HYPERSPECTRAL IMAGE CLUSTERING USING A NOVEL EFFICIENT ONLINE POSSIBILISTIC ALGORITHM

Spyridoula D. Xenaki[1,2], *Konstantinos D. Koutroumbas*[1], *Athanasios A. Rontogiannis*[1]

[1]IAASARS, National Observatory of Athens, GR-152 36, Penteli, Greece
[2]Department of Informatics and Telecommunications, National & Kapodistrian University of Athens, GR-157 84, Ilissia, Greece

ABSTRACT

In this paper a novel efficient online possibilistic clustering algorithm suitable for hyperspectral image clustering is proposed. The algorithm is an online version of the recently proposed adaptive possibilistic c-means (APCM) algorithm and inherits its basic advantage, that is the ability to adapt the involved parameters during its execution in order to track variations during the clustering formation. In addition, it embodies new procedures for creating new clusters or merging existing ones. The proposed algorithm is much more computationally efficient, compared to its batch ancestor with no degradation of the quality of the resulting clustering, while also, it compares favourably with other online related algorithms. Experimental results on a real data set corroborate the effectiveness of the proposed method.

Index Terms— possibilistic clustering, online clustering, parameter adaptivity, hyperspectral imaging

1. INTRODUCTION

Hyperspectral image (HSI) classification is a challenging task that has attracted considerable attention in the remote sensing literature during the last decades. One of the issues that frequently arises in the processing of HSIs is the partial or total lack of ground truth information. A common way to deal with this issue is to resort to unsupervised classification (*clustering*), where each pixel is represented by a vector (spectral signature) containing its spectral values in the observed bands and the goal is to identify spectrally homogeneous regions in the current HSI. Applying clustering to HSIs becomes much more challenging due to (a) the high spectral correlation of contiguous bands that tend to form overlapping clusters and (b) their high dimensionality, as well as, their (usually) very large size, which increase dramatically both computation and memory requirements.

Recently, the clustering problem of HSIs has gain significant attention, e.g. [2], [3] and [4]. However, most well known such clustering schemes are of batch nature, that is, they need to process the whole data set at each iteration before updating their parameters. Thus, they are considered extremely demanding in processing large sized and high dimensional data sets, as is the case with HSIs. One way to deal with this issue is to resort to online techniques, where parameter updating takes place after the consideration of each single data vector (pixel).

In this paper, we propose a novel online clustering algorithm, called *online adaptive possibilistic c-means* (O-APCM), which is an extension of the batch APCM algorithm presented in [1]. As is usually the case in all online algorithms, data vectors (pixels) are being processed one by one and their impact is memorized to suitably defined parameters; thus O-APCM is released from the noose of storing all the hyperspectral data cube and using it at each iteration, as is the case with the batch schemes. The basic feature that O-APCM inherits from its ancestor is the adaptation of the involved parameters during its execution, which makes the algorithm flexible in tracking variations during the clustering formation. In addition, O-APCM has its own novel mechanisms that allow (a) the creation of new clusters and (b) the merging of clusters, as the algorithm evolves, where necessary. As a result, O-APCM starts with a zero number of clusters and creates or merges clusters dynamically, as pixels are processed sequentially. This makes O-APCM, in principle, able to achieve high quality clusterings on HSIs. It is worth noting that other online clustering algorithms have also been presented in the bibliography, e.g. [5], [6], [7], albeit they have not been applied for HSI processing. In addition, all of them require knowledge of the exact number of physical clusters beforehand, which is kept fixed during their execution and they employ a random initialization of cluster representatives, which, obviously, affects the final clustering result.

The rest of the paper is organized as follows. In Section 2, a brief description of APCM algorithm is given. In Section 3, the proposed online APCM (O-APCM) clustering algorithm is presented in detail. Finally, in Section 4 the prerformance of O-APCM is assessed against batch APCM and online k-means [6] in a real AVIRIS HSI data set. Concluding remarks are provided in Section 5.

This research has been financed by the PHySIS project (http://www.physis-project.eu/), contract no. 640174, within the H2020 Framework Program of the European Commission.

