Artificial Intelligence Review
https://doi.org/10.1007/s10462-019-09705-8

Hybrid computational intelligence algorithms and their applications to detect food quality

Lavika Goel[1] · Sundaresan Raman[1] · Subham Swastik Dora[1] · Anirudh Bhutani[1] · A. S. Aditya[1] · Abhinav Mehta[1]

Abstract
Food security is a major problem faced today.With primitive storage facilities, especially in developing countries, it often leads to extensive losses. This work aims to develop algorithms based on vision data to assess the food quality and deploy them in food storage facilities to detect early signs of spoilage. This paper presents various segmentation techniques for finding spoilt food. Novel optimization techniques have been developed and implemented to improve K-means clustering and multilevel thresholding. A hybrid of moth flame optimization (MFO) and gravitational search algorithm (GSA) has been developed. Also, in another hybrid, particle swarm optimization (PSO) was also incorporated along with MFO and GSA. Both the hybrids performed better than the individual algorithms and the MFO–GSA–PSO hybrid performed better than the MFO–GSA hybrid on the benchmark functions. Segmented images using optimized K-means were used for feature extraction using local binary patterns (LBP). Multiclass support vector machine was used for classification which gave an accuracy of 81% for features from segmented images obtained using MFO–GSA hybrid and 83.33% for that using MFO–GSA–PSO hybrid. Results of simple linear iterative clustering superpixels for segmentation have also been discussed. The segmented clusters are then used to judge the rottenness of the food. Classification using LBP and Haralick features of the segmented image obtained using graphs over superpixels gave an accuracy of 81.7% and 78% respectively.

Keywords Moth flame optimization · Gravitational search algorithm · Particle swarm optimization · Simple linear iterative clustering (SLIC) superpixels · K-means clustering · Multilevel clustering · Support vector machines

1 Introduction

Swarm intelligence-based algorithms have proven to perform well for optimization problems. Algorithms like ant colony optimization, gravitational search algorithm (GSA), and moth flame optimization (MFO) have been successfully applied on

✉ Lavika Goel
lavika.goel@pilani.bits-pilani.ac.in

Extended author information available on the last page of the article

Published online: 30 April 2019

optimization problems. GSA is an optimization algorithm inspired by Newton's law of gravitation. Moth flame algorithm is another optimization algorithm inspired by movement of moths towards light. There are shortcomings of both these methods. The proposed algorithms aim to address the shortcomings of each of these methods by complementing it with the strengths of the other. Particle swarm optimization can also be incorporated so that the search agents can communicate with each other their results through a global optimum. In the first hybrid only MFO and GSA are used and in the second hybrid MFO, GSA and PSO are used. The optimization methods also aim to reduce the convergence of the optimal value to acceptable error range with fewer iterations to make the optimization more efficient. The hybrid algorithms presented in this paper are high level metaheuristic algorithms. They consist of self-contained metaheuristics executed in sequence. As per the taxonomy discussed by Talbi (2002), on the flat level they are homogeneous, global and general. They comprise of different metaheuristics which work on the entire search space and all the algorithms solve the same optimization problem.

Our work uses these hybrid algorithms to detect the quality of different food items. Automated detection of food quality can assist the food storage facilities to detect rotten foods and get rid of them before the disease is spread to other food items. These hybrids are used to improve the results of segmentation. The hybrid optimization algorithms have been successfully incorporated in K-means and multilevel thresholding segmentation algorithms. The aim is to improve K-means clustering using the proposed algorithm. K-means algorithm initializes the K-mean points randomly and hence the results (i.e. mean squared error of each data-point from the cluster point it is assigned to) vary each time the algorithm is applied. Optimizing K-means using the hybrid algorithm can produce uniform results each time.

Using the brute force approach for multilevel thresholding is a computationally expensive process as it finds multiple thresholds, the value of each can vary from 0 to 255. The proposed optimization techniques use a population of search agents to find the best threshold values. The search agents individually find the optimal values and in case of MFO–GSA–PSO hybrid, communicate to others resulting in a swarm behavior which leads to finding the optimal values quickly. Also, when the color space increases these can perform comparatively better. With higher number of color spaces these techniques can be applied to each dimension separately.

Another approach frames graphs on superpixels in order to separate the disease cluster by using color distance edges between different superpixels. Simple linear iterative clustering (SLIC) has been used for dividing the image into various superpixels. Using SLIC, all the superpixels can be divided into groups, where each group represents a complete graph. After segmentation feature extraction is done using local binary patterns (LBP) and grey level co-occurrence matrix (GCLM). Each cluster is labeled as per the disease using which a model is generated. Model thus created is tested on the test images.

Section 2 lists the literature survey on this topic. Section 3 describes the algorithms used to construct the hybrid. Section 4 is the application of both the hybrids in K-means and multilevel thresholding. Section 5 describes how segmentation can be achieved using superpixels. Section 6 gives the results of both the hybrids on benchmark functions as well as results of optimized K-means and multilevel thresholding. Classification results are also given in this section. Section 7 presents a brief analysis of all the algorithms. Section 8 concludes the work and describes the future scope.

2 Related work

A lot of work has been done to improve food safety and food quality inspection of food items like fruits (Dubey and Jalal 2012a, b), bakery (Abdullah et al. 2000), meat (Joo et al. 2013) etc. Sensor-based techniques and vision-based techniques are two broad classifications in this area. Sensor-based techniques employ devices like e-nose. (Echeverría et al. 2004) used electronic nose responses registered by seven different sensors to classify the apples using principal component analysis. Baietto and Wilson (2015) has discussed the various approaches taken by researches in the field of food analysis using electronic nose. A complete list of all the fruits and their respective volatile compounds that are easy to detect and affect the aroma and flavor has been mentioned. Berna et al. (2004) and Gomez et al. (2008) discussed the use of electronic nose to study shelf life of tomatoes. Sanaeifara et al. (2016) used e-nose signals to determine chemical and physical properties of banana.

Vision-based techniques consider texture, shape, size, irregular spots etc. of the food items to determine if a food item is spoilt or not. Multispectral imaging is also employed in several cases. Multispectral imaging has been used in the past to evaluate the quality of bi colored apples (Unay et al. 2012). Disease detection on apples has also been done using machine vision (Rehkugler and Throop 1986). Fuzzy models have also been used for classification of cookies. Size, shape, baked dough color and related features were extracted to train the classification model. Based on ratings of people four fuzzy models were developed (Davidson et al. 2001). Among the four, good results were obtained for the Mamdani inference system and Sugeno inference system. Artificial neural networks have also been used to determine quality of meat (Prevolnik et al. 2011). A machine vision approach to predict just the quality of tomatoes has been discussed in Srivastava et al. (2014). The algorithm consisted of 3 steps mainly pre-processing, segmentation and scaling, and recognition and classification. Clustering algorithms were used for segmentation and color based and texture-based features were used to train the classifier. The system classifying the tomatoes into 2 classes, gained an overall accuracy of 92%. Ojala et al. (2002) used local binary pattern (LBP) histogram for rotation invariant texture classification on apples. Local binary pattern technique has gained profuse popularity recently and has found applications in various fields including face recognition, dynamic texture recognition and shape localization. Dubey and Jalal (2012a, b) presented a clustering-based thresholding approach. The images are initially converted to LAB color space and the K-means clustering is applied in AB space. The resulting clusters divide the image based on color. Working on the heuristic that darkest cluster will always contain the disease, the approach was successful in separating the diseases from the apples to some extent. Global color histogram (GCH), color coherence vector (CCV) and local binary pattern (LBP) features were used to train the classifier. Support vector machine was used for classification into 4 classes resulting in 93% accuracy.

Nano sensors are also employed nowadays. Nanoparticles like silver-based nanoparticles, cerium oxide nanoparticles, gold-based nanoparticles (Bulbul et al., 2015) etc. have been used by researchers to detect food contamination. Duncan (2011) discussed the use of silver nanoparticles as potent antimicrobial agents. It also discussed use of nano sensors and nanomaterial-based assays for the detection of food-relevant analytes. Biosensors (Mutlu 2011) and Nano-biosensors (Yang et al. 2016) have also been used for food safety and food monitoring. Instead of using fruits, Sannakki et al. (2013) used leaves for disease detection in grapes. This way disease prediction can be done even without removing the fruit or the vegetable from the tree, making it on site and nondestructive method for disease

detection. Two level clusters were used, first to separate leaf from the background and the second level to segment the diseases portion from the leaf. A neural network was trained for first level of clustering.

A lot of nature inspired optimization algorithms have been developed. Mirjalili (2015), Rashedi et al. (2009) and Kennedy (2011) discussed moth flame optimization, gravitational search algorithm and particle swarm optimization respectively. There have been recent advances in moth flame optimization. Ghada et al. (2016) proposed two new versions of the MFO to select optimal feature subset for classification purposes. In one of them the logarithmic spiral was replaced with a hyperbolic spiral and in another by Archimedes spiral (a member Archimedean or Arithmetic spiral family). The new versions performed better than the original one but not by much. Buch et al. (2017) also modified the MFO by suggesting paths of moths in new spirals to solve optimal power flow problem.

