Extreme Self-paced Learning Machine for On-orbit SAR Images Change Detection

Shuyuan Yang¹, QuanWei Gao¹, Zhi Liu¹, Yuteng Gao², Zhixi Feng¹

¹School of Artificial Intelligence, Xidian University, China
²Electrical Engineering Department, Northwestern Polytechnical University, China

ABSTRACT With the rapid development of earth observation satellites, on-orbit data processing technology is becoming more and more desirable. In this paper, a new on-orbit Synthetic Aperture Radar (SAR) images change detection method is proposed via a new Extreme Self-paced Learning Machine (ESLM). First, a reflectivity-spatial affinity is defined to measure the similarity between two segmented super-pixels, to identify the initial three groups of pixels: strictly changed, strictly unchanged and fuzzy pixels. Then a new extreme self-paced learning machine is developed, by gradually selecting the most confident changed pixels and predicting the changed pixels in an incremental pattern. Moreover, both the labeled and unlabeled samples are explored to realize semi-supervised classification. Different with the available methods, ESLM works in a self-paced learning pattern and achieves accurate detection, for it can automatically choose the training samples and explore unlabeled samples to enhance the online prediction of changes. Therefore, ESLM has the characteristic of accurate and robust detection, parameter free, low-complexity and rapid implementation, which is very suitable for on-orbit processing. Some experiments are taken on five real benchmark datasets, and the results verify the effectiveness of ESLM.

INDEX TERMS Change detection, synthetic aperture radar, extreme self-paced learning machine, affinity propagation super-pixel clustering, manifold regularizer

I. INTRODUCTION

Change detection from multi-temporal Synthetic Aperture Radar (SAR) images aims to identify changes in images of the same scene taken at different times [1], which has extensive applications in many civil and military fields [2]. With the rapid development on earth observation satellites in recent years, increasing amount of spaceborne SAR data are collected. Thus, it is desirable to develop on-orbit change detection techniques, which can detect changes on the aircraft and transmit only the changes to the ground station. Compared with the on-ground change detection, on-orbit detection can avoid the storage, compression and transmission of a large amount of SAR data, so is potential in coping with the explosion of remote sensing data.

Making an analysis on the on-orbit change detection technologies, we will find that they should have the following characteristics:

1) High automation degree. Different with the on-ground processing, on-orbit change detection algorithms require little involvement of human operation. In order to automatically implement on the spacecraft, on-orbit detection algorithms need to have the capability of self-learning and self-evolution, since no manual involvement is allowed in the detection process.

2) Low complexity. The spacecraft often has limitations on the power consumption and device volume. For the low power consumption and small size limitation of aerospace platform, the on-orbit detection algorithms need to have the characteristic of simple principle, rapid processing and high efficiency.

3) Robustness and flexibility. It is well known that the on-board processing needs robust and adaptive detection algorithms that can work for SAR images with different resolutions, looks and working wavebands. Consequently, developing automated, rapid and robust on-orbit methods, is the future trend of change detection technology.

Having a glance over the available multi-temporal SAR images change detection approaches, we will find that they can be mainly categorized into unsupervised and supervised ones. Unsupervised methods first calculate a “Difference
Map” (DM) and then locate the changes from it using some distribution priors [3-6] or clustering algorithms [7-9][31]. For example, the data distributions of “changed” and “unchanged” components in the map are explored by casting various kinds of priors, including generalized Gaussian distribution [3], hidden Bivariate Gamma distribution [4], Gaussian mixture distribution [5], Markov chains [6] and so on [23-25]. Then the changes are detected via statistical estimation and priors. However, these methods often cast too rigid assumption on data, and the detection results rely heavily on the validity of priors. Different with these works, some methods utilize clustering algorithms, such as K-means clustering [5][30], splitting clustering [6], fuzzy clustering [7] and mean-shift clustering [8], graph non-negative matrix factorization [31], to distinguish “changed” and “unchanged” components. However, several parameters, such as the clusters number and splitting levels, should be manually tuned. Moreover, due to the presence of speckle noises, unsupervised change detection methods always suffer from high detection error ratio and sensitivity to noises.