978-0-9928-6265-7/16/$31.00 ©2016 IEEE

2. BRIEF REVIEW OF APCM

Let $X = \{\mathbf{x}_i \in \Re^\ell,\ i = 1, \ldots, N\}$ be the set of the N pixel vectors of an HSI in the ℓ-dimensional feature space (bands) and $\Theta = \{\boldsymbol{\theta}_j \in \Re^\ell,\ j = 1, \ldots, m\}$ be a set of m vectors that will be used for the representation of the clusters formed by the pixels in X. Let $U = [u_{ij}],\ i = 1, \ldots, N,\ j = 1, \ldots, m$ be an $N \times m$ matrix whose (i, j) entry corresponds to the *degree of compatibility* of the i-th pixel $\mathbf{x}_i$ with the j-th cluster, denoted by C_j and represented by the vector $\boldsymbol{\theta}_j$. In APCM, we have that $u_{ij} \in [0, 1],\ i = 1, \ldots, N,\ j = 1, \ldots, m,\ \max_{j=1,\ldots,m} u_{ij} > 0$ and $0 < \sum_{i=1}^{N} u_{ij} < N$ and the following objective function has to be minimized:

$$J_{APCM}(\Theta, U) = \sum_{i=1}^{N}\sum_{j=1}^{m}\left[u_{ij}\|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2 + \gamma_j\left(u_{ij}\ln u_{ij} - u_{ij}\right)\right] \tag{1}$$

with respect to $\boldsymbol{\theta}_j$'s and u_{ij}'s. This leads to the following updating equations for u_{ij}'s and $\boldsymbol{\theta}_j$'s:

$$u_{ij} = \exp\left(-\frac{\|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2}{\gamma_j}\right) \quad (2) \qquad \boldsymbol{\theta}_j = \frac{\sum_{i=1}^{N} u_{ij}\mathbf{x}_i}{\sum_{i=1}^{N} u_{ij}} \quad (3)$$

with $\gamma_j = \frac{\hat{\eta}}{\alpha}\eta_j$, where parameter η_j is a measure of the mean absolute deviation of the current form of cluster C_j, $\hat{\eta}$ is a constant defined as the minimum among all *initial* η_j's, $\hat{\eta} = \min_j \eta_j$, and α is a user defined positive parameter. Loosely speaking, $\sqrt{\gamma_j}$ may be considered as a measure of the "radius" of cluster C_j (for the rationale behind the expression of γ_j's, see [1]).

The initialization of $\boldsymbol{\theta}_j$'s is carried out using the final cluster representatives obtained by the FCM algorithm, executed with m_{ini} clusters, where m_{ini} is an overestimation of the true number of clusters m ($m_{ini} > m$). Then the initialization of η_j's is carried out as follows:

$$\eta_j = \frac{\sum_{i=1}^{N} u_{ij}^{FCM}\|\mathbf{x}_i - \boldsymbol{\theta}_j\|}{\sum_{i=1}^{N} u_{ij}^{FCM}}, \tag{4}$$

where $\boldsymbol{\theta}_j$'s and u_{ij}^{FCM}'s are the final estimates of FCM.

At each iteration of APCM, η_j's are adapted, according to the following update equation:

$$\eta_j = \frac{1}{n_j}\sum_{\mathbf{x}_i : u_{ij} = \max_{r=1,\ldots,m} u_{ir}} \|\mathbf{x}_i - \boldsymbol{\mu}_j\|, \tag{5}$$

where n_j denotes the number of the pixels $\mathbf{x}_i$ that are most compatible with cluster C_j and $\boldsymbol{\mu}_j$ is their mean vector. Finally, APCM incorporates a mechanism for eliminating clusters, as described in Algorithm 1. Note that APCM is a batch clustering scheme meaning that the whole data set is used for updating the various algorithmic quantities in each iteration.

Algorithm 1 $[\Theta, H, m]$ = APCM(X, m_{ini}, α)

Input: X, m_{ini}, α

$t = 0$

▷ *Initialization phase*

Initialize: $\boldsymbol{\theta}_j(t)$ *via FCM*

Initialize: $\eta_j(t) = \frac{\sum_{i=1}^{n} u_{ij}^{FCM}\|\mathbf{x}_i - \boldsymbol{\theta}_j(t)\|}{\sum_{i=1}^{n} u_{ij}^{FCM}},\ j = 1, \ldots, m$

Set: $\hat{\eta} = \min_{j=1,\ldots,m_{ini}} \eta_j(t)$

$m(t) = m_{ini}$

repeat

▷ *Update U and Θ*

  $u_{ij}(t) = \exp\left(-\frac{\alpha}{\hat{\eta}}\frac{\|\mathbf{x}_i - \boldsymbol{\theta}_j(t)\|^2}{\eta_j(t)}\right),\ \forall i, j$