These along with other nature inspired algorithms have been successfully applied for data clustering. Hatamlou et al. (2012) used gravitational search algorithm to improve upon K-means clustering. Hatamlou et al. (2011) also described the use of gravitational search algorithm for data clustering. Mingru et al. (2014) and Ahmadyfard and Modares (2008) discussed particle swarm optimization for data clustering. Li et al. (2015) achieved image segmentation using dynamic particle swarm optimization and K-means clustering. Kumar et al. (2011) described color-based image segmentation using fuzzy c-Means clustering. Co-occurrence statistical techniques for multispectral satellite imagery has been used to determine initial number of clusters in K-means clustering algorithm (Koonsanit et al., 2012).

Osuna-Enciso et al. (2013) illustrated the use of nature inspired algorithms for multilevel thresholding. The results of three algorithms i.e. Particle swarm optimization, artificial bee colony optimization and differential evolution have been discussed. Goel et al. (2012a, b, c) described biogeography-based optimization for land cover feature extraction. Goel et al. (2012a, b, c) discussed models of species abundance in biogeography-based optimization technique. Goel et al. (2012a, b, c) described hybrid bio-inspired optimization for land cover feature extraction.

3 Optimization algorithms

Even with the existence of several optimization algorithms new algorithms are continuously being developed. This is because of the No-Free-Lunch theorem which basically states that that a algorithm can solve only a class of optimization problems. Swarm based optimization algorithms can avoid local optima. Swarm-based optimization algorithm generally consists of a population of individuals. Some examples include moth flame optimization, particle swarm optimization, gravitational search algorithm, biogeography-based optimization etc. All the optimization algorithms used in the hybrids are swarm based.

3.1 Moth flame optimization

Moth flame optimization is also inspired by nature. It is also a swarm optimization algorithm, which mimics the locomotion of moths around flames. Moths can navigate in a straight line by maintaining a constant angle with the moon, since the moon is very far away. These moths

show the same behavior on encountering artificial lights. On maintaining a fixed angle with a close enough light source, moths end up in a spiral motion ending at the light source.

The MFO algorithm mimics this motion. The moths are the search agents and the flames are the best solution found by the moths so far. The coordinates of both the moths and flames are tentative solutions. The algorithm starts with n moths and n flames. Each moth travels around a flame in a logarithmic spiral, S given by Eq. (1).

$$S\left(M_i, F_i\right) = D_i e^{bt} \cos\left(2\pi t\right) + F_i \tag{1}$$

where S is the logarithmic spiral, M_i is the position of moth i, F_i is the position of flame is associated with moth i, D_i and b are constants which determine the shape and size of the spiral.

Fitness is associated to every flame calculated by a fitness function. The moths are initialized with random coordinates leading to better exploration of the search space. To ensure better exploitation, finding optima around a good solution, the no. of flames is reduced with each iteration by eliminating lesser fit flames. So, now since the no. of moths remains the same, more moths move around the fitter flames to find a better optimum around the fitter flames. Finally, only one flame remains whose coordinates are the solution obtained.

3.2 Gravitational search algorithm

Gravitational search algorithm (GSA) is based on Newton's law of gravitation. Here, the search agents are masses scattered across the hyperspace. These masses move according to the laws of gravity, all converging towards the heavier masses. Since mass is directly proportional to fitness, convergence to optimal solution is achieved. Initially, GSA starts with random search agents with associated mass calculated by a fitness function. The mass is related to the proximity of the search agent to the solution i.e. greater the mass, closer is the search agent to the solution. Gravitational search algorithm is a global algorithm. All the masses attract each other and as a result move towards each other. But since heavier objects have less acceleration, they move slowly and produce stronger gravitational field. Hence the search agents converge to larger masses, thus moving closer to optimal solution. Gravitational search algorithm involves three kinds of masses:

1. Active gravitational mass,M_a which measures the strength of the gravitational field. Strength of the gravitational field generated is directly proportional to the mass of the object.
2. Passive gravitational mass,M_p relates to the amount of force experienced by an object in a gravitational field. Higher the value of M_p the greater the force experienced by the object.
3. Inertial mass, M_i which measures the resistance of an object to changing its state of motion on application of a force. Objects with higher inertial mass offer greater resistance to changing their state of motion by applying external force.

The above mentioned masses are calculated using fitness function. These masses are then used to calculate the force and acceleration on each search agent using the following Eqs. (2) and (3). It should be noted that in actual Newton's gravitational law the force is calculated by dividing the masses by R^2 instead of R.

$$F_{ij} = G \times \left(M_{aj} \times M_{pi}\right)/R \tag{2}$$

$$a_i = \sum_{j=1 \text{ and } j \neq i}^{n} F_{ij}/M_{ii} \tag{3}$$

n is the total no. of objects, F_{ij} if the force on object i by object j, M_{aj} is the active gravitational mass of object j, M_{pj} is the active gravitational mass of object i, a_i is the acceleration of object i, G is the gravitational constant, R is the distance between object i and j.

The motion of the search agent is according to the gravitational forces along with some randomization. Randomness is achieved by multiplying each component of the force and the magnitude of distance by a random number lying between 0 and 1. This enables the GSA to explore more of the search space and avoid local optima.

3.3 Particle swarm optimization

Particle swarm optimization is also a population (swarm) based algorithm which is based on the movement in animal societies without any leader like a flock of birds finding food. In these each member of the group searches for food randomly. The members communicate with each other to find the best position. If one of the members finds a location which has more food available, it will inform other members of the flock. Particle swarm optimization method simulates these movements. The search agents are the members of the flock.

Each search agent is associated with a local optimum fitness value and there exists global optimum fitness value. The motion of the search agent is dictated by these values.

The velocity of the search agent at time t is given by Eq. (4).

$$v(t) = v(t-1) + c_1 r_1 (p(t-1) - x(t-1)) + c_2 r_2 (g(t-1) - x(t-1)) \tag{4}$$

$v(t)$ is the velocity of the concerned search agent at time t, $v(t-1)$ is the velocity of the concerned search agent at time $t-1$, $p(t-1)$ is the best position as per fitness value obtained by the concerned particle at time $t-1$, $g(t-1)$ is the global best position as per fitness value at time $t-1$, r_1, r_2 are constant parameters, c_1, c_2 are random numbers between 0 and 1.

The position of the search agent at time t is given by Eq. (5).

$$x(t) = v(t) + x(t-1) \tag{5}$$

$v(t)$ is the velocity of the concerned search agent at time t, $x(t-1)$ is the position of the concerned search agent at time $t-1$.

As clear from the algorithm search agents with better fitness move slowly as compared to search agents with higher fitness. The former achieves exploitation and the latter achieves exploration.

3.4 Hybrid of moth flame optimization and gravitational search algorithm

The hybrid algorithm uses the exploration of the moth flame optimization (MFO), and the efficient exploitation of the gravitational search algorithm (GSA) (Sarma et al. 2017). MFO explores according to logarithmic spiral motion, while the GSA uses the linear motion for locomotion. Thus, both the properties of the individual algorithms are utilized by the hybrid algorithm and they complement each other. The fitness measure is mass of each search agent for GSA while the fitness measure for MFO is distance to the fittest flame. Thus, the hybrid uses the strength of both MFO and GSA and hence is a better fit. The algorithm involves the following steps:

1. The population of search agents is initialized to random locations in the search space.
2. Next, fitness of every search agent is calculated using a fitness function.
3. The flames are then sorted according to their fitness and the moths are associated serially to their corresponding flames. The first moth is assigned to the fittest flame, second to second best flame and so on.
4. Then the search agents move around their corresponding flames in a spiral motion. After that, masses are updated. Moths (with masses) then attract each other moving closer to the fitter flames.

This process is repeated for steps 3 and 4 for required no. of iterations.

Locomotion for each search agent is first through the logarithmic spiral. This is the locomotion technique moths follow around a flame. This is followed by the gravitational pull of each search agent towards each other using the standard Newton's laws.

Figure 1 shows the pseudo code for MFO–GSA–PSO hybrid. It first initializes the moths in the matrix M which are the search agents. Also, the maximum no. of iterations is set. Then, the fitness of the moths is calculated. The moths are then sorted according to fitness and stored in another matrix F which represents the initial flames. Then all the n moths move in all the d dimensions according to the logarithmic motion given in Eq. (1). F_i is the flame associated with moth M_i. Mass is taken equal to the fitness. Then the force and acceleration are calculated as per Eqs. (2) and (3) respectively and motion as per gravitational search algorithm is performed. Mass is taken equal to the fitness. To calculate net force on $M(i)$, forces by each of the other object $M(k)$ are summed up.

