ is the velocity ratio for phase 1 in which the predator is moving faster than the prey; ⊗ notation represents the entry-wise multiplications; the multiplication of R_B by prey simulates the movement of prey; P is a constant number which is equal to 0, 5 and finally, R is a vector of uniform random numbers in [0, 1]

ELSE IF: unit velocity ratio

$$\frac{\text{Maximum Cycle Number}}{3} < \text{Cycle} < 2.\frac{\text{Maximum Cycle Number}}{3}. \tag{7}$$

For the first half of the population $i = 1, 2, \ldots, n/2$ update prey based on,

$$\overrightarrow{\text{prey}_i} = \overrightarrow{\text{prey}_i} + P.\vec{R} \otimes \overrightarrow{\text{stepsize}_i}, \tag{8}$$

where, $\overrightarrow{\text{stepsize}_i} = \overrightarrow{R_L} \otimes (\overrightarrow{Elite_i} - \overrightarrow{R_L} \otimes \overrightarrow{\text{prey}_i})$ is the velocity ratio for phase 2 in which both predator and prey are moving at the same pace; $\overrightarrow{R_L}$ represents a vector of random numbers produced from Lévy distribution;

For the second half of the population $i = (n/2) + 1, \ldots, n$ update prey based on

$$\overrightarrow{\text{prey}_i} = \overrightarrow{\text{Elite}_i} + P.CF \otimes \overrightarrow{\text{stepsize}_i}, \tag{9}$$

where, $\overrightarrow{\text{stepsize}_i} = \overrightarrow{R_B} \otimes (\overrightarrow{R_B} \otimes \overrightarrow{\text{Elite}_i} - \overrightarrow{\text{prey}_i})$ is the velocity ratio for phase 2 in which both predator and prey are moving at the same pace; the multiplication of R_B and *Elite* simulates the movement of predator in Brownian manner and finally, $CF = (1 - (\text{Cycle}/\text{Max Cycle}))^{(2.(\text{Cycle}/\text{Max Cycle}))}$ is an adaptive control parameter.

ELSE IF Cycle > 2.((Maximum Cycle Number)/3) : low velocity ratio

Update prey, $\overrightarrow{\text{prey}_i} = \overrightarrow{\text{Elite}_i} + P.CF \otimes \overrightarrow{\text{stepsize}_i}$

where, $\overrightarrow{\text{stepsize}_i} = \overrightarrow{R_L} \otimes (\overrightarrow{R_L} \otimes \overrightarrow{\text{Elite}_i} - \overrightarrow{\text{prey}_i})$ is the velocity ratio for phase 3 in which the predator is moving faster than the prey; the multiplication of $\overrightarrow{R_L}$ and *Elite* simulates the movement of predator in Brownian manner

END IF

Memorize the best solutions so far and update *Elite* matrix Apply *Fish Aggregating Devices (FADs)* effect given below for $r \le FADs$ and $r > FADs$, respectively, in order to prevent the algorithm to get stuck into the local minima.

$$\overrightarrow{\text{prey}_i} = \begin{cases} \overrightarrow{\text{prey}_i} + CF.\left[\overrightarrow{X_{\min}} + R \otimes \left(\overrightarrow{X_{\max}} - \overrightarrow{X_{\min}}\right)\right] \otimes \vec{U}, \\ \overrightarrow{\text{prey}_i} + [FA\ Ds\ .(1-r) + r].\left(\overrightarrow{\text{prey}_{r1}} - \overrightarrow{\text{prey}_{r2}}\right), \end{cases} \tag{10}$$

where, $FADs = 0.2$ is the probability of FADs effect on the optimization process; $\vec{U}$ is a binary vector constructed by generating a random vector in [0,1]; $\overrightarrow{X_{\min}}$ and $\overrightarrow{X_{\max}}$ are the vectors containing the lower and upper bounds of the dimensions; the r_1 and r_2 represent random indexes of prey matrix.

END WHILE

2.3. Tunicate Swarm Algorithm. Tunicate Swarm algorithm is a bioinspired metaheuristic optimization algorithm inspired from the swarm behaviors of tunicates during the foraging process. It was proposed by Kaur et al. in 2020 [51]. TSA algorithm has been constructed mathematically on two main behaviors of tunicates that are jet propulsion and swarm intelligence.

A detailed pseudocode for a bioinspired swarm intelligence based TSA algorithm can be given as follows:

Initialize the tunicate population, $\overrightarrow{P_p(x)}$

Calculate the fitness value of each search agent in the initial population

Cycle = 1

REPEAT

Determine the jet propulsion and swarm behaviors of tunicates by,

(i) Jet Propulsion Behavior

$$\overrightarrow{P\,D} = \left|\overrightarrow{FS} - \text{rand}\,[0,1].\overrightarrow{P_p(x)}\right|, \tag{11}$$

where; $\overrightarrow{FS}$ represents the fitness value vector of search agents;

(ii) Swarm Behavior

$$\overrightarrow{P_p(x)} = \begin{cases} \overrightarrow{FS} + \vec{A}.\overrightarrow{P\,D} \text{ if } \text{rand}\,[0,1] \ge 0.5, \\ \overrightarrow{FS} - \vec{A}.\overrightarrow{P\,D} \text{ if } \text{rand}\,[0,1] < 0.5, \end{cases} \tag{12}$$

where, $\vec{A}$ vector includes the variations to avoid conflicts among search agents.

Update the position of each search agent by using

```
    $$\overrightarrow{P_p(x+1)} = \frac{\overrightarrow{P_p(x)} + \overrightarrow{P_p(x+1)}}{2+c_1}, \quad (13)$$
    where, $c_1$ is a randomly produced number in [0, 1].
    Calculate the fitness value for updated search agents
    Eliminate the search agents placing out of the search space
    Cycle = Cycle + 1
UNTİL (termination criteria are met)
```

2.4. Mayfly Optimization Algorithm. The basic philosophy of Mayfly optimization algorithm is to simulate the flight behavior and the mating process of mayflies. Mayfly is a hybrid metaheuristic algorithm combining the major advantages of swarm intelligence and evolutionary algorithms. It was proposed by Zervoudakis and Tsafarakis in 2020 [52] as an effective swarm intelligence based metaheuristic optimization algorithm in terms of convergence rate.

The detailed pseudocode with mathematical expressions for a hybrid MA can be given as follows:

```
Initialize the male mayfly population, $x_i \in \{1, 2, \ldots, N\}$ and velocities $v_{mi}$
Initialize the female mayfly population, $y_i \in \{1, 2, \ldots, N\}$ and velocities $v_{fi}$
Calculate the fitness value of each possible solution in the initial population
Cycle = 1
REPEAT
Update velocities and solutions of males and females by using,
 (i) Male Mayflies
    $$v_{ij}^{t+1} = v_{ij}^{t} + \alpha_1 e^{-\beta . r_p^2}\left(p\text{best}_{ij}\, x_{ij}^{t}\right) + \alpha_2 . e^{-\beta . r_g^2}\left(g\text{best}_{ij} - x_{ij}^{t}\right), \quad (14)$$
    where, $v_{ij}^t$ represents the velocity of mayfly $i$ in dimension $j = 1, \ldots, n$ at time step $t$; $x_{ij}^t$ is the position of mayfly $i$ in dimension $j$ at time step $t$; $a_1 > 0$ and $a_2 > 0$ are the attraction constants; $p\text{best}_i$ is the best position of mayfly $i$ so far; $g$best represents the global best position so far; finally, $\beta$ is a fixed visibility coefficient used to limit a mayfly's visibility to others while $r_p$ is the Cartesian distance between $x_i$ and, and $r_g$ is the Cartesian distance between $x_i$ and $g$best.
 (ii) Female Mayflies
    $$v_{ij}^{t+1} = \begin{cases} v_{ij}^{t} + \alpha_2 e^{-\beta . r_{mf}^2}\left(x_{ij}^{t} - y_{ij}^{t}\right) \text{ if } f(y_i) > f(x_i), \\ v_{ij}^{t} + (fl.r) \text{ if } f(y_i) \le f(x_i), \end{cases} \quad (15)$$
    where; $y_{ij}^t$ is the position of female mayfly $i$ in dimension $j$ at time step $t$; $f\,l$ is a random walk coefficient, used when a female is not attracted by a male, so it flies randomly and $r$ is a random value in the interval of [−1, 1]; finally, $r_{mf}$ represents the Cartesian distance between male and female.
Calculate the fitness value of each new candidate solution
Rank the mayflies
Mate the mayflies by applying crossover operation as mentioned below,
    the crossover operation is realized mutually by using the following equations between the candidates selected from male and female populations.
    $$\text{offspring1} = L.\text{ male } + (1 - L).\text{ female}, \quad \text{offspring2} = L.\text{ female } + (1 - L).\text{ male}, \quad (16)$$
    where, male and female represent the male and female parents, respectively; $L$ is a random value within a specific interval.
Calculate the fitness value of each offspring
Separate offsprings to male and female populations randomly
Replace worst solutions with the new better ones
Update $p$best and $g$best
Cycle = Cycle + 1
UNTİL (termination criteria are met)
```

2.5. Chimp Optimization Algorithm. The Chimp optimization algorithm inspired by the individual intelligence and sexual motivation of chimps in their group hunting has been proposed by Khishe and Mosavi in 2020 [53]. The mathematical model of ChOA algorithm has been built on two main phases. While the intelligence phase includes four types of chimps entitled attacker, barrier, chaser, and driver, the hunting phase has been modelled by considering driving, chasing, blocking, and attacking behaviors of chimps.

The detailed pseudocode of a nature-inspired swarm intelligence based ChOA algorithm can be given as follows:

```
Initialize the chimp population, $x_i \in \{1, 2, \ldots, N\}$
Calculate the position of each chimp for driving by,
    $$d = \left| c.X_{\text{prey}}(t) - m.X_{\text{chimp}}(t) \right|. \quad (17)$$
Calculate the position of each chimp for chasing by,
    $$X_{\text{chimp}}(t+1) = X_{\text{prey}}(t) - a.d, \quad (18)$$
where $a$, $m$ and $c$ are the coefficient vectors; $X_{\text{chimp}}$ represents the position vector of a chimp while $X_{\text{prey}}$ is the vector of prey position; also for the descriptions of $a = 2.f.r_1 - f$ and $c = 2.r_2$; $f$ represents reduced nonlinearly from 2.5 to 0 through the cycles process, $r_1$ and $r_2$ are the random vectors in [0, 1]; finally, $m$ represents a chaotic vector simulating the sexual motivation of chimps in the hunting phase.
```

```
Divide chimps randomly into four independent groups as,
    $X_{\text{Attacker}}$: the best search agent
    $X_{\text{Chaser}}$: the second best search agent
    $X_{\text{Barrier}}$: the third best search agent
    $X_{\text{Driver}}$: the fourth best search agent
Cycle = 1
REPEAT
Calculate the fitness value of each chimp
  WHILE (Cycle < Maximum Cycles).
    For each chimp:
      Identify chimp groups
      Use the group strategy to update $f$, $m$ and $c$
      Use $f$, $m$ and $c$ to calculate the new $a$ and $d$
    End For
      for each search chimp:
        if ($\mu < 0.5$), $\mu$ is a randomly produced number in [0, 1]
          if ($|a| < 1$)
        Update the position of the current search agent by using,
            $X_{\text{chimp}}(t+1) = X_{\text{prey}}(t) - a.d$
          else if ( $|a| > 1$ ).
        Select a random search agent
          end if
else if ( $\mu > 0.5$),
        Update the position of the current search agent by using
```

$$X_{\text{chimp}}(t+1), \quad \begin{cases} X_{\text{prey}}(t) - a.d \text{ if } \mu < 0.5, \\ m(\text{chaotic value}) \text{ if } \mu \geq 0.5. \end{cases} \tag{19}$$

```
      end if
    end for
  Update $f$, $m$, $a$ and $c$
  Update $X_{\text{Attacker}}$, $X_{\text{Chaser}}$, $X_{\text{Barrier}}$ and $X_{\text{Driver}}$
  END WHILE
Cycle = Cycle + 1
UNTIL (termination criteria are met).
```

2.6. Slime Mould Algorithm. The Slime Mould algorithm used in this work is a biological-evolution inspired algorithm, which simulates the diffusion and foraging behavior of slime mould. It was introduced by Li et al. in 2020 [54] as an effective physics-based metaheuristic algorithm with the detailed pseudocode given below:

```
Initialize the positions of slime mould, $X_i \in \{1, 2, \ldots, n\}$
Cycle = 1
WHILE Cycle ≤ Maximum Cycle
Calculate the fitness value of all slime mould
Determine the $bestFitness$, $X_b$
Calculate the weight of slime mould ($W$) by using
```

$$\overrightarrow{W(\text{SmellIndex}(t))} = \begin{cases} 1 + r.\log\left(\dfrac{bF - S(i)}{bF - wF} + 1\right), \text{condition}, \\ \\ 1 - r.\log\left(\dfrac{bF - S(i)}{bF - wF} + 1\right), \text{others}, \end{cases} \tag{20}$$

```
where, $r$ denotes a randomly produced value in the interval of [0,1]; $bF$ represents the optimal fitness value obtained in the current cycle; $wF$ is the worst fitness value obtained in the current cycle; $S(i)$ represents the fitness value of the current location of slime mould; $condition$ indicates that $S(i)$ ranks first half of the population; finally, $SmellIndex$ denotes the sequence of fitness values sorted.
  FOR each search portion,
    Update the values of control parameters $p$, $vb$ and $vc$ to realize variations in the population in the intervals of $\tanh|S(i) - \text{Best Fitness so far}|$, $[-a, a]$ and $[-1, 1]$, respectively.
    Update positions of slime mould by
```

$$\overrightarrow{X^*} = \begin{cases} rand\,(UB - LB) + LB, rand < z, \\ \\ \overrightarrow{X_B(t)} + \overrightarrow{vb}.\left(W.\overrightarrow{X_A(t)} - \overrightarrow{X_B(t)}\right), r < p, \\ \\ \overrightarrow{vc}.\overrightarrow{X(t)}, r \geq p, \end{cases} \tag{21}$$

```
    where, $LB$ and $UB$ denote the lower and upper boundaries of search space; $z$ is a value taken from a test table given in [52] according to problem type.
  END FOR
Cycle = Cycle + 1
END WHILE
```

2.7. Archimedes Optimization Algorithm. The Archimedes Optimization algorithm is a nondeterministic physics-based metaheuristic algorithm, which is established on Archimedes' Principle. It was improved by Hashim et al. in 2020 [55] so as to simulate the relationship between an object immersed in a water and buoyant force applied on it.

