Multimedia Tools and Applications (2019) 78:34027–34063
https://doi.org/10.1007/s11042-019-08114-x

# Hyper-spectral image segmentation using an improved PSO aided with multilevel fuzzy entropy

Rupak Chakraborty[1] · Rama Sushil[2] · M. L. Garg[3]

Received: 5 August 2018 / Revised: 28 July 2019 / Accepted: 13 August 2019 /
Published online: 7 September 2019

## Abstract

This paper proposes a novel histogram-based multi-level segmentation scheme of hyper-spectral images. In the proposed scheme an Improved Particle Swarm Optimization (IPSO) algorithm is implemented as a nature-inspired evolutionary algorithm to overcome the draw-back of premature convergence and hence getting stuck in local optima problem of PSO. The high-dimension of PSO is decomposed into several one-dimensional problems and premature convergence is removed from each one-dimensional problem. This algorithm is further extended for replacing the worst particles by the fittest particles, determined by their fitness values. Multiple optimal threshold values have been evaluated based on fuzzy-entropy aided with the proposed algorithm. The performance of the IPSO is compared statistically with other global optimization algorithms namely Cuckoo Search (CS), Differential Evolution (DE), FireFly (FF), Genetic Algorithm (GA), and PSO. The produced segmented output of IPSO-fuzzy is then combined with the available ground truth values of image classes to train a Support Vector Machine (SVM) classifier via the composite kernel approach to improving the classification accuracy. This hybrid approach (IPSO-SVM) is then applied to popular hyper-spectral imageries acquired by AVRIS and ROSIS sensors. The final evaluated outcomes of the proposed scheme are also qualitatively compared to show its effectiveness over the other state-of-art global optimizers.

**Keywords** Improved particle swarm optimization (IPSO) · Multilevel-thresholding · Hyper-spectral imageries · Fuzzy entropy · Support vector machine (SVM) · Composite kernel

✉ Rupak Chakraborty
rupak.jis@gmail.com

Rama Sushil
rama.sushil@dituniversity.edu.in

M. L. Garg
dr.ml.garg@dituniversity.edu.in

[1] Department of Computer Science and Engineering, Bennett University, Greater Noida, India

[2] Department of Information Technology, DIT University, Dehradun, India

[3] Department of Computer Science and Engineering, DIT University, Dehradun, India

# 1 Introduction

Image segmentation is a fundamental step for meaningful analyzing and interpretation of an image. It is considered as a mandatory preprocessing step for extracting objects from its backgrounds in many computer vision oriented applications. Remote sensing Hyper-Spectral Imageries (HSI) have got importance as it played an important role to provide solutions in many applications like soil erosion monitoring, land-cover study, flood monitoring, assessment of forest resources etc. Unlike other spectral imaging, HSI collects information across the electromagnetic spectrum to obtain the spectrum of each pixel with the purpose of finding objects. As sensors capture a wide range of wavelengths of contiguous spectral bands so, it may cause a curse of dimensionality problem for effective segmentation [24]. Effective HSI segmentation still demands a challenging issue for the researchers. Segmentation denotes to put all homogeneous pixels in clusters concerning to some common features like color, intensity, and texture so that it can be analyzed properly. Thresholding approach is broadly accepted to achieve this goal to discriminate the objects from their background pixels.

In bi-level thresholding, the total image region is subdivided into two homogeneous pixel regions based on the histogram, minimum-variance, edge-detection etc. The reconstructed final image is a binary image where all pixels carry higher gray values than a threshold level, are put into a class whereas pixels having fewer values than a threshold level are put into another class. Some useful histogram, edge detection, minimum variance, interactive pixel classification bi-level thresholding techniques are surveyed in the literature [3, 41, 47, 55, 62]. Some global entropy-based bi-level segmentation techniques available in the literature like [15, 25, 29, 42, 48, 63]. These techniques are effective enough for bi-level thresholding, but not preferred for complex segmentation problems where multiple threshold values are desired. Adaptive neuro-fuzzy, hierarchical, histogram-based approaches for multilevel effective color image segmentation schemes are noted in the literature [8, 13, 28]. Multilevel approaches may still struggle because of its high time and working complexities [4, 27, 45, 61]. Researchers tried to overcome those issues by incorporating various exhaustive search process techniques to enhance the computational speed like [14, 54, 65] etc. Multilevel thresholding techniques with entropy maximization or minimization along with any meta-heuristic technique holds recent research interest [21].

Section 2 presents a literature survey related to meta-heuristic approaches, used to find multiple threshold values and a brief survey regarding supervised classification. Section 3 formulates the problem using fuzzy entropy for multilevel segmentation. Section 4 solves the multilevel problem by generating optimal solutions using the proposed Improved Particle Swarm Optimization (IPSO) technique. Dataset, experimental parameter setup are covered up in Section 5. The simulated results of the unsupervised segmentation scheme with its performance evolution along with hybrid classification obtained by different meta-heuristics are presented in Section 6. Finally, Section 6 draws a conclusion of the paper.

# 2 Related works

A meta-heuristic procedure is used to generate a search algorithm to provide the best solution to an optimization problem with limited computation capacity. As these approaches provide a better and easy sample set of solutions for a wide variety of complex problems, so it has been flourishing since almost last decade. These techniques are quite easy to generate optimal solutions to NP-hard problems as users don't need to have detailed knowledge regarding the selection of the initial population. Some well-known derivative-free optimizing techniques like Genetic Algorithm (GA) [11, 43], Firefly (FF) [38, 44], Artificial Bee Colony (ABC) [7, 23, 26], Cuckoo search algorithm [6, 56], Differential Evolution (DE) [50] based approaches to find optimal solutions of meta-heuristic problems are surveyed in the literature.

Particle Swarm Optimization (PSO) is another popular stochastic global optimization algorithm of recent category [12, 17, 40, 52]. The algorithm was inspired by the real-life swarm behaviors and has become an effective tool for multilevel problem optimization. But the main drawback of this basic PSO is that it may get stuck in sub-optimal solution regions and the problem may increase further in high-dimensional spaces as its success depends on the combination of global exploration and local exploitation features during the optimization process. Researchers identified this problem and tried to improve the algorithm. Cooperative and comprehensive learning- based PSO algorithm surveyed in the literature [5, 35, 60]. Further literature reveals that effective multi-swarm based PSO algorithm proposed by various researchers like [37, 68]. In the year 2004, Ratnaweera et al. [46] proposed a self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. They incorporated mutation concept into PSO first and later on social and cognitive parts of PSO had been updated to generate new velocity of each particle whenever stagnancy occurred. In the recent year, Nayak et al. [39] proposed a modified PSO aided with extreme learning machine mechanism for pathological brain detection. Modified PSO successfully optimized the hidden node parameters and combined MPSO-ELM overcame the drawback of single SVM or ELM based classifiers. So there is a scope to form an improved PSO algorithm so that premature convergence caused by the curse of dimensionality can be removed and as well as high complexity and slow rate of convergence problem can also be improved.

## 2.1 A brief survey of supervised classification

In supervised classification, remote sensing image data are analyzed quantitatively with the help of its known ground truth values. Challenge is to develop an approach to classify the spectral domain into regions so that it matches with ground cover classes and hence increase the accuracy of the technique. One lightweight sparsity-based algorithm, Basic Thresholding pixel-wise classifier is one of the approaches proposed by [58, 59] increased the accuracy of classification and reduced computational cost a lot. Some popular multiple learning kernel-based approaches for hyper-spectral data are surveyed in the literature [9, 10, 53] as well. A simple composite kernel-based approach that controlled the relative proportion of the spatial and segmented information found in the literature [31]. A multi-modal logistic regression and Markov random fields based hyper-spectral image segmentation produced effective results proposed by [30]. Some more approaches also exist in the literature which combined supervised and unsupervised classification to form hybrid approaches to obtain better accuracy and less computational cost [16, 57]. Mughees et al. [36] proposed boundary adjustment technique by merging spatial and spectral information of hyper-spectral data that enhanced the accuracy over the ground truth. Ghamsi et al. [18] proposed a Majority Voting approach to combine the segmented results of hyper-spectral images obtained by Fractional-Order Darwinian PSO as an optimization algorithm with the supervised classification map to generate high accuracy of classified objects. Sarkar et al. [51] proposed Differential Evolution (DE) based optimized segmentation of land cover images and then formed one composite kernel by SVM with the known training and test data to improve the performance of supervised classification. Some current CNN-based spatial-spectral classification approaches of hyper-spectral images have been surveyed as well in the literature. A novel deep pixel-pair method had been proposed to increase the accuracy over conventional CNN-based approaches [32]. In that approach, center and surrounding pixels had been combined and got trained by CNN and final results determined by the voting strategy. Li et al. [33] combined composite kernels (spatial+spectral) with Adaboost framework which later incorporated with weighted Extreme Learning Machine to increase accuracy in imbalanced large datasets. Zhang et al. [67] proposed a diverse region-based CNN classification framework to exhibit spatial-spectral context- sensitivity which resulted accurate pixel classification. To remove the curse of dimensionality and excessive computation time problem at the time of classifying spectral images, a collaborative classification framework had been proposed [66]. In that method, deep spectral features of HSI had been extracted by the trained CNN model and later effective statistical image features had been learned through contextual vectors which were formed by the combination of the rich spectral, spatial and statistical information of the images. In the year 2018, Xu et al. [64] proposed a two-branch CNN to extract spatial-spectral information from multi-band images. Extracted spatial-spectral information had been combined through a dual-tunnel branch of cascade network to achieve more accurate classification results. Li et al. [34] presented a survey paper of dimension reduction of HSI based on discriminant analysis. The approaches aimed to reduce the dimensionality of highly correlated spectral bands without losing important information. Experimental results were compared between four well-known approaches namely Kernel Discriminant Analysis, Kernel Local Fisher's Discriminant Analysis, Kernel Sparse Graph-based Discriminant Analysis, and Kernel Composite Graph-based Discriminant Analysis (KCGDA). In KCGDA, spectral-spatial information were integrated to achieve the highest accuracy. Motivated by the spectral-spatial classification approaches, one innovative spectral-spatial approach has been presented here to achieve highest accuracy. Though CNN based approaches generated higher accuracies for HSI dataset, but the computational cost for small datasets like HSI classification may discourage researchers. So, motivated by the above stated spatial-spectral combination approaches, we have proposed a composite spectral-spatial kernel-based application got trained by the conventional SVM classifier. The detailed contribution has been discussed next.

## 2.2 Author's contribution

The key contribution of our approach is to produce accurate, robust and effective unsupervised segmentation results which will later be given as input to the supervised SVM classifier for improving the classification accuracy. Fuzzy entropy has been generated for multi-level segmentation problem. As problem has been extended to convex optimization search space, so improving the stagnancy caused by PSO has been essentially required. To remove the stagnancy of the PSO problem, an Improved version of PSO has been proposed which communicates through the introduction of context parameter. Features have been extracted using Principal Component Analysis (PCA) and then Attribute Profile (APs) have been formed from each PC to finally form Extended Multi-Attribute Morphological Profiles (EMAPs) which has been used here as spatial information [31]. Segmentation Map obtained by multi-level fuzzy entropy and IPSO has been used here as spectral information. The combined spatial and spectral kernels have been trained by the SVM classifier to generate the final results. Step-wise contribution is determined like following:

- First of all the required parameters along with the threshold levels are set for the IPSO algorithm.

- Next, composite high-dimensional swarm (formed with the help of threshold level) is broken into several one-dimensional swarms in search space.
- In the proposed approach each swarm communicates with each other by exchanging information to find composite fitness of an entire system with the introduction of Context Parameter (CP) discussed in detail later.
- Next, each particle's new velocity is updated based on the Pbest information of any particle within a swarm to avoid premature convergence. Hence the new position of the particle is also updated for each generation by the tournament selection procedure.
- Membership function based fuzzy entropy is calculated from image histogram which helps to find best velocity and position of the particle stated in the above step.
- The segmented output as spectral information and known spatial information (obtained by the PCA and EMAP [31]) are then combined to form one composite Kernel (CK), based on the Gaussian radial basis function kernel with SVM is known as (SVM-CK) [31] is then considered to achieve better classification accuracy shown in the Fig. 1.

Next section formulates multilevel thresholding problem by the fuzzy entropy based objective function and is solved by using the proposed IPSO technique to get desired optimal thresholds.

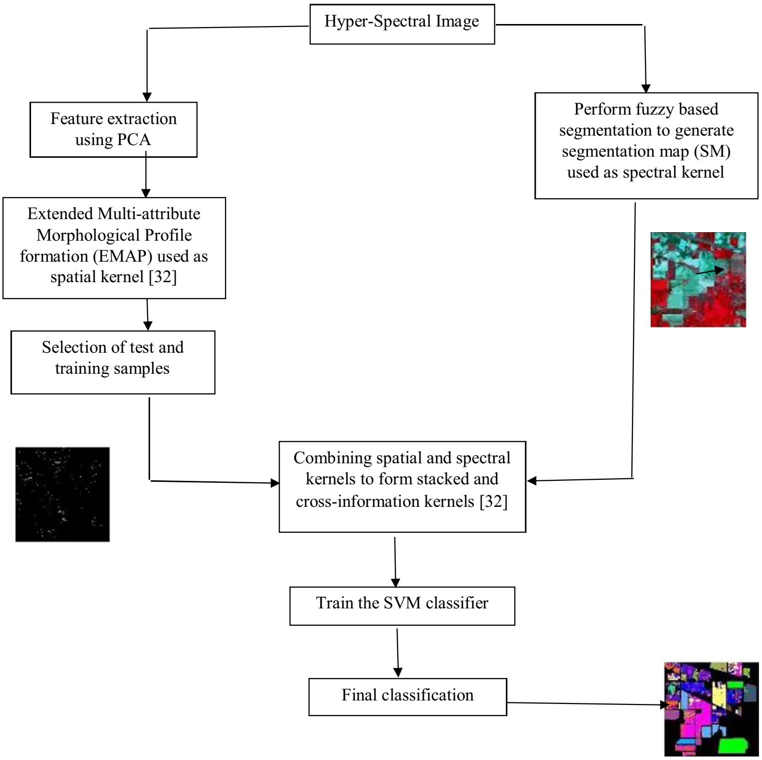

Hyper-Spectral Image
Feature extraction using PCA
Perform fuzzy based segmentation to generate segmentation map (SM) used as spectral kernel
Extended Multi-attribute Morphological Profile formation (EMAP) used as spatial kernel [32]
Selection of test and training samples
Combining spatial and spectral kernels to form stacked and cross-information kernels [32]
Train the SVM classifier
Final classification

**Fig. 1** Proposed methodology

## 3 Problem formulation

Multilevel thresholding of color image determines two or more optimum thresholds for each of the three components (red, green, blue) of the image. L*a*b* color space is used for segmentation problem than non-uniform color space like RGB to represent colors clearly. First of all, the RGB images have been converted to CIE L*a*b* color space using MATLAB. L* indicates lightness, a* red-green and b* blue-yellow color where each component value lies between 0 to 255. Let I is an original color image with size $M \times N$ and $r_i$ be the histogram of the image which is the combination of three components to store the color information i.e. $r_i = (L_i^*; a_i^*; b_i^*)$ for $i = (1, 2, ..L)$ where $L_i^*$, $a_i^*$, $b_i^*$ denote $i^{th}$ intensity value of the channels. L denotes the number of levels (0-256).

### 3.1 Multi-level fuzzy entropy

Traditional classical set (S) states that a collection of elements will either belong to or not belong to set S. But a fuzzy set is known as the extension of a classical set where an element can belong to partially also. Consider a fuzzy set F and can be defined as

$$F = \{(y, \mu_F(y)) \mid y \in Y\} \tag{1}$$

where $0 \leq \mu_F(y) \leq 1$ and $\mu_F(y)$ is called the membership function, which finds the closeness of x to F.