After that, since one iteration is completed the new positions of the flames are obtained by taking best m out of F and M which contain current flame and moth positions respectively and storing them back in F. m is the updated no. of flames. m can be calculated by any function as per the problem at hand but is successively reduced. The algorithm stops when only one flame remains which is the final solution.

3.5 Hybrid of moth flame optimization, gravitational search algorithm and particle swarm optimization

The hybrid optimization algorithm integrates all the above three algorithms. It combines the exploratory nature of moth flame optimization and exploitation by the gravitational search. Also, particle swarm optimization helps all the search agents to share their results through the global optima which stores the best solution obtained so far. So, it combines the strengths of all the three algorithms. The steps involved in the hybrid algorithm are as follows:

1. It starts with n no. of flames and moths.
2. Moths are sorted according to fitness in decreasing order. Flames are assigned to moths serially. The first moth in the list is assigned to the fittest flame, second moth to the second fittest flame and so on.
3. Moths then move around the associated flames in spiral motion.
4. Mass is calculated for each of the moths according to fitness and then they attract each other as per the gravitational search algorithm.
5. Then the moths undergo particle swarm optimization motion where the local optima are the associated flame for a moth and the global optima is the first (best) flame.

$M = initial\_moths\_posn( )$;
$F = sort(M)$; // $flames$
$GSA_{vel} = initial\_vel\_GSA$

while $iteration < \max\_iteration$ *do*
    update flame no.
    if $iteration == 1$
        $F = sort(M)$
    else
        $F = sort(M, F)$
    endif

    for $i = 1 : n$ *do*
        for $j = 1 : d$ *do*
            update r and t
            $D(i,j) = |F(i,j) - M(i,j)|$
            $M(i,j) = D(i,j) \times \exp(bt) \times \cos(2\pi t) + F(i,j)$
        endfor
    endfor

    $G = G_0 \times \left(\frac{1}{iteration}\right) \times b$

    for $i = 1 : n$ *do*
        $mass(i) = fitness(M(i))$
        for $k = 1 : n$ *do*
            $mass(k) = fitness(M(\mathrm{k}))$
            $R = |M(i) - M(k)|$
            $Force = Force + G \times \frac{mass(k)}{R}$
        endfor
        for $j = 1 : d$ *do*
            $GSA_{vel}(i,j) = GSA_{vel}(i,j) \times rand(0,1) + \frac{Force}{mass(i)}$
            $M(i,j) = M(i,j) \times rand(0,1) + GSA_{vel}(i,j)$
        endfor
    endfor

endwhile

Fig. 1 Pseudo code for MFO–GSA hybrid

6. Then, the best n out of the flames and new position of moths are chosen which are the new flames.
7. No. of flames is reduced and steps 3–7 are iterated until only one flame remains which is the required solution.

Figure 2 shows the pseudo code for MFO–GSA–PSO hybrid. It first initializes the moths in the matrix M which are the search agents. Also, the maximum no. of iterations is set. Then, the fitness of the moths is calculated. The moths are then sorted according to fitness and stored in another matrix F which represents the initial flames. Then, all the n moths move in all the d dimensions according to the logarithmic motion given in

$M = initial\_moths\_posn( )$;
$F = sort(M)$; // $flames$
$GSA_{vel} = initial\_vel\_GSA$
$PSO_{vel} = initial\_vel\_PSO$

while $iteration < \max\_iteration$ *do*
    update flame no.
    if $iteration == 1$
        $F = sort(M)$
    else
        $F = sort(M, F)$
    endif

    for $i = 1 : n$ *do*
        for $j = 1 : d$ *do*
            update r and t
            $D(i,j) = |F(i,j) - M(i,j)|$
            $M(i,j) = D(i,j) \times \exp(bt) \times \cos(2\pi t) + F(i,j)$
        endfor
    endfor

    $G = G_0 \times \left(\frac{1}{iteration}\right) \times b$

    for $i = 1 : n$ *do*
        $mass(i) = fitness(M(i))$
        for $k = 1 : n$ *do*
            $mass(k) = fitness(M(\mathrm{k}))$
            $R = |M(i) - M(k)|$
            $Force = Force + G \times \frac{mass(k)}{R}$
        endfor
        for $j = 1 : d$ *do*
            $GSA_{vel}(i,j) = GSA_{vel}(i,j) \times rand(0,1) + \frac{Force}{mass(i)}$
            $M(i,j) = M(i,j) \times rand(0,1) + GSA_{vel}(i,j)$
        endfor
    endfor

    for $i = 1 : n$ *do*
        for $j = 1 : d$ *do*
            $PSO_{vel}(i,j) = PSO_{vel}(i,j) + a \times rand(0,1) \times (F(i,j) - M(i,j)$
                $+a \times rand(0,1) \times (F(1,j) - M(i,j))$
            $M(i,j) = M(i,j) \times rand(0,1) + PSO_{vel}(i,j)$
        endfor
    endfor

endwhile

Fig. 2 Pseudo code of MFO–GSA–PSO hybrid

Eq. (1). F_i is the flame associated with moth M_i. Mass is taken equal to the fitness. Then the force and acceleration are calculated as per Eqs. (2) and (3) respectively and motion as per gravitational search algorithm is performed. Mass is taken equal to the fitness. To calculate net force on $M(i)$, forces by each of the other object $M(k)$ are summed up. After that the moths are moved according to PSO algorithm. The velocity is obtained as

per Eq. (4) and the position is updated as per Eq. (5). Here the global optima is $F(1)$, the first flame in the flame matrix F, which is the best fit flame.

After that, since one iteration is completed the new positions of the flames are obtained by taking best m out of F and M which contain current flame and moth positions respectively and storing them back in F. m is the updated no. of flames. m can be calculated by any function as per the problem at hand but is successively reduced. The algorithm stops when only one flame remains which is the final solution.

4 Application of hybrid optimization algorithms

Both MFO–GSA and MFO–GSA–PSO hybrid were used to improve K-means algorithm and multilevel thresholding. The performances of both these algorithms were also compared with each other as well as the standard algorithms.

4.1 Optimizing K-means segmentation

The standard K-means algorithm was improved using MFO–GSA and MFO–GSA–PSO hybrid to find better clusters. This optimizing K-means segmentation technique treats the K mean points as search agents or moths and finds the initial cluster points. Then classical K-means is applied. The steps of the classical K-means include the following:

1. K points are randomly selected from the given set of pixels which constitute the initial K-mean points. The value of pixel in each dimension is selected separately from the given set of points. Hence the selected pixel may not be in the image but each of the components in the pixel must be.
2. Every point in the dataset is assigned to one of the K-means cluster which resembles the most to the data point. Euclidean distance between pixels is the basis of similarity. Every point is assigned to exactly one cluster centroid C_k.
3. The new means of the clusters are then calculated based on the points assigned to the cluster in the previous iteration.

Steps 2 and 3 are iterated until convergence is reached i.e. when cluster assigned to every data point does not change with iterations.

In the pseudo code in Fig. 3 the search agents are the cluster points. Hence, each search agent denotes a possible combination of K points which are the cluster points. Now, n search agents are randomly selected and are subjected to MFO and GSA motion respectively for a fixed no. of iterations. After each iteration, the variance of the best solution is calculated and if the variance is less than a threshold value classical K-means is applied with the coordinates obtained by the hybrid algorithms as initial clusters.

Similarly, in the pseudo code in Fig. 4, initial cluster points are selected by MFO–GS–PSO Hybrid and then classical K-means is applied.

This is an improvement over the classical K-means which selects the initial cluster points randomly. This can cause the classical K-means algorithm to get stuck in local optima. Selection of initial cluster points by the hybrid algorithm gives a good tentative solution for the classical K-means algorithm which can then be applied to give a better solution.

4.2 Multilevel Thresholding using Hybrid Optimization

Multilevel thresholding is optimized using the MFO–GSA and MFO–GSA–PSO hybrid. First, uniformly distributed noise is added to the original image i.e.$F_{ij} = f_{ij} + n_{ij}$.