A detailed code of a physics-based AOA consisting mathematical explanations can be given as follows:

```
Randomly create an initial population consisting of immersed objects with random positions, densities, volumes and accelerations by using
```

$$\begin{aligned}&\text{positions: } O_i = lb_i + \text{rand}.\left(ub_i - lb_i\right),\\ &\quad i = 1, 2, \ldots, N,\\ &\text{density}\,(de\ n)\text{: } de\ n_i = \text{rand},\\ &\text{volume}\,(vol)\text{: } vol_i = \text{rand},\\ &\text{accerelation}\,(acc)\text{: } acc_i = lb_i + \text{rand}.\left(ub_i - lb_i\right),\end{aligned} \tag{22}$$

where; O_i is the i^{th} object in the population; lb_i and ub_i represents the lower and upper bounds of the search space, respectively; *rand* is a randomly generated D-dimensional vector having values between [0,1]; the updated density, volume, acceleration determines the new position of an object.

Evaluate initial population and determine the solution with the best fitness value

Cycle = 1

WHILE Cycle ≤ Maximum Cycle

FOR each object i

Update density and volume by the using the equations

$$\begin{aligned}de\ n_i^{t+1} &= de\ n_i^t + \text{rand}.\left(de\ n_{best} - de\ n_i^t\right),\\ vol_i^{t+1} &= vol_i^t + \text{rand}.\left(vol_{best} - vol_i^t\right),\end{aligned} \tag{23}$$

where; $de\ n_{\text{best}}$ and vol_{best} are the volume and *density* associated with the best object found so far; and *rand* is a uniformly distributed random number.

Update transfer and density decreasing factors TF and d,

$$\begin{aligned}TF &= e^{\left(t - t_{\max}/t_{\max}\right)},\\ d^{t+1} &= e^{\left(t - t_{\max}/t_{\max}\right)} - \left(\frac{t}{t_{\max}}\right).\end{aligned} \tag{24}$$

if $TF \le 0.5$: *Exploration phase*

Update acceleration by using

$$acc_i^{t+1} = \frac{de\ n_{mr} + vol_{mr}.acc_{mr}}{de\ n_i^{t+1}.vol_i^{t+1}}. \tag{25}$$

Normalize acceleration by using

$$acc_{i-norm}^{t+1} = u.\frac{acc_i^{t+1} - \min(acc)}{\max(acc) - \min(acc)} + l, \tag{26}$$

where; u and l represents the range of normalization and set to 0.9 and 0.1, respectively. $acc_{i-\text{norm}}^{t+1}$ determines the percentage of step that each agent will change.

Update position by taking $C_1 = 1$ or 2 ,

$$x_i^{t+1} = x_i^t + C_1.\text{rand}.acc_{i-\text{norm}}^{t+1}.d.\left(x_{\text{rand}} - x_i^t\right). \tag{27}$$

else $TF > 0.5$: Exploitation phase

Update acceleration by using the following equation:

$$acc_i^{t+1} = \frac{de\ n_{\text{best}} + vol_{\text{best}}.acc_{\text{best}}}{de\ n_i^{t+1}.vol_i^{t+1}}. \tag{28}$$

Normalize acceleration by using

$$acc_{i-\text{norm}}^{t+1} = u.\frac{acc_i^{t+1} - \min(acc)}{\max(acc) - \min(acc)} + l. \tag{29}$$

Update direction flag F by using the rule by taking $C_4 = 0.5$ or 1 and $P = 2.r\text{and} - C_4$,

$$F = \begin{cases} +1 \text{ if } P \le 0.5,\\ -1 \text{ if } P > 0.5.\end{cases} \tag{30}$$

Update position by using the equation given below for $C_2 = 6$ and T is a constant which is directly proportional to *TF*,

$$x_i^{t+1} = x_{\text{best}}^t + F.C_2.\text{rand}.acc_{i-\text{norm}}^{t+1}.d.\left(T.x_{\text{best}} - x_i^t\right). \tag{31}$$

end if

END FOR

Evaluate each object and select the one with the best fitness value

Cycle = Cycle + 1

END WHILE (until the termination criteria are met)

2.8. *Equilibrium Optimizer.*

Equilibrium optimizer, which was proposed by Faramarzi in 2020, is a novel physics-based metaheuristic algorithm inspired by dynamic source and sink models used to estimate equilibrium states [56].

A detailed pseudocode of a physics based EO algorithm can be given as follows:

Randomly create an initial population consisting of particles by using

$$C_i^{\text{initial}} = C_{\min} + \text{rand}_i.\left(C_{\max} - C_{\min}\right),\ i = 1, 2, \ldots, n, \tag{32}$$

where, C_i^{initial} is the initial concentration vector of the i^{th} particle; $C_{\min}$ and $C_{\max}$ respectively represents the lower and upper values for the dimensions; finally, $rand_i$ is a randomly generated vector in the interval of [0,1].

Evaluate each particle and sort them according to fitness values to assign equilibrium candidates where, the particles nominated as equilibrium candidates construct an equilibrium pool vector,

$$\overrightarrow{C_{eq.\text{pool}}} = \left\{\overrightarrow{C_{eq(1)}}, \overrightarrow{C_{eq(2)}}, \overrightarrow{C_{eq(3)}}, \overrightarrow{C_{eq(4)}}, \overrightarrow{C_{eq(ave)}}\right\}. \tag{33}$$

Determine the free parameters as $a_1 = 2$, $a_2 = 2$ and $GP = 0.5$; where, a_1 parameter controls the exploration ability of the algorithm; a_2 controls the exploitation ability of the algorithm; in all the simulations in this work and were taken as 2 and 1, respectively, as proposed in [54]; finally, *GP* is the generation probability used to control the update rate of the population.

```
Cycle = 1
WHILE Cycle ≤ Maximum Cycle
  FOR $i = 1, 2, \ldots, n$
  Calculate the fitness value of ith particle
    if fit$(\overrightarrow{C_i})$ < fit $(\overrightarrow{C_{eq(1)}})$
      Replace, $\overrightarrow{C_{eq(1)}}$ with $\overrightarrow{C_i}$ and fit $(\overrightarrow{C_{eq(1)}})$ with fit $(\overrightarrow{C_i})$
    else if fit $(\overrightarrow{C_i})$ > fit $(\overrightarrow{C_{eq(1)}})$ and fit $(\overrightarrow{C_i})$ < fit $(\overrightarrow{C_{eq(2)}})$
      Replace, $\overrightarrow{C_{eq(2)}}$ with $\overrightarrow{C_i}$ and fit $(\overrightarrow{C_{eq(2)}})$ with fit $(\overrightarrow{C_{eq(2)}})$
    else if fit $(\overrightarrow{C_i})$ > fit $(\overrightarrow{C_{eq(1)}})$ and fit$(\overrightarrow{C_i})$ > fit $(\overrightarrow{C_{eq(2)}})$
      and fit$(\overrightarrow{C_i})$ > fit $(\overrightarrow{C_{eq(3)}})$
      Replace, $\overrightarrow{C_{eq(3)}}$ with $\overrightarrow{C_i}$ and fit $(\overrightarrow{C_{eq(3)}})$ with fit $(\overrightarrow{C_i})$
    else if fit $(\overrightarrow{C_i})$ > fit $(\overrightarrow{C_{eq(1)}})$ and fit $(\overrightarrow{C_i})$ > fit $(\overrightarrow{C_{eq(2)}})$
      fit $(\overrightarrow{C_i})$ > fit $(\overrightarrow{C_{eq(1)}})$ and fit $(\overrightarrow{C_i})$ < fit $(\overrightarrow{C_{eq(4)}})$
      Replace, $\overrightarrow{C_{eq(4)}}$ with $\overrightarrow{C_i}$ and fit $(\overrightarrow{C_{eq(4)}})$ with fit $(\overrightarrow{C_i})$
    End if
  END FOR
Calculate $\overrightarrow{C_{ave}} = \left\{(\overrightarrow{C_{eq(1)}} + \overrightarrow{C_{eq(2)}} + \overrightarrow{C_{eq(3)}} + \overrightarrow{C_{eq(4)}})/4\right\}$ as in [56].
Construct the equilibrium pool as shown below:
```

$$\overrightarrow{C_{eq.\text{pool}}} = \left\{\overrightarrow{C_{eq(1)}}, \overrightarrow{C_{eq(2)}}, \overrightarrow{C_{eq(3)}}, \overrightarrow{C_{eq(4)}}, \overrightarrow{C_{eq(ave)}}\right\}. \quad (34)$$

```
Accomplish memory saving (if Cycle >1).
Assign $t = (1 - (\text{Cycle/Max Cycle}))^{(a_2.(\text{Cycle/Max Cycle}))}$
FOR
  Randomly choose one candidate from the equilibrium vector
  Generate random vectors of $\overrightarrow{\lambda}$, $\overrightarrow{r}$ and construct $\overrightarrow{F}$ by
```

$$\overrightarrow{F} = a_1.\,\text{sign}(\overrightarrow{r} - 0.5).\left(e^{\overrightarrow{\lambda}.t} - 1\right), \quad (35)$$

```
  where, $\overrightarrow{\lambda}$ is the control parameter called turnover rate; sign$(\overrightarrow{r} - 0.5)$ component effects on the direction of exploration and exploitation.
  Construct, $\overrightarrow{GCP} = \begin{cases} 0.5r_1 \text{ if } r_2 \geq GP, \\ 0 \text{ if } r_2 < GP. \end{cases}$
  Construct, $\overrightarrow{G_0} = \overrightarrow{GCP}.(\overrightarrow{C_{eq}} - \overrightarrow{\lambda}.\overrightarrow{C})$
  Construct generation rate, $\overrightarrow{G} = \overrightarrow{G_0}.\overrightarrow{F}$
  Update concentrations by using the following expression, $\overrightarrow{C} = \overrightarrow{C_{eq}} + (C - \overrightarrow{C_{eq}}).\overrightarrow{F} + \overrightarrow{G}/\overrightarrow{\lambda}.v(1 - \overrightarrow{F})$
  END FOR
Cycle = Cycle + 1
END WHILE
```

2.9. Retinal Vessel Segmentation. The Digital Retinal Images for Vessel Extraction (DRIVE) database [57] is the first open access database preferred in this work. The images in this database have been obtained by CanonCR5 3CCD camera under a 45° angle with a contrast of 565×584 pixels. The simulations have been realized on a total of 40 retinal images, half of which were used for training and the other half were used for testing.

The Structured Analysis of the Retina (STARE) [58] database is the second database preferred in this work. While constructing this database, a TopCon TRV-50 fundus camera has been used under a 35° angle with a contrast of 700×605 pixels. This database also contains 10 healthy retinal images and 10 diseased retinal images.

In the simulation, the pixels have been clustered as vessel pixels and nonvessel pixels. In order to create pixel intervals for both vessel pixels and nonvessel pixels, the number of cluster centers has been chosen as $k = 2$.

The optimal pixel values, each of which represents an optimal cluster center, have been found by using the improved JS, MPA, TSA, MA, ChOA, SMA, AOA, and EO algorithms.

At the beginning of the optimization process, each pixel value is assigned to the closest clustering centers based on the fitness values obtained. The fitness value of each possible cluster center can be calculated by using the mean squared error (MSE) function given as follows:

$$MSE = \frac{1}{M}\sum_{i=1}^{M}(f_i - y_i)^2, \quad (36)$$

where M is the total pixel number, f_i is the nearest clustering center value to the i^{th} pixel, and finally, the value of pixel i has been represented by y_i. From (36), it can be concluded that the clustering performance depends directly on the distance between the pixel values and their related clustering centers.

The encoding strategy used in the algorithms has a direct effect on optimization performance, and it should also be compatible with the optimization problem. During the metaheuristic based clustering processes improved in this work, the optimized cluster centers are obtained as real numbers. As a result of this, a Real-Valued Encoding Strategy has been applied to all algorithms. In this encoding scheme, for a clustering problem including k cluster centers, possible solutions as shown below are produced by the algorithms at each cycle (Table 1).

The four common control parameters of the metaheuristic algorithms used in this work are the Population Size, Maximum Cycle Number, $X_{\max}$ and $X_{\min}$. In the simulations, for each algorithm, the Population Sizes have been chosen as 10, and the Maximum Cycle Numbers have been chosen as 100. Also, $X_{\max}$ and $X_{\min}$ represent the highest and lowest pixel values of the relevant image, respectively.

Table 1: Real-valued encoding.

CC_1	CC_2	...	CC_k

where CC_k is a real value representing an optimized clustering center and taking values in the interval of [0,255].

The other control parameter values of the algorithms used in the simulations are given in Table 2.

3. Results

The retinal images emerging after segmentation process applied to the images in Figures 1(a) and 1(b) by using the clustering based JS, MPA, TSA, MA, ChOA, SMA, AOA, and EO algorithms improved in this work have been given in Figure 5. Similarly, the resulting images obtained after PSO, GWO, and DE based segmentation have been given in Figure 6. As seen from the figures, some pixels belonging to the image background but having pixel value close to the vessels may incorrectly be defined as vessel pixels. In order to overcome these difficulties encountered in segmentation, some postprocessing methods have been proposed in the literature. However, these postprocessing methods have not been used in this work in order to find out the pure performance of the algorithms. In general, it can be concluded that all the algorithms have performances too close to each other in terms of clustering. The simulation results represent that JS, MPA, TSA, MA, ChOA, SMA, AOA, and EO algorithms can distinguish the close pixel values with high accuracy similar to PSO, GWO, and DE algorithms. Figure 7 shows the clustering performance of the most recently proposed algorithms for the Figures 2(a) and 2(b) taken from the STARE database. Also, in Figure 8, the retinal images produced by PSO, GWO, and DE algorithms for STARE database have been demonstrated. The results represent that the algorithms produce poor performance in STARE database compared to DRIVE database. However, the simulation results prove that metaheuristic algorithms produce similar but a bit better performance than the conventional algorithms in literature in terms of clustering.