In this paper we have chosen trapezoidal membership function of a fuzzy set to calculate the membership of k segmented regions, $\mu_1, \mu_2, ....\mu_k$ by using $2 \times (k-1)$ unknown fuzzy parameters. Parameters are chosen as $a_1, b_1..., a_k - 1, b_k - 1$, where $0 \leq a_1 \leq b_1 \leq ... \leq a_{k-1} \leq b_{k-1} \leq L - 1$ displayed in the Fig. 2. Hence following membership function can be defined for k-level thresholding:

$$\mu_1(n) = \begin{cases} 1 n \leq a_1 \\ \frac{n-b_1}{a_1-b_1} a_1 \leq n \leq b_1 \\ 0 n > b_1 \end{cases} \tag{2}$$

$$\mu_{k-1}(n) = \begin{cases} 0 n \leq a_{k-2} \\ \frac{n-a_{k-2}}{b_{k-2}-a_{k-2}} a_{k-2} < n \leq b_{k-2} \\ 1 b_{k-2} < n \leq a_{k-2} \\ \frac{n-b_{k-1}}{a_{k-1}-b_{k-1}} a_{k-1} < n \leq b_{k-1} \\ 0 n > b_{k-1} \end{cases} \tag{3}$$

$$\mu_k(n) = \begin{cases} 1 n \leq a_{k-1} \\ \frac{n-a_k}{b_k-a_k} a_{k-1} < n \leq b_{k-1} \\ 1 n > b_{k-1} \end{cases} \tag{4}$$

The maximum fuzzy entropy for each of k-level segmentation can be expressed as:

$$\left.\begin{array}{l} H_1 = -\sum\limits_{i=0}^{L-1} \frac{r_i * \mu_1(i)}{R_1} * ln(\frac{r_i * \mu_1(i)}{R_1}), \\ H_2 = -\sum\limits_{i=0}^{L-1} \frac{r_i * \mu_2(i)}{R_2} * ln(\frac{r_i * \mu_2(i)}{R_2}), \\ \quad . \\ H_k = -\sum\limits_{i=0}^{L-1} \frac{r_i * \mu_k(i)}{R_k} * ln(\frac{r_i * \mu_k(i)}{R_k}), \end{array}\right\} \tag{5}$$

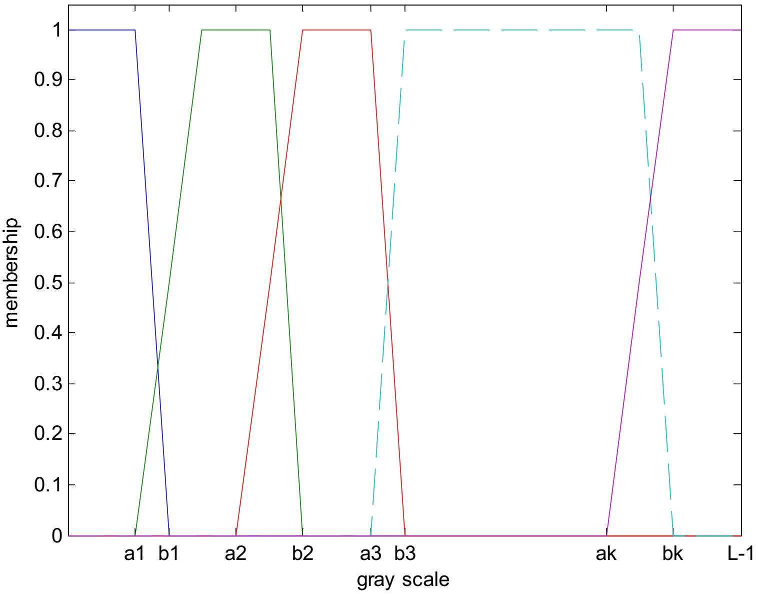


**Fig. 2** Fuzzy membership function for k-level thresholding

where, $R_1 = \sum_{i=0}^{L-1} r_i * \mu_1(i)$, $R_2 = \sum_{i=0}^{L-1} r_i * \mu_2(i)$, ...., $R_k = \sum_{i=0}^{L-1} r_i * \mu_k(i)$ Next, the optimum value of parameters can be achieved by maximizing the total entropy described in the following equation:

$$\psi(a_1, b_1, ...a_{k-1}, b_{k-1}) = max([H_1(t) + H_2(t)] + .. + H_k(t)]) \tag{6}$$

For easy computation, two dummy thresholds are $t_0 = 0$ and $t_k = L - 1$ are introduced with $t_0 < t_1 < ... < t_{k-1} < t_k$. The computational complexity for this multilevel problem is very expensive and high $O(L^{k-1})$. It is also noted that growth of complexity is proportional to the increase of levels [20]. So, IPSO as a global optimization technique is proposed to optimize the (6) by reducing the time complexity and hence produce best optimal solutions in exhaustive search space. Multiple $(k - 1)$ number of thresholds using fuzzy parameters can be obtained by the following formula:

$$t_1 = \frac{(a1 + b1)}{2}, t_2 = \frac{(a2 + b2)}{2}, .., t_{k-1} = \frac{(a_{k-1} + b_{k-1})}{2} \tag{7}$$

## 4 Overview of particle swarm optimization (PSO) algorithm

In PSO algorithm, particles (multiple solutions) are put into multidimensional spaces and the fitness of each particle is evaluated in each iteration. Here the new velocity, as well as new position of a particle, is calculated on the based on its present velocity and position at the start of a new iteration. The distance between particle's present and the best position called as **pb** whereas the distance between the present position of a particle and the position of the best particle among the total swarms is called as the **gb**. Assume it is required to solve a D-dimensional optimization problem by minimizing the objective function $f(x)$ given as

$$minf(s), s = [s_1, s_2, ....s_D] \tag{8}$$

where D denotes the number of parameters to be optimized. Here in basic PSO algorithm, a swarm of P "particles" is flown in a D-dimensional search space in a random manner to find the optimum solution of a fitness function. Let $S_a = (S_a^1, S_a^2, ....S_a^D)$ represents the position, $V_a = (V_a^1, V_a^2, ....V_a^D)$ the velocity and $pb_a = (pb_a^1, pb_a^2, ....pb_a^D)$ the best previous position obtained so far for every $i^{th}$ particle which consists of the lowest fitness values. Similarly, the best position noted for each iteration in the entire search space is denoted by $gb = (gb^1, gb^2, ....gb^D)$. The velocity $V_a^d$ of each particle in $d^{th}$ dimension for each iteration is determined by the following equation:

$$\begin{aligned} V_a^d &= w * V_a^d + c_x * r1_a^d * (pb_a^d - S_a^d) + c_y * r2_a^d * (gb^d - S_a^d) \\ V_a^d &= sign(V_a^d) * min(abs(V_a^d), V_{max}^d) \end{aligned} \tag{9}$$

The additive influence of $V_i^d$ found in the previous iteration known as "momentum" component, individual weighting distance of particles from $Pb_a^d$ known as a cognitive component and its distance from $gb^d$ known as social component, $c_x$ and $c_y$ are the acceleration coefficients, $r1, r2$ are the random numbers randomly generated from 0 to 1, w is the inertia weight contains a high value at initial then gradually decreases according to the following equation:

$$w = w_{initial} - \theta_{IW} \times it \tag{10}$$

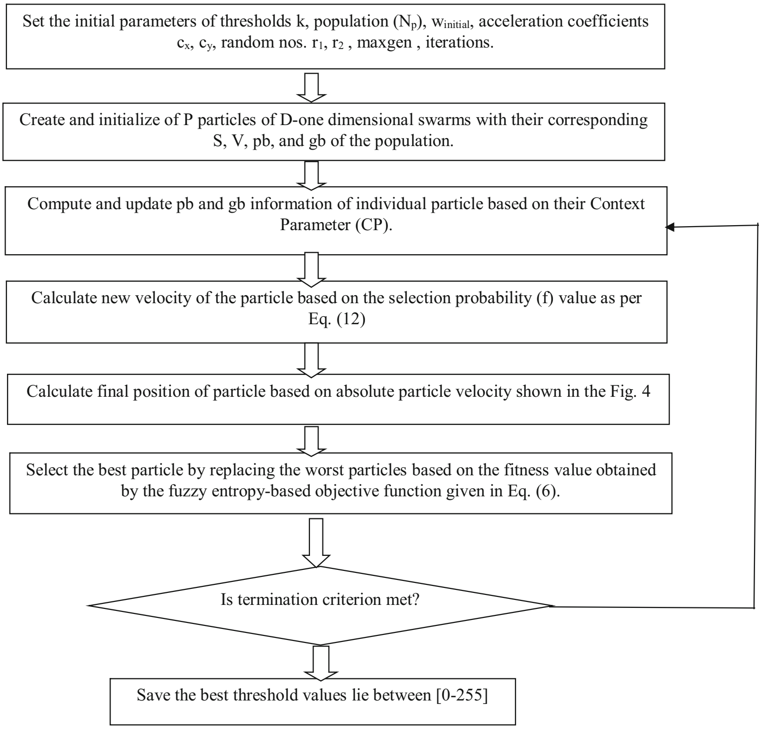

Set the initial parameters of thresholds k, population (N_p), w_initial, acceleration coefficients c_x, c_y, random nos. r_1, r_2 , maxgen , iterations.
Create and initialize of P particles of D-one dimensional swarms with their corresponding S, V, pb, and gb of the population.
Compute and update pb and gb information of individual particle based on their Context Parameter (CP).
Calculate new velocity of the particle based on the selection probability (f) value as per Eq. (12)
Calculate final position of particle based on absolute particle velocity shown in the Fig. 4
Select the best particle by replacing the worst particles based on the fitness value obtained by the fuzzy entropy-based objective function given in Eq. (6).
Is termination criterion met?
Save the best threshold values lie between [0-255]

**Fig. 3** Block-diagram of the proposed IPSO algorithm

$w_{initial}$ is the inertia weight at the starting condition, $it$ is the present iteration count, $\theta_{IW}$ is the slope of inertia weight variation. The formula to get the best position of a particle in PSO is:

$$S_i^d = S_i^d + V_a^d \tag{11}$$

However, PSO algorithm suffers from two basic problems like "curse of dimensionality" and a tendency of premature convergence which is quite common in several optimization algorithms. This type of problem leads particle's motion to get stuck and hence algorithm get prone to local optima convergence. As a result of that, some particles will gain low fitness values although their dimension values lie very close to the global optimal solution. Cooperation operation can remove this problem by acquiring information from "best" dimensions and preventing useful information from being unnecessarily discarded.

### 4.1 Motivation for improved PSO

"Curse of dimensionality" problem can be resolved by dividing a composite high-dimensional swarm into several one-dimensional swarms, which will cooperate with each other dynamically by exchanging the information to form composite fitness of an entire system. High-dimensional space can be divided into several one-dimensional spaces, so cooperation between each one-dimension can help to save the most useful information to

**Algorithm 1** Pseudo code of the proposed improved PSO (IPSO) technique.

```
BEGIN
DEFINE $CP(b, y) \equiv (P_1.gb, P_2.gb, ..., P_{b-1}.gb, y, P_{b+1}.gb, .....P_D.gb,)$
Create and initialize P particles for each of the D one-dimensional swarms with their corresponding S, V and pb and the gb of the population
REPEAT:
  FOR b=1 to D
    IF ($b_{th}$ swarm got stagnated for last few iterations)
      Find $P_{replace}$ "best" particles and $P_{replace}$ worst particles in the $b_{th}$ swarm on the basis of their fitness values
      Destroy those $P_{replace}$ worst particles and replace them by clones of $P_{replace}$ "best" particles
    ENDIF
    FOR i=1 to P
      IF $f(\mathbf{CP}(b, P_b.S_a)) < f(CP(b, P_b.pb_a))$ then $P_b.pb_a = P_b.S_a$
      IF $f(\mathbf{CP}(b, P_b.S_a)) < f(CP(b, P_b.gb))$ then $P_b.gb = P_b.S_a$
      ENDIF
      ENDIF
      Now calculate new velocity of particle in the following manner
      $P_b.V_a \leftarrow w * P_b.V_a + c_a * (r1)_i * (P_b.pb_{f_a} - P_b.S_a)$ Where f will be selected according to Step 1- Step 4 mentioned in the algorithm
      Previous equation reduces to $P_b.V_a \leftarrow sign(P_b.V_a) * min(abs(P_b.V_a), P_b.V_{max})$
      Calculate final position of particle: $P_b.S_a \leftarrow P_b.S_a + P_b.V_a$
    END FOR
  END FOR
UNTIL termination condition meets
END
```

accelerate convergence. Further premature convergence, caused by traditional PSO algorithm may be overcome by modifying the new velocity of each particle based on its **pb** information of within the swarm. In the next subsection, the detailed Improved Particle Swarm Optimization (IPSO) is proposed to overcome both the problems like "curse of dimensionality" and the tendency of premature convergence at a time. Figure 3 describes the block-diagram of the algorithm where as Algorithm 1 illustrates the detail working of the algorithm with code snippet.

### 4.2 Proposed improved particle swarm optimization (IPSO)

In this proposed approach, a D-dimensional problem is decomposed into D-one dimensional swarms where each swarm consists of S particles. A final global solution is evaluated by aggregating all the **gb** solutions achieved from each individual swarm. The fitness evaluation of particles is done based on the introduction of a new parameter called Context parameter (denoted by **CP**) that will be used to exchange information among all the individual swarms. The size of Context Parameter (CP) for a D-dimensional problem is equal to D-dimension itself. Here, when a $b^{th}$ swarm is active, then the **CP** is formed by remaining $(D-1)$ swarm's **gb** values. Then the $b^{th}$ row of the **CP** is filled by each particle of $b^{th}$ swarm one by one. Each such **CP** is calculated for finding its composite fitness. So the **pb** value $(P_b.pb_a)$ for the $a^{th}$ particle and the **gb** solution $(P_b.gb)$ for $b^{th}$ swarm are determined in that manner so that they not only depend on the performance of the $b^{th}$ swarm alone. Now the velocity and position of each particle in the $b^{th}$ swarm (denoted by $P_b.S_a$ and $P_b.V_a$) are updated based on these pb and gb solutions. Lastly the $b^{th}$ entry of the **CP** is filled by the newly calculated $P_b.gb$ and this process is continued for each $b^{th}$ swarm, till all the relevant context parameters are filled one by one. So, in brief, it can be concluded that the total search space is divided into D subspaces for D individual swarms and these swarms communicate with each other through their corresponding **CP** to determine their individual $P_b.pb_a$ and $P_b.gb$. The final context vector is calculated by concatenating all the evaluated $P_b.gb$ determined across all the swarms.

However, this above mentioned technique may still be get trapped in suboptimal locations within search space, where all individual solutions may fail to produce a better solution every time. To overcome this problem, we propose some modifications to the above-mentioned approach to determine new velocities and positions of each particle in each individual swarm in subspaces. So, the created problem of stagnation because of premature convergence is seriously taken care off, by permitting each particle to adjust its velocity (so, the position also), based on the **pb** information of any particle within the swarm. This helps to discourage premature convergence within the swarm strongly.

Now to select the particle, whose **pb** can be used to find the new velocity of any given particle within a given one-dimensional swarm is shown in the Algorithm 2. Hence, the new velocity update relation for each particle in a given swarm is demonstrated as

$$P_b.V_a \leftarrow w * P_b.V_a + c'_a * rand_a * (P_b.pb_{f_a} - P_b.S_a) \tag{12}$$

where $f_a$ decides which particle's pb should be followed by this $a^{th}$ particle. It is defined in the Steps 1-4 in the Algorithm 2.

**Algorithm 2** Algorithm for ”Best” particle selection.

```
1 Generate a random number in the range [0,1]. If it is higher than ”selection probability”
  (f) then particle will learn from its own pb, otherwise it will learn from another
  particle’s pb and will follow the tournament selection procedure, shown in steps 2-4.
2 Putting the current particle aside, choose two particles randomly from the population.
3 Now compare the fitness values of these two particle’s pb and choose the particle
  With higher fitness.
4 The particle found with better fitness is chosen and its pb is used to evaluate the
  velocity of the current particle under consideration.
```

### 4.3 “Best” particles cloning and “Worst” particle destruction

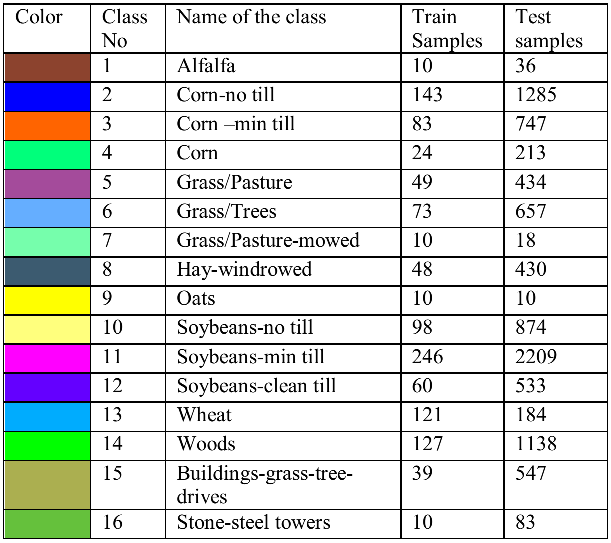
This module is employed to each one-dimensional local swarm for further improvement of this optimization strategy. As the local swarms may get be stagnated for the last few iterations because of almost insignificant improvements in the fitness values of the **gb** particle for the given swarm, then particles are needed to sort according to their **pb** values. Assume $P_{replace}$ are the number of particles identified based on their lowest fitness values, corresponding to their **pb** positions (called “Worst” particles), to be replaced. Similarly, $P_{replace}$ number of particles are also being identified in this swarm, based on their highest fitness values, corresponding to their **pb** positions (called as “Best” particles). These particles will be used to replace the worst particles identified so far. Now a superior swarm will be formed to obtain better optimization performance in the search space. At the end of this approach, it can be found that each swarm contains $P_{replace}$ number of positions where two particles are present simultaneously. So these particles are “cloned” to each other and this process is repeated for each one-dimensional swarm. We have considered 50% of the particles ($P_{replace} = P/2$) are declared as “Worst” whereas remaining 50% are designated as “Best” particles.

