Multispectal Image Classification Using Rough Set Theory and Particle Swam Optimization

Chih-Cheng Hung, Hendri Purnawan, Bor-Chen Kuo* and Scott Letkeman
Southern Polytechnic State University, Marietta, GA 30060 USA
*National Taichung University, Taichung, Taiwan

Abstract

This chapter provides an exploration of the rough set theory and particle swarm optimization for multispectral image classification. The rough set theory, particle swarm optimization and K-means clustering algorithm are briefly described. Two multispectral image classification algorithms based on the rough set theory and particle swarm optimization are proposed: Algorithm 1 is a multispectral image classification approach based on the rough set theory which uses upper and lower bounds for the class description, and Algorithm 2 is a hybrid rough K-means algorithm for image classification. The rough set theory is used to extract classification rules and establish the lower and upper bounds for data clustering with the K-means algorithm. This algorithm is able to deal with vagueness in image date, but since its ability to determine some key parameters is limited, partial swarm optimization must subsequently be used to locate optimal values for those parameters. Experimental results show that the proposed algorithms perform well and improve the classification in the blurred and vague areas of the image. A comparison of Algorithm 1 with the parallelepiped classifier, where the former uses the concept of cuts and the later uses the maximum and minimum values, is performed. Preliminary experimental results show that the proposed classifiers are effective for multispectral image classification.

1. Introduction

In the real world, data representation is most often imperfect, in the sense that the data may be either incomplete or redundant. Philosophers, logicians and mathematicians have dealt with this problem for a long time. In recent years, propelled by the advent of the computer, the problem of imperfect knowledge has been becoming an important topic for computer scientists engaged in artificial intelligence research, especially those involved with knowledge discovery from databases, expert systems, and pattern recognition.

Our research is focused on rough set as a tool for image processing, or more precisely, for image segmentation. Many techniques for image segmentation have been developed over time. There are clustering, edge detection, region growing and even more advanced techniques that use neural networks. In general, image segmentation techniques can be categorized as supervised or unsupervised. Supervised techniques require previously known truth data for training purposes, while unsupervised techniques have no such requirement

In this chapter, the classical rough set theory is reviewed in section 2. Particle swarm optimization is then introduced in section 3. The Davies-Bouldin measure for cluster validity is also described in this section. The K-means algorithm is briefly sketched in section 4. Multispectral image classification using rough set theory is discussed in section 5. A hybrid algorithm which combines the K-means algorithm, rough set and particle swarm optimization is given in section 6. Experimental results are shown in section 7. The conclusion and future work then follow.

2. Rough Set Theory

Rough set theory [5] is a mathematical tool that deals with the uncertainty of the data. The theory consists of finite sets, equivalence relations and cardinality concepts. As the theory matures and more applications reap the benefits of the concept, an abundance of related theorems and algorithms are being incorporated to extend rough sets theory.

It was introduced by Pawlak in the early 1980's and has been argued to overlap with other theories, such as statistics, evidence theory and fuzzy set. Furthermore, rough set is said to complement fuzzy set, a theory introduced by Zadeh in the early period. Rough set and fuzzy set were both introduced to deal with imprecise information however; fuzzy set deals with vagueness, while rough set deals with coarseness. Rough set does not need as much preliminary knowledge about the data where as fuzzy set requires knowledge of the possible values in advance. Basically, when using rough set, the data itself is used to come up with the approximation in order to deal with the imprecision within. It can therefore be considered a self-sufficient discipline.

Rough set mainly deals with data analysis in table format. The approach is generally to pre-process the data in the table and then to analyze them. Reducts are extracted with an algorithm and finally rules are generated based on the reducts. Rough set does not support analog values in the table attributes; therefore discretization must be performed in advance in order to evaluate the table. The following subsections will use a simple example to illustrate the concept of rough set theory.

2.1 Information Systems

In essence, an information system is a set of objects represented in a data table (attribute - value system). Each row contains an object and each column represents a measurable attribute for each object. Formally, an information system is a pair $\mathcal{A} = (U, A)$ where U is a non- empty finite set of objects representing the universe and A is a non-empty finite set of attributes such that $a: U \rightarrow V_a$ for every $a \in A$. The set V_a is the set of values for a.

Object Index	*Salary*	*Age*
u_1	80	30
u_2	30	23
u_3	80	40
u_4	50	45
u_5	80	55
u_6	50	45
u_7	30	60
u_8	100	35

Table 1. shows an information system which is a collection of salary and age attributes.

2.2 Decision Systems

If an information system has an additional attribute, namely a decision attribute, then it becomes a decision system. The decision attribute is associated with the object classification outcome, and it may depend on several other attributes. Formally, a decision system is a piece of information whose form is $\mathcal{A}$ = (U, A ∪ {d}), where d ∈A is the decision attribute. A decision attribute called "Class" has been added as shown in Table 2, where M and E denote Manager and Employee, respectively. The table was modified from the original [12].

Object Index	*Salary*	*Age*	Class
u_1	80	30	M
u_2	30	23	M
u_3	80	40	E
u_4	50	45	M
u_5	80	55	E
u_6	50	45	E
u_7	30	60	M
u_8	100	35	E

Table 2. A decision system where each row is classified into a class. Salary and Age are the condition attributes, while class is the decision attribute.

2.3 Indiscernibility

Objects in information and decision systems may be indistinguishable from one another based on a set of attributes B that belongs to A (B ⊆ A). A set of objects is indiscernible or equivalent when their attributes are related by an equivalence relation. An equivalence relation is a relation on a set B when it is:

1. Reflexive (if a R a, then R is reflexive).
2. Symmetric (if a R b then b R a, then R is symmetric).
3. Transitive (if a R b and b R c, then a R c, thus R is transitive).

For an information system $\mathcal{A}$ = (U, A), there is an equivalence relation for any of the sets B ⊆ A. The equivalence relation can be formalized as

$$IND_A(B) = \{(x,x') \in U^2 \mid \forall a \in B, \quad a(x) = a(x')\}.$$

Referring to table 4, the *IND* relations for *Salary* can be written as shown.

$$IND(Salary) = \{\{u_1, u_3, u_5, u_8\}, \{u_2, u_4, u_6, u_7\}\}$$

It is impossible to write the *IND* relations for *Salary* until discretization is completed.

2.4 Discretization

Discretization is not directly related to rough set theory. It is simply a preprocessing technique. Discretization is associated with information loss. In general, when it is too coarse (i.e. longer interval), there is too much information loss or noise in the data. However, it is better for the classification capability of unseen objects. When the discretization is more fine (i.e. shorter interval), less noise exists in data, but classification capability of unseen objects may be impaired.

In our decision system table, both Salary and Age need to be discretized. The set of possible Salary and Age values, respectively referred to as *s* and *a* from here on, is given by

$$V_s = [15, 120)$$
$$V_a = [18, 65)$$

The lower and upper bounds of the attribute's interval are extended to cover possible values. For example, the Age attribute is extended to include likely working ages from age 18 through 65.

The set of values of *s* and *a* in U is

$$s(U) = \{30,50,80,100\}$$
$$a(U) = \{23, 30, 35, 40, 45, 55, 60\}$$

The intervals obtained for *s* are

$$[30, 50); [50, 80); [80, 100)$$

The intervals obtained for *a* are

$$[23, 30); [30, 35); [35, 40);$$
$$[40, 45); [45, 55); [55, 60).$$

Boundary intervals such as [15, 30) and [100, 120) should not be used since one can not discern anything for this data set.

The intervals introduce a set of cuts, which are defined as (s, c) where $c \in V_s$ and (a, c) where $c \in V_a$. If the cut is taken based on the mid-point of each interval, the set of cuts **P** obtained for *s* and *a* are respectively

(s, 40); (s, 65); (s, 90);
(a, 26.5); (a, 32.5); (a, 37.5);
(a, 42.5); (a, 50); (a, 57.5)

The next step is to find the set of minimal cuts that can discern all of the objects that are needed. It turns out that the problem of finding the irreducible set of cuts **P** in the decision system is NP-complete while the effort to find the optimal set of cuts **P** in a decision system is NP-hard [5].