3.1. Performance Measures. Firstly, a detailed statistical performance comparison based on SE, SP, ACC, and F-score has been realized for the algorithms improved. Then, the convergence rate, MSE, CPU time, and standard deviation performances of the algorithms have been compared at each other.

The mathematical expressions of SE, SP, ACC, and F-score parameters can be given as follows:

$$\begin{aligned} SE &= \frac{TP}{(TP+FN)},\\ SP &= \frac{TN}{(TN+FP)},\\ ACC &= \frac{(TP+TN)}{(TP+FN+TN+FP)},\\ F-\text{score} &= \frac{TP}{[TP+1/2(FP+FN)]}, \end{aligned} \tag{37}$$

where TP is the correctly estimated vessel pixel number, FN is the number of vessel pixels incorrectly defined as background pixels, TN is the correctly estimated background pixel number, and finally FP is the number of background pixels incorrectly defined as vessel pixels.

3.2. Performance Analysis for DRIVE and STARE Databases. In Tables 3 and 4, the results relevant to sensitivity, specificity, and accuracy produced by the novel algorithms for 20 different images are given for DRIVE and STARE databases, respectively. Similarly, the results obtained for PSO, GWO, and DE algorithms have been given in Table 5 for DRIVE database and in Table 6 for STARE database.

In addition to the SE, SP, and ACC parameters, the *F*-score values obtained for each algorithms have also been given in Tables 7 and 8, respectively, for DRIVE and STARE databases.

For a fair comparison and the results to be more understandable, the mean values have also been calculated and added to the Tables. The results obtained for SE prove that all the algorithms can successfully classify the vessel pixels. On the other hand, higher SP values obtained in the simulations correspond to a successful distinguishing of the vessel pixels and background pixels. Finally, each algorithm can be expressed as successful enough in classifying both the vessel pixels and background due to the ACC values obtained. The F-score statistical parameter provides a single score that balances both the concerns of sensitivity and precision. The higher and similar F-score values produced by the algorithms prove the classifying success on whole image.

As mentioned before, for the statistical results to be more understandable, the mean values of the all statistical parameters have been combined in Tables 9 and 10. As seen from the statistical results given in these two tables, ChOA and PSO algorithms have produced a bit better results, respectively, for DRIVE and STARE databases. It can generally be said that the metaheuristic algorithms show similar clustering performances and can successfully be used for clustering based biomedical image processing.

In Tables 11 and 12, a detailed statistical performance comparison has been realized between the metaheuristic approaches analyzed in this work and the studies published in literature [59–78]. As seen from Table 11 given for DRIVE database, while the algorithms other than ChOA produce close results to [61] in terms of sensitivity, the ChOA algorithm produces a bit better sensitivity value than all the other algorithms. Also, due to their higher SE values, the performances of the JS, MPA, TSA, MA, ChOA, SMA, AOA, EO, PSO, GWO, and DE algorithms can be stated as too similar to [61, 76] for STARE database. On the other hand, the JS, MPA, TSA, MA, ChOA, SMA, AOA, EO, PSO, GWO, and DE algorithms can provide high SP performance as in the other studies in literature for both databases. Finally, in both databases, the ACC values obtained for each of the metaheuristic approaches seem similar to [61] but a bit better than the other algorithms. When the accuracy values obtained are evaluated, it can clearly be seen that the performance of the JS, MPA, TSA, MA, ChOA, SMA, AOA, EO,

Table 2: Control parameter values used in the simulations.

Algorithm	Control parameters
JS	(i) No other control parameters except for four common control parameters given in the text.
MPA	(i) Fish Aggregating Devices (FADs) = 0, 2
TSA	(i) Parameter $P_{\min} = 1$ (ii) Parameter $P_{\max} = 4$
MA	(i) Inertia weight, $g = 0{,}8$ (ii) Visibility coefficient, $\beta = 0,\ 2$ (iii) Nuptial dance, $d = 5$ (iv) Random flight, $f_1 = 1$ (v) Mutation rate, mu = 0, 01
ChOA	(i) m = chaotic vector (ii) f is reduced nonlinearly from 2,5 to 0 through the iteration process.
SMA	(i) Parameter $z = 0{,}03$
AOA	(i) Control variables $C_1 = 2,\ C_2 = 6,\ C_3 = 1,\ C_4 = 2$
EO	(i) Generation rate, GP = 0,5 (ii) Control parameters $a_1 = 2,\ a_2 = 1$
PSO	(i) Inertia factor, $w = 0{,}72$ (ii) Cognitive factor, $c_1 = 1{,}49$ (iii) Social factor, $c_2 = 1{,}49$
GWO	(i) The random parameters A and C assist candidate solutions to have hyper-spheres. (ii) Exploration and exploitation are guaranteed by the adaptive values of A and a.
DE	(i) Crossover rate = 0,8 (ii) Scaling factor = 0,2

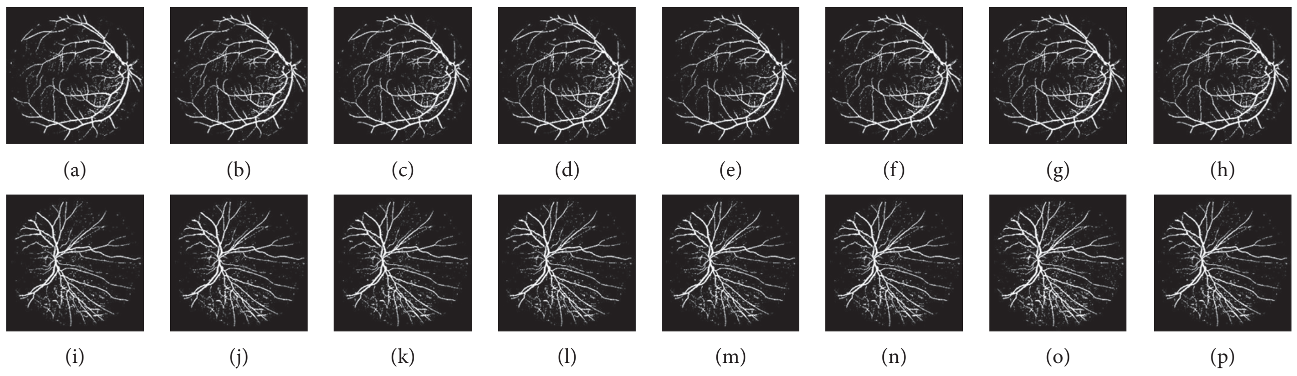


Figure 5: Retinal images obtained after applying segmentation to the images in Figures 1(a) and 1(b) by using the clustering based JS, MPA, TSA, MA, ChOA, SMA, AOA, and EO algorithms. (a) JS. (b) MPA. (c) TSA. (d) MA. (e) ChOA. (f) SMA. (g) AOA. (h) EO. (i) JS. (j) MPA. (k) TSA. (l) MA. (m) ChOA. (n) SMA. (o) AOA. (p) EO.

PSO, GWO, and DE algorithms in terms of pixel classification is higher than the conventional gradient based algorithms.

The retinal images contained in the DRIVE and STARE databases differ from each other in terms of contrast, brightness, vessel structures, and image sizes. Therefore, the analyses of the statistical performances of the improved algorithms on each database also become important [79]. The performance of each algorithm for DRIVE and STARE databases has been given in terms of mean SE, SP, ACC, and F-score in Figure 9. It can be seen from the figure that the algorithms can produce similar statistical performances on two separate databases containing different images and having different structures.

The convergence rates of the algorithms have been compared in Figure 10 for the diseased retinal image taken from DRIVE database and the healthy retinal image taken from STARE database. As seen from Figure 10(a) obtained for the diseased retinal image, the MA algorithm converges to the minimum error value at highest convergence rate. Similarly, when Figure 10(b) obtained for the healthy retinal image is examined, it can clearly be seen that MA reaches to the minimum error value at the highest convergence rate. In general, all the metaheuristic algorithms have reached their optimal MSE values approximately at 15 cycles and MA has produced the minimum MSE value for both databases.

Due to the different optimization phases they include, the Convergence Rate metric alone will be inadequate in the performance analysis of metaheuristic algorithms. In order to reflect the actual computational requirements, another performance criterion called *Number of Function Evaluations* (*NFEs*) has also to be analyzed. The *NFEs* defines the

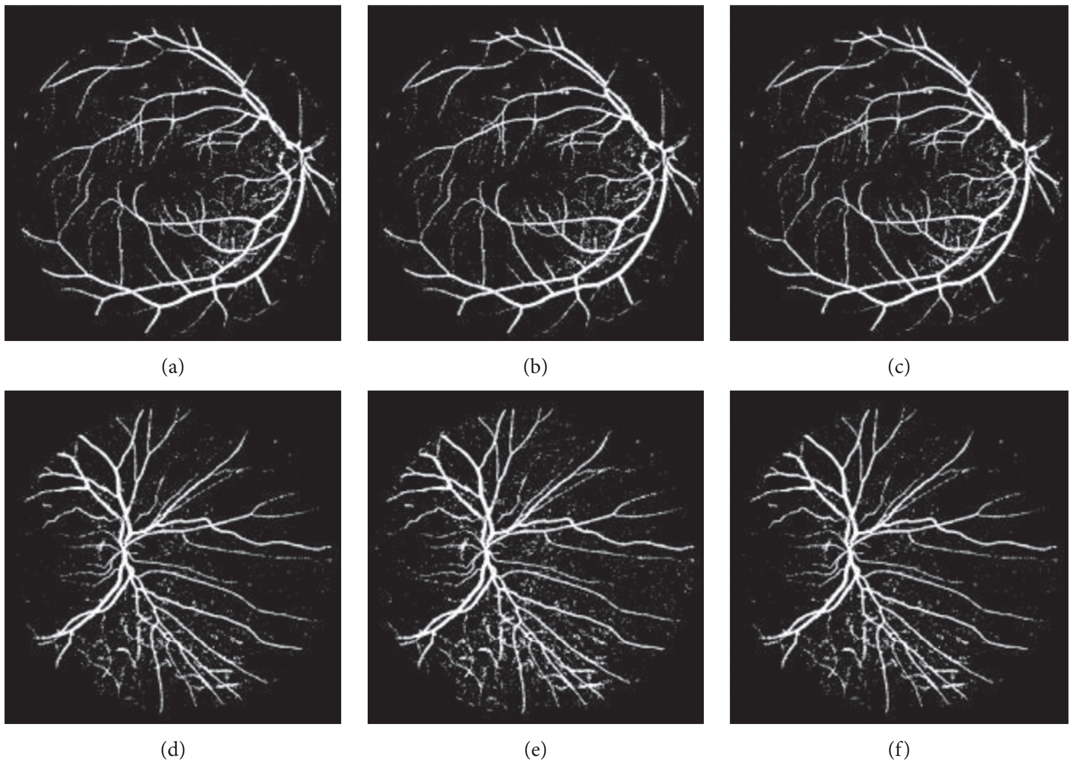


Figure 6: Retinal images obtained after applying segmentation to the images in Figures 1(a) and 1(b)) by using the clustering based PSO, GWO, and DE algorithms. (a) PSO. (b) GWO. (c) DE. (d) PSO. (e) GWO. (f) DE.

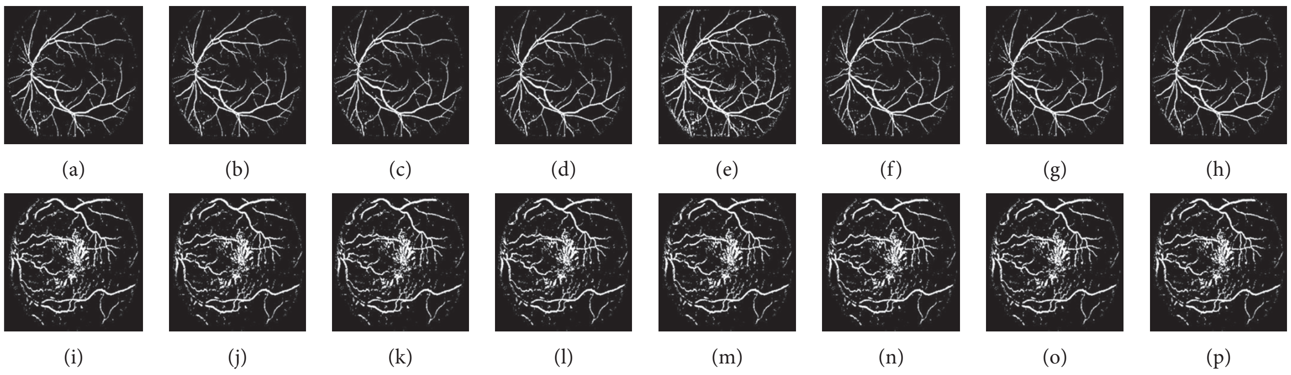


Figure 7: Retinal images obtained after applying segmentation to the images in Figures 2(a) and 2(b) by using the clustering based JS, MPA, TSA, MA, ChOA, SMA, AOA, and EO algorithms. (a) JS. (b) MPA. (c) TSA. (d) MA. (e) ChOA. (f) SMA. (g) AOA. (h) EO. (i) JS. (j) MPA. (k) TSA. (l) MA. (m) ChOA. (n) SMA. (o) AOA. (p) EO.