Our goal is to maximize the optimum k-dimensional vector $[t_0, t_1, t_2, ....t_k]$ by the proposed IPSO algorithm. But PSO is usually designed to solve minimization problem, so we solve the formulated problem of (6) by constructing fitness function as the reciprocal of it and then maximize $\psi(a_1, b_1, ...a_{k-1}, b_{k-1})$. Now the computational complexity of the fuzzy entropy based IPSO algorithm is calculated as $O(Np * 2(k-1) * G)$, where Np is the maximum populations, D denotes dimensionally which is equal to the number of threshold levels (in this case $2 * (k-1)$, 2 unknown fuzzy parameters), G is the maximum iteration. It is clearly seen the complexity of the multilevel problem has been reduced effectively in the exhaustive search space because of applying the meta-heuristic optimization algorithm.

## 5 Experimental setup

### 5.1 Datasets

Three well-known publicly available datasets namely Indian Pines, Pavia University and Salinas are used for our experiments. Indian Pines image taken over the agricultural Indian Pine test site of North-western Indiana in June 1992, by the Airborne Visible/Infrared

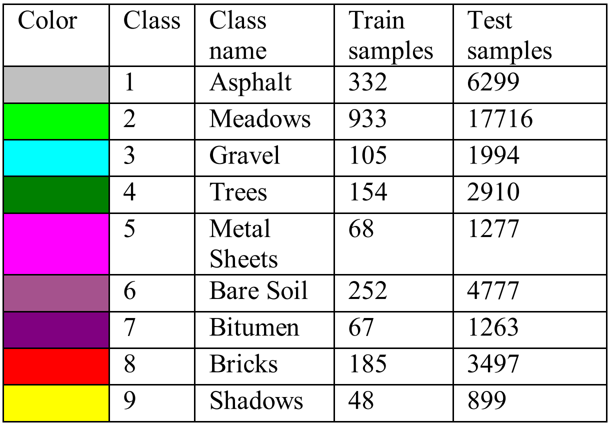


| Color | Class No | Name of the class | Train Samples | Test samples |
|---|---|---|---|---|
| 🟫 | 1 | Alfalfa | 10 | 36 |
| 🟦 | 2 | Corn-no till | 143 | 1285 |
| 🟧 | 3 | Corn –min till | 83 | 747 |
| 🟩 | 4 | Corn | 24 | 213 |
| 🟪 | 5 | Grass/Pasture | 49 | 434 |
| 🟦 | 6 | Grass/Trees | 73 | 657 |
| 🟩 | 7 | Grass/Pasture-mowed | 10 | 18 |
| ⬛ | 8 | Hay-windrowed | 48 | 430 |
| 🟨 | 9 | Oats | 10 | 10 |
| 🟨 | 10 | Soybeans-no till | 98 | 874 |
| 🟪 | 11 | Soybeans-min till | 246 | 2209 |
| 🟪 | 12 | Soybeans-clean till | 60 | 533 |
| 🟦 | 13 | Wheat | 121 | 184 |
| 🟩 | 14 | Woods | 127 | 1138 |
| 🟫 | 15 | Buildings-grass-tree-drives | 39 | 547 |
| 🟩 | 16 | Stone-steel towers | 10 | 83 |

**Fig. 4** Each class and its corresponding number of training and test samples of Indian Pines dataset

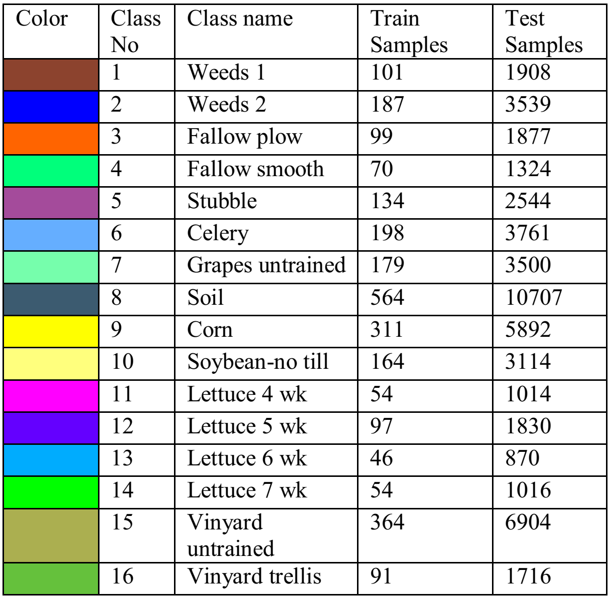
Imaging Spectrometer (AVIRIS) sensor mounted from an aircraft flown at 65,000 ft altitude. The image has 220 bands of size 145 × 145 with a spatial resolution of 20 m per pixel and a spectral coverage ranging from 0.4 to 2.5 $\mu$m. 20 water absorption bands no. 104–108, 150–163, and 220 were removed before experiments and therefore 200 out of 220 will be used for evaluation.

The University of Pavia image holding an urban area surrounding the University of Pavia was captured by the Reflective Optics System Imaging Spectrometer (ROSIS-03) optical



| Color | Class | Class name | Train samples | Test samples |
|---|---|---|---|---|
| ⬜ | 1 | Asphalt | 332 | 6299 |
| 🟩 | 2 | Meadows | 933 | 17716 |
| 🟦 | 3 | Gravel | 105 | 1994 |
| 🟩 | 4 | Trees | 154 | 2910 |
| 🟪 | 5 | Metal Sheets | 68 | 1277 |
| 🟪 | 6 | Bare Soil | 252 | 4777 |
| 🟪 | 7 | Bitumen | 67 | 1263 |
| 🟥 | 8 | Bricks | 185 | 3497 |
| 🟨 | 9 | Shadows | 48 | 899 |

**Fig. 5** Each class and its corresponding number of training and test samples of Pavia University dataset



| Color | Class No | Class name | Train Samples | Test Samples |
|---|---|---|---|---|
| 🟫 | 1 | Weeds 1 | 101 | 1908 |
| 🟦 | 2 | Weeds 2 | 187 | 3539 |
| 🟧 | 3 | Fallow plow | 99 | 1877 |
| 🟩 | 4 | Fallow smooth | 70 | 1324 |
| 🟪 | 5 | Stubble | 134 | 2544 |
| 🟦 | 6 | Celery | 198 | 3761 |
| 🟩 | 7 | Grapes untrained | 179 | 3500 |
| ⬛ | 8 | Soil | 564 | 10707 |
| 🟨 | 9 | Corn | 311 | 5892 |
| 🟨 | 10 | Soybean-no till | 164 | 3114 |
| 🟪 | 11 | Lettuce 4 wk | 54 | 1014 |
| 🟪 | 12 | Lettuce 5 wk | 97 | 1830 |
| 🟦 | 13 | Lettuce 6 wk | 46 | 870 |
| 🟩 | 14 | Lettuce 7 wk | 54 | 1016 |
| 🟫 | 15 | Vinyard untrained | 364 | 6904 |
| 🟩 | 16 | Vinyard trellis | 91 | 1716 |

**Fig. 6** Each class and its corresponding number of training and te st samples of Salinas dataset

sensor. The image contains 115 bands and spatial dimensions 610 ×340 with a spatial resolution of 1.3 m per pixel and a spectral coverage of 0.43 to 0.86 $\mu$m. 12 most noisy channels were removed and 103 bands are considered in our experiments.

The Salinas image was also captured by the AVIRIS sensor over Salinas Valley, CA, USA, and with a spatial resolution of 3.7 m per pixel and a spectral coverage of 0.4 to 2.5 $\mu$m. The image has 224 bands of size 512 × 217. Like other images 20 water absorption noise bands no. 108–112, 154–167, and 224 were discarded for better experiments. Indian Pines and Salinas images have 16 different classes whereas Pavia image carries 9 classes (Figs. 4, 5 and 6). The brief description of each dataset is given in the Table 1.

## 5.2 Parameters setup

Simulations of the proposed scheme are evaluated in MATLAB R2015a in a workstation with Intel $core^{TM}$ i3 3.2 GHz processor. Parameters setup of the proposed algorithm along with different optimization approaches like CS [1], DE [49], FF [44], GA [20], PSO [2]

**Table 1** Description of datasets

| Datasets | Size | Spatial resolution | Spectral coverage | Classes | Sensor | Bands |
|---|---|---|---|---|---|---|
| Indian Pines | 145×145×220 | 20 m | 0.4-2.5$\mu$m | 16 | AVIRIS | 200 |
| Salinas | 512×217×224 | 3.7 m | 0.4-2.5$\mu$m | 16 | AVIRIS | 204 |
| Pavia University | 610×340×115 | 1.5 m | 0.43-0.86$\mu$m | 9 | ROSIS | 103 |

**Table 2** Parameters and its values of used different algorithms in our experiment

| Algorithms | Parameters | Values |
|---|---|---|
| IPSO | D | 2∗(k-1) |
| | Population (Np) | 10 ∗ D |
| | $w_{initial}$ | 1.2 |
| | $c_x$ | 1.5 |
| | $c_y$ | 1.5 |
| | $r_1$ | [0-1] |
| | $r_2$ | [0-1] |
| | Max generation | 100 |
| | Iteration | D×1000 |
| CS | Mutation probability | 0.25 |
| | Scale factor | 1.5 |
| DE | Scaling Factor | 0.5 |
| | Max generation | 100 |
| | Iteration | D×1000 |
| FF | Randomization parameter($\alpha$) | 0.5 |
| | Light absorption coefficient at the source ($\gamma$) | 1.0 |
| | Inertial attractiveness ($\beta_0$) | 0.2 |
| GA | Crossover probability | 0.9 |
| | Mutation probability | 0.001 |
| | Encoding | Binary |
| PSO | $w_{min}, w_{max}$ | 0.4, 0.1 |
| | $C_1, C_2$ | 2 |

listed in the Table 2. All the tested algorithms run for 100 independent times where each run was carried out for the $D \times 1000$ (D denoted the dimension of search space) number of fitness evaluations. The dimension of the search space is calculated by the input segmentation level (k) which varies for multilevel problems. Segmentation levels are set from 10 to 14 to avoid under segmentation problem. Performance of the meta-heuristics in terms of quantitative measurements are tested by evaluating the best-mean fitness values $(f_{mean})$, standard deviations $(f_{std})$, and computation time (T) whereas the quality of segmentation

are evaluated by the known metrics like Peak Signal to Noise Ratio (PSNR), Structure similarity Index Measurement (SSIM).

## 6 Results and discussion

### 6.1 Unsupervised segmentation results

Computation speed, best-mean fitness value, and stability of the different optimization algorithms for segmentation level 10,12, and 14 of three specified images are compared and

**Table 3** Performance comparison of evolutionary algorithms for the Indiana Pines dataset

| L | Algo | T (in sec) | $f_{mean}$ | $f_{std}$ | PSNR | SSIM |
|---|---|---|---|---|---|---|
| 10 | IPSO | **31.206** | **36.207** | **0.612** | **20.431** | **0.935** |
| | CS | 31.442 | 35.901 | 0.734 | 18.8035 | 0.910 |
| | DE | 31.387 | 35.729 | 0.701 | 18.1220 | 0.888 |
| | FF | 31.309 | 35.577 | 0.692 | 19.591 | 0.906 |
| | GA | 31.222 | 34.932 | 0.616 | 18.533 | 0.896 |
| | PSO | 31.541 | 34.260 | 0.616 | 18.754 | 0.929 |
| 12 | IPSO | **38.822** | **40.202** | **0.585** | **22.952** | **0.943** |
| | CS | 38.840 | 39.625 | 0.641 | 19.913 | 0.918 |
| | DE | 38.845 | 39.407 | 0.637 | 20.321 | 0.907 |
| | FF | 38.837 | 38.892 | 0.642 | 21.951 | 0.927 |
| | GA | 38.835 | 38.079 | 0.607 | 22.596 | 0.910 |
| | PSO | 38.836 | 37.335 | 0.589 | 19.489 | 0.922 |
| 14 | IPSO | **40.334** | **44.619** | **0.310** | **22.377** | **0.951** |
| | CS | 40.402 | 44.427 | 0.565 | 19.732 | 0.898 |
| | DE | 40.417 | 43.627 | 0.577 | 19.305 | 0.808 |
| | FF | 40.372 | 43.554 | 0.316 | 19.738 | 0.915 |
| | GA | 40.355 | 42.932 | 0.316 | 21.169 | 0.947 |
| | PSO | 40.406 | 41.904 | 0.566 | 19.561 | 0.907 |

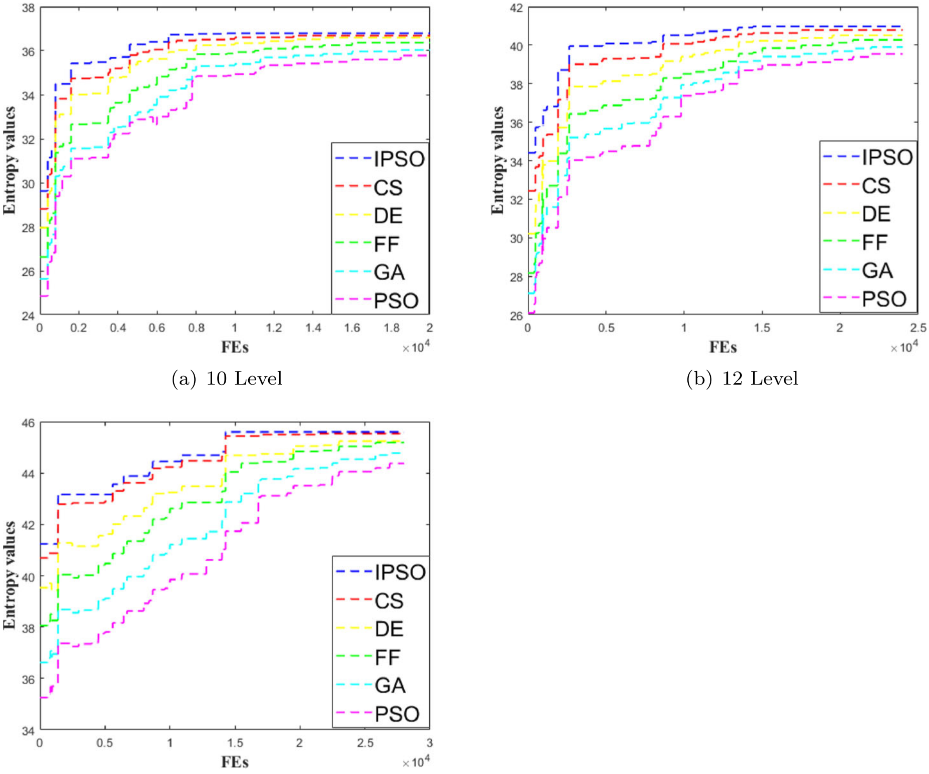
**Table 4** Performance comparison of evolutionary algorithms for the Pavia University dataset

| L | Algo | T (in sec) | $f_{mean}$ | $f_{std}$ | PSNR | SSIM |
|---|---|---|---|---|---|---|
| 10 | IPSO | **31.934** | **38.436** | **0.481** | **20.308** | **0.842** |
| | CS | 33.018 | 38.151 | 0.663 | 20.123 | 0.833 |
| | DE | 33.106 | 36.871 | 0.664 | 18.107 | 0.804 |
| | FF | 33.102 | 37.769 | 0.665 | 16.308 | 0.630 |
| | GA | 33.189 | 37.162 | 0.547 | 18.663 | 0.768 |
| | PSO | 33.221 | 36.392 | 0.489 | 17.440 | 0.697 |
| 12 | IPSO | **38.275** | **42.969** | **0.682** | **20.903** | **0.859** |
| | CS | 42.948 | 42.536 | 0.700 | 17.965 | 0.849 |
| | DE | 42.966 | 41.806 | 0.709 | 18.972 | 0.832 |
| | FF | 42.987 | 41.694 | 0.631 | 20.816 | 0.826 |
| | GA | 42.806 | 40.761 | 0.690 | 17.308 | 0.730 |
| | PSO | 42.584 | 40.135 | 0.690 | 17.875 | 0.729 |
| 14 | IPSO | **44.880** | **47.009** | **0.424** | **20.352** | **0.880** |
| | CS | 49.786 | 46.096 | 0.489 | 19.645 | 0.805 |
| | DE | 49.815 | 45.446 | 0.487 | 19.684 | 0.795 |
| | FF | 49.880 | 45.193 | 0.491 | 18.541 | 0.774 |
| | GA | 49.752 | 44.551 | 0.560 | 20.216 | 0.826 |
| | PSO | 49.805 | 43.484 | 0.637 | 20.101 | 0.832 |