The following are the steps of the algorithm:

$M = initial\_moths\_posn( )\ ;$
$F = sort(M)\ ;\ //\ flames$
$GSA_{vel} = initial\_vel\_GSA$

$while\ iteration < \max\_iteration\ do$
  $update\ flame\ no.$
  $if\ iteration == 1$
    $F = sort(M)$
  $else$
    $F = sort(M, F)$
  $endif$

  $for\ i = 1 : n\ do$
    $for\ j = 1 : d\ do$
      $update\ r\ and\ t$
      $D(i,j) = |F(i,j) - M(i,j)|$
      $M(i,j) = D(i,j)\ \times\ \exp(bt)\ \times \cos(2\pi t) + F(i,j)$
    $endfor$
  $endfor$

  $G = G_0\ \times \left(\frac{1}{iteration}\right)\ \times\ b$

  $for\ i = 1 : n\ do$
    $mass(i) =\ fitness(M(i))$
    $for\ k = 1 : n\ do$
      $mass(k) =\ fitness(M(\mathrm{k}))$
      $R\ =\ |M(i) - M(k)|$
      $Force\ =\ Force + G\ \times\ \frac{mass(k)}{R}$
    $endfor$
    $for\ j\ =\ 1 : d\ do$
      $GSA_{vel}(i,j)\ =\ GSA_{vel}(i,j)\ \times\ rand(0,1) + \frac{Force}{mass(i)}$
      $M(i,j) = M(i,j)\ \times\ rand(0,1) +\ GSA_{vel}(i,j)$
    $endfor$
  $endfor$

  $variance = \frac{1}{n} \times \sum \left(Fitness(F_i)\ -\ Fitness\left(F_{avg}\right)\ \right)^2$

  $if\ variance < threshold\ then$
    $Classical\_Kmeans()$
  $endif$

$endwhile$

Fig. 3 Pseudo code for K-means using MFO–GSA hybrid

$M = initial\_moths\_posn()$;
$F = sort(M)$; // $flames$
$GSA_{vel} = initial\_vel\_GSA$
$PSO_{vel} = initial\_vel\_PSO$
while $iteration < \max\_iteration$ *do*
    update flame no.
    if $iteration == 1$
        $F = sort(M)$
    else
        $F = sort(M, F)$
    endif
    for $i = 1 : n$ *do*
        for $j = 1 : d$ *do*
            update r and t
            $D(i,j) = |F(i,j) - M(i,j)|$
            $M(i,j) = D(i,j) \times \exp(bt) \times \cos(2\pi t) + F(i,j)$
        endfor
    endfor
    $G = G_0 \times \left(\frac{1}{iteration}\right) \times b$

    for $i = 1 : n$ *do*
        $mass(i) = fitness(M(i))$
        for $k = 1 : n$ *do*
            $mass(k) = fitness(M(k)))$
            $R = |M(i) - M(k)|$
            $Force = Force + G \times \frac{mass(k)}{R}$
        endfor
        for $j = 1 : d$ *do*
            $GSA_{vel}(i,j) = GSA_{vel}(i,j) \times rand(0,1) + \frac{Force}{mass(i)}$
            $M(i,j) = M(i,j) \times rand(0,1) + GSA_{vel}(i,j)$
        endfor
    endfor
    for $i = 1 : n$ *do*
        for $j = 1 : d$ *do*
            $PSO_{vel}(i,j) = PSO_{vel}(i,j) + a \times rand(0,1) \times (F(i,j) - M(i,j)$
                $+a \times rand(0,1) \times (F(1,j) - M(i,j))$
            $M(i,j) = M(i,j) \times rand(0,1) + PSO_{vel}(i,j)$
        endfor
    endfor

    $variance = \frac{1}{n} \times \sum \left(Fitness(F_i) - Fitness\left(F_{avg}\right)\right)^2$

    if $variance < threshold$ *then*
        $Classical\_Kmeans()$
    endif
endwhile

Fig. 4 Pseudo code of K-means using MFO–GSA–PSO hybrid

1. For each pixel (i, j) the initial value is chosen as the original value of the pixel at (i, j) or the value of noise at (i, j). This increases the randomness and hence, better accuracy is achieved.

```
procedure MFO_GSA or MFO_GSA_PSO Thresholding:
        I = ReadImage()
        N = Noise_Matrix(I) //with same dimensions as I
        I = I + N
        initialize()
        for i = 1 : max_iterations do
                while (size(population) > size(new_population)) do
                        MFO_GSA or MFO_GSA_PSO(population)
                        F_i = S_intercluster / S_intracluster
                        Sort(F[])
                endwhile
        endfor
endprocedure
```

Fig. 5 Pseudo code for thresholding using MFO–GSA or MFO–GSA–PSO

2. Fitness function is measured based on intra cluster similarity. Also, inter cluster similarity is avoided. Thus, the fitness is given by the ratio $S_{intercluster}$:$S_{intracluster}$.
3. The moths representing images/potential solutions undergo locomotion of both linear and spiral nature according to the hybrid algorithms. The best *n* moths among the new as well as old position of the moths are chosen, which become the flames in the next iteration. The flames are reduced with each iteration (by decrementing the value of *n*) and when only one flame remains it represents the solution/segmented image (see pseudo code in Fig. 5).

Larger images are handled by dividing the image into smaller images, generating final best of these smaller images using optimization and merging them to create final segmented image.

5 Using superpixels

Superpixel is an over-segmentation technique which develops an intermediate level image representation depicting all possible objects that can be there in the image. It finds a wide range of applications in object detection, recognition and tracking, 3D reconstruction and semantic segmentation. While there are various approaches to superpixels, three algorithms namely, normalized cuts (Shi and Malik 2000; Ren and Malik 2003), simple linear iterative clustering (SLIC) (Achanta et al. 2012) and superpixels extracted via energy driven sampling (SEEDS) (Van den Bergh et al. 2012) are most widely used. Derived from local K-means clustering, Simple linear iterative clustering initializes with all the superpixel centers on the image grid plane. Color cues from LAB color space and spatial distance together are used for clustering. The K-means being used is referred to as local because each cluster only considers the boundary pixels. This results in a regularity in pixel formation.

5.1 Using graphs over superpixels

Once superpixels have been extracted from the image, an undirected graph is established between different superpixels using color based cues as the weight for the edges. An

iterative process goes through each superpixel, finds out the Euclidean distance between neighboring superpixels using hue from HSV color space and B from LAB color space as metrics for the distance. H and B have been used in order to minimize the illumination differences and color noise. Going to each superpixel, adding it to other superpixels and then finding the number of connected components provides with the neighboring superpixels for each respective superpixel. Then, after finding Euclidean distance, if the distance is less than a fixed threshold, then an edge is established between those superpixels.

This approach goes through each superpixel for every respective superpixel, making it expensive complexity wise. A more optimized approach can also be used where only the neighboring superpixels are considered for the edges thus reducing the complexity. Since, SLIC-Superpixel algorithm provides only the mask for each superpixel, connected components were used to establish the spatial geometry of the superpixels. Using connected components one can tell if adding two superpixel masks is increasing the number of components, thus deciding the adjacent superpixels of each superpixel (Mehta 2016).

6 Results and implementation

Results of both the hybrids on the benchmark functions have been discussed along with results of optimized K-means and optimized multilevel thresholding. This section also presents the results of classification of diseased apples into various categories i.e. "normal", "rot", "blotch" and "scab". Local binary patterns and (LBP) and gray level co-occurrence matrix (GCLM) were used to extract features from the segmented images to be used for classification. LBP was also used in feature extraction for the image segmented using superpixels.

6.1 Benchmark functions

The performances of both MFO–GSA and MFO–GSA–PSO hybrid optimization algorithms are evaluated using a set of benchmark functions. There are two types of functions: unimodal functions and multi-modal functions as shown in Fig. 6. The accuracy of the optimization functions is tested using the unimodal functions test whereas the multimodal functions test the capability of the optimization algorithms to avoid local optima. Figure 7 shows the convergence curves for the MFO–GSA hybrid algorithm for unimodal functions whereas Fig. 8 shows convergence curves for MFO–GSA–PSO hybrid algorithm for multimodal functions. Benchmark functions were evaluated for MFO, GSA, MFO–GSA hybrid, MFO–GSA–PSO hybrid and genetic algorithm for 1000 iterations. Results for both the algorithms as well as other algorithms are tabulated in Table 1 for comparison. It should be noted that for all these the global minimum value is zero. So, closer the result of the optimization algorithm to zero, better is the solution. Comparing the results of MFO and GSA we find that in most of the cases MFO gave better results than GSA. It is consistent with the discussion that MFO is better in exploration than GSA. This can be further analyzed by looking at the convergence curves of MFO and GSA for multimodal functions in Fig. 9. The GSA can be trapped in local optima as can be seen from the convergence curves. Hence, MFO can better explore the entire search space and find better global optima. The hybrid of MFO–GSA gave better results than both because it uses the strength of both the algorithms. Exploration by MFO and exploitation by GSA renders it a better

Function	Dim	Range
$f_1(x) = \sum_{i=1}^{n} x_i^2$	100	[-100,100]
$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	100	[-10,10]
$f_3(x) = \sum_{i=1}^{n} \left(\sum_{j-1}^{i} x_j\right)^2$	100	[-100,100]
$f_4(x) = \max_i \{\lvert x_i \rvert, 1 \le i \le n\}$	100	[-100,100]
$f_5(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	100	[-30,30]
$f_6(x) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	100	[-100,100]
$f_7(x) = \sum_{i=1}^{n} i x_i^4 + random[0,1)$	100	[1.28,1.28]

a

Function	Dim	Range
$F_8(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i \rvert})$	100	[-500,500]
$F_9(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	100	[-5.12,5.12]
$F_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} cos(2\pi x_i)\right) + 20 + e$	100	[-32,32]
$F_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	100	[-600,600]
$F_{12}(x) = \frac{\pi}{n}\{10\sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2 \ [1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2\} + \sum_{i=1}^{n} u(x_i, 10,100,4)$ $y_i = 1 + \frac{x_i + 1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	100	[-50,50]
$F_{13}(x) = 0.1\{\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2[1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2[1 + \sin^2(2\pi x_n)]\} + \sum_{i=1}^{n} u(x_i, 5,100,4)$	100	[-50,50]

b

Fig. 6 Benchmark functions. **a** Unimodal, **b** multimodal

optimization algorithm. The addition of PSO algorithm to the MFO–GSA hybrid further improves it. The search agents in MFO–GSA–PSO Hybrid can communicate their results with each other to move towards the global optima. Hence MFO–GSA–PSO hybrid gave the best solutions over all the algorithms. Also, it is apparent that genetic algorithm is far less efficient in finding the global optima for relatively small no. of iterations. Hence, both the hybrids gave better results than the rest and MFO–GSA–PSO hybrid gave better results than MFO–GSA hybrid.