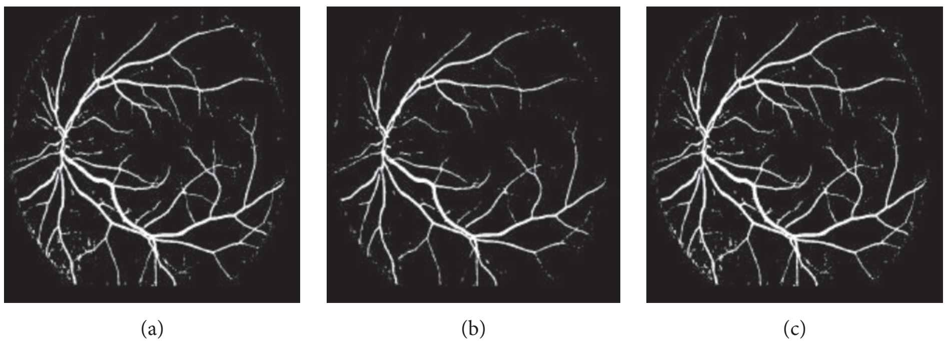


Figure 8: Continued.

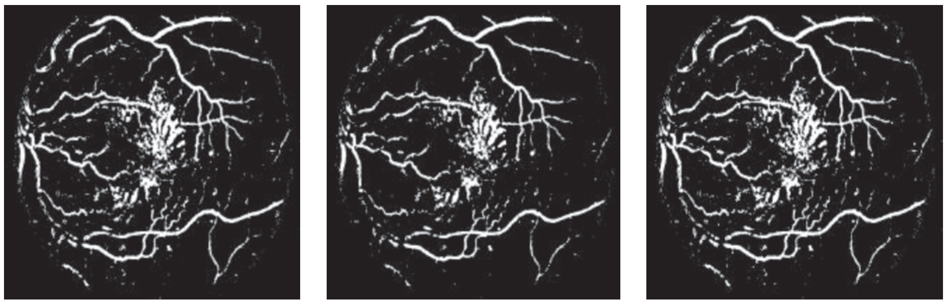


Figure 8: Retinal images obtained after applying segmentation to the images in Figures 2(a) and 2(b) by using the clustering based PSO, GWO, and DE algorithms. (a) PSO. (b) GWO. (c) DE. (d) PSO. (e) GWO. (f) DE.

number of times the objective function is evaluated during the optimization process, and it also provides more fair and objective results in terms of the performance analysis. In addition to the *NFEs* of each algorithm, the *Elapsed CPU times for NFEs* and the *Percentage of these times in the total CPU times* have also been calculated. The results related with the *NFEs* based analyses have been presented in Table 13. From the results, it can be concluded that the value of the *NFEs* parameter has a directly proportional effect on the CPU time. However, when Figure 10 and Table 13 have been evaluated simultaneously, it is also seen that the NFEs value does not cause an important variation on the minimum MSE values reached.

The standard deviation, which represents the stability of the algorithms, is one of the other important performance criteria that must be analyzed. If an algorithm can reach approximately similar MSE values at each cycle, the low standard deviation value of that algorithm will also be an indicator of its stability. The standard deviations of the algorithms for 20 random runs have been given in Table 14 for DRIVE and STARE databases.

For a more detailed analysis, the minimum MSE values reached and the CPU times required have also been added to Table 14. The close MSE values obtained for the algorithms correspond to a similar convergence and clustering performances for each algorithm. On the other hand, due to their lower standard deviation values for the images taken from both DRIVE and STARE databases, the MA and MPA algorithms seem a little more stable in terms of clustering when compared to other algorithms. Another important performance metric analyzed in this work is the CPU time value.

For a fair and realistic comparison, the convergence speed in addition to the CPU time parameter should also be analyzed. In this work, the simulations have been realized on an Intel i7-6700 CPU with 3,4 GHz frequency and 16 GB RAM. However, the operating system was chosen as 64 bit Windows 10 Pro. When the CPU time values obtained for the algorithms in both databases are examined, it is seen that although the MA is the algorithm with the highest convergence speed, it is obtained as the slowest algorithm in terms of CPU time. Similarly, the MPA algorithm, which is the second best algorithm (almost same as GWO) in terms of convergence speed, was obtained as the second worst algorithm in terms of the CPU time.

In order to statistically validate the numerical results obtained, the *Wilcoxon rank sum-test* which indicates the significant differences among algorithms has been applied [80]. The test results obtained on the RStudio-Software for the confidence interval of 95% ($p < 0.05$) have been given in Table 15. This table includes only the algorithms of which significant differences are ($p < 0.05$) obtained according to Wilcoxon rank sum-test results. The results obtained for DRIVE database indicate that the statistical performance of the MA algorithm is better than the other algorithms that produce statistically significant results. Similarly, it has also been observed for STARE database that the JS, MA, and MPA algorithms produce higher statistically significant results when compared to other algorithms. Finally, it is also seen for both databases that the AOA algorithm does not produce statistically significant results compared to other algorithms.

4. Discussion

Computer-aided biomedical image analysis provides effective approaches in diagnosing and treatment of diseases with high accuracy. Metaheuristic algorithms, as one of the most important computer aided approaches, have advantages such as global search ability, higher convergence rates, robustness, and flexibility.

The simulations have been realized by using JS, MPA, TSA, MA, ChOA, SMA, AOA, and EO algorithms, which are the most recently proposed metaheuristic algorithms in the literature. From the simulation results obtained for segmentation, it is seen that the metaheuristic algorithms produce similar and effective results for both healthy and diseased images in terms of convergence speed and error values reached. Furthermore, the results obtained in the detailed statistical analyses realized in this work prove the inherently stable, robust, and flexible behavior of the metaheuristic algorithms.

When the segmentation and statistical results obtained with high accuracy have been evaluated together, it can clearly be expressed that JS, MPA, TSA, MA, ChOA, SMA, AOA, and EO algorithms provide more effective and stable solutions in image analysis regardless of the image properties.

Table 3: The performance measures of the different images in the DRIVE database in terms of sensitivity, specificity, and accuracy.

	Sensitivity								Specificity								Accuracy							
Image	JS	MPA	TSA	MA	ChOA	SMA	AOA	EO	JS	MPA	TSA	MA	ChOA	SMA	AOA	EO	JS	MPA	TSA	MA	ChOA	SMA	AOA	EO
1	0.8913	0.8913	0.8913	0.8913	0.8913	0.8913	**0.996**	0.8913	0.9884	0.9884	0.9884	0.9884	0.9884	0.9884	**0.9996**	0.9884	0.9791	0.9791	0.9791	0.9791	0.9791	0.9791	**0.9993**	0.9791
2	0.7857	0.7857	0.7857	0.7857	0.7857	0.7857	0.8302	0.7857	0.9703	0.9703	0.9703	0.9703	0.9703	0.9703	0.9776	0.9703	0.9479	0.9479	0.9479	0.9479	0.9479	0.9479	0.9604	0.9479
3	0.8787	0.8787	0.8787	0.8787	0.8787	0.8787	0.8787	0.8787	0.9867	0.9867	0.9867	0.9867	0.9867	0.9867	0.9867	0.9867	0.976	0.976	0.976	0.976	0.976	0.976	0.976	0.976
4	0.723	0.723	0.723	0.723	0.723	0.723	0.723	0.723	0.9645	0.9645	0.9645	0.9645	0.9645	0.9645	0.9645	0.9645	0.9371	0.9371	0.9371	0.9371	0.9371	0.9547	0.9371	0.9371
5	0.9514	0.9514	0.9514	0.9514	0.9514	0.9514	0.8326	0.9514	**0.9955**	**0.9955**	**0.9955**	**0.9955**	0.9955	**0.9955**	0.9825	**0.9955**	**0.9918**	**0.9918**	**0.9918**	**0.9918**	0.9918	**0.9918**	0.9684	**0.9918**
6	**0.9572**	**0.9572**	**0.9572**	**0.9572**	0.9572	**0.9572**	0.9572	**0.9572**	0.9954	0.9954	0.9954	0.9954	0.9954	0.9954	0.9954	0.9954	0.9917	0.9917	0.9917	0.9917	0.9917	0.9917	0.9917	0.9917
7	0.8874	0.8874	0.8874	0.8874	0.9593	0.8874	0.8874	0.8874	0.9903	0.9903	0.9903	0.9903	**0.9967**	0.9903	0.9903	0.9903	0.9822	0.9822	0.9822	0.9822	0.994	0.9822	0.9822	0.9822
8	0.5506	0.5506	0.5506	0.5506	0.5506	0.5506	0.5506	0.5506	0.9449	0.9449	0.9449	0.9449	0.9449	0.9449	0.9449	0.9449	0.9019	0.9019	0.9019	0.9019	0.9019	0.9019	0.9019	0.9019
9	0.7818	0.7818	0.7818	0.7818	0.7818	0.7818	0.7818	0.7818	0.9761	0.9761	0.9761	0.9761	0.9761	0.9761	0.9761	0.9761	0.9568	0.9568	0.9568	0.9568	0.9568	0.9568	0.9568	0.9568
10	0.8932	0.8932	0.8932	0.8932	0.8932	0.9374	0.8932	0.8932	0.9908	0.9908	0.9908	0.9908	0.9908	0.9949	0.9908	0.9908	0.9831	0.9831	0.9831	0.9831	0.9831	0.9905	0.9831	0.9831
11	0.787	0.787	0.787	0.787	0.7499	0.787	0.8363	0.787	0.976	0.976	0.976	0.976	0.9706	0.976	0.9825	0.976	0.9569	0.9569	0.9569	0.9569	0.9474	0.9569	0.9685	0.9569
12	0.6624	0.6624	0.6624	0.6624	0.8609	0.6624	0.6624	0.6624	0.9574	0.9574	0.9574	0.9574	0.9861	0.9574	0.9574	0.9574	0.9243	0.9243	0.9243	0.9243	0.9747	0.9243	0.9243	0.9243
13	0.7992	0.7992	0.7992	0.7992	0.7407	0.7992	0.7992	0.7992	0.9725	0.9725	0.9725	0.9725	0.9621	0.9725	0.9725	0.9725	0.9516	0.9516	0.9516	0.9516	0.9338	0.9516	0.9516	0.9516
14	0.7439	0.7439	0.7439	0.7439	0.9137	0.7439	0.7439	0.7439	0.973	0.973	0.973	0.973	0.9924	0.973	0.973	0.973	0.9511	0.9511	0.9511	0.9511	0.9861	0.9511	0.9511	0.9511
15	0.9444	0.9444	0.9444	0.9444	0.9444	0.9444	0.8966	0.9444	0.9953	0.9953	0.9953	0.9953	0.9953	0.9953	0.9908	0.9953	0.9913	0.9913	0.9913	0.9913	0.9913	0.9913	0.9831	0.9913
16	0.8387	0.8387	0.8387	0.8387	0.8387	0.8387	0.8387	0.8387	0.9824	0.9824	0.9824	0.9824	0.9824	0.9824	0.9824	0.9824	0.9683	0.9683	0.9683	0.9683	0.9683	0.9683	0.9683	0.9683
17	0.7017	0.7017	0.7017	0.7017	0.874	0.7017	0.7017	0.7017	0.9672	0.9672	0.9672	0.9672	0.9886	0.9672	0.9672	0.9672	0.9409	0.9409	0.9409	0.9409	0.9791	0.9409	0.9409	0.9409
18	0.8972	0.8972	0.8972	0.8972	0.8972	0.8972	0.8972	0.8972	0.9885	0.9885	0.9885	0.9885	0.9885	0.9885	0.9885	0.9885	0.9794	0.9794	0.9794	0.9794	0.9794	0.9794	0.9794	0.9794
19	0.8161	0.8161	0.8161	0.8161	0.8161	0.8161	0.9312	0.8161	0.9758	0.9758	0.9758	0.9758	0.9758	0.9758	0.9919	0.9758	0.9572	0.9572	0.9572	0.9572	0.9572	0.9572	0.9856	0.9572
20	0.7477	0.7477	0.7477	0.7477	**0.9646**	0.7477	0.7477	0.7477	0.9667	0.9667	0.9667	0.9667	0.9963	0.9667	0.9667	0.9667	0.9412	0.9412	0.9412	0.9412	**0.9933**	0.9412	0.9412	0.9412
Mean	0.8119	0.8119	0.8119	0.8119	**0.8486**	0.8141	0.8192	0.8119	0.9778	0.9778	0.9778	0.9778	**0.9823**	0.9780	0.9790	0.9778	0.9604	0.9604	0.9604	0.9604	**0.9685**	0.9617	0.96254	0.9604

Bold values represent the best values obtained for the relevant metric.

Table 4: The performance measures of the different images in the STARE database in terms of sensitivity, specificity, and accuracy.