**Table 5** Performance comparison of evolutionary algorithms for the Salinas dataset

| L | Algo | T (in sec) | $f_{mean}$ | $f_{std}$ | PSNR | SSIM |
|---|---|---|---|---|---|---|
| 10 | IPSO | **31.879** | **31.384** | **0.734** | **22.496** | **0.929** |
| | CS | 32.320 | 30.821 | 0.836 | 21.087 | 0.861 |
| | DE | 32.385 | 30.710 | 0.846 | 21.087 | 0.861 |
| | FF | 32.410 | 30.523 | 0.848 | 20.889 | 0.859 |
| | GA | 32.219 | 29.991 | 0.793 | 21.005 | 0.865 |
| | PSO | 32.322 | 29.368 | 0.748 | 21.894 | 0.885 |
| 12 | IPSO | **36.855** | **34.377** | **0.658** | **24.202** | **0.938** |
| | CS | 37.387 | 34.078 | 0.748 | 23.753 | 0.904 |
| | DE | 37.398 | 33.556 | 0.747 | 23.865 | 0.900 |
| | FF | 37.421 | 33.497 | 0.728 | 23.952 | 0.912 |
| | GA | 37.310 | 32.818 | 0.665 | 20.889 | 0.859 |
| | PSO | 37.372 | 32.354 | 0.674 | 22.339 | 0.874 |
| 14 | IPSO | **43.402** | **38.422** | **0.751** | **25.944** | **0.949** |
| | CS | 45.144 | 37.689 | 0.761 | 22.795 | 0.902 |
| | DE | 45.162 | 37.578 | 0.760 | 22.501 | 0.867 |
| | FF | 45.237 | 36.964 | 0.825 | 23.154 | 0.928 |
| | GA | 45.132 | 36.451 | 0.774 | 23.952 | 0.912 |
| | PSO | 45.393 | 35.586 | 0.765 | 22.987 | 0.889 |

values are noted in the Tables 3, 4, and 5. The best values are marked as bold in these tables and results infer that the proposed algorithm produces best values for all the datasets in terms of time, fitness values and standard deviations. It is also to be noted that the computational complexity of IPSO is lesser compared to others even increasing levels (k). As the multilevel problem is treated as a maximization problem, so best mean fitness values suggest better working of the algorithm along with less standard deviations indicate robustness and stability of the algorithm. Further, convergence plots are generated for different datasets for 10, 12 and 14 levels. IPSO algorithm for Indian Pines image has converged at 11000 iterations whereas CS, DE, FF, GA, and PSO have started to converge at 13000, 14100, 14500, 16000, and 18500 iterations for level 10 segmentation, IPSO converged at 14000 and 15500 for level 12 and 14 segmentations where others not only struggled to converge fast but also produce less fitness values, hence trapped in premature convergence problem shown in the Fig. 7. Similarly, the fast convergence rate can be noticed for the Pavia and Salinas datasets in the Figs. 8 and 9. These figures state that proposed IPSO along with CS can only overcome the challenge to be get trapped in local optima problem where remaining can't. The segmented 10, 12, and 14 level images obtained by the various optimization algorithms along with false-color images of Indian Pines, Pavia, and Salinas are listed in the Figures 10, 11, and 12 respectively. Quality of the reconstructed images for different levels are judged by the PSNR and SSIM values mentioned at the right corner of the Tables 3, 4, and 5. PSNR and SSIM values for color images are calculated by the given following equation:



(a) 10 Level

(b) 12 Level

(c) 14 Level

**Fig. 7** Level wise comparison of convergence plots between evolutionary algorithms for “Indian Pines” dataset

Peak Signal to noise ratio:

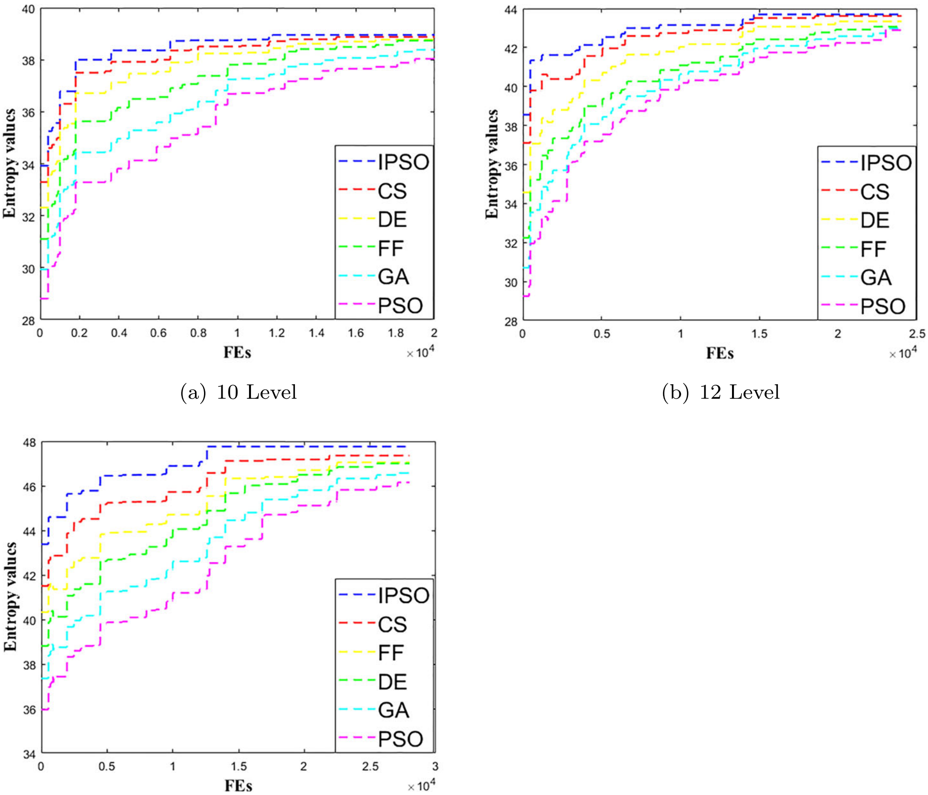
$$PSNR = 10 \times log_{10}\left(\frac{255^2}{RMSE}\right)(dB)$$

$$where\ RMSE = \frac{1}{R \times C}\sum_{i}^{R}\sum_{j}^{C}\{I(i,j) - I'(i,j)\}^2 \tag{13}$$

where $I$ and $I'$ are the original and segmented images and $R \times C$ is the size of the image and RMSE stands for root mean square error. Structural Similarity Index:

$$SSIM(x,y) = \frac{(2\mu_I\mu_{I'} + C_1)(2\sigma_{II'} + C_2)}{(\mu_I^2 + \mu_{I'}^2 + C_1)(\sigma_I^2 + \sigma_{I'}^2 + C_2)}$$

$$for\ an\ RGB\ image,\ SSIM = \sum_{c} SSIM(x^c, y^c) \tag{14}$$



(a) 10 Level

(b) 12 Level

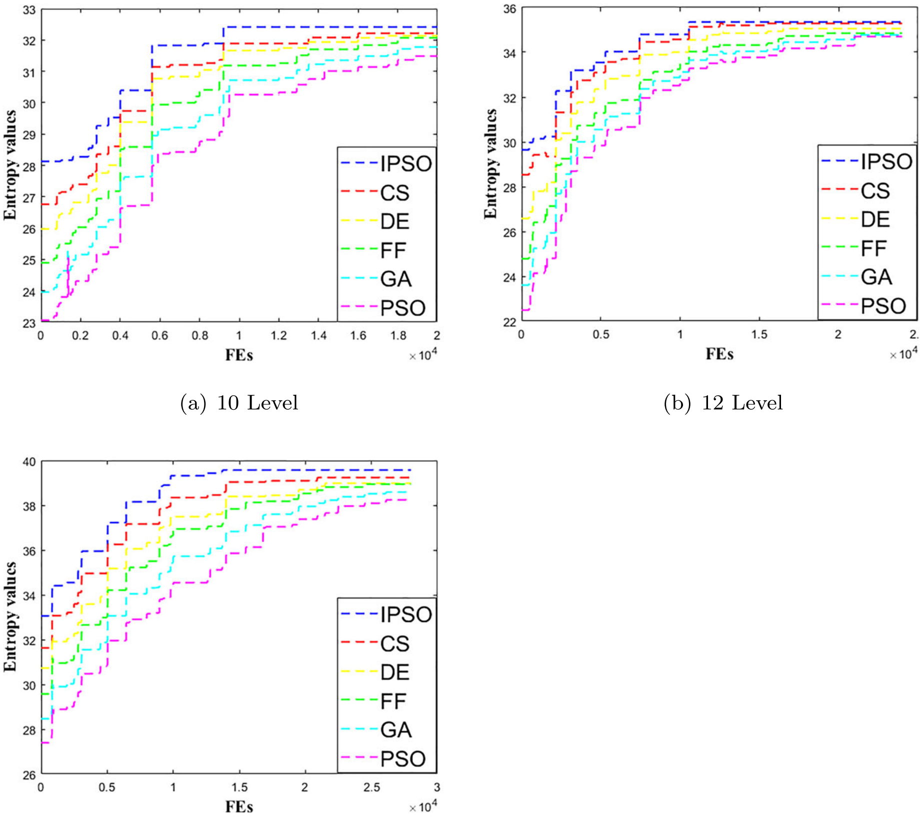
(c) 14 Level

**Fig. 8** Level wise comparison of convergence plots between evolutionary algorithms for “Pavia University” dataset

where $\mu_I$ and $\mu_{I'}$ are the mean values of the original image I and segmented image I’, $\sigma_I$ and $\sigma_{I'}$ are the standard deviation of the original image I and segmented image I’, $\sigma_{II'}$ is the cross correlation and C1 and C2 are constants, c indicates color.

## 6.2 Hybrid classification results

In our experiment, 10% data of Indian pines are chosen as training samples and remaining 90% from ground-truth pixels are used as test samples for supervised classification. 16 classes and its corresponding training and test samples of Indian Pines data are shown in the Fig. 4. For both Pavia and Salinas data, 5% ground-truth pixels are chosen as training data and remaining 95% work as a test data shown in the Figs. 5 and 6. Now, level-wise segmentation results are mapped with the EMAP data to form a multiple learning kernel which is used to train SVM classifier. The following four accuracy metrics are considered for checking the classification accuracy:



(a) 10 Level

(b) 12 Level

(c) 14 Level

**Fig. 9** Level wise comparison of convergence plots between evolutionary algorithms for "Salinas" dataset

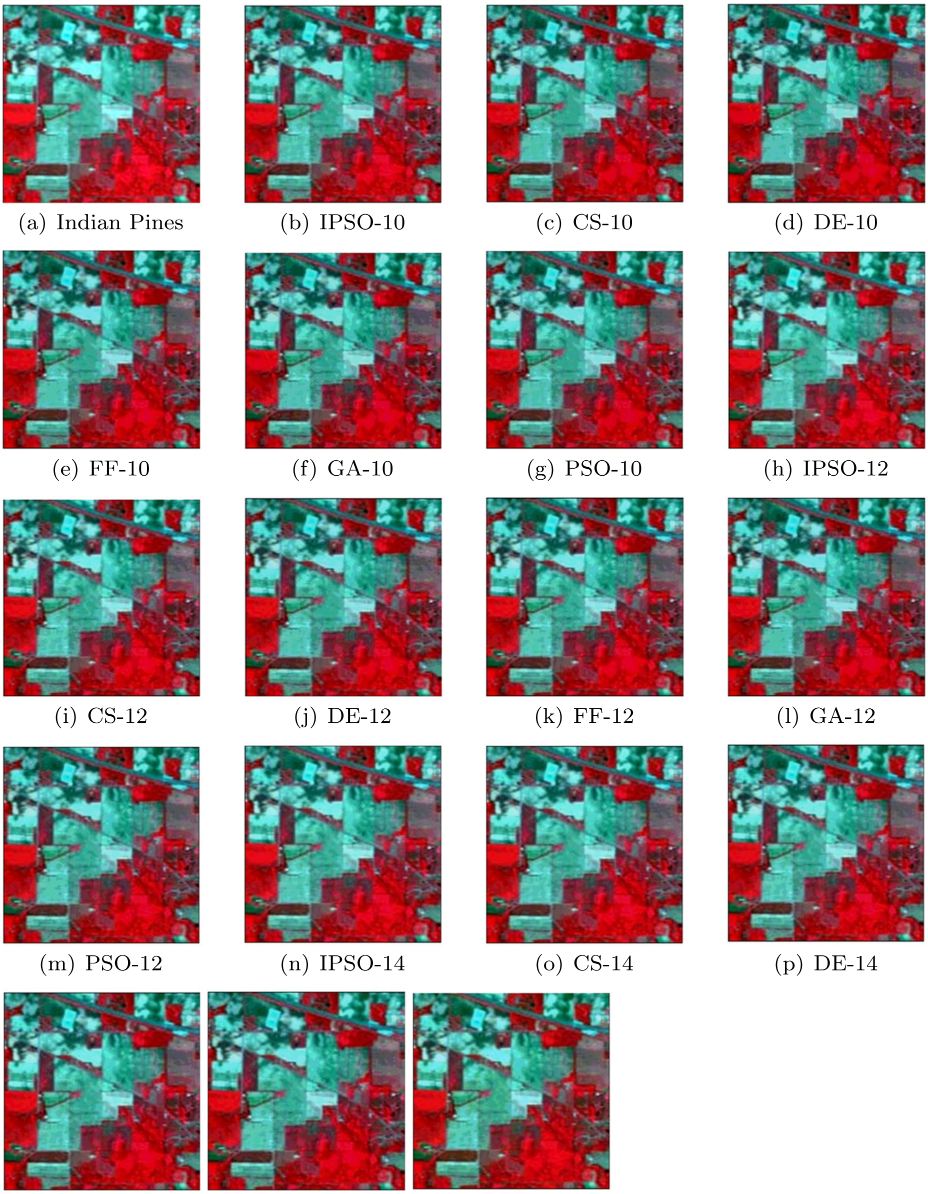
**Overall accuracy (OA)** OA calculates out of all the classes how much proportion of classes are mapped properly. It is measured in percentage and provides easiest accuracy information to users. It is calculated like following:

$$\frac{No.\,of\ correctly\ classified\ classes}{Total\ no.\,of\ classes} \times 100 \tag{15}$$

**Kappa index (KI)** KI is measured by a statistical test to evaluate the accuracy. It is performed based on assigning some random values and checked how classification works better than random. Outcomes lie between -1 to +1 which indicate accurate classification. Matlab function can evaluate KI effectively.