  $\boldsymbol{\theta}_j(t+1) = \sum_{i=1}^{N} u_{ij}(t)\mathbf{x}_i \Big/ \sum_{i=1}^{N} u_{ij}(t),\ j = 1, \ldots, m$

▷ *Possible cluster elimination*

  for $i \leftarrow 1$ **to** N **do**

    Determine: $u_{ir}(t) = \max_{j=1,\ldots,m(t)} u_{ij}(t)$

    Set: $label(i) = r$

  end for

  $p = 0$ *//number of removed clusters at iteration t*

  for $j \leftarrow 1$ **to** m **do**

    if $j \notin label$ **then**

      Remove: C_j (**renumber** accordingly Θ, U)

      $p = p + 1$

    end if

  end for

  $m(t+1) = m(t) - p$

▷ *Adaptation of η_j's*

  $\eta_j(t+1) = \frac{1}{n_j(t)}\sum_{\mathbf{x}_i : u_{ij}(t) = \max_{r=1,\ldots,m(t+1)} u_{ir}(t)} \|\mathbf{x}_i - \boldsymbol{\mu}_j(t)\|,\ j = 1, \ldots, m$

  $t = t + 1$

until the change in $\boldsymbol{\theta}_j$'s between two successive iterations becomes sufficiently small

return $\Theta = \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_m\}$, $H = \{\eta_1, \ldots, \eta_m\}$, m

3. THE PROPOSED ONLINE APCM METHOD

In this section, we describe in detail the proposed online APCM (O-APCM) clustering algorithm, which considers a single pixel per iteration and immediately updates its parameters. The cost function (1) in the online context becomes,

$$J(\Theta, U) = \sum_{i=1}^{t}\sum_{j=1}^{m(t)}\left[u_{ij}\|\mathbf{x}_i - \boldsymbol{\theta}_j(t)\|^2 + \gamma_j(t)\left(u_{ij}\ln u_{ij} - u_{ij}\right)\right] \tag{6}$$

where t denotes the current iteration of the algorithm, $\mathbf{x}_t$ is the current pixel to be processed and $m(t)$ is the number of clusters formed up to the t-th iteration.

3.1. Parameter initialization

As it is easily understood, the initialization of the parameters $\boldsymbol{\theta}_j$ and η_j in O-APCM cannot be impemented as in batch

APCM, due to the fact that the whole HSI cube is not available a-priori. Specifically, consider the first iteration of O-APCM, where the first cluster is created containing only the pixel $\mathbf{x}_1$. Obviously, it is not feasible to extract information about the "size" of this cluster from just a single pixel, in order to initialize its parameter η_1. Therefore, in the initialization phase of O-APCM, we run the batch APCM on a small sample of, say K, pixels (e.g. $K = 100$), with an overestimated number of clusters, m_{ini}. After its convergence, APCM ends up with m ($< m_{ini}$) number of clusters and their final estimates, $\boldsymbol{\theta}_j$'s and η_j's, are used as the current state of O-APCM. Also, we set $\hat{\eta} = \min_j \eta_j$ (as in APCM) and then we run O-APCM for the remaining pixels of the HSI cube.

3.2. Parameter adaptation - Cluster creation

In O-APCM, this part refers to (a) the computation of the degree of compatibility u_{tj}, $j = 1, \ldots, m(t-1)$ of the current pixel $\mathbf{x}_t$ with all clusters and the adaptation of all cluster representatives $\boldsymbol{\theta}_j$'s, $j = 1, \ldots, m(t-1)$, taking into consideration only the corresponding u_{tj}'s, and (b) the adaptation of the parameters η_r and $\boldsymbol{\mu}_r$ of the cluster C_r, with $u_{tr} = \max_{j=1,\ldots,m(t-1)} u_{tj}$, i.e. of the most compatible cluster to pixel $\mathbf{x}_t$.