Image	Sensitivity								Specificity								Accuracy							
	JS	MPA	TSA	MA	ChOA	SMA	AOA	EO	JS	MPA	TSA	MA	ChOA	SMA	AOA	EO	JS	MPA	TSA	MA	ChOA	SMA	AOA	EO
1	0.7498	0.7498	0.7498	0.7498	0.7498	0.7498	0.7498	0.7498	0.9742	0.9742	0.9742	0.9742	0.9742	0.9742	0.9742	0.9742	0.9533	0.9533	0.9533	0.9533	0.9533	0.9533	0.9533	0.9533
2	0.6876	0.6876	0.6876	0.6876	0.6876	0.6876	0.6876	0.6876	0.9701	0.9701	0.9701	0.9701	0.9701	0.9701	0.9701	0.9701	0.9454	0.9454	0.9454	0.9454	0.9454	0.9454	0.9454	0.9454
3	0.6423	0.6423	0.6423	0.6423	0.6423	0.6423	0.6423	0.6423	0.9568	0.9568	0.9568	0.9568	0.9568	0.9568	0.9568	0.9568	0.9229	0.9229	0.9229	0.9229	0.9229	0.9229	0.9229	0.9229
4	0.3513	0.3513	0.3513	0.3513	0.3513	0.3513	0.3513	0.3513	0.8508	0.8508	0.8508	0.8508	0.8508	0.8508	0.8508	0.8508	0.7575	0.7575	0.7575	0.7575	0.7575	0.7575	0.7575	0.7575
5	0.8407	0.8407	0.8407	0.8407	0.8407	0.8407	0.9193	0.8407	0.9825	0.9825	0.9825	0.9825	0.9825	0.9825	0.9918	0.9825	0.9685	0.9685	0.9685	0.9685	0.9685	0.9685	0.9852	0.9685
6	0.8211	0.8211	0.8211	0.8211	0.8211	0.8211	0.9524	0.8211	0.981	0.981	0.981	0.981	0.981	0.981	0.9956	0.981	0.9656	0.9656	0.9656	0.9656	0.9656	0.9656	0.9919	0.9656
7	0.9324	**0.9945**	**0.9945**	**0.9945**	0.9324	0.9324	0.9301	**0.9945**	0.9937	**0.9995**	**0.9995**	**0.9995**	0.9937	0.9937	0.9934	**0.9995**	0.9884	**0.9991**	**0.9991**	**0.9991**	0.9884	0.9884	0.988	**0.9991**
8	**0.9987**	0.8955	0.8955	0.8955	0.8955	0.8955	0.8955	0.8955	**0.999**	0.9906	0.9906	0.9906	0.9906	0.9906	0.9906	0.9906	**0.9998**	0.9828	0.9828	0.9828	0.9828	0.9828	0.9828	0.9828
9	0.9281	0.9281	0.9281	0.9281	0.9281	0.9281	0.9281	0.9281	0.9932	0.9932	0.9932	0.9932	0.9932	0.9932	0.9932	0.9932	0.9876	0.9876	0.9876	0.9876	0.9876	0.9876	0.9876	0.9876
10	0.8143	0.8143	0.8143	0.8143	0.8143	0.8143	0.8143	0.8143	0.9796	0.9796	0.9796	0.9796	0.9796	0.9796	0.9796	0.9796	0.9633	0.9633	0.9633	0.9633	0.9633	0.9633	0.9633	0.9633
11	0.9377	0.9377	0.9377	0.9377	0.7301	0.9377	0.9377	0.9377	0.995	0.995	0.995	0.995	0.9726	0.995	0.995	0.995	0.9907	0.9907	0.9907	0.9907	0.9503	0.9907	0.9907	0.9907
12	0.8163	0.8163	0.8163	0.8163	0.8934	0.8163	0.8163	0.8163	0.9785	0.9785	0.9785	0.9785	0.9885	0.9785	0.9785	0.9785	0.9615	0.9615	0.9615	0.9615	0.9792	0.9615	0.9615	0.9615
13	0.8813	0.8813	0.8813	0.8813	0.8813	0.8813	0.8752	0.8813	0.9883	0.9883	0.9883	0.9883	0.9883	0.9883	0.9876	0.9883	0.9787	0.9787	0.9787	0.9787	0.9787	0.9787	0.9775	0.9787
14	0.9822	0.9822	0.9822	0.8918	**0.9822**	**0.9822**	0.8918	0.9822	0.9986	0.9986	0.9986	0.9905	**0.9986**	**0.9986**	0.9905	0.9986	0.9973	0.9973	0.9973	0.9825	**0.9973**	**0.9973**	0.9825	0.9973
15	0.7968	0.7968	0.7968	0.7968	0.7968	0.7968	0.6621	0.7968	0.9794	0.9794	0.9794	0.9794	0.9794	0.9794	0.9596	0.9794	0.9625	0.9625	0.9625	0.9625	0.9625	0.9625	0.9278	0.9625
16	0.9095	0.9095	0.9095	0.9095	0.9095	0.9095	0.9095	0.9095	0.9913	0.9913	0.9913	0.9913	0.9913	0.9913	0.9913	0.9913	0.9841	0.9841	0.9841	0.9841	0.9841	0.9841	0.9841	0.9841
17	0.9449	0.9449	0.9449	0.9449	0.9449	0.9449	**0.9612**	0.9449	0.9955	0.9955	0.9955	0.9955	0.9955	0.9955	**0.9969**	0.9955	0.9916	0.9916	0.9916	0.9916	0.9916	0.9916	**0.9942**	0.9916
18	0.7593	0.7593	0.7593	0.7593	0.7593	0.7593	0.9547	0.7593	0.9748	0.9748	0.9748	0.9748	0.9748	0.9748	0.9961	0.9748	0.9544	0.9544	0.9544	0.9544	0.9544	0.9544	0.9929	0.9544
19	0.9055	0.9055	0.9055	0.9055	0.9055	0.9055	0.9055	0.9055	0.9896	0.9896	0.9896	0.9896	0.9896	0.9896	0.9896	0.9896	0.9813	0.9813	0.9813	0.9813	0.9813	0.9813	0.9813	0.9813
20	0.8817	0.8817	0.8817	0.9014	0.8817	0.9014	0.9014	0.8817	0.9869	0.9869	0.9869	0.9813	0.9869	0.9893	0.9893	0.9869	0.9764	0.9764	0.9764	0.9807	0.9764	0.9807	0.9807	0.9764
Mean	0.8290	0.8270	0.8270	0.8234	0.8173	0.8249	**0.8343**	0.8270	0.9779	0.9778	0.9778	0.9771	0.9769	0.9776	**0.9785**	0.9778	0.9615	0.9612	0.9612	0.9607	0.9595	0.9609	**0.9625**	0.9612

Bold values represent the best values obtained for the relevant metric.

Table 5: The performance measures of the different images in the DRIVE database in terms of sensitivity, specificity and accuracy for PSO, GWO and DE.

Image	Sensitivity			Specificity			Accuracy		
	PSO	GWO	DE	PSO	GWO	DE	PSO	GWO	DE
1	0.8913	0.8606	0.8913	0.9884	0.9847	0.9884	0.9791	0.9724	0.9791
2	0.7857	0.8302	0.8302	0.9703	0.9776	0.9776	0.9479	0.9604	0.9604
3	0.8787	0.5392	0.8787	0.9867	0.923	0.9867	0.976	0.868	0.976
4	0.723	0.7793	0.723	0.9645	0.9735	0.9645	0.9371	0.9527	0.9371
5	0.9514	0.9514	0.9514	0.9955	0.9955	0.9955	**0.9918**	0.9918	0.9918
6	**0.9572**	0.8605	0.9572	0.9954	0.9836	0.9954	0.9917	0.9707	0.9917
7	0.8874	0.7331	0.8874	0.9903	0.9727	0.9903	0.9822	0.9504	0.9822
8	0.5506	0.8672	0.5506	0.9449	0.9892	0.9449	0.9019	0.98	0.9019
9	0.7818	0.8251	0.7818	0.9761	0.9817	0.9761	0.9568	0.9669	0.9568
10	0.8932	0.8932	0.718	0.9908	0.9908	0.9705	0.9831	0.9831	0.9467
11	0.787	**0.9879**	0.787	0.976	**0.9989**	0.976	0.9569	**0.998**	0.9569
12	0.6624	0.6824	0.6624	0.9574	0.9609	0.9574	0.9243	0.9305	0.9243
13	0.7992	0.6489	0.7992	0.9725	0.9426	0.9725	0.9516	0.9013	0.9516
14	0.7439	0.9292	0.7439	0.973	0.9939	0.973	0.9511	0.9887	0.9511
15	0.9444	0.5551	0.9444	0.9953	0.9393	0.9953	0.9913	0.8932	0.9913
16	0.8387	0.7084	0.8387	0.9824	0.9631	0.9824	0.9683	0.9345	0.9683
17	0.7017	0.7017	0.7017	0.9672	0.9672	0.9672	0.9409	0.9409	0.9409
18	0.8972	0.7438	0.8972	0.9885	0.9663	0.9885	0.9794	0.9405	0.9794
19	0.8161	0.5084	0.8161	0.9758	0.9037	0.9578	0.9572	0.839	0.9572
20	0.7477	0.7464	**0.9646**	**0.9967**	0.9665	**0.9963**	0.9412	0.9408	**0.9933**
Mean	0.8119	0.7676	**0.8162**	**0.9793**	0.9687	0.9778	0.9604	0.9451	**0.9619**

Bold values represent the best values obtained for the relevant metric.

Table 6: The performance measures of the different images in the STARE database in terms of sensitivity, specificity, and accuracy for PSO, GWO, and DE.

Image	Sensitivity			Specificity			Accuracy		
	PSO	GWO	DE	PSO	GWO	DE	PSO	GWO	DE
1	0.7205	0.9611	0.7205	0.9702	0.9968	0.9702	0.9461	0.9941	0.9461
2	0.6876	0.8461	0.6876	0.9701	0.9878	0.9701	0.9454	0.9774	0.9454
3	0.6423	**0.9761**	0.6423	0.9568	0.998	0.9568	0.9229	**0.9963**	0.9229
4	0.6634	0.6634	0.3513	0.954	0.954	0.8508	0.9191	0.9191	0.7575
5	0.8407	0.7572	0.8407	0.9825	0.9708	0.9825	0.9685	0.9479	0.9685
6	0.8211	0.7104	0.8789	0.981	0.9649	0.9879	0.9656	0.9374	0.9779
7	**0.9945**	0.7999	**0.9945**	**0.9995**	0.9785	**0.9995**	**0.9991**	0.9612	**0.9991**
8	0.8955	0.7947	0.8955	0.9906	0.9795	0.9906	0.9828	0.9627	0.9828
9	0.9281	0.9281	0.9281	0.9932	0.9932	0.9932	0.9876	0.9876	0.9876
10	0.8143	0.925	0.8143	0.9796	0.9927	0.9796	0.9633	0.9866	0.9633
11	0.9377	0.7654	0.9377	0.995	0.9772	0.995	0.9907	0.9584	0.9907
12	0.8163	0.8163	0.7862	0.9785	0.9785	0.9742	0.9615	0.9615	0.9539
13	0.8813	0.7034	0.8813	0.9883	0.9643	0.9883	0.9787	0.9362	0.9787
14	0.9822	0.7102	0.8918	0.9986	0.9687	0.9905	0.9973	0.9434	0.9825
15	0.7968	0.9393	0.7968	0.9794	0.9947	0.9794	0.9625	0.9902	0.9625
16	0.9905	0.9095	0.9095	0.9913	0.9913	0.9913	0.9841	0.9841	0.9841
17	0.9449	0.9612	0.9449	0.9955	**0.9969**	0.9955	0.9916	0.9942	0.9916
18	0.7593	0.7593	0.7593	0.9748	0.9748	0.9748	0.9544	0.9544	0.9544
19	0.9095	0.9055	0.9095	0.9896	0.9896	0.9896	0.9813	0.9813	0.9813
20	0.8817	0.6438	0.7621	0.9869	0.9482	0.9701	0.9764	0.9095	0.9468
Mean	**0.8454**	0.8237	0.8166	**0.9827**	0.9800	0.9764	**0.9689**	0.9641	0.9588

Bold values represent the best values obtained for the relevant metric.

Table 7: The performance measures of the different images in the DRIVE database in terms of *F*-score.

Image	*F*-score										
	JS	MPA	TSA	MA	ChOA	SMA	AOA	EO	PSO	GWO	DE
1	0.7831	0.7831	0.7831	0.7831	0.7831	0.7831	0.7853	0.7831	0.7831	0.7638	0.7831
2	**0.7904**	**0.7904**	**0.7904**	**0.7904**	**0.7904**	**0.7904**	**0.8065**	**0.7904**	**0.7904**	**0.8065**	**0.8065**
3	0.7043	0.7043	0.7043	0.7043	0.7043	0.7043	0.7043	0.7043	0.7043	0.6055	0.7043
4	0.7221	0.7221	0.7221	0.7221	0.7221	0.7632	0.7221	0.7221	0.7221	0.7162	0.7221
5	0.766	0.766	0.766	0.766	0.766	0.766	0.7322	0.766	0.766	0.766	0.766
6	0.711	0.711	0.711	0.711	0.711	0.711	0.711	0.711	0.711	0.723	0.711
7	0.7089	0.7089	0.7089	0.7089	0.722	0.7089	0.7089	0.7089	0.7089	0.6757	0.7089
8	0.587	0.587	0.587	0.587	0.587	0.587	0.587	0.587	0.587	0.7029	0.587
9	0.6563	0.6563	0.6563	0.6563	0.6563	0.6563	0.6563	0.6563	0.6563	0.6912	0.6563
10	0.7574	0.7574	0.7574	0.7574	0.7574	0.7392	0.7574	0.7574	0.7574	0.7574	0.6835
11	0.7418	0.7418	0.7418	0.7418	0.7244	0.7418	0.7555	0.7418	0.7418	0.7546	0.7418
12	0.6591	0.6591	0.6591	0.6591	0.7161	0.6591	0.6591	0.6591	0.6591	0.6575	0.6591
13	0.7246	0.7246	0.7246	0.7246	0.7055	0.7246	0.7246	0.7246	0.7246	0.6058	0.7246
14	0.706	0.706	0.706	0.706	0.7471	0.706	0.706	0.706	0.706	0.7623	0.706
15	0.7037	0.7037	0.7037	0.7037	0.7037	0.7037	0.6886	0.7037	0.7037	0.5825	0.7037
16	0.7843	0.7843	0.7843	0.7843	0.7843	0.7843	0.7843	0.7843	0.7843	0.7046	0.7843
17	0.6701	0.6701	0.6701	0.6701	0.7101	0.6701	0.6701	0.6701	0.6701	0.6701	0.6701
18	0.7538	0.7538	0.7538	0.7538	0.7538	0.7538	0.7538	0.7538	0.7538	0.6981	0.7538
19	0.7691	0.7691	0.7691	0.7691	0.7691	0.7691	0.7661	0.7691	0.7691	0.5967	0.7691
20	0.6696	0.6696	0.6696	0.6696	0.7083	0.6696	0.6696	0.6696	0.6696	0.7055	0.7083
Mean	0.7184	0.7184	0.7184	0.7184	**0.7261**	0.7195	0.7174	0.7184	0.7184	0.6972	0.7174

Bold values represent the best values obtained for the relevant metric.

Table 8: The performance measures of the different images in the STARE database in terms of *F*-score.