**Mean accuracy (MA)** This accuracy calculates the ratio of correctly classified pixels to the total number of pixels in each class. MA score is measured with the help of ground truth values as per the following equation:

$$MA = \frac{TP}{TP + FN} \tag{16}$$



(a) Indian Pines (b) IPSO-10 (c) CS-10 (d) DE-10

(e) FF-10 (f) GA-10 (g) PSO-10 (h) IPSO-12

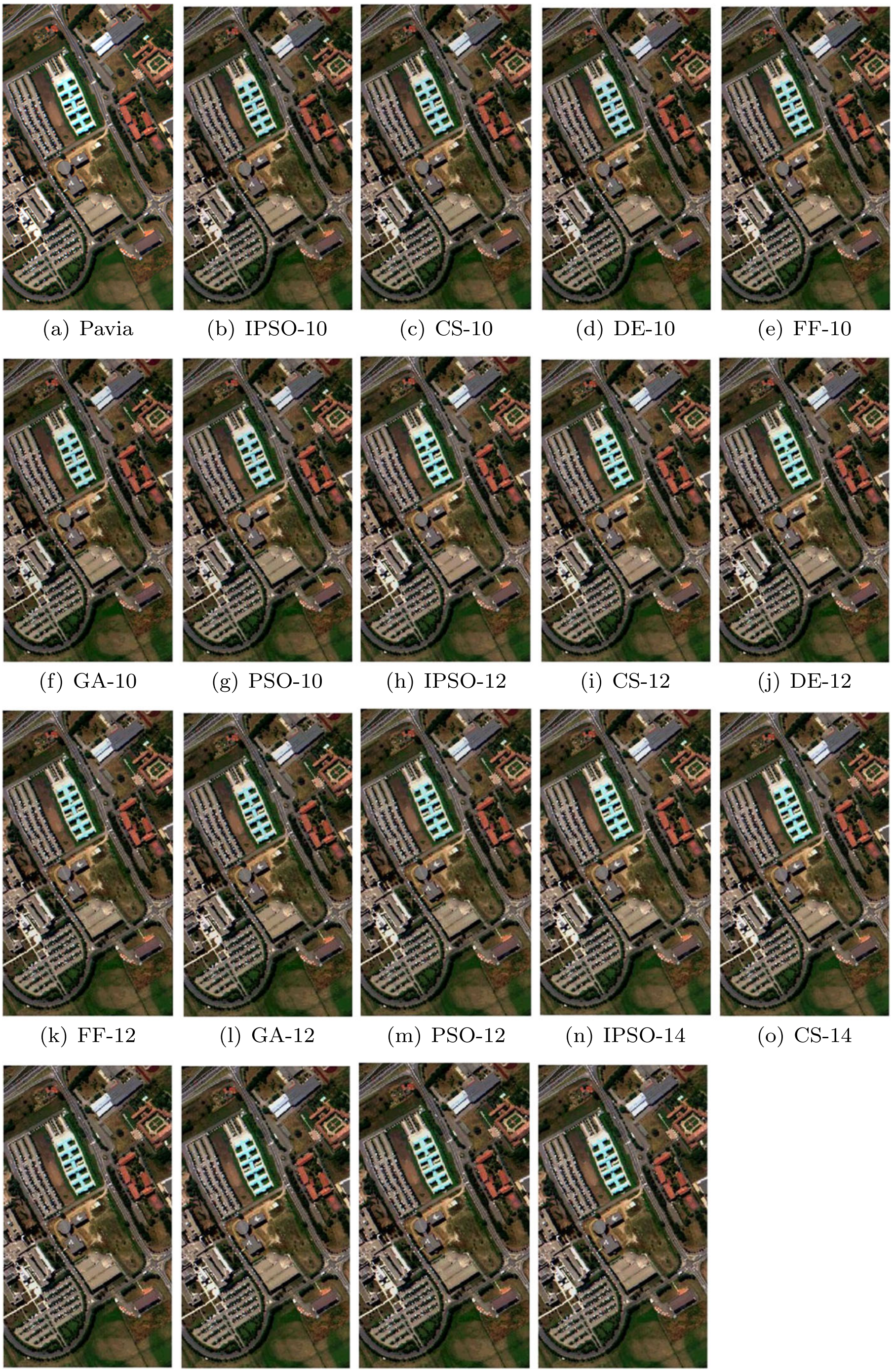
(i) CS-12 (j) DE-12 (k) FF-12 (l) GA-12

(m) PSO-12 (n) IPSO-14 (o) CS-14 (p) DE-14

(q) FF-14 (r) GA-14 (s) PSO-14

**Fig. 10** Indian Pines image with 10,12,14 level segmented images by evolutionary algorithms using fuzzy entropy

where TP is the number of true positives and FN is the number of false negatives in the classes. MA is the average Accuracy of all classes in that particular image.



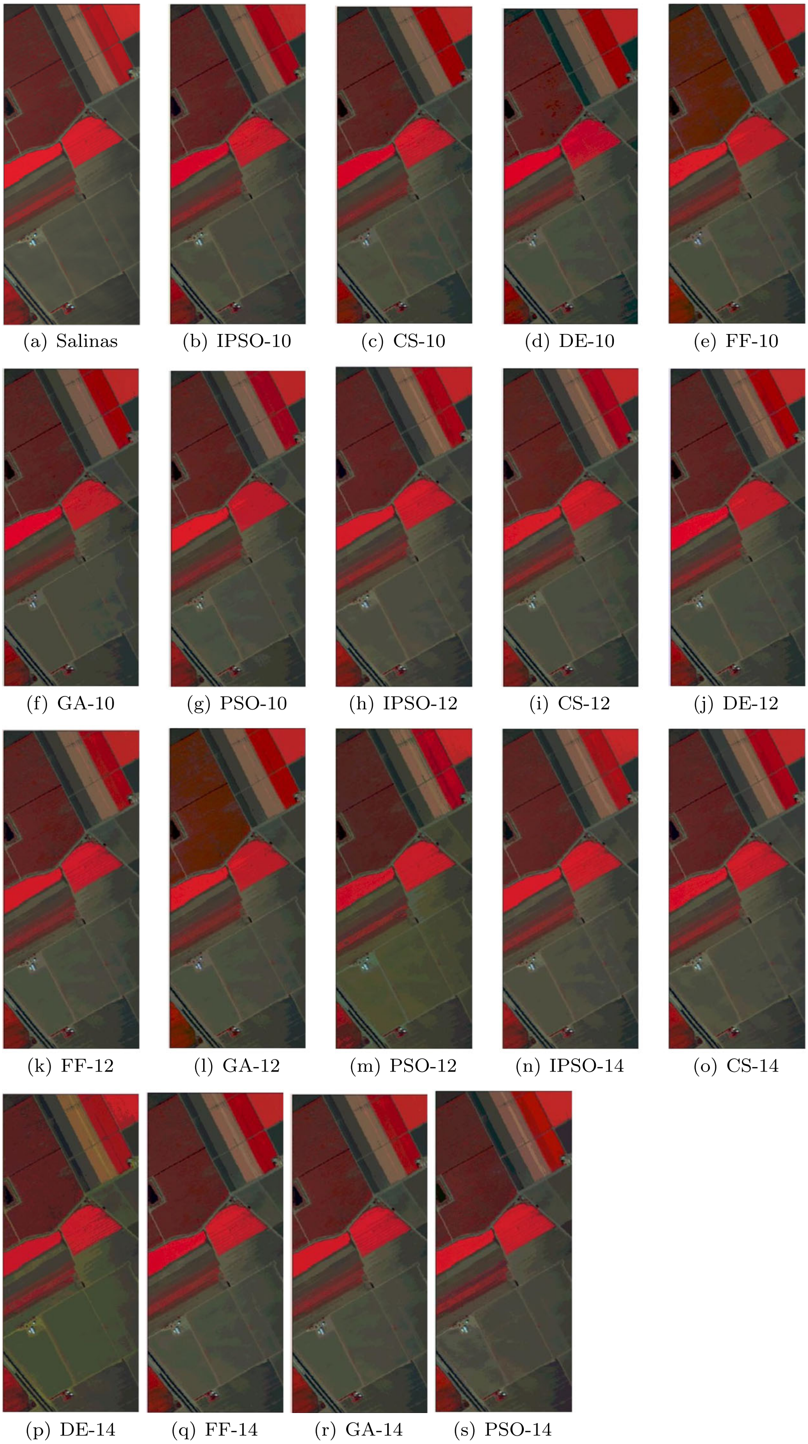
(a) Pavia (b) IPSO-10 (c) CS-10 (d) DE-10 (e) FF-10

(f) GA-10 (g) PSO-10 (h) IPSO-12 (i) CS-12 (j) DE-12

(k) FF-12 (l) GA-12 (m) PSO-12 (n) IPSO-14 (o) CS-14

(p) DE-14 (q) FF-14 (r) GA-14 (s) PSO-14

**Fig. 11** Pavia University image with 10,12,14 level segmented images by evolutionary algorithms using fuzzy entropy



(a) Salinas (b) IPSO-10 (c) CS-10 (d) DE-10 (e) FF-10

(f) GA-10 (g) PSO-10 (h) IPSO-12 (i) CS-12 (j) DE-12

(k) FF-12 (l) GA-12 (m) PSO-12 (n) IPSO-14 (o) CS-14

(p) DE-14 (q) FF-14 (r) GA-14 (s) PSO-14

**Fig. 12** Salinas image with 10,12,14 level segmented images by evolutionary algorithms using fuzzy entropy

**Table 6** Performance comparison between SVM and SVM with meta-heuristic algorithms (SVMCK) over Indiana Pines dataset

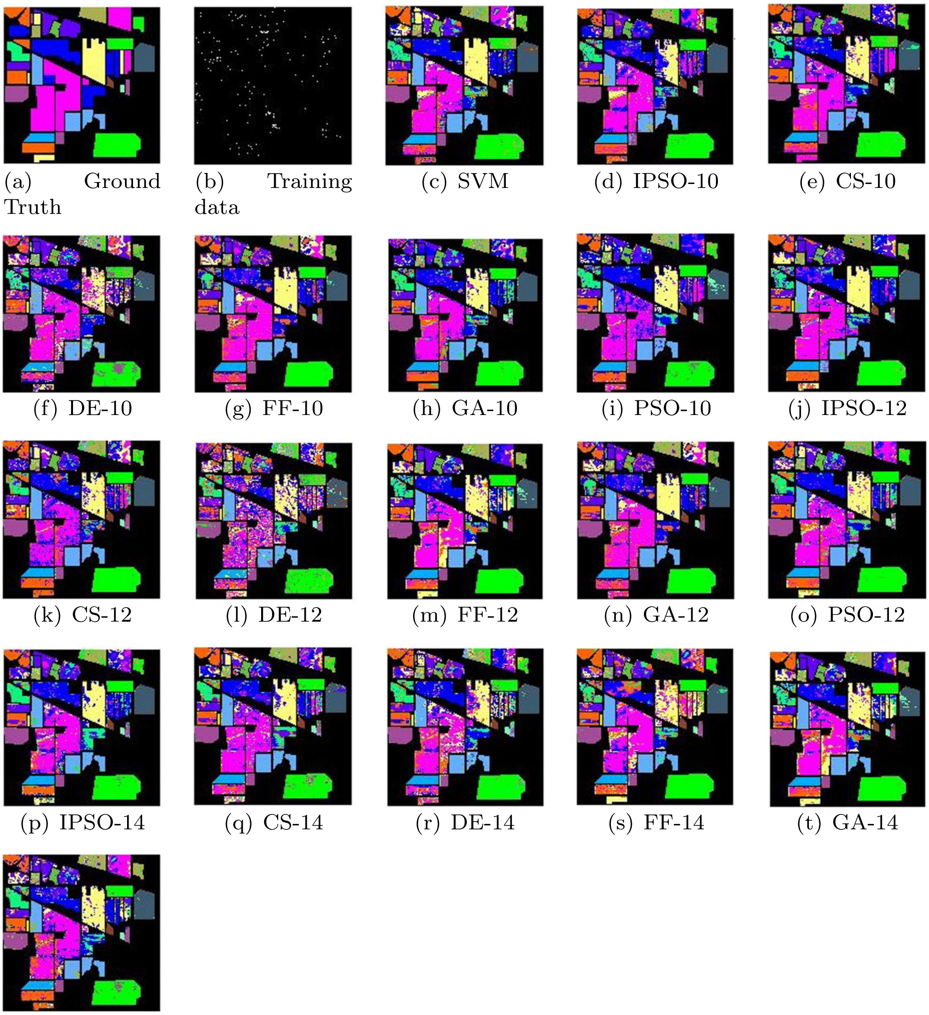
| Class | SVM | 10 Level | | | | | | 12 Level | | | | | | 14 Level | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | IPSO | CS | DE | FF | GA | PSO | IPSO | CS | DE | FF | GA | PSO | IPSO | CS | DE | FF | GA | PSO |
| 1 | 0.681 | 0.977 | 0.840 | 0.772 | 0.909 | 0.954 | 0.954 | 0.931 | 0.886 | 0.909 | 0.818 | 0.954 | 0.840 | 0.977 | 0.863 | 0.886 | 0.818 | 0.886 | 0.954 |
| 2 | 0.390 | 0.650 | 0.536 | 0.478 | 0.573 | 0.516 | 0.608 | 0.573 | 0.521 | 0.398 | 0.487 | 0.543 | 0.526 | 0.531 | 0.640 | 0.511 | 0.542 | 0.527 | 0.415 |
| 3 | 0.640 | 0.657 | 0.759 | 0.474 | 0.540 | 0.508 | 0.577 | 0.609 | 0.509 | 0.423 | 0.697 | 0.771 | 0.706 | 0.690 | 0.438 | 0.503 | 0.585 | 0.525 | 0.530 |
| 4 | 0.906 | 0.598 | 0.758 | 0.571 | 0.696 | 0.705 | 0.897 | 0.870 | 0.924 | 0.638 | 0.584 | 0.750 | 0.848 | 0.830 | 0.843 | 0.638 | 0.678 | 0.665 | 0.897 |
| 5 | 0.868 | 0.895 | 0.835 | 0.870 | 0.889 | 0.839 | 0.509 | 0.876 | 0.887 | 0.809 | 0.843 | 0.878 | 0.901 | 0.907 | 0.905 | 0.848 | 0.903 | 0.702 | 0.835 |
| 6 | 0.899 | 0.934 | 0.956 | 0.765 | 0.970 | 0.884 | 0.921 | 0.911 | 0.884 | 0.883 | 0.940 | 0.884 | 0.864 | 0.967 | 0.913 | 0.903 | 0.926 | 0.860 | 0.868 |
| 7 | 0.937 | 0.937 | 0.875 | 0.937 | 1.000 | 0.937 | 1.000 | 0.937 | 0.875 | 1.000 | 1.000 | 1.000 | 0.937 | 0.875 | 0.875 | 0.812 | 0.937 | 0.875 | 1.000 |
| 8 | 0.787 | 0.974 | 0.964 | 0.906 | 0.977 | 0.966 | 0.949 | 0.979 | 0.983 | 0.714 | 0.889 | 0.910 | 0.926 | 0.972 | 0.983 | 0.768 | 0.966 | 0.920 | 0.977 |
| 9 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 |
| 10 | 0.495 | 0.724 | 0.659 | 0.530 | 0.698 | 0.733 | 0.723 | 0.662 | 0.737 | 0.652 | 0.776 | 0.830 | 0.767 | 0.792 | 0.771 | 0.610 | 0.685 | 0.852 | 0.716 |
| 11 | 0.648 | 0.560 | 0.732 | 0.618 | 0.552 | 0.543 | 0.657 | 0.602 | 0.439 | 0.367 | 0.655 | 0.655 | 0.657 | 0.586 | 0.517 | 0.460 | 0.431 | 0.662 | 0.643 |
| 12 | 0.597 | 0.655 | 0.601 | 0.571 | 0.677 | 0.807 | 0.417 | 0.571 | 0.663 | 0.329 | 0.7003 | 0.490 | 0.634 | 0.759 | 0.645 | 0.543 | 0.759 | 0.660 | 0.786 |
| 13 | 0.990 | 0.995 | 0.995 | 0.920 | 0.980 | 0.990 | 0.995 | 0.990 | 0.990 | 0.901 | 0.995 | 0.965 | 0.985 | 0.990 | 0.990 | 0.980 | 0.970 | 0.990 | 0.995 |
| 14 | 0.955 | 0.890 | 0.915 | 0.736 | 0.907 | 0.978 | 0.750 | 0.769 | 0.927 | 0.806 | 0.873 | 0.940 | 0.855 | 0.779 | 0.925 | 0.728 | 0.846 | 0.919 | 0.968 |
| 15 | 0.700 | 0.770 | 0.783 | 0.529 | 0.837 | 0.575 | 0.673 | 0.764 | 0.781 | 0.362 | 0.900 | 0.664 | 0.773 | 0.781 | 0.745 | 0.381 | 0.627 | 0.729 | 0.645 |
| 16 | 0.964 | 0.988 | 0.882 | 0.870 | 0.847 | 0.976 | 0.941 | 0.941 | 0.858 | 0.917 | 0.929 | 0.929 | 0.988 | 0.917 | 0.976 | 0.858 | 1.000 | 0.964 | 0.952 |
| OA | 0.686 | **0.757** | 0.723 | 0.632 | 0.713 | 0.703 | 0.690 | **0.743** | 0.678 | 0.555 | 0.735 | 0.701 | 0.735 | **0.727** | 0.711 | 0.606 | 0.669 | 0.724 | 0.717 |
| KI | 0.645 | **0.726** | 0.694 | 0.586 | 0.677 | 0.666 | 0.651 | 0.665 | 0.641 | 0.502 | 0.701 | 0.711 | 0.703 | **0.696** | 0.677 | 0.559 | 0.630 | 0.689 | 0.681 |
| MA | 0.717 | **0.771** | 0.722 | 0.636 | 0.716 | 0.741 | 0.675 | **0.791** | 0.784 | 0.654 | 0.772 | 0.716 | 0.769 | **0.821** | 0.815 | 0.706 | 0.814 | 0.772 | 0.722 |
| IoU | 0.560 | **0.626** | 0.585 | 0.488 | 0.561 | 0.563 | 0.540 | **0.659** | 0.643 | 0.443 | 0.566 | 0.561 | 0.590 | **0.671** | 0.649 | 0.505 | 0.600 | 0.566 | 0.569 |