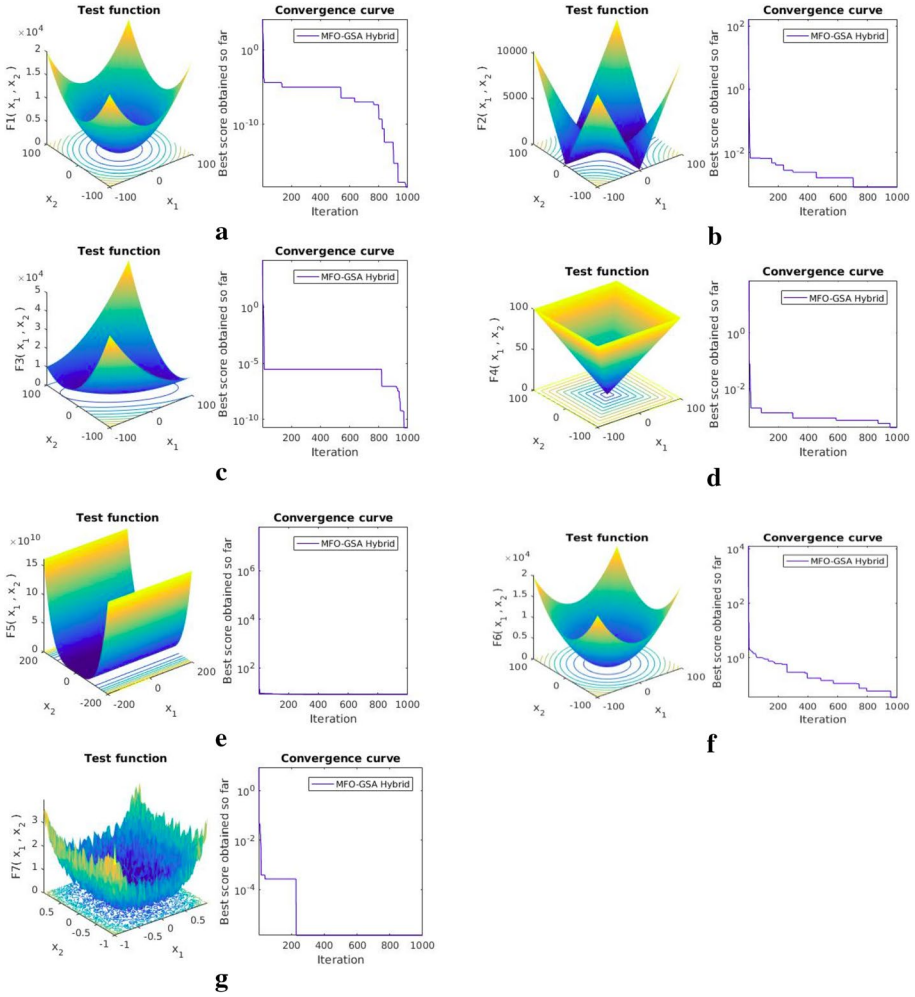


Fig. 7 Depicting the convergence curves for benchmark functions F1, F2, F3, F4, F5, F7 for MFO–GSA Hybrid for unimodal functions

6.2 Optimized K-means

Mean squared error (MSE) of every data point from its cluster point is calculated for the test datasets. Set 1 and Set 2 were custom datasets whereas iris and cancer were taken from University of California, Irvine (UCI) repository. In all the test datasets optimized K-means (using both MFO–GSA and MFO–GSA–PSO hybrids) performed better or equal to the standard K-means as evident from the mean-squared error value in Table 2. It shows that finding the initial K-means points using the hybrid algorithms is a good idea to avoid the local optima. Hence the optimized K-means fared better. The initial clusters were found using only 100 iterations of the hybrid algorithms. So, optimized K-means is reasonably fast by the virtue of the already fast K-means clustering. Hence, the classical K-means algorithm was modified and improved by the hybrids.

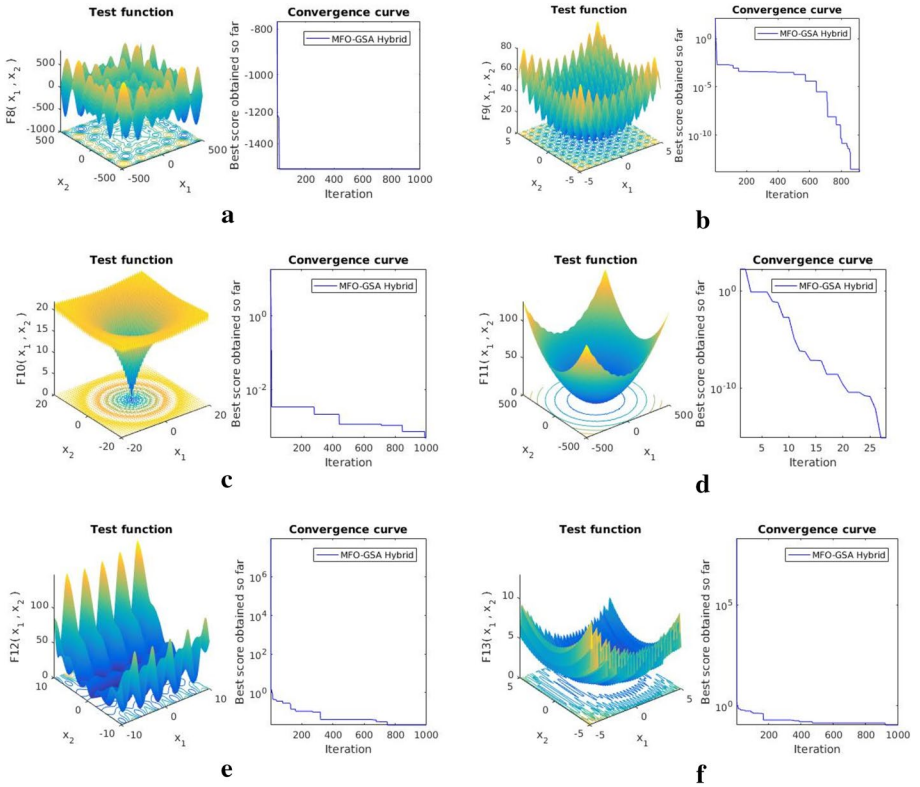


Fig. 8 Depicting the convergence curves for benchmark functions F8, F9, F10, F11, F12, F13 for MFO–GSA–PSO hybrid for multimodal functions

Table 1 Comparing optimum values for MFO, GSA, GA and hybrid algorithms for 1000 iterations

Function	MFO	GSA	MFO–GSA hybrid	MFO–GSA–PSO hybrid	Genetic algorithm
F1	6.6092e−21	1.321152	5.9991e−17	1.8394e−20	21886.03
F2	2.6592e−15	7.715564	0.0010437	0.0012333	56.51757
F3	7.2411e−09	736.5616	2.2439e−17	3.4189e−19	37010.29
F4	0.0043443	12.97988	0.00031035	0.00053415	59.14331
F5	15.7733	77360.4184	8.0849	7.2914	313.21418
F6	0.12108	2.86418	0.32572	0.054564	52.64496
F7	0.0046347	1.03951	2.8854e−05	1.1736e−05	20,964.83
F8	−2879.4219	−102.5649	−2471.4722	−1583.564	−13.37504
F9	10.9445	25.46556	0	0	2.14891
F10	4.4409e−15	5.32221e−10	0.00011668	0.00047837	5.4981e−5
F11	2.2639e−9	6.1948e−5	2.9199e−14	0	7.4198
F12	8.4387e−32	1.286e−25	0.24027	0.20087	0.2959
F13	1.992	12.4944	2.9821	0.11238	4.5195