Image	*F*-score										
	JS	MPA	TSA	MA	ChOA	SMA	AOA	EO	PSO	GWO	DE
1	0.601	0.601	0.601	0.601	0.601	0.601	0.601	0.601	0.601	0.5897	0.601
2	0.5594	0.5594	0.5594	0.5594	0.5594	0.5594	0.5594	0.5594	0.5594	0.5708	0.5594
3	0.568	0.568	0.568	0.568	0.568	0.568	0.568	0.568	0.568	0.6102	0.568
4	0.4385	0.4385	0.4385	0.4385	0.4385	0.4385	0.4385	0.4385	0.5233	0.5233	0.4385
5	0.6418	0.6418	0.6418	0.6418	0.6418	0.6418	0.6711	0.6418	0.6418	0.6137	0.6418
6	0.6522	0.6522	0.6522	0.6522	0.6522	0.6522	0.6681	0.6522	0.6522	0.5906	0.6652
7	0.7231	0.7248	0.7248	0.7248	0.7231	0.7231	0.724	0.7248	0.7248	0.7015	0.7248
8	0.7123	0.7098	0.7098	0.7098	0.7098	0.7098	0.7098	0.7098	0.7098	0.6986	0.7098
9	**0.7271**	**0.7271**	**0.7271**	**0.7271**	**0.7271**	**0.7271**	**0.7271**	**0.7271**	**0.7271**	**0.7271**	**0.7271**
10	0.6191	0.6191	0.6191	0.6191	0.6191	0.6191	0.6191	0.6191	0.6191	0.5992	0.6191
11	0.7131	0.7131	0.7131	0.7131	0.6903	0.7131	0.7131	0.7131	0.7131	0.6682	0.7131
12	0.6996	0.6996	0.6996	0.6996	0.7227	0.6996	0.6996	0.6996	0.6996	0.6996	0.6842
13	0.6367	0.6367	0.6367	0.6367	0.6367	0.6367	0.6582	0.6367	0.6367	0.5642	0.6367
14	0.6969	0.6969	0.6969	0.6976	0.6969	0.6969	0.6976	0.6969	0.6969	0.6783	0.6976
15	0.6686	0.6686	0.6686	0.6686	0.6686	0.6686	0.6335	0.6686	0.6686	0.6865	0.6686
16	0.6938	0.6938	0.6938	0.6938	0.6938	0.6938	0.6938	0.6938	0.6938	0.6938	0.6938
17	0.7082	0.7082	0.7082	0.7082	0.7082	0.7082	0.7097	0.7082	0.7082	0.7097	0.7082
18	0.5525	0.5525	0.5525	0.5525	0.5525	0.5525	0.5635	0.5525	0.5525	0.5525	0.5525
19	0.4678	0.4678	0.4678	0.4678	0.4678	0.4678	0.4678	0.4678	0.4678	0.4678	0.4678
20	0.5552	0.5552	0.5552	0.5788	0.5552	0.5788	0.5788	0.5552	0.5552	0.4092	0.516
Mean	0.6317	0.6317	0.6317	0.6329	0.6316	0.6328	0.6350	0.6317	**0.6359**	0.6177	0.6296

Bold values represent the best values obtained for the relevant metric.

Table 9: The statistical performances of the JS, MPA, TSA, MA, ChOA, SMA, AOA, EO, PSO, GWO, and DE algorithms for the retinal images taken from DRIVE database.

Method	Sensitivity	Specificity	Accuracy	*F*-score
JS	0.81193	0.977885	0.96049	0.71843
MPA	0.81193	0.977885	0.96049	0.71843
TSA	0.81193	0.977885	0.96049	0.71843
MA	0.81193	0.977885	0.96049	0.71843
ChOA	**0.84862**	**0.98237**	**0.9685**	**0.7261**
SMA	0.81414	0.97809	0.96174	0.719575

Table 9: Continued.

Method	Sensitivity	Specificity	Accuracy	*F*-score
AOA	0.81928	0.979065	0.962545	0.717435
EO	0.81193	0.977885	0.96049	0.71843
PSO	0.81193	0.979385	0.96049	0.71843
GWO	0.7676	0.968735	0.94519	0.697295
DE	0.81624	0.977815	0.9619	0.717475

Bold values represent the best values obtained for the relevant metric.

Table 10: The statistical performances of the JS, MPA, TSA, MA, ChOA, SMA, AOA, EO, PSO, GWO, and DE algorithms for the retinal images taken from STARE database.

Method	Sensitivity	Specificity	Accuracy	*F*-score
JS	0.829075	0.97794	0.96154	0.631745
MPA	0.82702	0.97781	0.961225	0.631705
TSA	0.82702	0.97781	0.961225	0.631705
MA	0.823485	0.977125	0.9607	0.63292
ChOA	0.81739	0.9769	0.959555	0.631635
SMA	0.8249	0.97764	0.960905	0.6328
AOA	0.834305	0.978525	0.962555	0.635085
EO	0.82702	0.97781	0.961225	0.631705
PSO	**0.84541**	**0.98277**	**0.968945**	**0.635945**
GWO	0.823795	0.98002	0.964175	0.617725
DE	0.81664	0.976495	0.95888	0.62966

Bold values represent the best values obtained for the relevant metric.

Table 11: Performance comparison of JS, MPA, TSA, MA, ChOA, SMA, AOA, EO, PSO, GWO, and DE algorithms and other methods for DRIVE database.

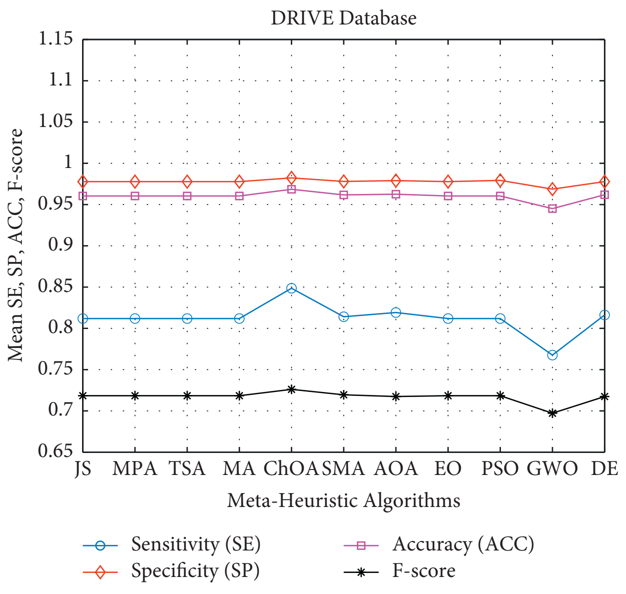
Method	Sensitivity	Specificity	Accuracy
You et al. [59]	0.7410	0.9751	0.9434
Roychowdhury et al. [60]	0.725	**0.983**	0.952
Wang et al. [61]	**0.8173**	0.9733	**0.9767**
Mendonca and Campilho [62]	0.7344	0.9764	0.9452
Azzopardi et al. [63]	0.765	0.970	0.944
Fraz et al. [64]	0.7406	0.9807	0.9480
Odstrcilik et al. [65]	0.706	0.9693	0.934
Marin et al. [66]	0.7068	0.9801	0.9452
Kaba et al. [67]	0.7466	0.968	0.941
Dash and Bhoi [68]	0.719	0.976	0.955
Argüello et al. [69]	0.7209	0.9758	0.9431
Asad et al. [70]	0.7388	0.9288	0.9028
Zhao et al. [71]	0.735	0.978	0.947
Zhang et al. [72]	0.7120	0.9724	0.9382
Imani et al. [73]	0.7524	0.9753	0.9523
Metaheuristic algorithms			
JS	0.81193	0.977885	0.96049
MPA	0.81193	0.977885	0.96049
TSA	0.81193	0.977885	0.96049
MA	0.81193	0.977885	0.96049
ChOA	**0.84862**	**0.98237**	**0.9685**
SMA	0.81414	0.97809	0.96174
AOA	0.81928	0.979065	0.962545
EO	0.81193	0.977885	0.96049
PSO	0.81193	0.979385	0.96049
GWO	0.7676	0.968735	0.94519
DE	0.81624	0.977815	0.9619

Bold values represent the best values obtained for the relevant metric.

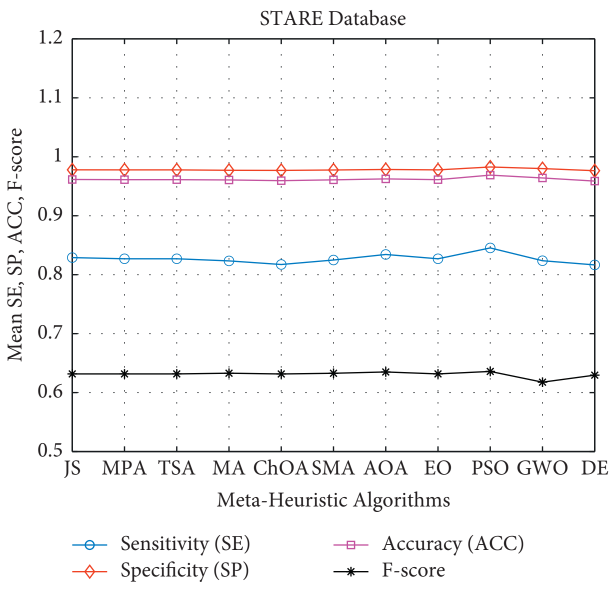
Table 12: Performance comparison of JS, MPA, TSA, MA, ChOA, SMA, AOA, and EO algorithms and other methods for STARE database.

Method	Sensitivity	Specificity	Accuracy
You et al. [59]	0.726	0.976	0.950
Roychowdhury et al. [60]	0.772	0.973	0.952
Wang et al. [61]	0.8104	**0.9791**	**0.9813**
Fraz et al. [64]	0.755	0.976	0.953
Odstrcilik et al. [65]	0.7847	0.9512	0.9341
Kaba et al. [67]	0.7619	0.967	0.9456
Argüello et al. [69]	0.7305	0.9688	0.9448
Zhao et al. [71]	0.719	0.977	0.951
Zhang et al. [72]	0.718	0.975	0.948
Imani et al. [73]	0.7502	0.9745	0.959
Mendonca et al. [74]	0.718	0.973	0.946
Xiao et al. [75]	0.715	0.974	0.948
Yin et al. [76]	**0.854**	0.942	0.933
Lázár and Hajdu [77]	0.725	0.975	0.949
Strisciuglio et al. [78]	0.801	0.972	0.954
Metaheuristic algorithms			
JS	0.829075	0.97794	0.96154
MPA	0.82702	0.97781	0.961225
TSA	0.82702	0.97781	0.961225
MA	0.823485	0.977125	0.9607
ChOA	0.81739	0.9769	0.959555
SMA	0.8249	0.97764	0.960905
AOA	0.834305	0.978525	0.962555
EO	0.82702	0.97781	0.961225
PSO	**0.84541**	**0.98277**	**0.968945**
GWO	0.823795	0.98002	0.964175
DE	0.81664	0.976495	0.95888

Bold values represent the best values obtained for the relevant metric.

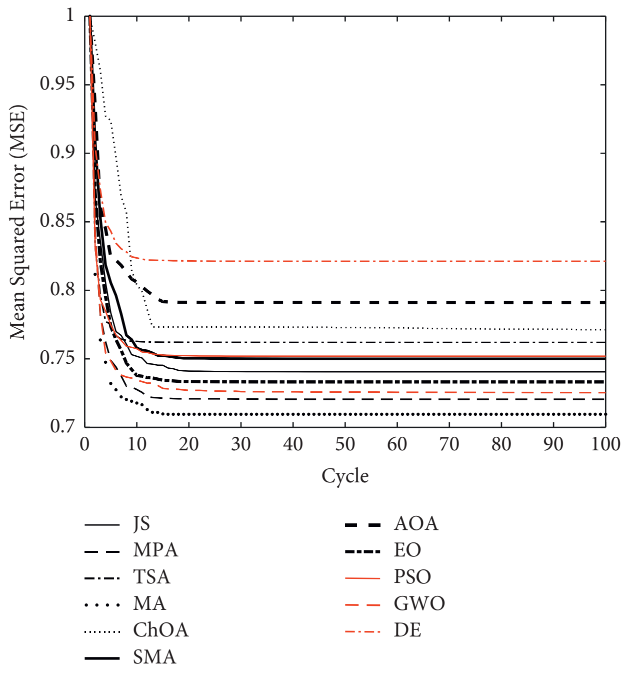


(a)

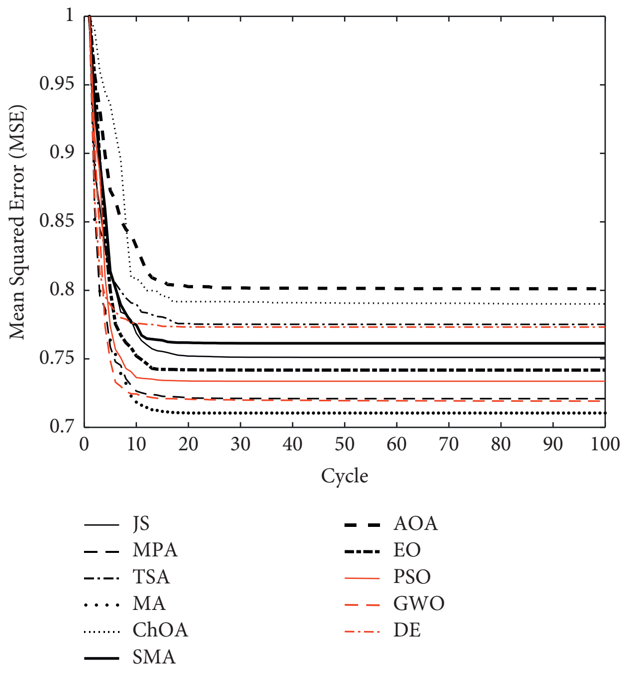


(b)

Figure 9: Statistical performance analyzes of the improved algorithms for DRIVE and STARE databases. (a) DRIVE database. (b) STARE database.



(a)



(b)

FIGURE 10: Convergence speeds of the algorithms. (a) Convergence rates to minimum MSE error values obtained in the simulations realized for Figure 1(b). (b) Convergence rates to minimum MSE error values obtained in the simulations realized for Figure 2(a).