**Table 7** Performance comparison between SVM and SVM with meta-heuristic algorithms (SVMCK) over Pavia University dataset

| Class | SVM | 10 Level | | | | | | 12 Level | | | | | | 14 Level | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | IPSO | CS | DE | FF | GA | PSO | IPSO | CS | DE | FF | GA | PSO | IPSO | CS | DE | FF | GA | PSO |
| 1 | 0.735 | 0.704 | 0.729 | 0.791 | 0.781 | 0.838 | 0.738 | 0.793 | 0.791 | 0.709 | 0.751 | 0.492 | 0.760 | 0.753 | 0.728 | 0.754 | 0.727 | 0.809 | 0.763 |
| 2 | 0.878 | 0.909 | 0.926 | 0.685 | 0.856 | 0.879 | 0.874 | 0.883 | 0.915 | 0.816 | 0.864 | 0.908 | 0.884 | 0.876 | 0.942 | 0.851 | 0.895 | 0.759 | 0.921 |
| 3 | 0.715 | 0.659 | 0.509 | 0.472 | 0.759 | 0.783 | 0.707 | 0.602 | 0.712 | 0.832 | 0.726 | 0.127 | 0.629 | 0.660 | 0.580 | 0.735 | 0.755 | 0.683 | 0.393 |
| 4 | 0.766 | 0.892 | 0.938 | 0.735 | 0.860 | 0.857 | 0.808 | 0.888 | 0.854 | 0.861 | 0.831 | 0.932 | 0.832 | 0.839 | 0.835 | 0.820 | 0.890 | 0.807 | 0.928 |
| 5 | 0.991 | 0.991 | 0.990 | 0.989 | 0.994 | 0.989 | 0.990 | 0.992 | 0.991 | 0.991 | 0.992 | 0.991 | 0.990 | 0.993 | 0.991 | 0.989 | 0.992 | 0.994 | 0.984 |
| 6 | 0.772 | 0.686 | 0.672 | 0.788 | 0.735 | 0.642 | 0.719 | 0.809 | 0.620 | 0.768 | 0.686 | 0.170 | 0.721 | 0.655 | 0.510 | 0.589 | 0.742 | 0.746 | 0.419 |
| 7 | 0.851 | 0.886 | 0.855 | 0.885 | 0.839 | 0.774 | 0.824 | 0.888 | 0.729 | 0.842 | 0.895 | 0.899 | 0.799 | 0.924 | 0.794 | 0.889 | 0.920 | 0.767 | 0.777 |
| 8 | 0.657 | 0.805 | 0.718 | 0.675 | 0.674 | 0.746 | 0.789 | 0.666 | 0.814 | 0.621 | 0.595 | 0.925 | 0.826 | 0.730 | 0.792 | 0.720 | 0.839 | 0.714 | 0.855 |
| 9 | 1.000 | 1.000 | 1.000 | 0.996 | 1.000 | 0.997 | 0.998 | 1.000 | 1.000 | 0.998 | 0.998 | 0.998 | 1.000 | 1.000 | 0.998 | 0.996 | 1.000 | 0.997 | 0.998 |
| OA | 0.814 | **0.832** | 0.829 | 0.728 | 0.817 | 0.829 | 0.820 | **0.837** | 0.814 | 0.791 | 0.801 | 0.726 | 0.827 | **0.845** | 0.818 | 0.794 | 0.845 | 0.774 | 0.806 |
| KI | 0.755 | **0.779** | 0.774 | 0.656 | 0.761 | 0.776 | 0.763 | **0.784** | 0.783 | 0.730 | 0.740 | 0.632 | 0.772 | 0.753 | 0.755 | 0.730 | 0.797 | 0.709 | 0.739 |
| MA | 0.714 | 0.771 | 0.723 | 0.846 | 0.716 | 0.785 | 0.822 | **0.898** | 0.827 | 0.892 | 0.808 | 0.710 | 0.829 | **0.824** | 0.799 | 0.790 | 0.771 | 0.808 | 0.811 |
| IoU | 0.520 | **0.661** | 0.531 | 0.709 | 0.534 | 0.583 | 0.660 | **0.798** | 0.691 | 0.805 | 0.657 | 0.534 | 0.634 | **0.726** | 0.638 | 0.626 | 0.642 | 0.657 | 0.706 |

**Table 8** Performance comparison between SVM and SVM with meta-heuristic algorithms (SVMCK) over Salinas dataset

| Class | SVM | 10 Level | | | | | | 12 Level | | | | | | **14 Level** | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | IPSO | CS | DE | FF | GA | PSO | IPSO | CS | DE | FF | GA | PSO | IPSO | CS | DE | FF | GA | PSO |
| 1 | 0.991 | 0.977 | 0.986 | 0.957 | 0.965 | 0.982 | 0.993 | 0.979 | 0.983 | 0.989 | 0.994 | 0.988 | 0.978 | 0.977 | 0.993 | 0.986 | 0.985 | 0.993 | 0.993 |
| 2 | 0.968 | 0.993 | 0.986 | 0.987 | 0.945 | 0.940 | 0.901 | 0.984 | 0.990 | 0.990 | 0.975 | 0.990 | 0.997 | 0.991 | 0.865 | 0.877 | 0.991 | 0.912 | 0.972 |
| 3 | 0.612 | 0.774 | 0.884 | 0.760 | 0.775 | 0.925 | 0.674 | 0.958 | 0.653 | 0.956 | 0.890 | 0.798 | 0.874 | 0.934 | 0.569 | 0.908 | 0.653 | 0.759 | 0.565 |
| 4 | 0.978 | 0.994 | 0.994 | 0.990 | 0.984 | 0.976 | 0.979 | 0.980 | 0.975 | 0.995 | 0.983 | 0.991 | 0.997 | 0.984 | 0.961 | 0.987 | 0.997 | 0.981 | 0.981 |
| 5 | 0.937 | 0.979 | 0.946 | 0.966 | 0.947 | 0.944 | 0.886 | 0.956 | 0.975 | 0.954 | 0.959 | 0.983 | 0.914 | 0.957 | 0.943 | 0.937 | 0.966 | 0.950 | 0.898 |
| 6 | 0.997 | 0.998 | 0.996 | 0.998 | 0.998 | 0.998 | 0.974 | 0.994 | 0.998 | 0.971 | 0.997 | 0.988 | 0.976 | 0.995 | 0.995 | 0.999 | 0.990 | 0.997 | 0.995 |
| 7 | 0.994 | 0.994 | 0.994 | 0.994 | 0.994 | 0.982 | 0.994 | 0.985 | 0.992 | 0.994 | 0.982 | 0.994 | 0.994 | 0.993 | 0.994 | 0.994 | 0.994 | 0.994 | 0.993 |
| 8 | 0.363 | 0.664 | 0.291 | 0.548 | 0.523 | 0.977 | 0.164 | 0.573 | 0.502 | 0.541 | 0.538 | 0.532 | 0.605 | 0.350 | 0.670 | 0.588 | 0.395 | 0.452 | 0.055 |
| 9 | 0.988 | 0.980 | 0.995 | 0.980 | 0.975 | 0.986 | 0.987 | 0.963 | 0.962 | 0.980 | 0.925 | 0.980 | 0.970 | 0.987 | 0.967 | 0.988 | 0.978 | 0.988 | 0.989 |
| 10 | 0.815 | 0.756 | 0.793 | 0.822 | 0.668 | 0.822 | 0.614 | 0.716 | 0.633 | 0.817 | 0.795 | 0.864 | 0.680 | 0.830 | 0.522 | 0.837 | 0.815 | 0.851 | 0.865 |
| 11 | 0.884 | 0.870 | 0.869 | 0.940 | 0.895 | 0.880 | 0.888 | 0.902 | 0.833 | 0.878 | 0.930 | 0.987 | 0.966 | 0.916 | 0.889 | 0.889 | 0.862 | 0.915 | 0.905 |
| 12 | 1.000 | 0.995 | 0.994 | 0.923 | 0.909 | 1.000 | 0.992 | 0.972 | 0.982 | 0.989 | 0.993 | 0.885 | 0.924 | 0.983 | 0.999 | 0.993 | 0.997 | 0.994 | 0.997 |
| 13 | 0.984 | 0.990 | 0.973 | 0.980 | 0.989 | 0.990 | 0.990 | 0.977 | 0.985 | 0.993 | 0.989 | 0.986 | 0.979 | 0.981 | 0.990 | 0.982 | 0.984 | 0.981 | 0.986 |
| 14 | 0.887 | 0.868 | 0.929 | 0.903 | 0.842 | 0.879 | 0.838 | 0.920 | 0.876 | 0.855 | 0.900 | 0.899 | 0.893 | 0.902 | 0.848 | 0.900 | 0.875 | 0.896 | 0.889 |
| 15 | 0.689 | 0.503 | 0.815 | 0.641 | 0.699 | 0.914 | 0.816 | 0.648 | 0.732 | 0.584 | 0.766 | 0.613 | 0.653 | 0.704 | 0.517 | 0.603 | 0.804 | 0.650 | 0.931 |
| 16 | 0.830 | 0.976 | 0.948 | 0.928 | 0.938 | 0.901 | 0.863 | 0.976 | 0.924 | 0.927 | 0.941 | 0.975 | 0.947 | 0.912 | 0.853 | 0.779 | 0.895 | 0.895 | 0.918 |
| OA | 0.781 | **0.828** | 0.798 | 0.825 | 0.809 | 0.767 | 0.738 | 0.828 | 0.808 | 0.819 | 0.835 | 0.822 | 0.808 | **0.821** | 0.794 | 0.798 | 0.808 | 0.802 | 0.753 |
| KI | 0.758 | **0.809** | 0.778 | 0.803 | 0.788 | 0.745 | 0.712 | **0.818** | 0.787 | 0.799 | 0.810 | 0.802 | 0.809 | 0.778 | 0.801 | 0.770 | 0.778 | 0.780 | 0.729 |
| MA | **0.905** | 0.878 | 0.857 | 0.830 | 0.894 | 0.867 | 0.836 | **0.894** | 0.779 | 0.826 | 0.816 | 0.825 | 0.801 | **0.871** | 0.868 | 0.840 | 0.852 | 0.816 | 0.825 |
| IoU | 0.734 | **0.740** | 0.698 | 0.675 | 0.661 | 0.735 | 0.703 | 0.716 | 0.712 | 0.670 | 0.656 | 0.661 | 0.740 | **0.744** | 0.718 | 0.702 | 0.690 | 0.656 | 0.670 |



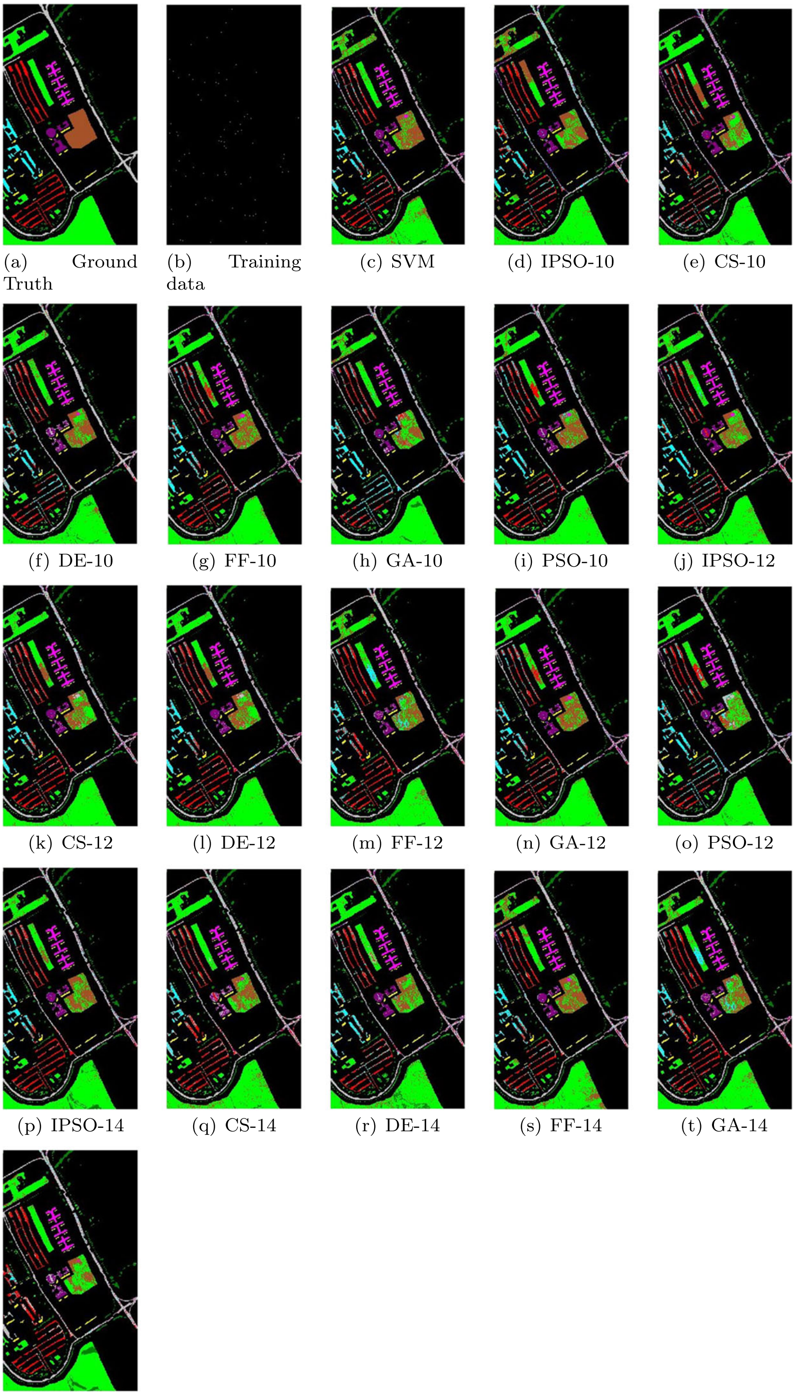
(a) Ground Truth
(b) Training data
(c) SVM
(d) IPSO-10
(e) CS-10
(f) DE-10
(g) FF-10
(h) GA-10
(i) PSO-10
(j) IPSO-12
(k) CS-12
(l) DE-12
(m) FF-12
(n) GA-12
(o) PSO-12
(p) IPSO-14
(q) CS-14
(r) DE-14
(s) FF-14
(t) GA-14
(u) PSO-14

**Fig. 13** Ground truth image, Training data, SVM classified output, SVM with 10,12 and 14 level classified output (SVMCK) using chosen meta-heuristic algorithms of Indian Pines dataset

**Intersection over union (IoU)** IoU also known as the Jaccard similarity coefficient and is often used in statistical accuracy measurement. Like MA, IoU is also computed with the ground truth images as per the below equation:

$$\frac{TP}{TP + FP + FN} \tag{17}$$

where TP, FP, and FN describes true positive, false positive, and false negative predictions of images respectively. Mean IoU is the average IoU score of all classes in the particular image.