However, there are heuristics that can be used to find the optimal set of cuts **P** in practical time. One of them is the Maximal Discernability heuristic [1], [5], which is demonstrated here. The algorithm to construct table $\mathcal{A}^*$ from $\mathcal{A}$ is listed in the following steps:

1. Each column in table $\mathcal{A}^*$ is a Boolean variable of the corresponding column in $\mathcal{A}$. If each pair of objects can be discerned by the Boolean variable, then assign value 1, else assign 0.
2. Choose a column from $\mathcal{A}^*$ that has a maximal number of 1's and delete all the rows which contain a 1 in the selected column.
3. Repeat step 2 and continue until all columns and rows are consumed.

$\mathcal{A}^*$	p_1^s	p_2^s	p_3^s	p_1^a	p_2^a	p_3^a	p_4^a	p_5^a	p_6^a	d
u1, u3	0	0	0	0	1	1	0	0	0	1
u1, u5	0	0	0	0	1	1	1	1	0	1
u1, u6	0	1	0	0	1	1	1	0	0	1
u1, u8	0	0	1	0	1	0	0	0	0	1
u2, u3	1	1	0	1	1	1	0	0	0	1
u2, u5	1	1	0	1	1	1	1	1	0	1
u2, u6	1	0	0	1	1	1	1	0	0	1
u2, u8	1	1	1	1	1	0	0	0	0	1
u3, u4	0	1	0	0	0	0	1	0	0	1
u3, u7	1	1	0	0	0	0	1	1	1	1
u4, u5	0	1	0	0	0	0	0	1	0	1
u4, u6	0	0	0	0	0	0	0	0	0	1
u4, u8	0	1	1	0	0	1	1	0	0	1
u5, u7	1	1	0	0	0	0	0	0	1	1
u6, u7	1	0	0	0	0	0	0	1	1	1
u7, u8	1	1	1	0	0	1	1	1	1	1
new	0	0	0	0	0	0	0	0	0	0

Table 3. Decision System A*.

The following example clarifies the process of constructing table $\mathcal{A}^*$ from $\mathcal{A}$ mentioned in step 1 of the algorithm. Each cut previously obtained is assigned a Boolean variable, which in turn is used as a condition attribute in table $\mathcal{A}^*$.

For example, (s, 40) is assigned Boolean variable p_1^s . Each object pair in $\mathcal{A}^*$, is derived from table $\mathcal{A}$ by looking at the decision attribute. Objects that do not have the same decision attribute should be paired up. For example, u_1 and u_3 are paired up since they have different decision values (i.e. M and E).

The resulting table $\mathcal{A}^*$ created from $\mathcal{A}$ is shown in Table 3.

The optimal cut chosen is (s, 65), (a, 32.5) and (a, 50). These cuts are then used to discretize the decision table 2. The rank can be assigned using the following rules:

1 If s < 65, value 0 is assigned to s, else assign value of 1.
2 If a < 32.5, value 0 is assigned to a.
3 If (32.5 ≤ a < 50), value 1 is assigned to a.
4 If a ≥ 50, value 2 is assigned to a.

A discretized table can be produced by applying the condition to each analog value in the table.

Index	*Salary*	*Age*	Class
u_1	1	0	M
u_2	0	0	M
u_3	1	1	E
u_4	0	1	M
u_5	1	2	E
u_6	0	1	E
u_7	0	2	M
u_8	1	1	E

Table 4. Discretized Decision System.

2.5 Lower and Upper Approximations in Rough Set and Accuracy

Let U be the non-empty finite set and R be an equivalence relation. The pair $\mathcal{A}$ = (U, R) is an approximation space. The equivalence relation R on U leads to a partition of the objects in the universe U. The idea here is to partition the objects that have the same outcome, or in other words, to partition objects that have the same decision attribute. However, this may not always be as easy as stated. There will be objects with the same condition attributes (in the same equivalence class), but different decision attributes. Therefore one can not define every set precisely.

In cases where the set can not be defined precisely, it can be approximated. This is where rough set emerges. Let us assume that there is an information system $\mathcal{A}$ = (U, A), a set of attributes B $\subseteq$ A, and a set of objects X $\subseteq$ U. Using the set of attributes B, one can approximate the objects X into:

1. Lower Approximation: the set of objects that can be classified as member X with certainty. Formally stated as

$$\underline{B}(X) = \bigcup \{E_i \in U^B : E_i \subseteq X\} \tag{2.1}$$

2. Upper Approximation: the set of objects that can possibly be classified as a member X. Formally stated as

$$\overline{B}(X) = \bigcup \{E_i \in U^B : E_i \cap X \neq \{\}\} \tag{2.2}$$

Between the lower and upper approximation, one can define the set of objects that cannot be classified into X decisively. This set is also known as the *B-boundary region* of X.

There is a coefficient that reflects the accuracy of approximation,

$$\alpha_B(X) = \frac{|\underline{B}(X)|}{|\overline{B}(X)|} \tag{2.3}$$

where |X| denotes the cardinality of X ≠ ∅. When α_B = 1, the X is crisp with respect to B, otherwise, X is rough with respect to B.

For our example, the boundary region would be for object u_4 and u_6 since they can not be discerned. The lower and upper approximations can be written as

$$\underline{B}(M) = \{u_1, u_2, u_4, u_7\}$$
$$\overline{B}(M) = \{u_1, u_2, u_4, u_6, u_7\}$$

$$\alpha_B(M) = \frac{|\underline{B}(M)|}{|\overline{B}(M)|} = \frac{4}{5} = 0.8$$

In general, the value of α reflects the accuracy of decision rules obtained.

2.6 Reducts

One way to increase computation efficiency is to reduce the size of data by reducing attributes that need to be taken into account. Only attributes that do not contribute to the classification result can be omitted such that the indiscernibility relation remains intact. The set of remaining attributes is the minimal set and is called a reduct.

Although finding the equivalence class is a relatively straightforward computation process, finding reducts with minimal attributes is known to be NP-hard. Fortunately, there are heuristics that allow minimal reducts to be computed in reasonable time.

2.7 Discernibility Matrix

Computing the reducts of an information system $\mathcal{A}$ = (U, A) can be started by creating the indiscernibility matrix. This matrix is a symmetric *n* x *n* matrix where each entry c_{ij} is defined as

$$c_{ij} = \{a \in A \mid a(x_i) \neq a(x_j)\} \quad for \quad i, j = 1, \ldots, n \tag{2.4}$$

Each cell in the matrix holds the set of attributes where objects x_i and x_j are discernable. The cell would have empty set when:

1 $x_i = x_j$, that is case for diagonal cells.

2 For decision systems, when the decision attribute of objects x_i and x_j are equal, or formally, $d(x_i) = d(x_j)$.

2.8 Discernibility Functions

Based on the discernibility matrix, a discernibility function can be immediately obtained. It is constructed using Boolean expressions from the discernibility matrix, defined as

$$f_A(a_1^*,...,a_m^*) = \wedge \{\vee \ c_{ij}^* \mid 1 \le j \le i \le n, \ \ c_{ij} \ne \{\}\} \tag{2.5}$$

where $a_1^*,...,a_m^*$ are related to attributes $a_1,...,a_m$. The attributes may be transformed during discretization process. $c_{ij}^* = \{a^* \mid a \in c_{ij}\}$ is the set of Boolean variables.

Once the discernibility function f_A is formed, it can be further developed using Boolean algebra simplification.

2.9 Decision Rules

When applying rough set for supervised learning, we need to construct a set of rules from the training data, such that new or unseen objects can be separated into known classes.
A basic method for forming the decision rules is begun by finding the reducts of the decision table. Then for each reducts $R = \{r_1,...,r_n\}$, we generate the decision rule by taking the conjunction $(r_1 = r_1(u)) \wedge .... \wedge (r_2 = r_2(u))$ as the predecessor. Next we take the decision attribute d with value $d(u)$ as the successor and format the predecessor and successor values as

$$(r_1 = r_1(u)) \wedge .... \wedge (r_2 = r_2(u)) \Rightarrow d = d(u) \tag{2.6}$$

Rule induction is about deciding which attributes should be included in the predecessor of the rule. Rules obtained can always be minimized, but it will introduce noise and may poorly classify the unseen objects.