TABLE 13: Number of function evolutions (NFEs) based convergence analyzes of the metaheuristic algorithms.

Algorithm	Total NFEs	Elapsed time for NFEs (seconds)	
		Figure 1(b)	Figure 2(a)
JS	1010	3.192 s (%80.90 of the total CPU time)	3.988 s (%82.10 of the total CPU time)
MPA	2000	6.346 s (%81.53 of the total CPU time)	7.886 s (%82.67 of the total CPU time)
TSA	1000	3.264 s (%82.19 of the total CPU time)	4.011 s (%82.88 of the total CPU time)
MA	3120	9.752 s (%80.21 of the total CPU time)	12.043 s (%81.23 of the total CPU time)
ChOA	1000	3.244 s (%81.30 of the total CPU time)	4.060 s (%82.58 of the total CPU time)
SMA	1000	3.238 s (%81.14 of the total CPU time)	4.073 s (%82.61 of the total CPU time)
AOA	1010	3.419 s (%82.25 of the total CPU time)	4.112 s (%82.26 of the total CPU time)
EO	1000	3.151 s (%81.40 of the total CPU time)	3.976 s (%82.22 of the total CPU time)
PSO	1010	3.230 s (%80.79 of the total CPU time)	4.083 s (%82.15 of the total CPU time)
GWO	1010	3.313 s (%82.05 of the total CPU time)	4.105 s (%82.83 of the total CPU time)
DE	1010	3.221 s (%80.01 of the total CPU time)	4.067 s (%81.71 of the total CPU time)

TABLE 14: Performance comparison of the JS, MPA, TSA, MA, ChOA, SMA, AOA, and EO algorithms for DRIVE and STARE databases.

		DRIVE	STARE
JS	Minimum MSE	**0.7421**	0.7507
	Standard deviation	1.03908$e-05$	**3.11027e – 07**
	CPU time (second)	**3.9452**	4.8571
MPA	Minimum MSE	**0.7215**	0.7218
	Standard deviation	7.72127$e-09$	**4.84936e – 09**
	CPU time (second)	**7.7833**	9.5382
TSA	Minimum MSE	**0.7645**	0.7782
	Standard deviation	0.020777359	**0.01876178**
	CPU time (second)	**3.9710**	4.8395
MA	Minimum MSE	**0.7058**	0.71
	Standard deviation	2.54232$e-10$	**2.5175e – 10**
	CPU time (second)	**12.1566**	14.8251

Table 14: Continued.

		DRIVE	STARE
ChOA	Minimum MSE	**0.7726**	0.7928
	Standard deviation	**0.013645648**	0.308106679
	CPU time (second)	**3.9897**	4.9164
SMA	Minimum MSE	**0.7512**	0.7619
	Standard deviation	**0.000223977**	7.81482$e-05$
	CPU time (second)	**3.9906**	4.9299
AOA	Minimum MSE	**0.7933**	0.8
	Standard deviation	**0.112297412**	0.150191448
	CPU time (second)	**4.1567**	4.9382
EO	Minimum MSE	**0.7328**	0.7416
	Standard deviation	2.51$e-07$	**9.04214e – 06**
	CPU time (second)	**3.8706**	4.8354
	Minimum MSE	0.752	**0.7337**
PSO	Standard deviation	**2.58125e – 08**	3.06854$e-08$
	CPU time (second)	**3.9976**	4.9700
	Minimum MSE	0.7253	**0.719**
GWO	Standard deviation	**5.45881e – 06**	9.51344$e-06$
	CPU time (second)	**4.0375**	4.9556
	Minimum MSE	0.8212	**0.7732**
DE	Standard deviation	**2.15721e – 05**	3.41578$e-05$
	CPU time (second)	**4.0257**	4.9771

Bold values represent the best values obtained for the relevant metric.

Table 15: The Wilcoxon sum-rank test results of the metaheuristic algorithms ($p < 0.05$).

	Algorithm	Better than
DRIVE database (Figure 1(b))	AOA	
	ChOA	AOA (6.6063$e-08$)
	DE	AOA (3.3675$e-08$) ChOA (3.4729$e-08$) SMA (2.7754$e-06$) TSA (3.4729$e-08$) GWO (3.3254$e-09$)
	EO	AOA (2.3615$e-08$) ChOA (2.4387$e-08$) SMA (4.6093$e-08$) TSA (2.4387$e-08$) GWO (2.0841$e-09$)
	JS	AOA (2.8636$e-08$) ChOA (2.9550$e-08$) SMA (1.0764$e-06$) TSA (2.9550$e-08$) GWO (2.6874$e-09$)
	MA	AOA (7.7176$e-09$) ChOA (8.0065$e-09$) DE (9.5756$e-03$) EO (4.0127$e-02$) JS (1.9799$e-02$) PSO (3.9817$e-02$) SMA (8.0065$e-09$) TSA (8.0065$e-09$) GWO (4.6827$e-10$)
	MPA	AOA (1.0876$e-08$) ChOA (1.1267$e-08$) DE (3.5354$e-02$) SMA (1.1267$e-08$) TSA (1.1267$e-08$) GWO (7.4275$e-10$)
	PSO	AOA (2.3274$e-08$) ChOA (2.4037$e-08$) SMA (2.4037$e-08$) TSA (2.4037$e-08$) GWO (2.0445$e-09$)
	SMA	AOA (6.6063$e-08$) ChOA (1.4509$e-11$) TSA (2.9018$e-11$) GWO (2.1025$e-07$)
	TSA	AOA (6.6063$e-08$) ChOA (1.4509$e-11$)
	GWO	AOA (7.7176$e-09$) ChOA (8.0065$e-09$) TSA (8.0065$e-09$)
STARE database (Figure 2(a))	AOA	
	ChOA	AOA (6.7860$e-08$)
	DE	AOA (4.9511$e-08$) ChOA (1.3011$e-05$) SMA (1.9188$e-03$) TSA (4.9511$e-08$) GWO (4.7792$e-08$)
	EO	AOA (3.4782$e-08$) ChOA (1.0345$e-05$) SMA (9.2244$e-07$) TSA (3.4782$e-08$) GWO (3.3520$e-08$)
	JS	AOA (1.1267$e-08$) ChOA (4.9771$e-06$) DE (3.3383$e-03$) EO (4.3208$e-02$) SMA (1.5720$e-08$) TSA (1.1267$e-08$) GWO (1.0799$e-08$)
	MA	AOA (8.0065$e-09$) ChOA (3.9879$e-06$) DE (9.3030$e-04$) EO (9.5844$e-03$) SMA (7.9919$e-09$) TSA (8.0065$e-09$) GWO (7.6609$e-09$)
	MPA	AOA (8.0065$e-09$) ChOA (3.9879$e-06$) DE (9.3030$e-04$) EO (9.5844$e-03$) SMA (7.9919$e-09$) TSA (8.0065$e-09$) GWO (7.6609$e-09$)
	PSO	AOA (1.5124$e-08$) ChOA (6.0247$e-06$) DE (6.9581$e-03$) SMA (1.5098$e-08$) TSA (1.5124$e-08$) GWO (1.4517$e-08$)
	SMA	AOA (6.7860$e-08$) ChOA (1.5968$e-05$) TSA (6.7860$e-08$) GWO (1.5634$e-05$)
	TSA	AOA (1.4509$e-11$) ChOA (1.5983$e-05$)
	GWO	AOA (6.5690$e-08$) ChOA (1.5634$e-05$) TSA (6.5690$e-08$)

5. Conclusion

In this work, JS, MPA, TSA, MA, ChOA, SMA, AOA, and EO algorithms, which are among the most recently proposed metaheuristic algorithms in literature, have been improved as clustering based and applied to the retinal vessel segmentation. From the simulation results, it has been observed that the performances of the algorithms in terms of convergence speed and the MSE value are close to each other. As a result of the detailed statistical analyses, including the sensitivity, specificity, and accuracy parameters, it is seen that JS, MPA, TSA, MA, ChOA, SMA, AOA, and EO algorithms present high robustness and stability in retinal image analyses. If the algorithms are evaluated separately, the MA and MPA algorithms seem a bit more stable than the other algorithms in terms of clustering based retinal vessel segmentation due to their lower standard deviation values. Consequently, it is clearly seen from the results obtained that the performances of the JS, MPA, TSA, MA, ChOA, SMA, AOA, and EO algorithms in terms of clustering are too similar, and they can efficiently be used in retinal vessel segmentation.

In future research, firstly, the novel algorithms used in this work will be improved so as to obtain the optimal threshold values in segmentation of biomedical images with high accuracy. Secondly, the performance of each algorithm will be analyzed in functional diffusion magnetic resonance imaging in the area of ear diseases. Then, improvement strategies will be applied to each algorithm for the purpose of enhancing the sensitivity, specificity, accuracy, and *F*-score of the algorithms. Finally, hardware implementations for retinal vessel segmentation will be developed on a selected microprocessor.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] J. S. Chou and D. N. Truong, “Multistep energy consumption forecasting by metaheuristic optimization of time-series analysis and machine learning,” *International Journal of Energy Research*, vol. 45, no. 3, pp. 4581–4612, 2020.

[2] M. A. A. Al-qaness, A. I. Saba, A. H. Elsheikh et al., “Efficient artificial intelligence forecasting models for COVID-19 outbreak in Russia and Brazil,” *Process Safety and Environmental Protection*, vol. 149, pp. 399–409, 2021.

[3] X. Sun, G. Wang, L. Xu, H. Yuan, and N. Yousefi, “Optimal performance of a combined heat-power system with a proton exchange membrane fuel cell using a developed marine predators algorithm,” *Journal of Cleaner Production*, vol. 284, Article ID 124776, 2021.

[4] A. Shukla and A. K. Gupta, “Damping enhancement of DFIG integrated power system by coordinated controllers tuning using marine predators algorithm,” *Control Applications in Modern Power System*, vol. 710, pp. 165–176, 2020.

[5] A. Tharwat and W. Schenck, “A conceptual and practical comparison of PSO-style optimization algorithms,” *Expert Systems with Applications*, vol. 167, Article ID 114430, 2021.

[6] G. Nhu Nguyen, N. Ho Le Viet, G. Prasad Joshi, and B. Shrestha, “Intelligent tunicate swarm-optimization-algorithm-based lightweight security mechanism in internet of health things,” *Computers, Materials & Continua*, vol. 66, no. 1, pp. 551–562, 2020.

[7] H. Chen, W. Li, and X. Yang, “A whale optimization algorithm with chaos mechanism based on quasi-opposition for global optimization problems,” *Expert Systems with Applications*, vol. 158, Article ID 113612, 2020.

[8] H. Kaur, A. Rai, S. S. Bhatia, and G. Dhiman, “MOEPO: a novel Multi-objective Emperor Penguin Optimizer for global optimization: special application in ranking of cloud service providers,” *Engineering Applications of Artificial Intelligence*, vol. 96, Article ID 104008, 2020.

[9] L. Chen, C. Xu, H. Song, and K. Jermsittiparsert, “Optimal sizing and sitting of EVCS in the distribution system using metaheuristics: a case study,” *Energy Reports*, vol. 7, pp. 208–217, 2021.

[10] J. Zhao and Z. M. Gao, “The improved mayfly optimization algorithm with Chebyshev map,” *Journal of Physics*, vol. 1684, no. 1, 2020.

[11] Q. Askari, M. Saeed, and I. Younas, “Heap-based optimizer inspired by corporate rank hierarchy for global optimization,” *Expert Systems with Applications*, vol. 161, Article ID 113702, 2020.

[12] M. Filip, T. Zoubek, R. Bumbalek et al., “Advanced computational methods for agriculture machinery movement optimization with applications in sugarcane production,” *Agriculture*, vol. 10, no. 10, pp. 434–520, 2020.

[13] M. Kaur, R. Kaur, N. Singh, and G. Dhiman, “SChoA: a newly fusion of sine and cosine with chimp optimization algorithm for HLS of datapaths in digital filters and engineering applications,” *Engineering with Computers*, 2021.

[14] M. Mostafa, H. Rezk, M. Aly, and E. M. Ahmed, “A new strategy based on slime mould algorithm to extract the optimal model parameters of solar PV panel,” *Sustainable Energy Technologies and Assessments*, vol. 42, no. 111, Article ID 100849, 2020.

[15] M. Abdel-Basset, V. Chang, and R. Mohamed, “HSMA_WOA: a hybrid novel Slime mould algorithm with whale optimization algorithm for tackling the image segmentation problem of chest X-ray images,” *Applied Soft Computing*, vol. 95, Article ID 106642, Oct. 2020.

[16] Y. Feng, S. Deb, G.-G. Wang, and A. H. Alavi, “Monarch butterfly optimization: a comprehensive review,” *Expert Systems with Applications*, vol. 168, Article ID 114418, 2021.

[17] A. Mohammadi-Balani, M. Dehghan Nayeri, A. Azar, and M. Taghizadeh-Yazdi, “Golden eagle optimizer: golden eagle optimizer: a nature-inspired metaheuristic algorithm,” *Computers & Industrial Engineering*, vol. 152, Article ID 202150, 2021.

[18] M. Abdel-Basset, R. Mohamed, and M. Abouhawwash, “Balanced multi-objective optimization algorithm using improvement based reference points approach,” *Swarm and Evolutionary Computation*, vol. 60, Article ID 100791, 2021.

[19] S. I. Seleem, H. M. Hasanien, and A. A. El-Fergany, “Equilibrium optimizer for parameter extraction of a fuel cell dynamic model,” *Renewable Energy*, vol. 169, pp. 117–128, 2021.

[20] D. S. Fong, L. Aiello, and T. W. Gardner, “Retinopathy in diabetes,” *Diabetes Care*, vol. 27, no. 1, pp. 84–87, 2004.

[21] M. M. Fraz, P. Remagnino, A. Hoppe et al., "Blood vessel segmentation methodologies in retinal images - a survey," *Computer Methods and Programs in Biomedicine*, vol. 108, no. 1, pp. 407–433, 2012.