(a) Ground Truth

(b) Training data

(c) SVM

(d) IPSO-10

(e) CS-10

(f) DE-10

(g) FF-10

(h) GA-10

(i) PSO-10

(j) IPSO-12

(k) CS-12

(l) DE-12

(m) FF-12

(n) GA-12

(o) PSO-12

(p) IPSO-14

(q) CS-14

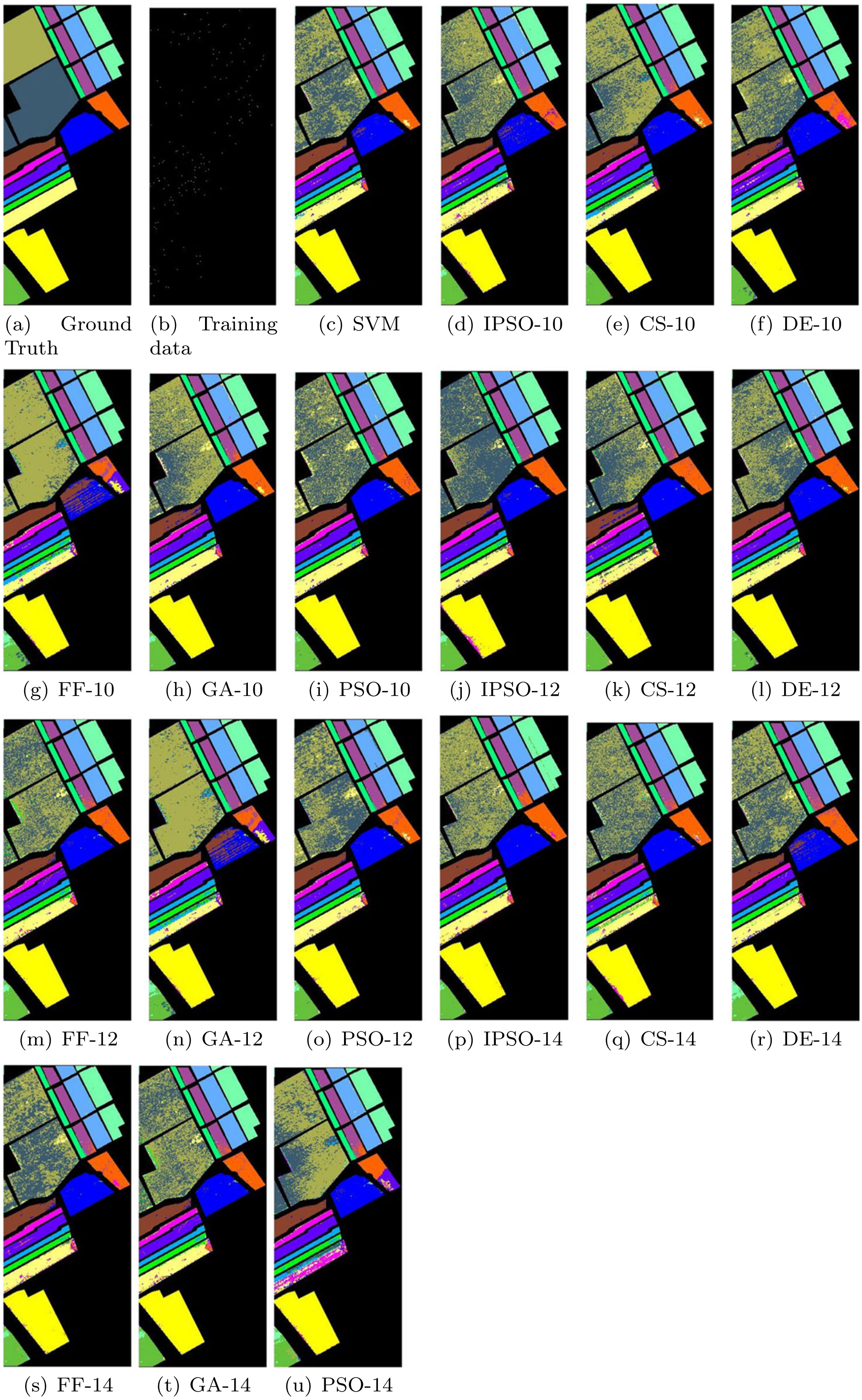
(r) DE-14

(s) FF-14

(t) GA-14

(u) PSO-14

**Fig. 14** Ground truth image, Training data, SVM classified output, SVM with 10,12 and 14 level classified output (SVMCK) using chosen meta-heuristic algorithms of Pavia University dataset

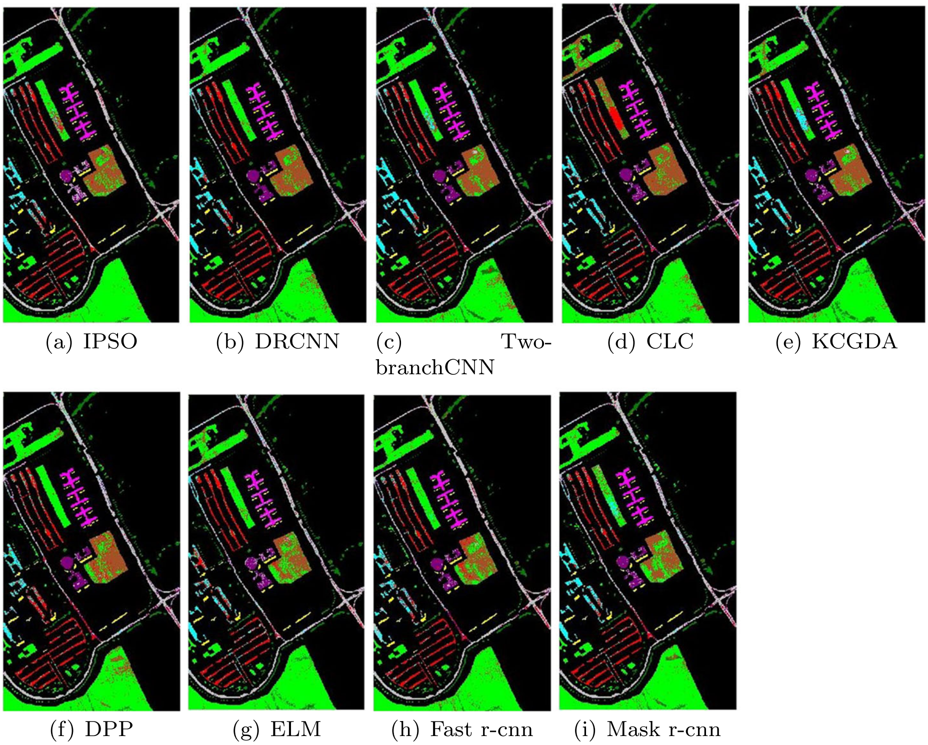

(a) Ground Truth (b) Training data (c) SVM (d) IPSO-10 (e) CS-10 (f) DE-10

(g) FF-10 (h) GA-10 (i) PSO-10 (j) IPSO-12 (k) CS-12 (l) DE-12

(m) FF-12 (n) GA-12 (o) PSO-12 (p) IPSO-14 (q) CS-14 (r) DE-14

(s) FF-14 (t) GA-14 (u) PSO-14

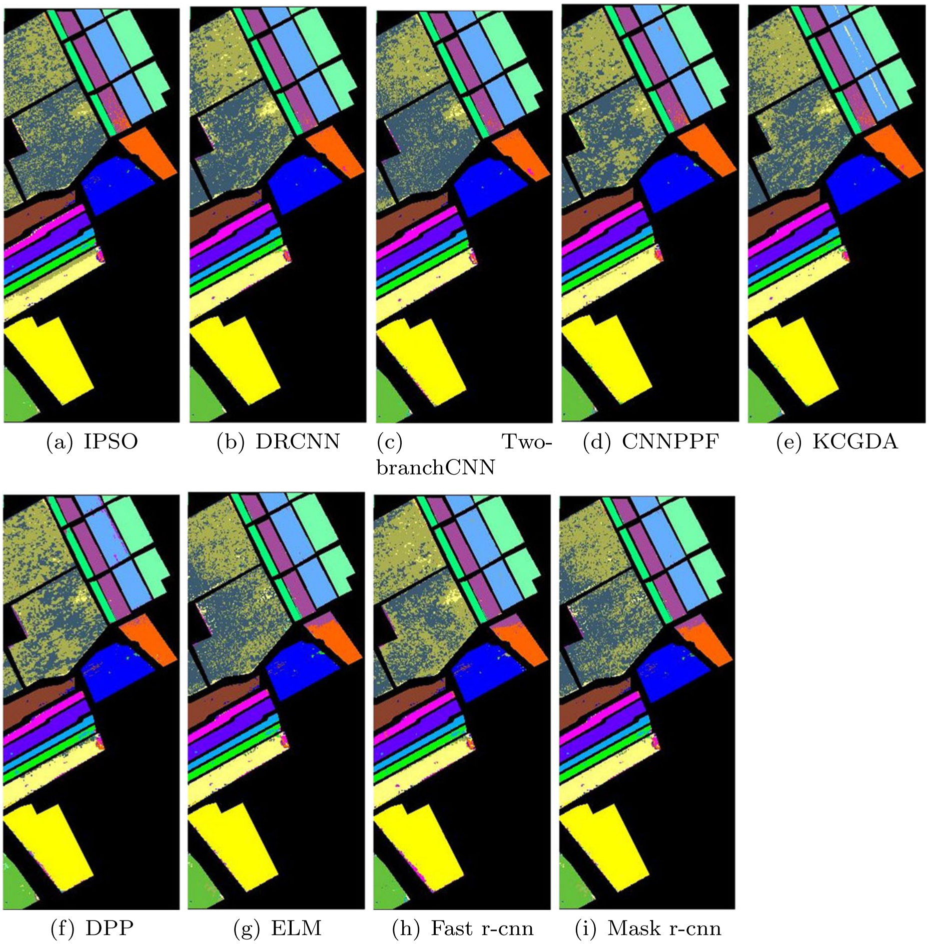
**Fig. 15** Ground truth image, Training data, SVM classified output, SVMCK 10,12 and 14 level classified output using chosen meta-heuristic algorithms of Salinas dataset


(a) IPSO (b) DRCNN (c) Two-branchCNN (d) CLC (e) KCGDA

(f) DPP (g) ELM (h) Fast r-cnn (i) Mask r-cnn

**Fig. 16** Comparison with CNN-based approaches over 50% training data of Pavia dataset

Table 6 displays the comparative results of traditional SVM and different meta-heuristic algorithms based SVMCK results for levels 10,12 and 14 of Indian Pines data. It is observed that the results produced by SVMCK with optimization algorithms are better than traditional SVM. Similarly, composite kernel-based classification results of Pavia and Salina datasets are shown in the Tables 7 and 8 respectively. All these results conclude that the composition of spatial and spectral information based kernel has produced more accurate and better classification than conventional SVM. All the accuracy parameters which have produced the best values for the proposed algorithm for every level of segmentation have been marked as bold. Finally, the outcomes of there datasets along with their ground-truth and training data are displayed in the Figs. 13, 14, and 15. The results are further compared with some current proposed CNN-based approaches like Deep Pixel Pair (DPP) [32], Extreme Learning Machines (ELM) [33], Diverse-Region based CNN [67], Collaborative Classification [66], Two-branch CNN [64], Kernel Composite Grapg Discriminant Analysis (KCGDA) [34], region-based fast R-CNN [19], and Mask R-CNN [22]. The parameters of the R-CNN based approaches have been set by the guidelines given in the manuscript [19]. As CNN performs better with large data, so above-stated approaches are tested along with the proposed approach on Pavia and Salinas dataset. To compare the outcome of the



(a) IPSO (b) DRCNN (c) Two-branchCNN (d) CNNPPF (e) KCGDA

(f) DPP (g) ELM (h) Fast r-cnn (i) Mask r-cnn

**Fig. 17** Comparison with CNN-based approaches over 50% training data of Salinas dataset

proposed algorithm with CNN-based approaches, 50% data instead of 5% have been trained and tested for both the dataset. The outcomes of the proposed technique has been passed by the morphological filter for comparing with CNN-based approaches. The detailed outcome of the proposed approach along with the mentioned CNN-based approaches for Pavia and Salinas dataset have been shown in the Figs. 16 and 17 respectively. Results of the accuracy parameters like OA, KI, MA, IoU are shown along with the individual accuracy of various classes in the Tables 9 and 10. Best values produced by the proposed algorithm are marked as bold and it is observed that the outcomes of the proposed scheme are marginal better as compared to the CNN-based approaches.

**Table 9** Performance comparison of the proposed scheme with CNN-based algorithms for the Pavia dataset of 50% training data

| Class | IPSO | DRCNN [67] | Two-branch-CNN [64] | CLC [66] | KCGDA [34] | DPP [32] | ELM [33] | Fast R-CNN [19] | Mask R-CNN [22] |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.9925 | 0.9825 | 0.9785 | 0.9627 | 0.8627 | 0.7558 | 0.7802 | 0.7842 | 0.8069 |
| 2 | 0.9997 | 0.9995 | 0.9990 | 0.9985 | 0.9311 | 0.9079 | 0.9093 | 0.8904 | 0.9182 |
| 3 | 0.9950 | 0.9920 | 0.9875 | 0.9850 | 0.6687 | 0.5508 | 0.5580 | 0.7501 | 0.5929 |
| 4 | 0.9982 | 0.9963 | 0.9957 | 0.9942 | 0.7889 | 0.7961 | 0.8673 | 0.7868 | 0.7911 |
| 5 | 0.9955 | 0.9935 | 0.9905 | 0.9820 | 0.9767 | 0.9737 | 0.8470 | 0.9797 | 0.9722 |
| 6 | 0.9820 | 0.9730 | 0.9524 | 0.9127 | 0.7673 | 0.7090 | 0.6490 | 0.6418 | 0.6538 |
| 7 | 0.9829 | 0.9742 | 0.9655 | 0.9574 | 0.5936 | 0.9217 | 0.7920 | 0.8347 | 0.8484 |
| 8 | 0.9955 | 0.9942 | 0.9902 | 0.9854 | 0.8133 | 0.8236 | 0.8196 | 0.8238 | 0.7859 |
| 9 | 0.9999 | 0.9997 | 0.9995 | 0.9993 | 0.9872 | 0.9839 | 0.9957 | 0.9712 | 0.9776 |
| OA | **0.9934** | 0.9893 | 0.9894 | 0.9752 | 0.8603 | 0.8328 | 0.8271 | 0.8276 | 0.8343 |
| KI | **0.9825** | 0.9789 | 0.9745 | 0.9624 | 0.8141 | 0.7793 | 0.7706 | 0.7717 | 0.7797 |
| MA | **0.9840** | 0.9709 | 0.9625 | 0.9550 | 0.8211 | 0.8150 | 0.8020 | 0.8144 | 0.8157 |
| IoU | **0.7621** | 0.7440 | 0.7425 | 0.7219 | 0.6915 | 0.6638 | 0.6501 | 0.6726 | 0.6606 |

**Table 10** Performance comparison of the proposed scheme with CNN-based algorithms for the Salinas dataset of 50% training data

| Class | IPSO | DRCNN [67] | Two-branch-CNN [64] | CLC [66] | KCGDA [34] | DPP [32] | ELM [33] | Fast R-CNN [19] | Mask R-CNN [22] |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.9879 | 0.9812 | 0.9722 | 0.9650 | 0.9800 | 0.9705 | 0.9760 | 0.9760 | 0.9775 |
| 2 | 0.9995 | 0.9991 | 0.9990 | 0.9988 | 0.9865 | 0.9736 | 0.9742 | 0.9769 | 0.9691 |
| 3 | 0.9935 | 0.9922 | 0.9921 | 0.9875 | 0.9949 | 0.9379 | 0.8057 | 0.8143 | 0.8103 |
| 4 | 0.9985 | 0.9979 | 0.9955 | 0.9933 | 0.9798 | 0.9574 | 0.9978 | 0.9978 | 0.9906 |
| 5 | 0.9755 | 0.9601 | 0.9655 | 0.9524 | 0.9104 | 0.9955 | 0.9824 | 0.9831 | 0.9895 |
| 6 | 0.9822 | 0.9719 | 0.9724 | 0.9657 | 0.9937 | 0.9392 | 0.9914 | 0.9919 | 0.9878 |
| 7 | 0.9712 | 0.9092 | 0.9910 | 0.9824 | 0.9936 | 0.9938 | 0.9944 | 0.9941 | 0.9899 |
| 8 | 0.9990 | 0.9881 | 0.9785 | 0.9755 | 0.7014 | 0.5165 | 0.6744 | 0.6962 | 0.5198 |
| 9 | 0.9965 | 0.9972 | 0.9955 | 0.9902 | 0.9984 | 0.9793 | 0.9860 | 0.9871 | 0.9654 |
| 10 | 0.9988 | 0.9555 | 0.9621 | 0.9421 | 0.6992 | 0.8305 | 0.8947 | 0.9030 | 0.8641 |
| 11 | 0.9659 | 0.9752 | 0.9650 | 0.9521 | 0.8800 | 0.9149 | 0.9272 | 0.9272 | 0.9140 |
| 12 | 0.9888 | 0.9775 | 0.9555 | 0.9421 | 0.9979 | 0.9958 | 0.9969 | 0.9969 | 0.9948 |
| 13 | 0.9997 | 0.9996 | 0.9950 | 0.9902 | 0.9790 | 0.9901 | 0.9823 | 0.9823 | 0.9868 |
| 14 | 0.9955 | 0.9929 | 0.9910 | 0.9825 | 0.9236 | 0.8745 | 0.9189 | 0.9189 | 0.8972 |
| 15 | 0.9953 | 0.9815 | 0.9822 | 0.9722 | 0.6000 | 0.7198 | 0.6480 | 0.6686 | 0.7835 |
| 16 | 0.9999 | 0.9996 | 0.9995 | 0.9888 | 0.9744 | 0.9583 | 0.9215 | 0.9271 | 0.9677 |
| OA | **0.9904** | 0.9799 | 0.9820 | 0.9738 | 0.8527 | 0.8316 | 0.8590 | 0.8677 | 0.8411 |
| KI | **0.9722** | 0.9601 | 0.9611 | 0.9521 | 0.8359 | 0.8135 | 0.8434 | 0.8530 | 0.7797 |
| MA | **0.9810** | 0.9719 | 0.9722 | 0.9655 | 0.9120 | 0.9092 | 0.9170 | 0.9077 | 0.9102 |
| IoU | **0.7859** | 0.7835 | 0.7841 | 0.7795 | 0.7774 | 0.7265 | 0.7729 | 0.7684 | 0.7434 |

## 7 Conclusion and future work

An improved PSO based multi-level image thresholding scheme has been proposed. Among different entropies, membership function-based fuzzy entropy has been chosen as an objective function because of its easiness and effectiveness over color images to calculate threshold values. IPSO has successfully overcome the challenges of premature convergence compared to other known popular optimization algorithms. The composite-kernel based approach has generated better classification outcomes with respect to the accuracy parameters. The approach has been further compared with some CNN-based approaches and results infer that the values are closely following to them. This proposed approach can be applied to different types of machine-learning based classification fields like Optical Character Recognition, object identification in multiple frames of video surveillance problems etc. Like CNN, this approach can be applied on some larger dataset like ImageNet, COCO (Common Objects in Context) to recognize objects. Further, some parameters can be modified of the proposed IPSO to form kernel so that accuracy can reach to the higher side.