Once the rules are obtained, they can be used to classify the objects that were unseen before. The basic steps involved can be outlined as follows [1].

1. Apply the existing rules to the new objects so that it can determine which rules actually are a fit to the new objects.
2. If none of the rules are matched, then fallback a must be chosen, or the objects would be classified as undefined.
3. If more than one rule is applicable, then a negotiation among the rules must be performed to decide which one to be used.

For the discernibility function extracted from the decision table 2, we obtain the following sets of decision rules by:

1. If (a < 32.5), then d = M
2. If (s > 65) and (32.5 ≤ a < 50), then d = E
3. If (s < 65) and (32.5 ≤ a < 50), then d = M
4. If (s > 65) and (a ≥ 50), then d = E
5. If (s < 65) and (32.5 ≤ a < 50), then d = E
6. If (s < 65) and (a ≥ 50), then d = M

3. Particle Swarm Optimization

PSO was originally introduced by Kennedy and Eberhart [21]. The algorithm was inspired by a sociological observation of a flock of birds behavior while searching for food. Each member of the flock moves with a direction and speed influence by its own previous state and that of the as a whole flock.

PSO consists of a swarm (collection) of particles searching through the solution space. Each particle holds information that can potentially become the solution. Each particle has a position and velocity that are mutually affecting those of other particles. Each particle will adjust its parameter according to the swarm's best outcome, while still considering its own experience. Therefore, at any instance, the following information is maintained by each particle.

- x_i, the current position of the particle;
- v_i, the current velocity of the particle; and
- y_i, the personal best position of the particle (*pbest*); the best position visited so far by the particle.
- $\hat{y}$, the global best position of the swarm (*gbest*); the best position visited so far by the entire swarm.

The search performed by the swarm is either to maximize or minimize the objective function *f(x)*. The personal best position (*pbest*) is obtained by evaluating the following.

$$y_i(t+1) = \begin{cases} y_i(t) & \textit{if } f(x_i(t+1)) \geq f(y_i(t)) \\ x_i(t+1) & \textit{if } f(x_i(t+1)) < f(y_i(t)) \end{cases} \tag{3.1}$$

The global best position (*gbest*) is obtained by using

$$\hat{y}(t) \in \{y_0, y_1, \ldots, y_s\} = \min\{f(y_0(t)), f(y_1(t)), \ldots, f(y_s(t))\} \tag{3.2}$$

After each iteration, the current position (x_i) and velocity (v_i) are recalculated using

$$v_i(t+1) = \omega v_i(t) + c_1 r_1(t)(y_i(t) - x_i) + c_2 r_2(t)(\hat{y}(t) - x_i(t)) \tag{3.3}$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{3.4}$$

where ω is the inertia weight which reflects the memory of previous velocities. $y_i(t) - x_i$ (cognitive component) represents the particle's own experience as to where the best solution is. $\hat{y}(t)-x_i$ (social component) represents the direction of the entire swarm towards the best solution. The c_1 and c_2 are acceleration constants. $r_1(t)$, $r_2(t)$ are in the distribution of $U(0,1)$ which will be a random number between 0 and 1.

In image classification, the PSO algorithm is used to optimize the objective functions that are mainly to:

- Minimize the distance between pixels and cluster means for each cluster.
- Maximize the distance between clusters.

In unsupervised training, there is no prior knowledge of the number of clusters. Therefore the cluster validity is determined by the objective functions. In the algorithm, the Davies-Bouldin index is used as the means to evaluate the result of each iteration.

3.1 Cluster Validity – Davies-Bouldin Index.

The accuracy or validity of the classification results need to be measured using certain criteria. As a prerequisite, a set of objects needs to possess a natural group structure. In our image classification algorithm outlined in section 5, the Davies-Bouldin (DB) index is used as the aid in parameter tuning. Our objective function is to minimize the DB index, since a smaller index value indicates compact and well-separated clusters. The similarity index between two clusters C_i and C_j can be expressed as [17]

$$R_{ij} = \frac{s_i + s_j}{d_{ij}} \tag{3.5}$$

where s_i and s_j are a measure of distance within a cluster, and d_{ij} is the distance between cluster i and j. The s_i is defined as [17]

$$s_i = \left(\frac{1}{n_i}\sum_{x\in C_i}\left\|\mathbf{x}-\mathbf{m}_i\right\|^r\right)^{1/r} \tag{3.6}$$

where n_i is the number of pixels in the cluster C_i .The distance between two clusters d_{ij} is defined as [17]

$$d_{ij} = \left(\sum_{k=l}^{l}\left|m_{ik} - m_{jk}\right|^q\right)^{1/q} \tag{3.7}$$

where l is the number of clusters and m represents the mean distance.
Let R_i be defined as [17]

$$R_i = {}_{j=1,\ldots,m,\, j\neq i} \max R_{ij}, \qquad i = 1,\ldots,m \tag{3.8}$$

Then the DB index is defined as [17]

$$DB_m = \frac{1}{m}\sum_{i=1}^{m} R_i \tag{3.9}$$

4. The K-means algorithm for multispectral image classification

The K-means algorithm is one of the simplest and most efficient unsupervised learning algorithms to solve clustering problems in image segmentation. In this algorithm, random cluster means are assigned and repeatedly modified throughout the process in order to minimize the squared error function. Suppose there are N pixels in an image to be classified into m clusters. Each pixel v_i , where $1 \le i \le N$, is assigned to one of the clusters c_j , where $1 \le j \le m$, based on squared Euclidean distance of each pixel to each cluster mean.

$$d(v,c) = \sum_{i=1}^{N}\sum_{j=1}^{m}(v_i - c_j)^2 \tag{3.10}$$

Upon the completion of the assignment, each new cluster mean is calculated using

$$c_j = \frac{\sum_{v\in c} v_i}{n} \tag{3.11}$$

where $1 \le i \le N$, and n is the number of pixels in cluster c_j. The process ends when c_j stabilizes.

The weakness of K-means is that it is dependent on the initial selection of the cluster means and it may be trapped into locally optimal results. However, running the algorithm repeatedly and randomly selecting different sets of cluster means may offset the problem. In a paper by Hung and Germany [19] it is shown that the local optimal results may also be avoided by assigning the cluster means based on distribution of patterns in histogram of an image.

5. Multispectral Image Classification using Rough Set Theory

Multi-spectral images can be analyzed using rough set theory. However, since all the attribute values are analog, the discretization process is required. Multispectral images contain multiple bands, for example the RGB color band.

Object Index	R	G	B	Class
u_1	149	148	143	1
u_2	154	155	150	1
u_3	159	160	155	1
u_4	174	171	164	2
u_5	164	161	154	2
u_6	179	183	186	3
u_7	159	165	163	3
u_8	178	184	182	3

Table 5. Decision System Table for multi-spectral image.

The values of the condition attributes are obtained from the image data shown in figure 1, while the values of the decision attributes are obtained from 'ground truth' data..
Each object has three condition attributes, Red (R), Green (G) and Blue (B) which are associated with a decision attribute. The decision attributes signify the following:

1 Class 1 represents land
2 Class 2 represents village
3 Class 3 represents water.

The value of each attribute ranges from 0 to 255, hence the training data from Table 5 can be expressed as:

V_R={0, 149, 154, 159, 164, 174, 178, 179, 255}
V_G={0, 148, 155, 160, 161, 165, 171, 183, 184, 255}
V_B={0, 143, 150, 154, 155, 163, 164, 182, 186, 255}

Based on the above intervals, the following set of cuts are obtained.