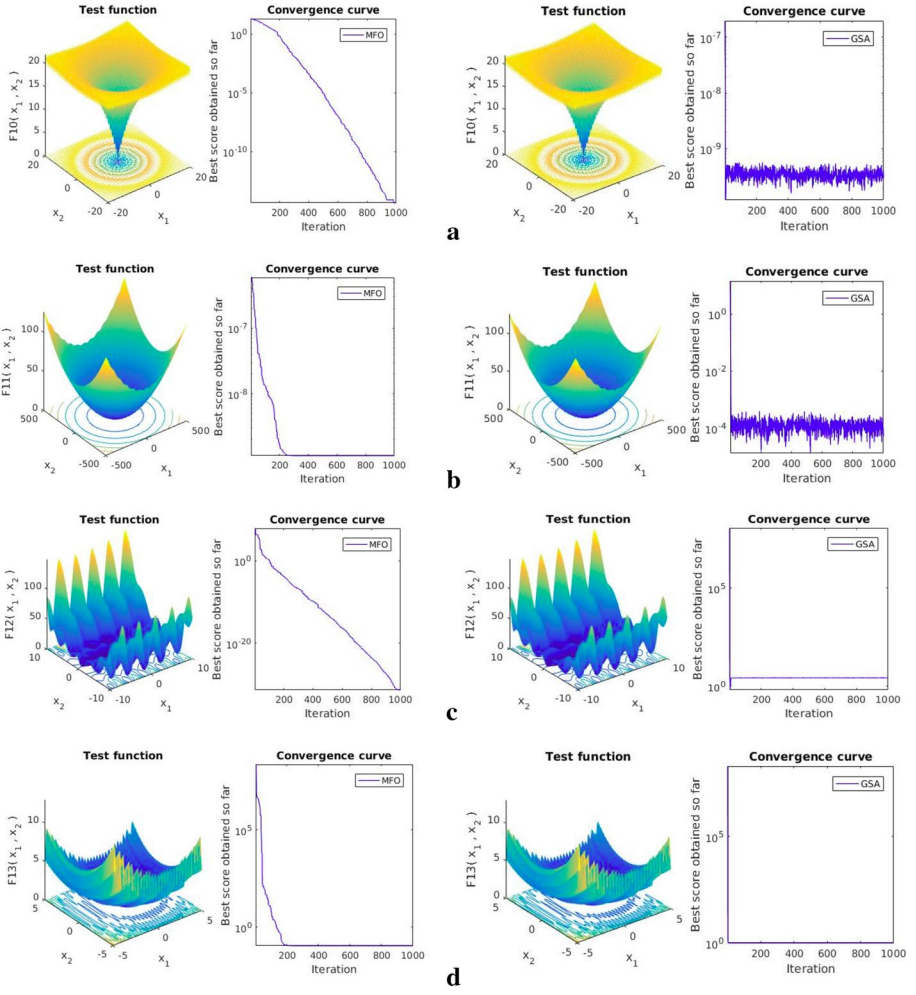


Fig. 9 Comparing the convergence curves of MFO (left) and GSA (right) for multimodal functions

Table 2 Comparing MSE of K-means and optimized K-means

Dataset	K-means	K-means using MFO–GSA	K-means using MFO–GSA–PSO
Set 1	1.11	1.11	–
Set 2	41.59	39.54	–
Iris	0.54	0.52	0.52
Cancer	29.12	27.46	28.2

The optimized K-means (using both MFO–GSA and MFO–GSA–PSO hybrids) was then used for segmentation of images in the apple dataset. The results are shown in Fig. 10 for MFO–GSA hybrid and in Fig. 11 for MFO–GSA–PSO hybrid. It can be seen clearly in

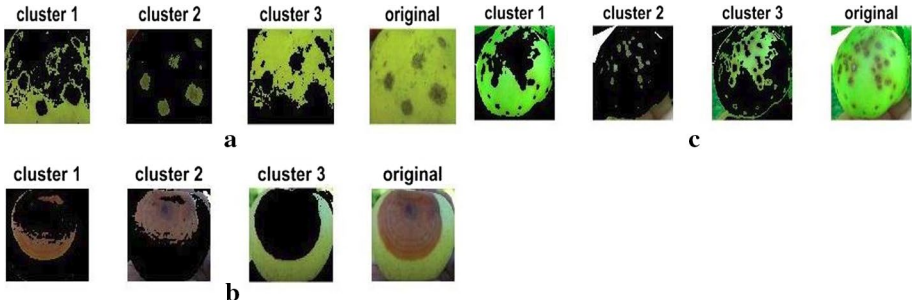


Fig. 10 Segmentation by K-means using MFO–GSA on "blotch", "rot" and "scab" respectively. The diseased clusters can be identified as cluster 2 in all cases

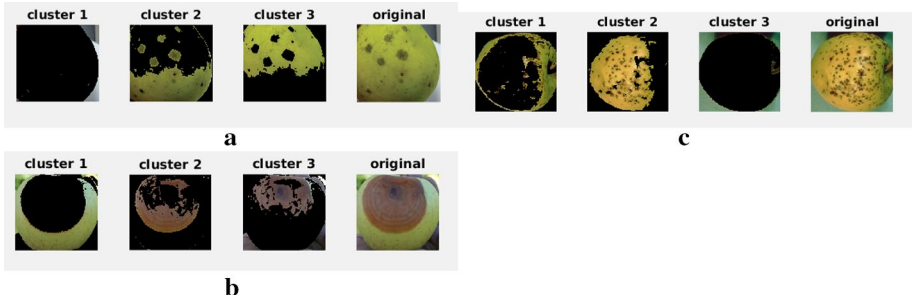


Fig. 11 Segmentation by K-means using MFO–GSA–PSO on "blotch", "rot" and "scab" respectively. The diseased clusters can be identified as cluster 2 in all cases

both the figures that the diseased portion belonged to cluster 2 in all the cases. Also, the diseased portion is in clear contrast with the background. Hence, the diseased option of the apple was successfully isolated using optimized K-means.

Two additional datasets i.e. echocardiogram and glass identification were evaluated for K-means using MFO–GSA–PSO and the MSE was 1.9191e+03 and 961.9 respectively. These datasets were also taken from UCI repository.

6.3 Multilevel thresholding using optimization

Unlike K-means it does not involve randomization and the whole search space is explored to obtain the best thresholds. Gray scale Images involve only one dimension with values ranging from 0 to 255 and hence the algorithm worked very fast for these images. This technique can even be used in higher dimensions like with color spaces RGB, LAB, HSV etc.

Calculating thresholds through the image and real time filters is a manual process. The proposed thresholding technique automates it removing the role of human and the error associated with it.

The results of multilevel thresholding using MFO–GSA are shown in Fig. 12 and using MFO–GSA–PSO is shown in Fig. 13. In these, images in LAB color space were first

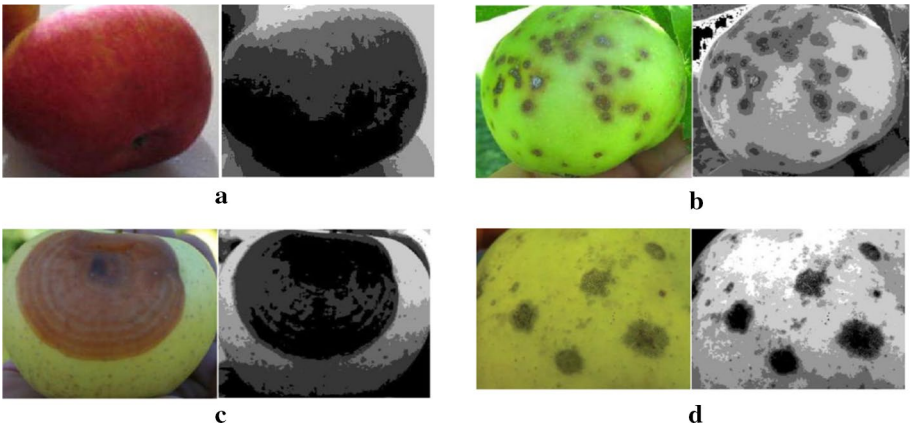


Fig. 12 Segmentation by multilevel thresholding using MFO–GSA

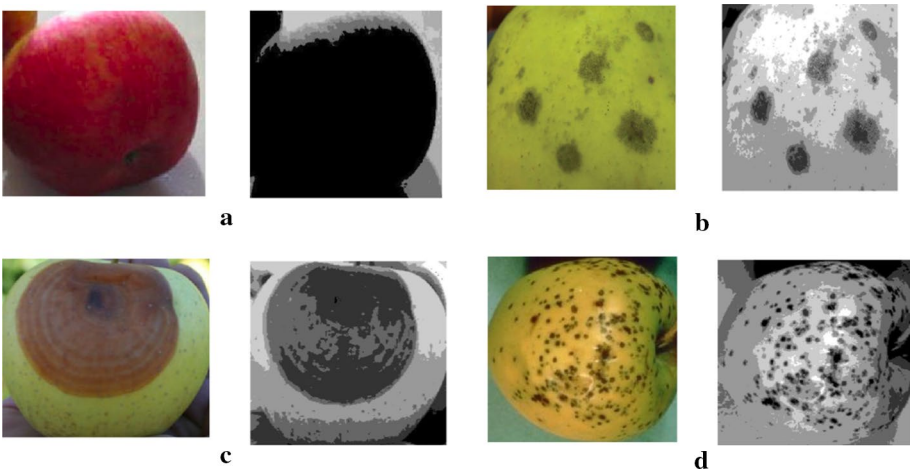


Fig. 13 Segmentation by multilevel thresholding using MFO–GSA–PSO

converted to grayscale and the thresholding was done. LAB color space images were chosen to reduce the effect of luminance. It is seen that the diseased portion of the apples is separated to different darker region than the rest of the apple.