**Acknowledgments** The authors would like to thank Mehmet Altan Toksöz, Student Member, IEEE, for helping by providing the color map information of ground-truth classes as source code and Prof. Jun Li, School of Geography and Planning, Sun Yat-Sen University, China for kindly providing the partial source codes of SVM-based the composite kernel used in this paper. The authors also would like to show their gratitude to Prof. P. Gamba and Prof. D. A. Landgrebe for kindly providing the data sets used in this paper.

## References

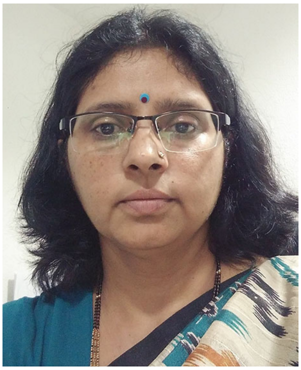
1. Agrawal S, Panda R, Bhuyan S, Panigrahi BK (2013) Tsallis entropy based optimal multilevel thresholding using cuckoo search algorithm. Swarm Evol Comput 11:16–30
2. Akay B (2013) A study on particle swarm optimization and artificial bee colony algorithms for multilevel thresholding. Appl Soft Comput 13(6):3066–3091
3. Arifin AZ, Asano A (2006) Image segmentation by histogram thresholding using hierarchical cluster analysis. Pattern Recogn Lett 27(13):1515–1521
4. Arora S, Acharya J, Verma A, Panigrahi PK (2008) Multilevel thresholding for image segmentation through a fast statistical recursive algorithm. Pattern Recogn Lett 29(2):119–125
5. Baskar S, Alphones A, Suganthan P, Liang J (2005) Design of yagi–uda antennas using comprehensive learning particle swarm optimisation. IEE Proceedings-Microwaves, Antennas and Propagation 152(5):340–346
6. Bhandari AK, Singh VK, Kumar A, Singh GK (2014) Cuckoo search algorithm and wind driven optimization based study of satellite image segmentation for multilevel thresholding using Kapur's entropy. Expert Syst Appl 41(7):3538–3560
7. Bhandari AK, Kumar A, Singh GK (2015) Modified artificial bee colony based computationally efficient multilevel thresholding for satellite image segmentation using Kapur's, Otsu and Tsallis functions. Expert Syst Appl 42(3):1573–1601
8. Boskovitz V, Guterman H (2002) An adaptive neuro-fuzzy system for automatic image segmentation and edge detection. IEEE Trans Fuzzy Syst 10(2):247–262
9. Camps-Valls G, Bruzzone L (2005) Kernel-based methods for hyperspectral image classification. IEEE Trans Geosci Remote Sens 43(6):1351–1362
10. Camps-Valls G, Gomez-Chova L, Muñoz-Marí J, Vila-Francés J, Calpe-Maravilla J (2006) Composite kernels for hyperspectral image classification. IEEE Geosci Remote Sens Lett 3(1):93–97
11. Cao L, Bao P, Shi Z (2008) The strongest schema learning ga and its application to multilevel thresholding. Image Vis Comput 26(5):716–724
12. Chander A, Chatterjee A, Siarry P (2011) A new social and momentum component adaptive pso algorithm for image segmentation. Expert Syst Appl 38(5):4998–5004
13. Cheng HD, Sun Y (2000) A hierarchical approach to color image segmentation using homogeneity. IEEE Trans Image Process 9(12):2071–2082

14. Dong L, Yu G, Ogunbona P, Li W (2008) An efficient iterative algorithm for image thresholding. Pattern Recogn Lett 29(9):1311–1316
15. Du J (2008) Property of tsallis entropy and principle of entropy increase. arXiv:08023424
16. Fauvel M, Tarabalka Y, Benediktsson JA, Chanussot J, Tilton JC (2013) Advances in spectral-spatial classification of hyperspectral images. Proc IEEE 101(3):652–675
17. Gao H, Xu W, Sun J, Tang Y (2010) Multilevel thresholding for image segmentation through an improved quantum-behaved particle swarm algorithm. IEEE Trans Instrum Meas 59(4):934–946
18. Ghamisi P, Couceiro MS, Martins FM, Benediktsson JA (2014) Multilevel image segmentation based on fractional-order darwinian particle swarm optimization. IEEE Trans Geosci Remote Sens 52(5):2382–2394
19. Girshick R (2015) Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp 1440–1448
20. Hammouche K, Diaf M, Siarry P (2008) A multilevel automatic thresholding method based on a genetic algorithm for a fast image segmentation. Comput Vis Image Underst 109(2):163–175
21. Hammouche K, Diaf M, Siarry P (2010) A comparative study of various meta-heuristic techniques applied to the multilevel thresholding problem. Eng Appl Artif Intel 23(5):676–688
22. He K, Gkioxari G, Dollár P, Girshick R (2017) Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp 2961–2969
23. Horng MH (2011) Multilevel thresholding selection based on the artificial bee colony algorithm for image segmentation. Expert Syst Appl 38(11):13,785–13,791
24. Hughes G (1968) On the mean accuracy of statistical pattern recognizers. IEEE Trans Inf Theory 14(1):55–63
25. Kapur JN, Sahoo PK, Wong AK (1985) A new method for gray-level picture thresholding using the entropy of the histogram. Comput Vis Graph Image Process 29(3):273–285
26. Karaboga D, Gorkemli B, Ozturk C, Karaboga N (2014) A comprehensive survey: artificial bee colony (abc) algorithm and applications. Artif Intell Rev 42(1):21–57
27. Kittler J, Illingworth J (1986) Minimum error thresholding. Pattern Recogn 19(1):41–47
28. Kurugollu F, Sankur B, Harmanci AE (2001) Color image segmentation using histogram multithresholding and fusion. Image Vis Comput 19(13):915–928
29. Li CH, Lee C (1993) Minimum cross entropy thresholding. Pattern Recogn 26(4):617–625
30. Li J, Bioucas-Dias JM, Plaza A (2012) Spectral–spatial hyperspectral image segmentation using subspace multinomial logistic regression and Markov random fields. IEEE Trans Geosci Remote Sens 50(3):809–823
31. Li J, Marpu PR, Plaza A, Bioucas-Dias JM, Benediktsson JA (2013) Generalized composite kernel framework for hyperspectral image classification. IEEE Trans Geosci Remote Sens 51(9):4816–4829
32. Li W, Wu G, Zhang F, Du Q (2017) Hyperspectral image classification using deep pixel-pair features. IEEE Trans Geosci Remote Sens 55(2):844–853
33. Li L, Wang C, Li W, Chen J (2018) Hyperspectral image classification by adaboost weighted composite kernel extreme learning machines. Neurocomputing 275:1725–1733
34. Li W, Feng F, Li H, Du Q (2018) Discriminant analysis-based dimension reduction for hyperspectral image classification: a survey of the most recent advances and an experimental comparison of different techniques. IEEE Geosci Remote Sens Mag 6(1):15–34
35. Liang JJ, Qin AK, Suganthan PN, Baskar S (2006) Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Trans Evol Comput 10(3):281–295
36. Mughees A, Chen X, Tao L (2016) Unsupervised hyperspectral image segmentation: merging spectral and spatial information in boundary adjustment. In: 2016 55th Annual Conference of the Society of instrument and control engineers of Japan (SICE). IEEE, pp 1466–1471
37. Mukhopadhyay S, Banerjee S (2012) Global optimization of an optical chaotic system by chaotic multi swarm particle swarm optimization. Expert Syst Appl 39(1):917–924
38. Naidu M, Kumar PR, Chiranjeevi K (2017) Shannon and fuzzy entropy based evolutionary image thresholding for image segmentation. Alexandria Engineering Journal
39. Nayak DR, Dash R, Majhi B (2018) Discrete ripplet-ii transform and modified pso based improved evolutionary extreme learning machine for pathological brain detection. Neurocomputing 282:232–247
40. Önüt S, Tuzkaya UR, Doğaç B (2008) A particle swarm optimization algorithm for the multiple-level warehouse layout design problem. Comput Indu Engineering 54(4):783–799
41. Otsu N (1979) A threshold selection method from gray-level histograms. IEEE Trans Syst Man Cybern 9(1):62–66
42. Pal NR (1996) On minimum cross-entropy thresholding. Pattern Recogn 29(4):575–580

43. Pare S, Bhandari AK, Kumar A, Singh GK, Khare S (2015) Satellite image segmentation based on different objective functions using genetic algorithm: a comparative study. In: 2015 IEEE International conference on digital signal processing (DSP). IEEE, pp 730–734
44. Pare S, Bhandari A, Kumar A, Singh G (2017) A new technique for multilevel color image thresholding based on modified fuzzy entropy and Lévy flight firefly algorithm. Computers & Electrical Engineering
45. Perez A, Gonzalez RC (1987) An iterative thresholding algorithm for image segmentation. IEEE Trans Pattern Anal Mach Intell 6:742–751
46. Ratnaweera A, Halgamuge SK, Watson HC (2004) Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Trans Evol Comput 8(3):240–255
47. Revol C, Jourlin M (1997) A new minimum variance region growing algorithm for image segmentation. Pattern Recogn Lett 18(3):249–258
48. Sahoo PK, Arora G (2004) A thresholding method based on two-dimensional Renyi's entropy. Pattern Recogn 37(6):1149–1161
49. Sarkar S, Paul S, Burman R, Das S, Chaudhuri SS (2014) A fuzzy entropy based multi-level image thresholding using differential evolution. In: International conference on swarm, evolutionary, and memetic computing. Springer, pp 386–395
50. Sarkar S, Das S, Chaudhuri SS (2015) A multilevel color image thresholding scheme based on minimum cross entropy and differential evolution. Pattern Recogn Lett 54:27–35
51. Sarkar S, Das S, Chaudhuri SS (2016) Hyper-spectral image segmentation using Rényi entropy based multi-level thresholding aided with differential evolution. Expert Syst Appl 50:120–129
52. Sathya P, Kayalvizhi R (2010) Pso-based tsallis thresholding selection procedure for image segmentation. Int J. Comput Appl 5(4):39–46
53. Schölkopf B, Smola AJ (2002) Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT Press
54. Sezgin M, Taşaltın R (2000) A new dichotomization technique to multilevel thresholding devoted to inspection applications. Pattern Recogn Lett 21(2):151–161
55. Sezgin M et al (2004) Survey over image thresholding techniques and quantitative performance evaluation. J Electron Imag 13(1):146–168
56. Suresh S, Lal S (2016) An efficient cuckoo search algorithm based multilevel thresholding for segmentation of satellite images using different objective functions. Expert Syst Appl 58:184–209
57. Tarabalka Y, Chanussot J, Benediktsson JA (2010) Segmentation and classification of hyperspectral images using watershed transformation. Pattern Recogn 43(7):2367–2379
58. Toksöz MA, Ulusoy I (2016) Classification via ensembles of basic thresholding classifiers. IET Comput Vis 10(5):433–442
59. Toksöz MA, Ulusoy I (2016) Hyperspectral image classification via basic thresholding classifier. IEEE Trans Geosci Remote Sens 54(7):4039–4051
60. Van den Bergh F, Engelbrecht AP (2004) A cooperative approach to particle swarm optimization. IEEE Trans Evol Comput 8(3):225–239
61. Wang S, Chung Fl, Xiong F (2008) A novel image thresholding method based on parzen window estimate. Pattern Recogn 41(1):117–129
62. Weszka JS (1978) A survey of threshold selection techniques. Comput Graph Image Process 7(2):259–265
63. Wong AK, Sahoo PK (1989) A gray-level threshold selection method based on maximum entropy principle. IEEE Trans Syst Man Cybern 19(4):866–871
64. Xu X, Li W, Ran Q, Du Q, Gao L, Zhang B (2018) Multisource remote sensing data classification based on convolutional neural network. IEEE Trans Geosci Remote Sens 56(2):937–949
65. Zahara E, Fan SKS, Tsai DM (2005) Optimal multi-thresholding using a hybrid optimization approach. Pattern Recogn Lett 26(8):1082–1095
66. Zhang M, Li W, Du Q (2017) Collaborative classification of hyperspectral and visible images with convolutional neural network. J Appl Remote Sens 11(4):042,607
67. Zhang M, Li W, Du Q (2018) Diverse region-based cnn for hyperspectral image classification. IEEE Trans Image Process 27(6):2623–2634
68. Zheng H, Jie J, Hou B, Fei Z (2014) A multi-swarm particle swarm optimization algorithm for tracking multiple targets. In: 2014 IEEE 9th Conference on industrial electronics and applications (ICIEA). IEEE, pp 1662–1665

**Publisher's note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

**Rupak Chakraborty** pursuing his PhD degree from DIT University, Dehradun, India, under the joint supervision of Dr. Rama Sushil and Dr. M.L Garg and also working as an Assistant professor in Computer Science and Engineering department in Bennett University as well, obtained his M.Tech degree from JIS College of Engineering (West Bengal, India) in the year of 2011. His research area includes image processing and support vector machine. His publication till date includes three international conferences and two journals and also holding almost 7 years of teaching experience as an Assistant Professor in IT department.



**Prof. Rama Sushil** obtained her PhD degree from IIT Roorkee. She has more than 18 years of teaching and research experience, published a book on “Artificial Intelligence” and chapters on “cloud computing” & “M Commerce”, many research papers published in peer reviewed int. Journals of repute. She has presented her research work in Japan, Italy, USA & Taiwan. Currently serving as head of Information Technology department at DIT University, Dehradun, India and guiding PhD research scholars in the field of Cloud, cryptography and image processing.

**Dr. M. L. Garg**, presently Professor, Department of Computer Science & Engineering at DIT University, Dehradun, India, has obtained his PhD degree in Computer Science & Engineering from Thapar Institute of Engineering & Technology (Deemed University), Patiala, India with collaborative research work at IIT Delhi, in the year 1992. He has a total experience of about 37 years out of which 16 years is in academics and 21 years is in Punjab State Electricity Board. Three researchers have completed and 4 others are pursuing their Ph.D under his supervision. He has been the Dean Faculty of Engineering for a period of 5 years at Shri Mata Vaishno Devi University, Katra, India. His area of research includes Fuzzy Logic Genetic Algorithms & Knowledge Representation and Reasoning. He has published about thirty research articles in peer reviewed International and National Journals and Conferences. He is on the Editorial Board of the International Journal of Cloud Computing and Super Computing (IJCS), Canada.