For the R attribute:
(r, 151.5); (r, 156.5); (r, 161.5); (r, 169); (r,176); (r,178.5)
For the G attribute:
(g, 151.5); (g, 157.5); (g, 160.5); (g, 163); (g, 168); (g, 177); (g, 183.5)
For the B attribute:
(b, 146.5); (b, 152); (b, 154.4); (b, 159); (b,163.5); (b, 173); (b, 184)

The optimal set of cuts needs to be selected now. There are many ways to perform the selection. For decision table $\mathcal{A}$ = (U, A $\cup$ {d}), a local method can be used as [1]:

Input: The consistent decision table $\mathcal{A}$.
Output: The semi-minimal set of cuts $\mathcal{D}$ consistent with $\mathcal{A}$.
Method: Initialize the binary tree variable T with the empty tree. Label the root by the set of all objects U and fix the status of the root to be unready.

```
while there is a leaf marked by unready do
    begin for any unready leave N of the tree T
        begin
            if objects labeling N have the same decision value then
            begin
                Replace the object set at N by its common decision
                Change the status of N to ready.
            end
            else
            begin
                Compute the value $W^N(a,c)$ for all cuts from $C_A$ and search for cut
                $(a^*, c^*)$ maximizing the function $W^N(.)$ i.e.
                $(a^*,c^*) = \arg \max_{(a,c)} W^N(a,c)$
                Replace the label of N by $(a^*, c^*)$ and mark it as ready;
                Create two new nodes N1 and N2 with status unready as the left and right
                subtrees of N, where:
                $N_1 = \{u \in N : a^*(u) < c^*\}$ and
                $N_2 = \{u \in N : a^*(u) \geq c^*\}$
            end
        end
    end
return T
```

By applying the algorithm above to the image data as shown in Table 6, the following details are derived. For each cut of the R, G and B attributes, we find the cut that yields the maximum number of pairs. The search gives us (g, 160.5) as the optimal solution which yields 15 pairs.

The cut (g, 160.5) divides the set into two, $X_1 = \{u_1, u_2, u_3\}$ and $X_2 = \{u_4, u_5, u_6, u_7, u_8\}$. Notice that X_1 actually consists of objects of the same class, so the search ends. The search continues for X_2. Three sets of cuts are found from the R, G and B attributes for X_2. All of the cuts, (r, 176), (g, 177) and (b, 173) yield the same number of objects (4 pairs). We only need to select one, and the one chosen is (r, 176). Again, this cut divides the set into two, $Y_1 = \{u_4, u_5, u_7\}$ and $Y_2 = \{u_6, u_8\}$. Y_2 consists of objects of the same class, so the search ends. The search continues for Y_1. The cut that can discern the most from Y_1 is (r, 161.5).
The cut (r, 161.5) divides Y_1 into two sets, $Z_1 = \{u_4, u_5\}$ and $Z_2 = \{u_7\}$. The search ends since both sets contain objects of the same class.

The set of cuts selected are:

(r, 161.5); (r, 176.0)
(g, 160.5)

It appears that our data set only requires two attributes to be fully discerned. Note that different discretization methods will obtain different results. For example, if a naïve algorithm was used, the B attribute will be considered in generating the cuts.

Using the cuts, a discretized table is subsequently generated. The asterisk in the B column indicates that it is not needed to discern the classes. This, however, will not be the case when the training set grows larger.

Object Index	R	G	B	Class
u_1	0	0	*	1
u_2	0	0	*	1
u_3	0	0	*	1
u_4	1	1	*	2
u_5	1	1	*	2
u_6	2	1	*	3
u_7	0	1	*	3
u_8	2	1	*	3

Table 6. Discretized Decision System Table for multispectral image.

Based on Table 6, the following rules are generated:
1 If (g < 160.5), then d = 1.
2 If (161.5 ≤ r < 176), then d = 2.
3 If (r ≥ 176), then d = 3.
4 If (r < 161.5) and (g ≥ 160.5), then d = 3.

6. The Hybrid Rough K-means Algorithm and Particle Swarm Optimization for Multispectral Image Classification

The K-means clustering method is categorized as a hard clustering method. Using K-means to classify images that have obscured or blurred boundaries will not bring a satisfactory result. There are many methods proposed to deal with this. The fuzzy C-means [22] and genetic K-means [23] algorithms are two examples.

Rough K-means is a recently proposed method that deals with the coarseness of the information. In gray image classification the challenge is on segmenting the blurred boundaries between clusters. Using rough sets theory, an image can be represented as sets of lower and upper approximation. The rough K-means model for our proposed image segmentation algorithm is adapted from [20].

$$c_j = \begin{cases} w_{lower} * \dfrac{\sum_{v \in \underline{A}(x)} v_j}{|\underline{A}(x)|} + w_{upper} * \dfrac{\sum_{v \in (\overline{A}(x) - \underline{A}(x))} v_j}{|\overline{A}(x) - \underline{A}(x)|}, & if \quad \overline{A}(x) - \underline{A}(x) \neq \varnothing \\ w_{lower} * \dfrac{\sum_{v \in \underline{A}(x)} v_j}{|\underline{A}(x)|}, otherwise \end{cases} \tag{6.1}$$

Each image pixel can be classified into lower or upper approximations. Following basic rough set properties:
- A pixel can be part of only one lower approximation

- If a pixel is part of a lower approximation, then it is also part of the upper approximation
- If a pixel does not belong to any lower approximation, then it belongs to two or more upper approximations.

Applying rough set into K-means requires the formula to include lower and upper approximations. The formula, as shown below, includes the weighing factor w_{lower} and w_{upper}. Let v be a pixel vector and $d(v, c_i)$ be the distance between the pixel and the mean of cluster i. Let

$$d(v,c_i) = \min_{1 \le j \le k} d(v,c_j) \tag{6.2}$$

and

$$T = \{j : d(v,c_i) - d(v,c_j) \le \mathit{threshold} \quad \mathit{and}\, i \ne j\} \tag{6.3}$$

In order to correctly classify a pixel, the following *classification criteria* are used:

1. If T is not an empty set, then the pixel is classified as an upper approximation of both clusters i and j.
2. If T is an empty set, the pixel is classified as a lower approximation for cluster i. It will also be classified as an upper approximation for cluster i.

To summarize, the following are steps to perform the rough K-means algorithm [26]:

1. Initialize K clusters randomly.
2. Select w_{lower} and a threshold value.
3. For each cluster, find d using Equation 6.2 and T using Equation 6.3.
4. Classify the pixel using the *classification criteria*.
5. Calculate the new cluster center (mean) using Equation 6.1.
6. If every cluster converges, then stop. Otherwise, repeat step 3.

The parameters involved are w_{lower}, w_{upper} and the threshold. The sum of w_{lower} and w_{upper} will always be one. These parameters are set manually by trial and error. Since it is not trivial to come up with good parameter values, this is the major disadvantage for this method. In order to adjust these parameters automatically, this algorithm needs to be improved using automatic tuning mechanism. The PSO algorithm alleviates the limitation by automatically searching and modifying the parameters during the image segmentation process.

The proposed algorithm that combines rough K-means and PSO algorithm is outlined as follows [26]:

1. Initialize the mean of each cluster.
2. Initialize a number of particles where each of the particles is randomly assigned with w_{lower} and the threshold.
3. Find the minimum pair of distance of x to all clusters, $d(x\text{-}c_i)$. Then assign the pixel according to the following criteria.
    - If the difference of the distance $d(x\text{-}c_i)$ – $d(x\text{-}c_j)$ is less than the threshold, then the pixel belongs to upper approximation of both clusters c_i and c_j.
    - Otherwise, the pixel x belongs to lower approximation of cluster c_i.

4. Calculate the DB index of each particle. Save the DB index of each particle and compare them with those of other particles. Find the global best index and tune the lower approximation and thresholds of each particle according to the following guidelines.
    - If the personal best DB index equals the global best DB index, then lower the threshold so that it includes only the pixels that are definitely in the lower approximation.
    - If the personal best DB index is greater than the global best DB index, then adjust the w_{lower} and the threshold toward the particle with the global best DB index.
5. Calculate the new mean for each cluster.
6. Repeat steps 3, 4 and 5 until all particles converge.