[22] J. J. Wang, B. Taylor, T. Y. Wong et al., "Retinal vessel diameters and obesity: a population-based study in older persons," *Obesity*, vol. 14, no. 2, pp. 206–214, 2006.

[23] M. Foracchia, E. Grisan, and A. Ruggeri, "Extraction and Quantitative Description of Vessel Features in Hypertensive Retinopathy Fundus Images," *Abstracts of the 2nd CAFIA workshop*, Springer, Manhattan, NY, USA, 2001.

[24] P. Mitchell, H. Leung, J. J. Wang et al., "Retinal vessel diameter and open-angle glaucoma," *Ophthalmology*, vol. 112, no. 2, pp. 245–250, 2005.

[25] K. Goatman, A. Charnley, L. Webster, and S. Nussey, "Assessment of automated disease detection in diabetic retinopathy screening using two-field photography," *PLoS ONE*, vol. 6, no. 12, Article ID e27524, 2011.

[26] A. H. Asad, E. Elamry, A. E. Hassanien, and M. F. Tolba, "New global update mechanism of ant colony system for retinal vessel segmentation," in *Proceedings of the 13th International Conference on Hybrid Intelligent Systems (HIS 2013)*, pp. 222–228, Gammarth, Tunisia, December 2013.

[27] K. Verma, P. Deep, and A. G. Ramakrishnan, "Detection and classification of diabetic retinopathy using retinal images," *In Proc. Annu. IEEE INDICON, India*, pp. 1–6, 2011.

[28] C. Heneghan, J. Flynn, M. OKeefe, and M. Cahill, "Characterization of changes in blood vessel width and tortuosity in retinopathy of prematurity using image analysis," *Medical Image Analysis*, vol. 6, no. 4, pp. 407–429, 2002.

[29] J. Lowell, A. Hunter, D. Steel, A. Basu, R. Ryder, and R. L. Kennedy, "Measurement of retinal vessel widths from fundus images based on 2-D modeling," *IEEE Transactions on Medical Imaging*, vol. 23, no. 10, pp. 1196–1204, 2004.

[30] A. Haddouche, M. Adel, M. Rasigni, J. Conrath, and S. Bourennane, "Detection of the foveal avascular zone on retinal angiograms using Markov random fields," *Digital Signal Processing*, vol. 20, no. 1, pp. 149–154, 2010.

[31] E. Grisan and A. Ruggeri, "A divide and impera strategy for automatic classification of retinal vessels into arteries and veins," in *Proceedings of the 25th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (IEEE Cat. No.03CH37439)*, pp. 890–893, Cancun, Mexico, September 2003.

[32] J. J. Kanski, *Clinical Ophthalmology*, 6th edition, 2007.

[33] D. B. Archer, "Diabetic retinopathy: some cellular, molecular and therapeutic considerations," *Eye*, vol. 13, no. 4, pp. 497–523, 1999.

[34] B. Antal and A. Hajdu, "An ensemble-based system for microaneurysm detection and diabetic retinopathy grading," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 6, pp. 1720–1726, 2012.

[35] M. D. Abràmoff, J. C. Folk, D. P. Han et al., "Automated analysis of retinal images for detection of referable diabetic retinopathy," *JAMA Ophthalmology*, vol. 131, no. 3, pp. 351–357, 2013.

[36] M. Foracchia, E. Grisan, and A. Ruggeri, "Detection of optic disc in retinal images by means of a geometrical model of vessel structure," *IEEE Transactions on Medical Imaging*, vol. 23, no. 10, pp. 1189–1195, Oct. 2004.

[37] M. B. Çetinkaya and H. Duran, "A detailed and comparative work for retinal vessel segmentation based on the most effective heuristic approaches," *Biomedical Engineering/Biomedizinische Technik*, vol. 66, no. 2, pp. 181–200, 2020.

[38] H. Jia, J. Ma, and W. Song, "Multilevel thresholding segmentation for color image using modified moth-flame optimization," *IEEE Access*, vol. 7, Article ID 44097, 2019.

[39] L. Xu, H. Jia, C. Lang, X. Peng, and K. Sun, "A novel method for multilevel color image segmentation based on dragonfly algorithm and differential evolution," *IEEE Access*, vol. 7, Article ID 19502, 2019.

[40] A. K. Bhandari, I. V. Kumar, and K. Srinivas, "Cuttlefish algorithm-based multilevel 3-D otsu function for color image segmentation," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 5, pp. 1871–1880, 2020.

[41] Z. Zhang and J. Yin, "Bee foraging algorithm based multi-level thresholding for image segmentation," *IEEE Access*, vol. 8, Article ID 16269, 2020.

[42] E. R. Esparza, L. A. Zanella-Calzada, D. Oliva et al., "An efficient Harris hawks-inspired image segmentation method," *Expert Systems with Applications*, vol. 155, pp. 1–29, 2020.

[43] T. Kurban, P. Civicioglu, R. Kurban, and E. Besdok, "Comparison of evolutionary and swarm based computational techniques for multilevel color image thresholding," *Applied Soft Computing*, vol. 23, pp. 128–143, 2014.

[44] M. Braovic´, D. Stipanicev, and L. Šeric´, "Retinal blood vessel segmentation based on heuristic image analysis," *Computer Science and Information Systems*, vol. 16, no. 1, pp. 227–245, 2018.

[45] S. J. Mousavirad, G. Schaefer, and H. E. Komleh, "A benchmark of population-based metaheuristic algorithms for high-dimensional multi-level image thresholding," in *Proceedings of the . 2019 IEEE Congress on Evolutionary Computation (CEC)*, pp. 2394–2401, Wellington, New Zealand, June 2019.

[46] A. Sharma and S. Sehgal, "Image segmentation using firefly algorithm," in *Proceedings of the 2016 International Conference on Information Technology (InCITe) - The Next Generation IT Summit on the Theme - Internet of Things: Connect your Worlds*, pp. 99–102, Noida, India, October 2016.

[47] P. Kandhway and A. K. Bhandari, "A water cycle algorithm-based multilevel thresholding system for color image segmentation using masi entropy," *Circuits, Systems, and Signal Processing*, vol. 38, no. 7, pp. 3058–3106, 2019.

[48] S. J. Mousavirad, H. Ebrahimpour-Komleh, and G. Schaefer, "Effective image clustering based on human mental search," *Applied Soft Computing*, vol. 78, pp. 209–220, 2019.

[49] J.-S. Chou and D.-N. Truong, "A novel metaheuristic optimizer inspired by behavior of jellyfish in ocean," *Applied Mathematics and Computation*, vol. 389, Article ID 125535, 2021.

[50] A. Faramarzi, M. Heidarinejad, S. Mirjalili, and A. H. Gandomi, "Marine predators algorithm: a nature-inspired metaheuristic," *Expert Systems with Applications*, vol. 152, Article ID 113377, 2020.

[51] S. Kaur, L. K. Awasthi, A. L. Sangal, and G. Dhiman, "Tunicate Swarm Algorithm: a new bio-inspired based metaheuristic paradigm for global optimization," *Engineering Applications of Artificial Intelligence*, vol. 90, Article ID 103541, 2020.

[52] K. Zervoudakis and S. Tsafarakis, "A mayfly optimization algorithm," *Computers & Industrial Engineering*, vol. 145, Article ID 106559, 2020.

[53] M. Khishe and M. R. Mosavi, "Chimp optimization algorithm," *Expert Systems with Applications*, vol. 149, Article ID 113338, 2020.

[54] S. Li, H. Chen, M. Wang, A. A. Heidari, and S. Mirjalili, "Slime mould algorithm: a new method for stochastic optimization,"

Future Generation Computer Systems, vol. 111, pp. 300–323, 2020.

[55] F. A. Hashim, K. Hussain, E. H. Houssein, M. S. Mabrouk, and W. Al-Atabany, "Archimedes optimization algorithm: a new metaheuristic algorithm for solving optimization problems," *Applied Intelligence*, vol. 51, no. 3, pp. 1531–1551, 2020.

[56] A. Faramarzi, M. Heidarinejad, B. Stephens, and S. Mirjalili, "Equilibrium optimizer: a novel optimization algorithm," *Knowledge-Based Systems*, vol. 191, Article ID 105190, 2020.

[57] C. Alonso-Montes, D. L. Vilarĩno, P. Dudek, and M. G. Penedo, "Fast retinal vessel tree extraction: a pixel parallel approach," *International Journal of Circuit Theory and Applications*, vol. 36, no. 5-6, pp. 641–651, 2008.

[58] A. D. Hoover, V. Kouznetsova, and M. Goldbaum, "Locating blood vessels in retinal images by piecewise threshold probing of a matched filter response," *IEEE Transactions on Medical Imaging*, vol. 19, no. 3, pp. 203–210, 2000.

[59] X. You, Q. Peng, Y. Yuan, Y.-m. Cheung, and J. Lei, "Segmentation of retinal blood vessels using the radial projection and semi-supervised approach," *Pattern Recognition*, vol. 44, pp. 2314–2324, 2011.

[60] S. Roychowdhury, D. Koozekanani, and K. Parhi, "Blood vessel segmentation of fundus images by major vessel extraction and sub-image classification," *IEEE Journal of Biomedical and Health Informatics*, vol. 19, no. 3, p. 1, 2014.

[61] S. Wang, Y. Yin, G. Cao, B. Wei, Y. Zheng, and G. Yang, "Hierarchical retinal blood vessel segmentation based on feature and ensemble learning," *Neurocomputing*, vol. 149, pp. 708–717, 2015.

[62] A. M. Mendonca and A. Campilho, "Segmentation of retinal blood vessels by combining the detection of centerlines and morphological reconstruction," *IEEE Transactions on Medical Imaging*, vol. 25, no. 9, pp. 1200–1213, 2006.

[63] G. Azzopardi, N. Strisciuglio, M. Vento, and N. Petkov, "Trainable COSFIRE filters for vessel delineation with application to retinal images," *Medical Image Analysis*, vol. 19, no. 1, pp. 46–57, 2015.

[64] M. M. Fraz, P. Remagnino, A. Hoppe et al., "An ensemble classification-based approach applied to retinal blood vessel segmentation," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 9, pp. 2538–2548, 2012.

[65] J. Odstrcilik, R. Kolar, A. Budai et al., "Retinal vessel segmentation by improved matched filtering: evaluation on a new high-resolution fundus image database," *IET Image Processing*, vol. 7, no. 4, pp. 373–383, 2013.

[66] D. Marín, A. Aquino, M. E. Gegundez-Arias, and J. M. Bravo, "A new supervised method for blood vessel segmentation in retinal images by using gray-level and moment invariants-based features," *IEEE Transactions on Medical Imaging*, vol. 30, no. 1, pp. 146–158, 2011.

[67] D. Kaba, C. Wang, Y. Li, A. Salazar-Gonzalez, X. Liu, and A. Serag, "Retinal blood vessels extraction using probabilistic modelling," *Health Information Science and Systems*, vol. 2, no. 1, p. 2, 2014.

[68] J. Dash and N. Bhoi, "A thresholding based technique to extract retinal blood vessels from fundus images," *Future Computing and Informatics Journal*, vol. 2, no. 2, pp. 103–109, 2017.

[69] F. Argüello, D. L. Vilarĩno, D. B. Heras, and A. Nieto, "GPU-based segmentation of retinal blood vessels," *J. Real Time Image Process*, vol. 14, no. 4, pp. 1–10, 2014.

[70] A. H. Asad, A. T. Azar, and A. E. Hassaanien, "Ant colony-based system for retinal blood vessels segmentation," *Advances in Intelligent Systems and Computing*, pp. 441–452, 2012.

[71] Y. Q. Zhao, X. Hong Wang, X. Fang Wang, and F. Y. Shih, "Retinal vessels segmentation based on level set and region growing," *Pattern Recognition*, vol. 47, no. 7, pp. 2437–2446, 2014.

[72] B. Zhang, L. Zhang, L. Zhang, and F. Karray, "Retinal vessel extraction by matched filter with first-order derivative of Gaussian," *Computers in Biology and Medicine*, vol. 40, no. 4, pp. 438–445, 2010.

[73] E. Imani, M. Javidi, and H.-R. Pourreza, "Improvement of retinal blood vessel detection using morphological component analysis," *Computer Methods and Programs in Biomedicine*, vol. 118, no. 3, pp. 263–279, 2015.

[74] A. Mendonca, B. Dashtbozorg, and A. Campilho, "Segmentation of the vascular network of the retina," in *Image Analysis and Modeling in Ophthalmology*, E. Y. K. Ng, U. R. Acharya, A. Campilho, and J. S. Suri, Eds., pp. 85–109, CRC Press, 2014.

[75] Z. Xiao, M. Adel, and S. Bourennane, "Bayesian method with spatial constraint for retinal vessel segmentation," *Computational and Mathematical Methods in Medicine*, vol. 2013, pp. 1–9, Article ID 401413, 2013.

[76] B. Yin, H. Li, B. Sheng et al., "Vessel extraction from non-fluorescein fundus images using orientation-aware detector," *Medical Image Analysis*, vol. 26, no. 1, pp. 232–242, 2015.

[77] I. Lázár and A. Hajdu, "Segmentation of retinal vessels by means of directional response vector similarity and region growing," *Computers in Biology and Medicine*, vol. 66, no. 1, pp. 209–221, 2015.

[78] N. Strisciuglio, G. Azzopardi, M. Vento, and N. Petkov, "Multiscale blood vessel delineation using b-cosfire filters," *Computer Analysis of Images and Patterns*, pp. 300–312, 2015.

[79] F. A. Hashim, E. H. Houssein, K. Hussain, M. S. Mabrouk, and W. Al-Atabany, "A modified Henry gas solubility optimization for solving motif discovery problem," *Neural Computing & Applications*, vol. 32, no. 14, Article ID 10759, 2020.

[80] A. Arcuri and L. Briand, "A Hitchhiker's guide to statistical tests for assessing randomized algorithms in software engineering," *Software Testing, Verification and Reliability*, vol. 24, no. 3, pp. 219–250, 2012.