6.4 Classification of segmented images

Features extracted from the segmented images obtained by optimized K-means were used to classify images. Multi class support vector machine was used to classify apples into normal, scab, rot, and blotch.

6.4.1 Classification of segmented images obtained by K-means using MFO–GSA hybrid

Multi class support vector machine was used for classification. Segmentation results for some sample images have been shown in Fig. 10. The data set consists of the categories: "normal", "scab", "rot", and "blotch". After segmentation using optimized K-means and extracting color-based features, classification was done. The "blotch" was excluded since it overlapped with "scab".

An accuracy of 81% was achieved with the multiclass model for classification into "scab", "rot" and "normal" classes. Overlapping of the disease is the cause for the inaccuracy which on research was found to be very common.

6.4.2 Classification of segmented images obtained by K-means using MFO–GSA–PSO hybrid

Classification of images was done after the images were segmented by K-means using MFO–GSA–PSO. The segmented images are shown in Fig. 11. LBP was used to extract the features. Multiclass SVM model was used for classification. The test set contained total of 60 apples with 15 apples of each of the four categories. The confusion matrix is given below in Table 3. The diagonal elements of the matrix represent the no. of correctly predicted apples.

Hence for "blotch" 13 out of 15 apples were correctly predicted, for "rot"15 out of 15 "normal" apples were correctly predicted, for "scab" 7 out of 15 apples were correctly predicted and 15 out of 15 apples were correctly predicted. Accuracy of "blotch", "rot", "scab" and "normal" was 86.7%, 100%, 46.7% and 100% respectively. Overall accuracy came out to be 83.33%. The features of "scab" and "blotch" apples overlapped but the features of blotch were more apparent hence the "blotch" apples were classified correctly but the some of the "scab" apples were classified into "blotch".

6.4.3 Classification of images segmented using graphs over superpixels

Initially this approach gave around very less accuracy. The primary reason for such results could be accommodated to the fact that after the segmentation apart from the disease cluster, on average 70% of the image was black. This black region was overpowering the feature set, as the result of which the whole data had extremely low variability. Lack of variability between the classes resulted in a skewed classifier that classifies each class as normal to maximize the accuracy.

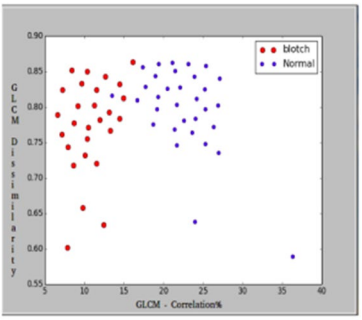
The same problem persisted with GLCM features. To tackle the problem, the modified LBP features mentioned earlier were used. Out of the two approaches, avoiding

Table 3 Confusion matrix for "blotch", "rot", "scab" and "normal"

Actual/predicted	Blotch	Rot	Scab	Normal
Blotch	13	1	1	0
Rot	0	15	0	0
Scab	4	4	7	0
Normal	0	0	0	15

	Normal	Rot	Scab
Normal	54	4	1
Rot	12	57	0
Scab	7	4	14

a Confusion matrix for superpixel approach



b GLCM dissimilarity vs correlation graph

Fig. 14 Confusion matrix for superpixel approach

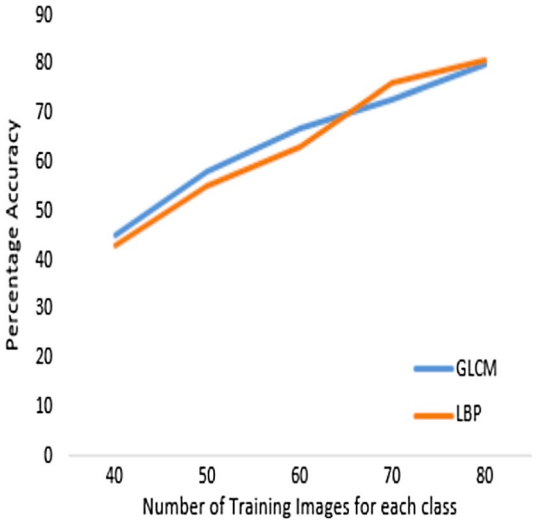


Fig. 15 Increase in accuracy for different features for increase in data

black pixels or making an average rectangle at center of the cluster, the latter gave better results. Here, the "blotch" apples were excluded since they overlapped with "scab".

After testing with modified local binary pattern and GLCM or Haralick features, local binary pattern gave 81.7% accuracy (confusion matrix is shown in Fig. 14), similarly Haralick features gave 78% accuracy. In case of local binary pattern since there were 59 features, as evident from Fig. 15, the accuracy increased with increase in training data.

7 Discussion and analysis

This paper presents two novel hybrid optimization techniques i.e. MFO–GSA and MFO–GSA–PSO. The latter gave better results for the benchmark functions. This implies that adding PSO motion to MFO–GSA leads to better results. This is because the search

agents can share their results using a global optimum obtained so far. These hybrid optimization techniques are used to improve upon the image segmentation process by K-means clustering and multilevel thresholding. It is important especially for multilevel thresholding since it is a computationally expensive process as it finds multiple thresholds, the value of each can vary from 0 to 255. The complexity of K-means clustering is O(*nkid*) where n = total elements, k = number of clusters, i = iterations, d = dimensions. This is reasonably efficient provided no. of iterations are less and K-means is not stuck in local optima. The proposed algorithm tackles both the shortcomings by initializing the 'k' centroids using MFO–GSA and MFO–GSA–PSO hybrid which gives a near optimal solution and it can then be exploited using K-means clustering. On comparing optimized K-means with classical K-means, the former gave better results with less error. However, the computational cost of K-means increases when we initialize the centroids using the optimization algorithms rather than doing it randomly. Time complexity of moth flame optimization, gravitational search algorithm and particle swarm optimization are $O(i(nlog(n) + nd))$, $O\left(i\left(n^2d\right)\right)$ and $O(i(nd))$ respectively, where 'i' is the number of iterations, 'n' is the total number of elements and 'd' is the dimension. Hence, overall time complexity for initializing the clusters is $O\left(i\left(nlog(n) + \left(n^2d\right)\right)\right)$. Hence if total elements are very large as compared to the number of iterations, this can increase the computational cost but if the number of iterations is relatively large than the increase in cost is not that significant.

Multilevel thresholding using Lab color space was done to minimize the effects of luminance. But this can easily be extended to other color spaces like HSV, RGB etc. and the performance can be compared as the problem complexity remains the same. In the classification results in sections above, it was evident that "blotch" and "rot" apples overlapped with each other. So, accuracy can be tremendously improved by ignoring one of these classes. To classify among these classes, some other data like odor also need to be incorporated.

In using graphs over superpixels, each superpixel is compared with all the rest of the superpixels to construct edges which increases its complexity although it can be optimized by comparing to only neighboring superpixels. Applying classification techniques on segmented image obtained by K-means using MFO–GSA hybrid and MFO–GSA–PSO hybrid gave an accuracy of 81% and 83.33% respectively. In the former blotch dataset is ignored. Classification using LBP and Haralick features of the segmented image obtained using graphs over superpixels gave an accuracy of 81.7% and 78% respectively. Here, superpixels were generated using SLIC algorithm.

8 Conclusion and future scope

The paper presented several approaches for detection of food quality in terms of segmentation algorithms. Both (MFO–GSA and MFO–GSA–PSO) hybrid algorithms have proven instrumental in the development. All the algorithms, optimized K-means, optimized multilevel thresholding, Lab based segmentation and SLIC superpixel algorithm have given promising results. There are ongoing advances in all the three algorithms i.e. MFO, GSA and PSO. Once better versions are well established, these can be used instead of the originals to construct the hybrid. Superpixel approach could be further developed and other algorithms like graph cut could also be incorporated and tested. Especially with respect to food items, odor plays a significant role in identifying rottenness, like fruits releasing

specific gases when rotten. Even though the current work presents a model based on vision data alone, incorporating odor data while building the model could improve the model. In the long-term, real time application of the proof of concepts presented in this paper could achieve the aim of reducing losses through spoilage in food storage facilities and improving food safety.