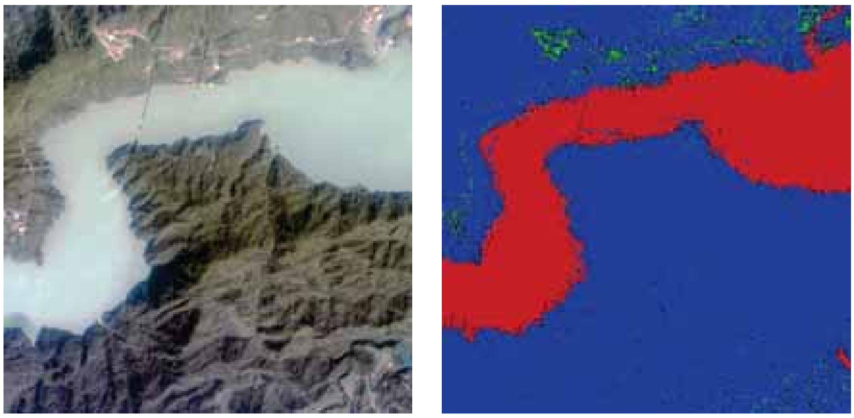
7. Experimental Results

To test the effectiveness of the proposed algorithms, multispectral and artificial images were used in our experiments. The original image is processed to obtain the multispectral information. Then the Rough Set Exploration System (RSES) software was used to process the image data [24]. A selected percentage of the image pixels were sampled for training purpose. Finally MATLAB was used to make the results viewable as an image. Experimental results are described in section 7.1. The experiment on the rough K-means algorithm is intended to show the effect of parameter selection on the results of the classification. Experimental results on the algorithm are shown in section 7.2.

7.1 Experimental Results on the Rough Set Theory

Due to the size of the table in our training sample (in the range of over 80,000 pixels), we need to resort to the decomposition tree feature of the RSES. This feature allows us to break the table into sections no larger than a predefined size. In this case, a size of 500 samples is selected as the maximum size of each leaf in the decomposition tree. These methods are further elaborated in [3] and [4]. During the decomposition process, the table is also discretized. A local method like the one outlined in Section 2 is chosen as the method for selecting the optimal cuts. Each leaf of the decomposition tree contains a set of rules that was dynamically created. The rules are then used to classify the unseen objects. Using RSES, there are two formats of output that the user can select: confusion matrix or classification results in table format.

After applying the rules to the pixels and obtaining the classification result, the reverse process is done using MATLAB to get the classified image. All pixels, including the unclassified ones, are assigned a specific color for visualization. The original image as shown in Figure 1(a) is a terrain image that has land, water and village. After the classification using rough set theory, the classified result is obtained in Figure 1(b). The confusion matrix with an average accuracy of 0.79 is shown in Table 7.



(a) (b)

Fig. 1. (a) An original satellite image, and (b) the classified result using the rough set theory. The color blue represent land, the red represents water and green represents village. Undefined objects are left as black pixels.

	Classified			
Actual		1	2	3
	1	181,838	2,264	1,052
	2	960	1,631	90
	3	2,083	124	67,381
	True Pos Rates	0.98	0.41	0.98

Table 7. Rough set classification accuracy assessment.

With the parallelepiped classification algorithm [25], the ordering of the classes affects the final result. The experimental results are shown in Figure 2. First we show the result of classification, where the order of classes are 1, 2, and 3 (respectively land, village and water). It is apparent that the RGB spectral signatures for village and water overlap. Since the order of analysis begins with village (2), most pixels of water (3) were classified as village. The confusion matrix for the results in Figure 2 is shown in Table 8 with an average accuracy of 0.6.

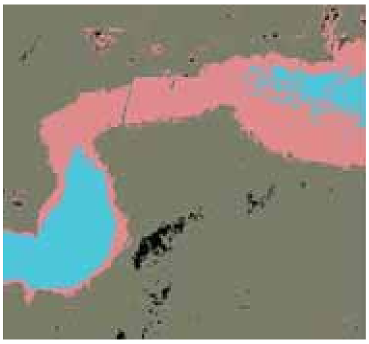


Fig. 2. A classification result of Figure 1(a) using the parallelepiped method.
Ordering of the classification for classes 1, 2, 3 (land, village and water)

	Classified			
Actual		1	2	3
	1	188,043	38	0
	2	2,113	1,305	0
	3	361	40,661	29,626
	True Pos Rates	0.99	0.38	0.42

Table 8. Parallelepiped classification accuracy assessment.
The classification ordering is class 1, 2, and 3 (land, village, and water)

The experiment is repeated for the parallelepiped classifier. The ordering is now started with classes 1, 3 and 2 (respectively land, water and village). Contrary to the result in Figure 2, now most pixels of the village area are classified as water. The confusion matrix for the result in Figure 3 is shown in Table 9 with an average accuracy of 0.72. The increase in the average accuracy, because of the misclassification of village, is mitigated by the number of its pixels overall.

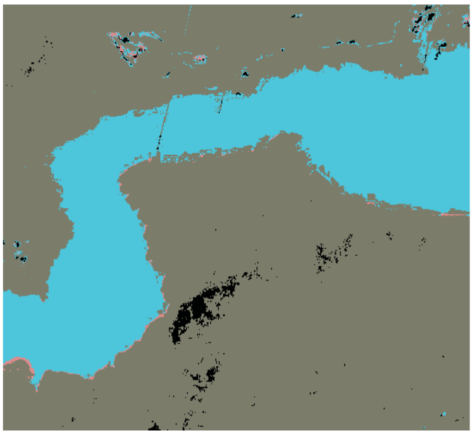


Fig. 3. A classification result of Figure 1(a) using the parallelepiped method. ordering 1,3 ,2

	Classified			
Actual		1	3	2
	1	188,043	38	0
	3	361	70,237	0
	2	2,113	689	616
	True Pos Rates	0.99	0.99	0.18

Table 9. Parallelepiped classification accuracy assessment.
The classification ordering is class 1, 3, and 2 (land, water, and village)

The following experiment requires ground truth data for accuracy assessment. The remote image sensing truth data was obtained from Dr. Su in the National Central University in Taiwan, while the ground truth data for the artificial images were created using custom software written in Java. The decision rules, which are required for classification of the image, are facilitated by RSES [24]. The process for remotely sensed images begins by sampling 30% of the image pixels as training data to create decision rules. The process to create decision rules follows the outline in Section 2. After obtaining the rules with RSES, they are used to classify the image. The image consists of approximately 262,000 pixels. Referring to Figure 4(a), class 1 is land, class 2 village and class 3 water. The average accuracy for the classification is 79%. The confusion matrix is shown in Table 10. The most difficult pixels to classify are the village pixels, as indicated by the small value of its true positive rate. The ground truth data and classified result are shown in Figure (b) and (c).

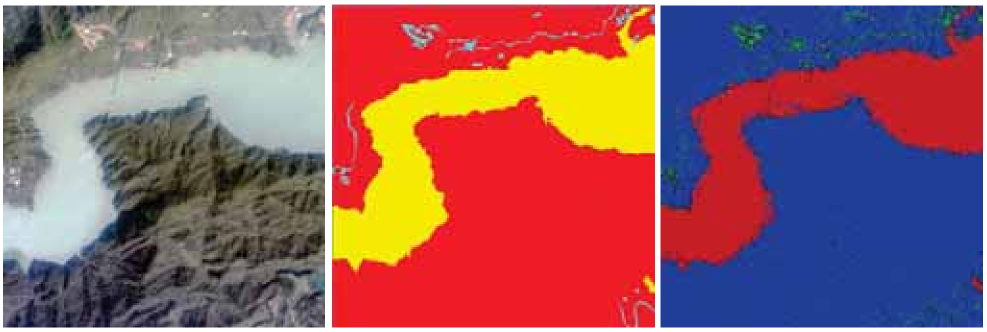
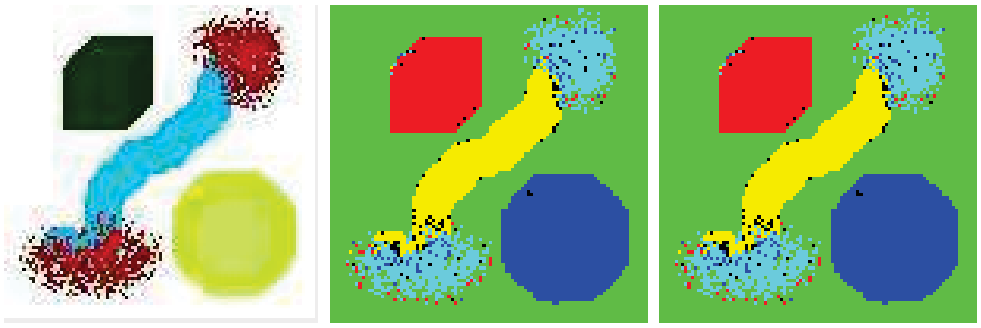


Fig. 4. (a) An original remote sensing image, (b) Ground Truth Data, and (c) Classification result.