Minimization of eq. (6) with respect to u_{tj} and $\boldsymbol{\theta}_j$, $j = 1, \ldots, m(t-1)$ results to

$$u_{tj} = \exp\left(-\frac{\|\mathbf{x}_t - \boldsymbol{\theta}_j(t-1)\|^2}{\gamma_j(t-1)}\right), \tag{7}$$

where $\gamma_j(t-1) = \frac{\hat{\eta}}{\alpha}\eta_j(t-1)$ and to the following time recursive equation for $\boldsymbol{\theta}_j(t)$,

$$\boldsymbol{\theta}_j(t) = \boldsymbol{\theta}_j(t-1) + \frac{u_{tj}}{U_j(t)}\left(\mathbf{x}_t - \boldsymbol{\theta}_j(t-1)\right), \tag{8}$$

where $U_j(t) = U_j(t-1) + u_{tj},\ j = 1, \ldots, m(t-1)$, respectively. Note that *all* cluster representatives are updated, during the processing of the current pixel $\mathbf{x}_t$. However, this is not the case with the parameters $\boldsymbol{\mu}_j$ and η_j of the clusters. Specifically, if u_{tr} is above a certain threshold, $thres$ (e.g. $thres = 1e-05$), which implies that $\mathbf{x}_t$ is not far enough from the pixels processed up to now so that to create a new cluster, only the parameters $\boldsymbol{\mu}_r$ and η_r of the most compatible to $\mathbf{x}_t$ cluster, C_r, are updated. Setting $I_{tj} = 0$, $j \neq r$ and $I_{tr} = 1$, we can easily get from (5) the following time-recursive formulas:

$$\boldsymbol{\mu}_j(t) = \boldsymbol{\mu}_j(t-1) + \frac{I_{tj}}{S_j(t)}\left(\mathbf{x}_t - \boldsymbol{\mu}_j(t-1)\right) \tag{9}$$

$$\eta_j(t) = \eta_j(t-1) + \frac{I_{tj}}{S_j(t)}\left(\|\mathbf{x}_t - \boldsymbol{\mu}_j(t)\| - \eta_j(t-1)\right) \tag{10}$$

where $S_j(t) = S_j(t-1) + I_{tj}$.

Let us now comment on the mechanism that creates new clusters, as the algorithm evolves. This procedure is activated when u_{tr} is less than $thres$. In this case, a new cluster is created, containing only $\mathbf{x}_t$, its corresponding parameters $\boldsymbol{\theta}$, $\boldsymbol{\mu}$ are set equal to $\mathbf{x}_t$, its parameters U, S are set to 1, while its parameter η is set to the minimum η among all current clusters. Finally, the number of clusters is increased by one.

3.3. Cluster merging

In online clustering schemes, the clustering result is usually dependent on the order in which the data (pixels) are processed. Therefore, depending on this, several clusters might be created, which nevertheless represent parts of the same physical cluster. Consequently, a mechanism that identifies and merges such clusters should be incorporated in an online clustering algorithm. To this end, every T iterations (e.g. $T = 100$), O-APCM considers all pairs of clusters and checks whether they exhibit overlapping; that is, it checks whether $\sqrt{\gamma_s} + \sqrt{\gamma_k} > \beta \cdot d(\boldsymbol{\theta}_s, \boldsymbol{\theta}_k)$, where $d(\boldsymbol{\theta}_s, \boldsymbol{\theta}_k)$ is the Euclidean distance between two cluster representatives $\boldsymbol{\theta}_s$, $\boldsymbol{\theta}_k$, (β controls the acceptable degree of overlapping and it is set equal to 1.1 in our case). If this is the case, these two clusters are merged into one cluster and the parameters of the newly formed cluster are defined as follows:

$$\boldsymbol{\theta}_{new} = \frac{U_s\boldsymbol{\theta}_s + U_k\boldsymbol{\theta}_k}{U_s + U_k} \tag{11}$$

$$U_{new} = U_s + U_k \tag{12}$$

$$\boldsymbol{\mu}_{new} = \frac{S_s\boldsymbol{\mu}_s + S_k\boldsymbol{\mu}_k}{S_s + S_k} \tag{13}$$

$$\eta_{new} = \frac{S_s\eta_s + S_k\eta_k}{S_s + S_k} \tag{14}$$

$$S_{new} = S_s + S_k \tag{15}$$

The basic steps of O-APCM are shown in Algorithm 2.