References

Abdullah MZ, Aziz SA, Mohamed AM (2000) Quality inspection of bakery products using a color-based machine vision system. J Food Qual 23(1):39–50

Achanta R, Shaji A, Smith K, Lucchi A, Fua P, Susstrunk S (2012) Slic superpixels compared to state-of-the-art superpixel methods. IEEE Trans Pattern Anal Mach Intell 34(11):2274–2282

Ahmadyfard A, Modares H (2008) Combining PSO and K-means to enhance data Clustering. In: 2008 International symposium on telecommunications, Tehran, pp 688–691

Baietto M, Wilson AD (2015) Electronic-nose applications for fruit identification, ripeness and quality grading. Sensors 15(1):899–931

Berna AZ, Lammertyn J, Saevels S, Di Natale C, Nicolaï BM (2004) Electronic nose systems to study shelf life and cultivar effect on tomato aroma profile. Sens Actuators B Chem 97(2):324–333

Buch H, Trivedi IN, Jangir P (2017) Moth flame optimization to solve optimal power flow with non-parametric statistical evaluation validation. Cogent Eng 4(1):1286731

Bulbul G, Hayat A, Andreescu S (2015) Portable nanoparticle-based sensors for food safety assessment. Sensors 15:30736–30758

Davidson VJ, Ryks J, Chu T (2001) Fuzzy models to predict consumer ratings for biscuits based on digital image features. IEEE Trans Fuzzy Syst 9(1):62–67

Dubey SR, Jalal AS (2012a) Detection and classification of apple fruit diseases using complete local binary patterns. In: 2012 Third international conference on computer and communication technology (ICCCT). IEEE

Dubey SR, Jalal AS (2012b) Adapted approach for fruit disease identification using images. Int J Comput Vis Image Process 2(3):44–58

Duncan TV (2011) Applications of nanotechnology in food packaging and food safety: barrier materials, antimicrobials and sensors. J Colloid Interface Sci 363(1):1–24

Echeverría G, Correa E, Ruiz-Altisent M, Graell J, Puy J, López L (2004) Characterization of Fuji apples from different harvest dates and storage conditions from measurements of volatiles by gas chromatography and electronic nose. J Agric Food Chem 52(10):3069–3076

Ghada T, Khorshid MH, Abou-El-Enien T (2016) Modified moth-flame optimization algorithms for terrorism prediction. Int J Appl Innovation Eng Manage (IJAIEM) 5(7):47–58

Goel L, Gupta D, Panchal VK (2012a) Dynamic model of blended biogeography-based optimization for land cover feature extraction. In: International conference on contemporary computing (IC3). Communications in computer and information sciences (CCIS-LNCS), vol 306. Springer, pp 8–19

Goel L, Gupta D, Panchal VK (2012b) Extended species abundance models of biogeography-based optimization. In: IEEE conference on computational intelligence, modelling and simulation (CIMSim). IEEE Xplore and CSDL, Kuantan, Malaysia, pp 7–12

Goel L, Panchal VK, Gupta D (2012c) Hybrid bio-inspired techniques for land cover feature extraction: a remote sensing perspective. Appl Soft Comput J 12(2012):832–849

Gomez AH, Wang J, Hu G, Pereira AG (2008) Monitoring storage shelf life of tomato using electronic nose technique. J Food Eng 85(4):625–631

Hatamlou A, Abdullah S, Nezamabadi-Pour H (2011) Application of gravitational search algorithm on data clustering, rough sets and knowledge technology. Lect Notes Comput Sci 2011:337–346

Hatamlou A, Abdullah S, Nezamabadi Pour H (2012) A combined approach for clustering based on K-means and gravitational search algorithms. Swarm Evolut Comput 6(2012):47–52

Joo ST, Kim GD, Hwang YH, Ryu YC (2013) Control of fresh meat quality through manipulation of muscle fiber characteristics. Meat Sci 95(4):828–836

Kennedy J (2011) Particle swarm optimization. Encyclopedia of machine learning. Springer, New York, pp 760–766

Koonsanit K, Jaruskulchai C, Eiumnoh N (2012) Determination of the initialization number of clusters in K-means clustering application using co-occurrence statistics techniques for multi-spectral satellite imagery. Int J Inf Electron Eng 2(5):785–789

Kumar TS, Mahesh Chandra M, Sreenivasa Murthy P (2011) Color based image segmentation using fuzzy c-means clustering. Int J Intell Electron Syst 5(2):47–51

Li H, He H, Wen Y (2015) Dynamic particle swarm optimization and K-means clustering algorithm for image segmentation. Optik Int J Light Electron Opt 126(24):4817–4822

Mehta A (2016) Disease prediction in apples using computer vision. Undergraduate thesis, Birla Institute of Technology and Science (BITS), Pilani, India

Mingru Z, Tang H, Guo J, Sun Y (2014) Data clustering using particle swarm optimization. Lect Notes Electr Eng Future Inf Technol 2014:607–612

Mirjalili S (2015) Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. Knowl Based Syst 89(2015):228–249

Mutlu M (2011) Biosensors in food processing, safety, and quality control. CRC, Boca Raton

Ojala T, Pietikainen M, Maenpaa T (2002) Multiresolution grayscale and rotation invariant texture classification with local binary patterns. IEEE Trans Pattern Anal Mach Intell 24(7):971–987

Osuna-Enciso V, Cuevas E, Sossa H (2013) A comparison of nature inspired algorithms for multi-threshold image segmentation. Expert Syst Appl 40(4):1213–1219

Prevolnik M, Škorjanc D, Čandek-Potokar M, Novič M (2011) Artificial neural networks - industrial and control engineering applications. InTech, pp 223–240

Rashedi E, Nezamabadi-Pour H, Saryazdi S (2009) GSA: a gravitational search algorithm. Inf Sci 179(13):232–248

Rehkugler GE, Throop JA (1986) Apple sorting with machine vision. Trans ASAE 29:1388–1397

Ren X, Malik J (2003) Learning a classification model for segmentation. In: Proceedings of the international conference on computer vision (ICCV)

Sanaeifara A, Mohtasebi SS, Ghasemi-Varnamkhastib M, Ahmadi H (2016) Application of MOS based electronic nose for the prediction of banana quality properties. Measurement 82:105–114

Sannakki SS, Rajpurohit VS, Nargund VB, Kulkarni P (2013) Diagnosis and classification of grape leaf diseases using neural networks. 2013 Fourth international conference on computing communications and networking technologies (ICCCNT), pp 1–5

Sarma A, Bhutani A, Goel L (2017) Hybridization of moth flame optimization and gravitational search algorithm and its application on detection of food quality. In: 2017 international conference on intelligent systems (IntelliSys). IEEE, pp 52–60

Shi J, Malik J (2000) Normalized cuts and image segmentation. IEEE Trans Pattern Anal Mach Intell (TPAMI) 22(8):888905

Srivastava S, Boyat S, Sadistap S (2014) A novel vision sensing system for tomato quality detection. Int J Food Sci 2014:1–11

Talbi EG (2002) A taxonomy of hybrid metaheuristics. J Heuristics 8(5):541–564

Unay D, Gosselin B, Kleynen O, Leemans V, Destain M, Debeir O (2012) Automatic grading of Bi-colored apples by multispectral machine vision. Comput Electron Agric 75(1):204–212

Van den Bergh M, Boix X, Roig G, de Capitani B, Van Gool L (2012) Seeds: superpixels extracted via energy-driven sampling. In: Proceedings of the european conference on computer vision (ECCV). Lecture notes in computer science, vol 7578. Springer, pp 1326

Yang T, Huang H, Zhu F, Lin Q, Zhang L, Liu J (2016) Recent progresses in nanobiosensing for food safety analysis. Sensors 16(7):1118

Affiliations

Lavika Goel[1] · Sundaresan Raman[1] · Subham Swastik Dora[1] · Anirudh Bhutani[1] · A. S. Aditya[1] · Abhinav Mehta[1]

Sundaresan Raman
sundaresan.raman@pilani.bits-pilani.ac.in

Subham Swastik Dora
f2014644@pilani.bits-pilani.ac.in

Anirudh Bhutani
f2013114@pilani.bits-pilani.ac.in

A. S. Aditya
f2013079@pilani.bits-pilani.ac.in

Abhinav Mehta
f2013183@pilani.bits-pilani.ac.in

[1] Department of Computer Science & Information Systems, Birla Institute of Technology and Science (BITS), Pilani, Rajasthan 333031, India