	Classified			
Actual		1	2	3
	1	181,838	2,264	1,052
	2	960	1,631	90
	3	2,083	124	67,381
	True Positive Rates	0.98	0.41	0.98

Table 10. Rough set classification accuracy assessment for remote sensing image.

The other experiment is performed on the artificial image that consists of several shapes, namely, a cube, a serpentine, two airbrush shapes and a round shape (Figure 5). Similarly, 30 % of the pixels in the image are used for training. After obtaining the decision rules, the image is classified. The artificial image has a total of 10000 pixels. Referring to Table 11, class 1 is the cube, class 2 is the connector of the airbrush images, class 3 is the airbrush images, class 4 is the round shape and class 5 is the background. Some difficulties occur while trying to obtain the ground truth, due to the inherent limitations of the image processing software. The results however, indicate that class 3, the airbrush shapes, has the most incorrectly identified pixels. The total accuracy is still about 99% as shown in Table 11.



(a) (b) (c)

Fig. 5. (a) Original artificial shapes (b) Image truth (c) Classification result

	Classified					
Actual		1	2	3	4	5
	1	833	0	0	0	1
	2	5	884	3	7	3
	3	0	13	907	3	2
	4	0	3	0	1,434	8
	5	0	0	2	3	5,878
	True Positive Rate	0.99	0.98	0.99	0.99	1

Table 11. Rough set classification accuracy assessment for artificial shapes.

7.2 Experimental Results of the Hybrid Rough K-Means and PSO

In the experiment shown in Figure 6.1 (b), the parameter w_{lower} is set to 0.55 and the threshold 0.45. The first parameter weights how much the previous calculated mean will affect the new mean and the second parameter adjusts the boundary region. In other words, the second parameter is the criteria limiting whether a pixel should be included in the upper approximation of a class. The higher the value of the threshold, the more less the criteria is constrained.

The experiment was done for several different combinations of w_{lower} and the threshold value. After careful inspection on the results shown in Figure 6(b) through Figure 6(f), it turns out that a monotonic increase or decrease of w_{lower} and the threshold does not guarantee improvement in the classification results. From Figure 6(b) to (c), the accuracy decreased. Although the threshold was reduced, the boundary area between land, village and river actually turns blurred. In the result of Figure 6(d) the accuracy improves. Also, from Figure 6(d) to (e) the accuracy decreases again, although not as badly as between Figure 6(b) to (c). The accuracy improves again in the results of Figure 6(f). These are strong indications that varying the parameters (w_{lower} and the threshold) do not guarantee that the best results can be predicted easily. As a matter of fact, the most optimal parameters can only be found empirically. This is exactly the shortcoming of the rough K-means algorithm and the problem is addressed using PSO to tune the parameters.

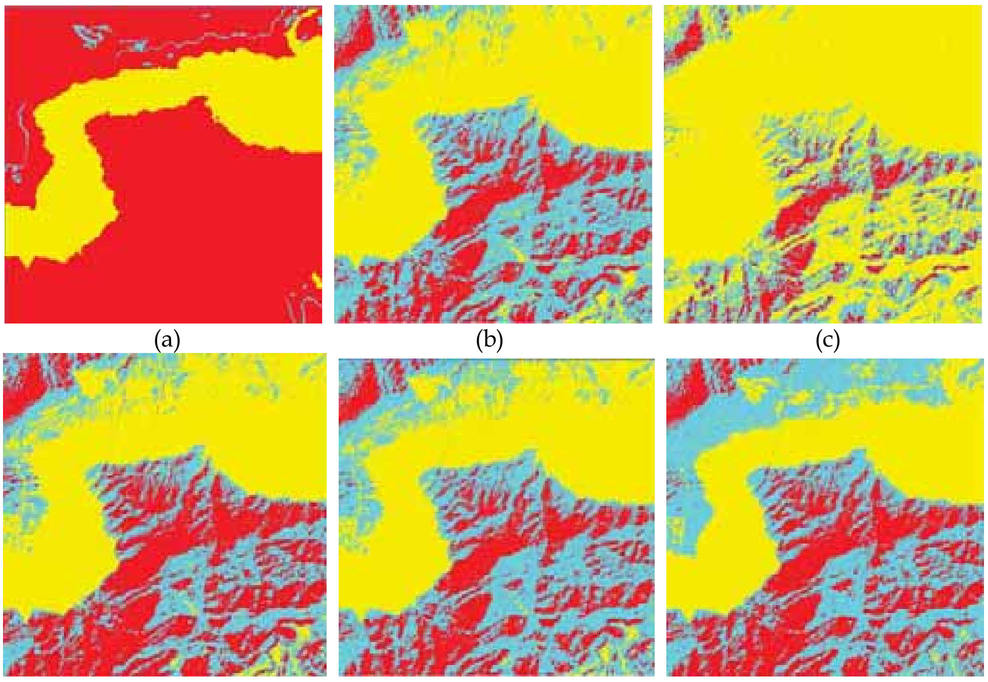


Fig. 6. Rough K-Means for remote sensing image. (a) Ground Truth, (b) w_{lower}= 0.55 and Threshold = 0.45 Accuracy = 44.95 %, (c) w_{lower} = 0.65 and Threshold = 0.35 Accuracy = 38.54 %, (d) w_{lower} = 0.75 and Threshold = 0.25 Accuracy = 49.18 %, (e) w_{lower} = 0.85 and Threshold = 0.15 Accuracy = 46.62 %, and (f) w_{lower} = 0.95 and Threshold = 0.05 Accuracy = 51.42 %

Results with similar consistency are obtained for the image in Figure 7. From Figure 7(b) to 7(c), we can see improvement visually. As the parameters change in one direction, the accuracy drops as shown in Figure 7(d). Finally the best value for the experiment is shown in figure 7(f). Looking at those classified results, it may lead us into thinking that increasing the w_{lower} and decreasing the threshold gives a better result. That is not necessarily the case, since doing so means that we are counting on the lower bound more and reducing the threshold value, while at the same time discounting the upper bound. At the extreme, where w_{lower} is almost 1 and threshold is almost zero, roughness is actually removed, and the set becomes crisp. This is also formulated in Equations 6.1, 6.2 and 6.3 earlier.