Algorithm 2 [Θ, Γ] = O-APCM(X, m_{ini}, α)

Input: X, m_{ini}, α

$t = 0$

▷ *Initialization*

$K = 100$, $thres = 1e-05$ **and** $T = 100$

[$\Theta(t), H(t), m(t)$]=APCM($X(:, 1:K), m_{ini}, \alpha$)

$\hat{\eta} = \min_{j=1,\ldots,m(t)} \eta_j(t)$

Set: $U_j(t) = 1$ **and** $S_j(t) = 1, j = 1, \ldots, m(t)$

while $t + K <$ DataSize **do**

  $t = t + 1$

  $\mathbf{x}_t = X(:, t + K)$

  $D_j = \|\mathbf{x}_t - \boldsymbol{\theta}_j(t-1)\|^2, j = 1, \ldots, m(t-1)$

  $u_{tj} = \exp(-\frac{\alpha}{\hat{\eta}}\frac{D_j}{\eta_j(t-1)}), j = 1, \ldots, m(t-1)$

▷ *Update cluster representatives*

  $U_j(t) = U_j(t-1) + u_{tj}, j = 1, \ldots, m(t-1)$

  $\boldsymbol{\theta}_j(t) = \boldsymbol{\theta}_j(t-1) + \frac{u_{tj}}{U_j(t)}(\mathbf{x}_t - \boldsymbol{\theta}_j(t-1)), \forall j$

  Determine: $u_{tr} = \max_{j=1,\ldots,m(t-1)} u_{tj}$

  if $u_{tr} < thres$ **then**

▷ *Create a new cluster*

$m(t) = m(t-1) + 1$
$\boldsymbol{\theta}_{m(t)}(t) = \mathbf{x}_t$
$\boldsymbol{\mu}_{m(t)}(t) = \mathbf{x}_t$
$\eta_{m(t)}(t) = \min_{j=1,\ldots,m(t-1)} \eta_j(t-1)$
$U_{m(t)}(t) = 1$ **and** $S_{m(t)}(t) = 1$
Set: $I_{tj} = 0, j = 1, \ldots, m(t)$
else
▷ *Update parameters of cluster* C_r
$m(t) = m(t-1)$
Set: $I_{tj} = 0, \forall j \neq r$ **and** $I_{tr} = 1$
end if
$S_j(t) = S_j(t-1) + I_{tj}, j = 1, \ldots, m(t)$
$\boldsymbol{\mu}_j(t) = \boldsymbol{\mu}_j(t-1) + \frac{I_{tj}}{S_j(t)} \left(\mathbf{x}_t - \boldsymbol{\mu}_j(t-1)\right), \forall j$
$\eta_j(t) =$
$= \eta_j(t-1) + \frac{I_{tj}}{S_j(t)} \left(\|\mathbf{x}_t - \boldsymbol{\mu}_j(t)\| - \eta_j(t-1)\right), \forall j$
▷ *Every* T *iterations perform the merging clusters procedure*
end while
return Θ, Γ

4. EXPERIMENTAL RESULTS

In this section, we test the proposed O-APCM method in a real HSI data set and illustrate the results. Moreover, we compare the results with those obtained from the batch APCM, as well as the online k-means (O-kmeans), presented in [6].