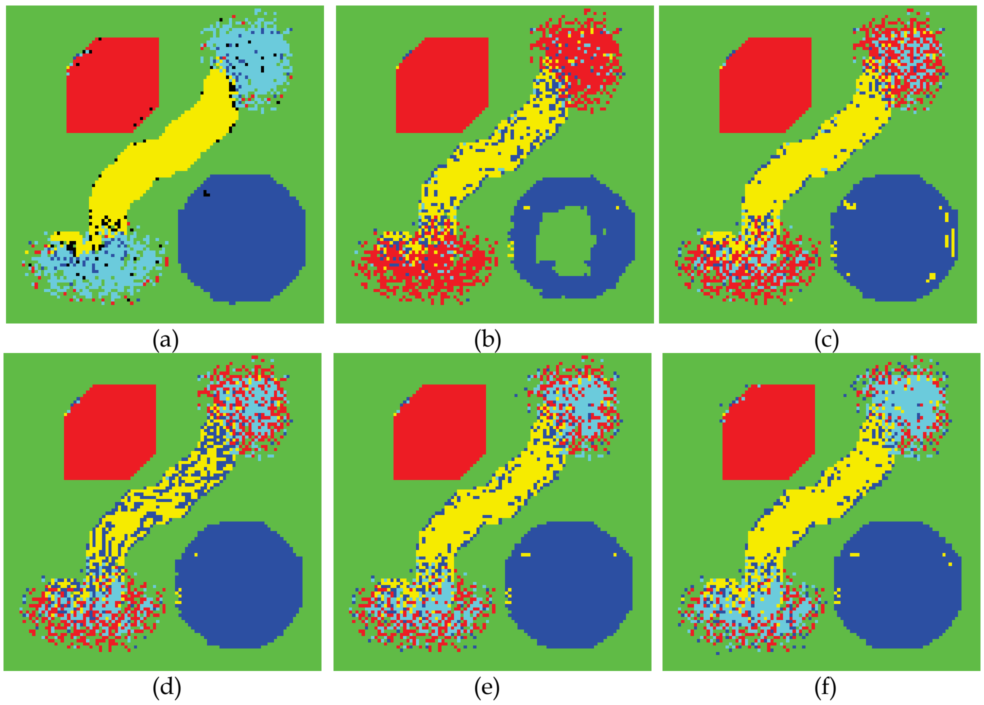


Fig. 7. Rough K-Means for artificial image. (a) Ground Truth, (b) w_{lower} = 0.55 and Threshold = 0.45 Accuracy = 68.17 % (c) w_{lower} = 0.65 and Threshold = 0.35 Accuracy = 82.83 %, (d) w_{lower} = 0.75 and Threshold = 0.25 Accuracy = 78.92 % (e) w_{lower} = 0.85 and Threshold = 0.15, Accuracy = 85.11 %, and (f) w_{lower} = 0.95 and Threshold = 0.05, Accuracy = 89.54 %.

Experiments using the K-means and rough K-means PSO algorithms are performed. For the comparison, the number of iteration is limited to 50 and the tolerance is set to 0.001. The result shown in Figure 8 is selected from the best outcome of 20 runs of the K-means and rough K-means algorithms. For the rough K-means PSO, 10 particles are used to explore the search space. Comparing the results of the K-means, rough K-means and rough K-means PSO algorithms, it reveals that although the improvement can be made, it is in the order of more or less 5 %. It is not very significant, but we should note that the rough K-means PSO achieve the optimal results independent of initial mean selections.

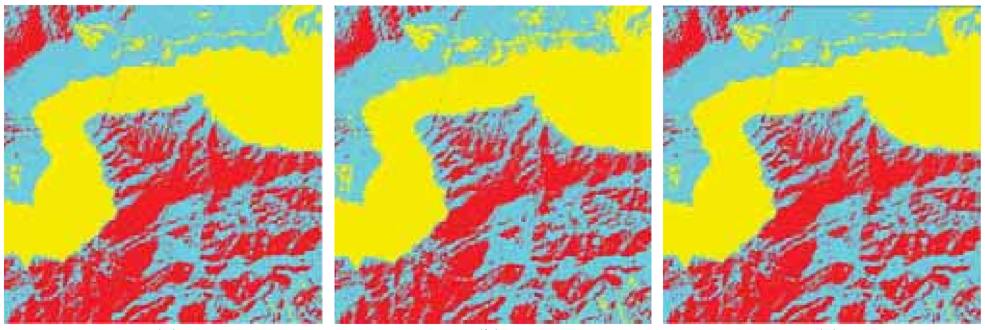


Fig. 8. Comparison of the results. (a) K-Means Accuracy = 56.26 %, (b) Rough K-Means w_{lower} = 0.95 and Threshold = 0.05, Accuracy = 51.42 %, and (c) Rough K-Means PSO, Accuracy = 59.96 %.

While running, the algorithm is tuned by keeping track of the DB index and adjusting the PSO particle accordingly to calculate the new mean. Figure 9 shows the DB index tracking for Figure 8(a) and 8(c).

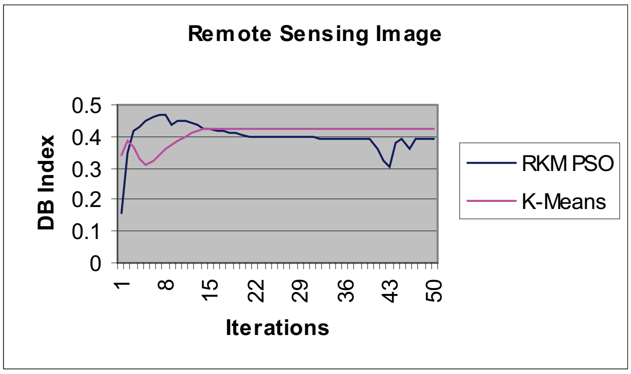


Fig. 9. K-means and rough K-means PSO DB index tracking for Figure 8(a) and 8(c).

Referring to Figure 9, it is apparent that the K-means algorithm eventually converges and locks into a certain mean value. The rough K-means PSO shows a better capability to search for solutions, because there are about 10 particles to keep track of global best and adjust the velocity towards the best solution in every iteration. Similar results are obtained from the remaining tests performed. The resulting improvement, however, is not as obvious as those shown in the artificial image (Figure 10). Part of the reason is because the artificial image does not have enough roughness. Hence, it is not difficult for K-means to perform well in this case. Figure 11 shows the tracking of the DB index.

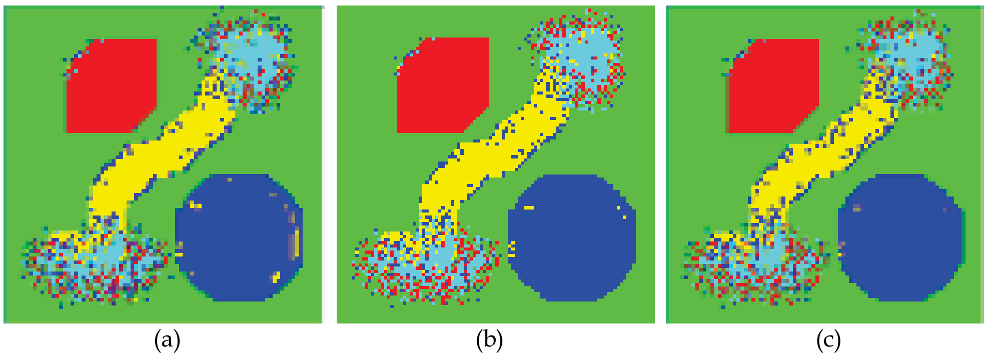


Fig. 10. Comparison of the results. (a) K-means Accuracy 90.12 %, (b) Rough K-means w_{lower} = 0.95 and Threshold = 0.05, Accuracy = 89.55 %, and (c) Rough K-means PSO, Accuracy = 90.65 %

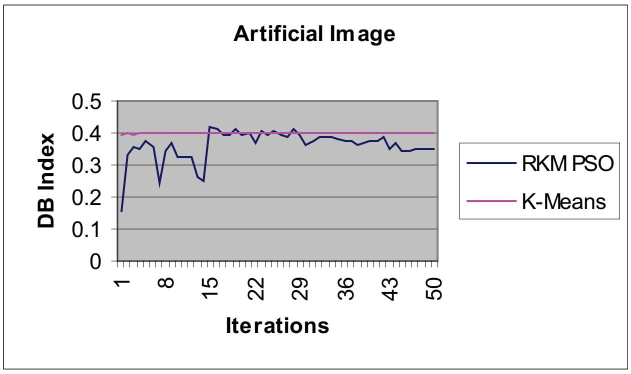


Fig. 11. The K-means and rough K-means PSO DB index tracking for the artificial image shown in Figure 10.

For the planet image, there is no ground truth data available. However, the visual inspection reveals improvement. Based on the results shown in Figure 12, we can see that the K-means algorithm actually has some difficulty in the segmentation of the blurred or rough boundaries. The rough K-means PSO however, appears to be able to discern the rough boundaries, and therefore comes up with a much more rounded shape for the planet. The outer shape of the planet appears sharper, more rounded and less distorted. Figure 13 shows the DB index tracking of the planet image.

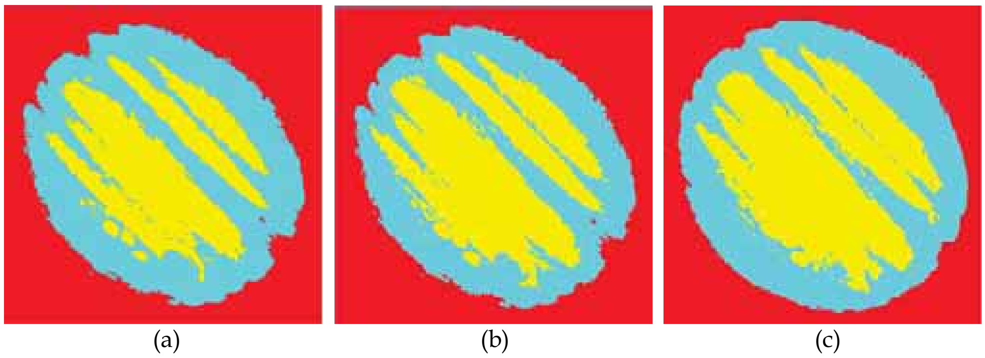


Fig. 12. Experimental results for planet image, (a) K-means, (b) rough K-means w_{lower} = 0.95 Threshold = 0.05, and (c) rough K-means PSO.

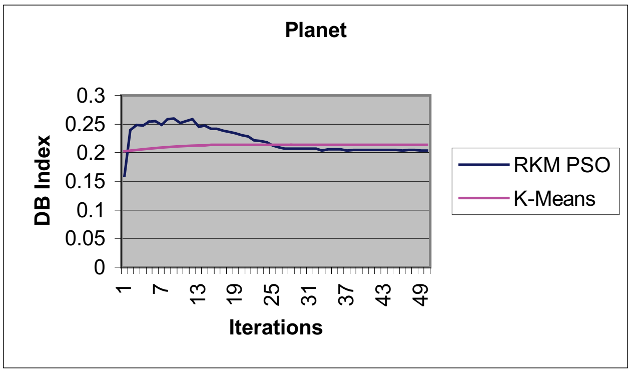


Fig. 13. The K-means and rough K-means PSO DB index tracking for the planet image shown in Figure 12.

8. Conclusions and Future Work

Image classification and segmentation by applying rough set theory may be approached from two different perspectives: unsupervised or supervised methods. From the experiment, it is generally shown that a supervised classification achieves better results as compared with the unsupervised methods. However, it should be noted that unsupervised classification may be preferred because it requires less prior knowledge. The K-means algorithm can be enhanced by using the rough set theory for image classification, however it has a practical limitation by itself, since the parameters (w_{lower} and the threshold value) are difficult to tune manually. To solve this problem, the PSO is used for tuning these parameters. The algorithm is tested on several and in general its improved noise immunity can be seen in the results especially when the images have rough boundaries or noisy details. For future work, the ant colony optimization and differential evolution algorithms will be explored for tuning the parameters in the rough K-Means algorithm.

9. References

[1] Bazan, J., H.S. Nguyen, S.H. Nguyen, P. Synak, and J. Wroblweski, "Rough Set Algorithms in Classification Proble," In L. Polkowski, S. Tsumoto, and T. Lin (Editors), Rough Set Methods and Applications. New York, Physica-Verlag Heidelberg New York (2000): pp 49-88.

[2] Bazan, J. and M. Szczuka, "The Rough Set Exploration System," In J. Peter and A. Skowron (Editors), Transactions on Rough Sets III. Springer (2005): pp 37-55.

[3] Bazan, J.and M. Szczuka, "RSES and RSESlib - A collection of tools for Rough Set Computations," Lecture Notes in Artificial Intelligence 3066, Heidelberg, Springer (2000): pp. 592-601.

[4] Kim, D., "Data classification based on tolerant rough set." Pattern Recognition, 34 (2001): pp. 1613-1624.

[5] Komorowski, J., Z. Pawlak, L. Polowski, and A. Skowron, "Rough Sets: A Tutorial," In S. K. Pal and A. Skowron (Editors), Rough Fuzzy Hybridization. Springer (1999): pp 3-98.

[6] Lillesand, T. M. and R. W. Kiffer, Remote Sensing and Image Interpretation (3rd ed), John Wiley & Sons, Inc., 1994.

[7] Lingras, P., "Unsupervised Rough Set Classification Using GAs," Journal of Intelligent Information Systems, 16. (2001): pp 215-228.

[8] Nguyen, H. S., Data Regularity analysis and applications in data mining, Ph.D thesis, Supervisor B. Chlebus, Warsaw University, 1999.

[9] Pawlak, Z., "Rough Set," International Journal of Information and Computer Science, 11. (1982): pp 341-356.

[10]Pawlak, Z., J. Grzymala-Busse, R. Slowinski and W. Ziarko, "Rough Sets," Communication of ACM, 38 (1995): pp 89-95.

[11] Schowengerdt, R. A., Remote Sensing: Models and Methods for Image Processing (2nd ed), Academic Press, 1997.

[12] Qin, Z., G. Wang, Y. Wu, and X. R. Xue, „A Scalable Rough Set Knowledge Reduction Algorithm," Heidelberg , Springer - Verlag, (2004): pp 445-454.

[13] Petrosino, A. and G. Salvi, "Rough Fuzzy set based scale space transforms and their use in image analysis." International Journal of Approximate Reasoning, 41 (2006): pp 212-228.

[14] Seul M., L. O'Gorman, and M. J. Sammon, Practical Algorithms for Image Analysis, Cambridge University Press, 2000.

[15] Sonka M., V. Hlavac, and R. Boyle, Image Processing, Analysis, and Machine Vision (2nd ed), Brooks/Cole Publishing Company, 1999.

[16] Das S., A. Abraham and S. K. Sakar, "A Hybrid Rough Set – Particle Swarm Algorithm for Image Classification." Hybrid Intelligent System 6th Conference, 6 (2006): pp 26.

[17] Theodoris S. and K. Koutroumbas, Pattern Recognition (3rd Ed), Academic Press, 2006.

[18] Jang Q. and S. S. R. Abidi, "A hybrid of conceptual clusters, rough sets and attribute oriented induction for inducing symbolic rules," Machine Learning and Cybernetics Conference, 9 (2005): pp 5573 – 5578.

[19] Hung, C. C. and G. Germany, "K-means and Iterative Selection Algorithms in Image Segmentation," in proceedings of IEEE Southeast Conference, Session 1: Software Development, Jamaica, West Indies, April 4 - 6, 2003.

[20] Lingras, P and C. West, "Interval Set Clustering of Web Users with Rough K-Means," Journal of Intelligent Information Systems, 23.1 (2004): pp 5-16.

[21] Kennedy, J and R. C. Eberhar, “Particle Swarm Optimization,” Proceedings of IEEE International Conference on Neural Networks, Perth, 1995.
[22] Pal, N. R., K. Pal, J. M. Keller and J. C. Bezdek, “A possibilistic fuzzy c-Means clustering algorithm,” IEEE Transaction on Fuzzy Systems, 13. 4 (2005): pp 517-530.
[23] Dariusz, M. and T. W. Sławomir, “Standard and Genetic *k*-means Clustering Techniques in Image Segmentation,” The 6th International Conference on Computer Information Systems and Industrial Management Applications, (2007): pp 299-304.
[24] Bazan, J., R. Latkowski, S. H. Nguyen, P. Synak, A. Wojna, M. Wojnarski and J. Wróblewski, Rough Set Exploration System Software. Version 2.2. http://logic.mimuw.edu.pl/~rses
[25] Hung, C. C., H. Purnawan, and B.-C. Kuo, “Multispectral Image Classification Using Rough Set Theory and the Comparison with Parallelepiped Classifier,” The IEEE International Conference on Geosciences and Remote Sensing (IGARSS), Barcelona, Spain, July 23-27, 2007 (CD).
[26] Hung, C. C. and H. Purnawan, “A ybrid Rough K-means Algorithm and Particle Swarm Optimization for Image Classification,” Springer-Verlag, Lecture Notes in Computer Science (LNAI 5317 pp. 585-593), MICAI 2008: Advances in Artificial Intelligence.