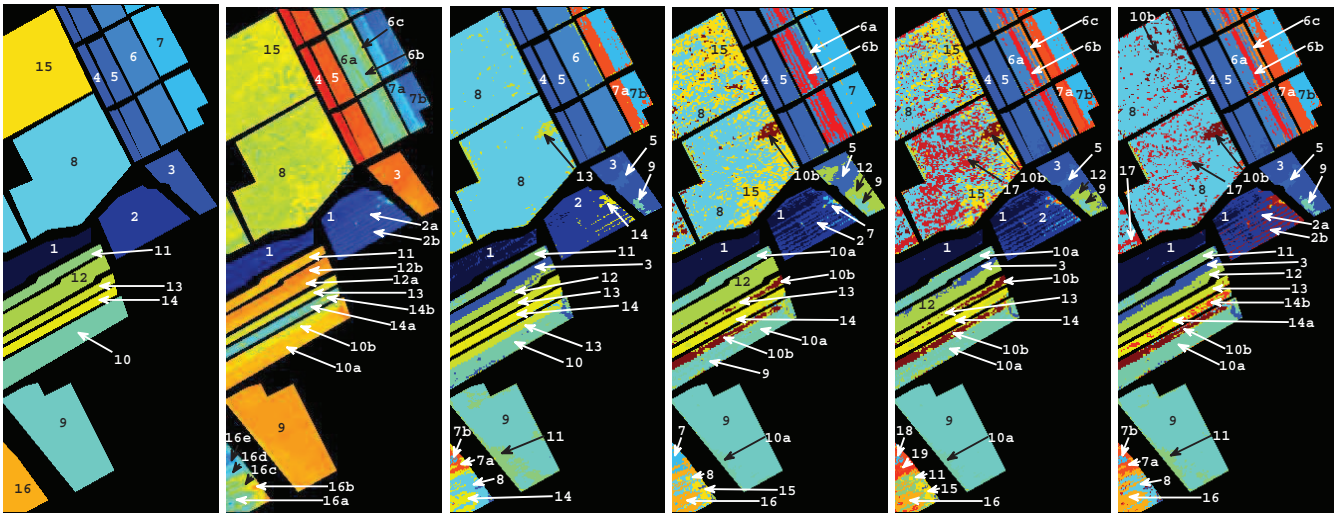
In this experiment, we consider an HSI that depicts a subscene of size 512×217 of the flightline acquired by the AVIRIS sensor over Salinas Valley, California [8]. The AVIRIS sensor generates 224 spectral bands across the range from 0.2 to 2.4 μm. The number of bands is reduced to 204 by removing 20 water absorption bands. For this HSI, only partial ground-truth information is available. Specifically, all pixels, except for those of black colour for which no ground truth is available, stem from the following 16 ground-truth classes: two types of "Broccoli" (classes 1, 2), three types of "Fallow" (classes 3, 4, 5), "Stubble" (class 6), "Celery" (class 7), "Grapes" (class 8), "Corn" (class 10), four types of "Lettuce" (classes 11, 12, 13, 14) and three types of "Vineyard" (classes 9, 15, 16), denoted by different colors in Fig. 1(a). Note that Fig. 1 depicts the best clustering obtained by each algorithm after the fine-tuning of its involved parameters.

Fig. 1(b) shows the 1st PC of the data set. As it can be deduced, there is some more detailed information not appearring in the ground truth in Fig. 1(a). Specifically, Fig. 1(b) demonstrates that (i) class 2 is formed by two distinguished from each other "subclasses" (2a and 2b), (ii) class 6 is formed by three distinguished from each other "subclasses" (6a, 6b and 6c), (iii) class 7 is formed by two "subclasses" (7a and 7b), (iv) class 10 is formed by two "subclasses" (10a and 10b), (v) class 12 is formed by two "subclasses" (12a and 12b), (vi) class 14 is formed by two "subclasses" (14a and 14b) and (vii) class 16 is formed by five "subclasses" (16a - 16e).

In order to measure the quality of clustering obtained by O-APCM, APCM and O-kmeans, we use the Success Rate (SR) metric, which measures the percentage of the points that have been correctly labeled by each algorithm with respect to the ground truth information. Table 1 shows the SR measure for all algorithms, where m_{ini} and m_{final} denote the initial and the final number of the obtained clusters, respectively. Due to the partial ground-truth labeling, SR is measured only over the pixels where ground truth class information is available. To avoid ending up with misleading decreased SR metric values, we adopt the following convention: First, in each of the 16 ground-truth classes, we identify structures (*subclasses*) with the aid of the 1st PC (Fig. 1(b)). Then, for a given cluster, produced by an algorithm, we determine the subclass to which it matches more. Then, for the computation of SR, we assign the pixels of this cluster to the class, from which the above subclass stems. Clearly, all the pixels of the cluster that do not fall in this class are considered as misclassified. In the case where a cluster does not match with any of the subclasses shown in the 1st PC (Fig. 1(b)), all of its pixels are considered as misclassified. Finally, the runtime of each algorithm is also provided in Table 1.

As shown in Fig. 1, APCM (with $m_{ini} = 30$ and $\alpha = 5$) concludes to 15 clusters. Specifically, APCM identifies correctly the two subclasses (7a and 7b) of class 7, but fails to distinguish cluster 12b from class 3 and cluster 10b from class 13. Moreover, APCM fails to uncover class 15. Online k-means (O-kmeans), when it is run with $m = 16$, identifies two subclasses (6a and 6b) in class 6 and the two subclasses (10a and 10b) in class 10. However, it fails to uncover class 3 and class 11. When O-kmeans is run with $m = 22$, it manages additionally to distinguish all subclasses (6a, 6b and 6c) of class 6 and class 3. However, O-kmeans still fails to distinguish class 11 and poorly distinguishes class 8. Finally, O-APCM (with $m_{ini} = 30$ and $\alpha = 0.6$) fails only in identifying class 15, however, it succeeds in distinguishing the subclasses (2a and 2b) of class 2, the subclasses (6a, 6b and 6c) of class 6, the subclasses (7a and 7b) of class 7, the subclasses (10a and 10b) of class 10 and the subclasses (14a and 14b) of class 14. The fact that the subclasses are not labeled as actual classes in the ground truth labeling, prevents the high discrimination ability of O-APCM, from being fully highlighted. Finally, O-APCM exhibits high robustness in the choice of parameter α.

As we can see in Table 1, O-APCM performs better than both O-kmeans and APCM in terms of SR. This may be attributed to the flexibility that O-APCM gains from its mechanisms for creating new clusters, as well as, for merging existing ones, compared to O-kmeans (where the number of clusters is fixed) and APCM (where the number of clusters may only be reduced). Compared to APCM, O-APCM is much more computationally efficient, while it is less fast than O-kmeans. However, as shown in Table 1, the performance of O-kmeans depends critically on m_{ini}, the actual value of which is unknown in most cases in practice.



(a) Ground truth (b) 1st PC (c) APCM (d) O-kmeans ($m = 16$) (e) O-kmeans ($m = 22$) (f) O-APCM

Fig. 1. (a) The ground truth of Salinas HSI, (b) the 1st PC and the clustering results of (c) APCM, (d) O-kmeans ($m = 16$), (e) O-kmeans ($m = 22$) and (f) O-APCM.

Table 1. Performance of clustering algorithms for the Salinas HSI data set.

	m_{ini}	m_{final}	$SR(\%)$	$Runtime$
O-kmeans	16	16	70.86	1.39sec
O-kmeans	22	22	65.50	1.60sec
APCM ($\alpha = 5$)	30	15	71.98	28.22min
O-APCM ($\alpha = 0.6$)	30	22	**73.78**	14.49sec

5. CONCLUSION

In this paper, a novel online adaptive possibilistic c-means clustering algorithm, called O-APCM, suitable for hyperspectral image clustering is proposed. The algorithm is an online implementation of the recently proposed adaptive possibilistc c-means (APCM) [1], and thus, it incorporates the relative parameter adaptation mechanism of the latter, which makes it capable to deal well with closely located and hardly distinguished clusters. Moreover, O-APCM embodies two new procedures for creating new clusters and merging existing ones, when necessary. As a result, O-APCM does not need a priori knowledge of the underlying number of physical clusters. In contrast, it creates or merges clusters dynamically, as data points (pixels) are considered sequentially. Due to its online nature, O-APCM requires very low computational time, compared to a batch mode implementation, where the whole data cube is considered at each iteration of the algorithm. This makes O-APCM applicable to HSI clustering even in cases where the size and the dimensionality are prohibitively increased for batch algorithms. Experimental results show that O-APCM offers high discrimination ability at a very low computational cost. The extension of O-APCM to non-stationary enviroments is a subject of current research.

REFERENCES

[1] S. D. Xenaki and K. D. Koutroumbas and A. A. Rontogiannis, "A novel adaptive possibilistic clustering algorithm", *IEEE Transactions on Fuzzy Systems*, 2015, to appear. DOI: 10.1109/TFUZZ.2015.2486806

[2] A. A. Naeini et al., "A comparison study between two hyperspectral clustering methods: KFCM and PSO-FCM", *Springer Computational Intelligence and Decision Making*, vol. 61, pp. 23-33, 2013.

[3] N. Gillis and D. Kuang and H. Park, "Hierarchical clustering of hyperspectral images using rank-two nonnegative matrix factorization", *IEEE Trans. on Geoscience and Remote Sensing*, vol. 53, pp. 2066-2078, 2014.

[4] C. Cariou and K. Chehdi, "Unsupervised nearest neighbors clustering with application to hyperspectral images", *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, pp. 1105-1116, 2015.

[5] S. Zhong, "Efficient online spherical k-means clustering", *IEEE International Joint Conference on Neural Networks (IJCNN)*, vol. 5, pp. 3180-3185, 2005.

[6] W. Barbakh and C. Fyfe, "Online clustering algorithms", *International Journal of Neural Systems (IJNS)*, vol. 18, pp. 185-194 , 2008.

[7] A. Choromanska and C. Monteleoni, "Online clustering with experts", *J. of Mach. Learn. Res.*, pp. 1-18, 2011.

[8] http://www.ehu.es/ccwintco/index.php?title=Hyperspectral Remote Sensing Scenes