Supervised methods formulate the change detection as a binary classification task, and utilize some labeled data (or samples) to identify the changes from the DM [10-11]. For example, a Support Vector Machine (SVM) with a difference-kernel and a Ratio Kernel (RK) is proposed, for accurate change detection [10]. Later an iterative label-information composite kernel based classifier is proposed for change detection with the guidance of anisotropic texture [11]. Due to the use of labeled samples, supervised methods often have higher detection accuracy [28-29]. In a recent work, a relationship learning approach is proposed, which establishes a classifier to learn the relationship between the changed class and unchanged class [12]. However, both training a good classifier and learning the relationship need a lot of labeled data, which will involve much human participation and result in high labeling cost.

A recent trend in machine learning is to integrate naturalistic learning in biological species into learning, such as continuous learning and self-learning[16]. Self-paced learning is inspired by children’s learning process, whose basic idea is to establish a simple model first and then gradually learn samples from "simple" to "complex". In order to realize automatic and efficient on-orbit change detection from multi-temporal SAR images, in this paper we develop a rapid and simple unsupervised change detection method via a new Extreme Self-Paced Learning Machine (ESLM). ESLM can gradually select the confident changed pixels and incrementally predict changed pixels in a semi-supervised mode. First, the DM of two multi-temporal SAR images is calculated by a log-ratio operator. Then the DM is segmented into some connected regions named super-pixels. Second, a reflectivity-spatial affinity is defined for two super-pixels to evaluate their similarity, and an Affinity Propagation Super-pixel Clustering (APSC) algorithm is designed to automatically cluster super-pixels. Based on APSC, three groups of pixels: strictly changed pixels, strictly unchanged pixels and fuzzy pixels, are extracted from the DM, to serve as the initial training data. The flowchart of the proposed method is shown in Fig.1.

**FIGURE 1.** The framework of on-orbit change detection via ESLM

Compared with the available change detection methods, the proposed method can not only work for on-orbit processing, but also have the following characteristics: 1) ESLM works in an incremental self-paced learning pattern, which can automatically choose the training samples itself from "simple" to "complex"; 2) ESLM explores unlabeled samples to enhance the online prediction accuracy of changes. So it can not only boost the performance of unsupervised methods, but also avoid high computational complexity of semi-supervised methods; 3) a multistage clustering is used to select initial training samples, which is of high automation degree. Consequently, ESLM has the characteristic of little manual participation, accurate detection and high robustness. Some experiments are taken on the Bern dataset acquired by the ERS-2 SAR, the Ottawa dataset acquired by a RADARSAT SAR, the Yellow River dataset acquired by Radarsat-2 sensor and two single-look SAR images pairs on the Wuhan dataset acquired by PALSAR without ground-truth. The experimental results verify the efficiency of ESLM.

**II. MULTISTAGE CLUSTERING**

In order to automatically extract the training samples for the subsequent incremental classification, in this section a multistage clustering scheme is proposed. The reflectivity-spatial affinity is first defined, and then a new APSC algorithm is introduced.

**A. REFLECTIVITY-SPATIAL AFFINITY OF SUPERPIXELS**

Given two co-registered SAR images $X_i = \{X_i(i,j)\}$ and $X_j = \{X_j(i,j)\}$ taken at different time of the same scene, where $X_i(i,j)$ and $X_j(i,j)$ are the pixels located at $(i,j)$ in the images, and $1 \leq i \leq I, 1 \leq j \leq J$. First, we generate the DM by a log-ratio operator [13],

$$
DM = \frac{\log(X_i + \varepsilon)}{\log(X_j + \varepsilon)}
$$

(1)
where $\varepsilon$ is a small positive constant to avoid the pixel values in $X(i = 1, 2)$ being zero. As soon as $DM$ is obtained, we first use the Simple Linear Iterative Cluster (SLIC) [14] to segment $DM$ into $N$ super-pixels $S_1, \ldots, S_N$. The number of super-pixels is set as $N = \lfloor I \times J / 100 \rfloor$. Then an APSC algorithm is advanced to acquire initial training samples of ESM, which need not define the number of clusters beforehand.

It is well known that with the increasing resolution of SAR images, they are not only a set of reflectivities but also a set of data with spatial organization. So in our work we define a reflectivity-spatial affinity $s(i, j)$ between the $i^{th}$ and $j^{th}$ super-pixel, to evaluate their similarity $s(i, j)$:

$$s(i, j) = s_{reflectivity}(i, j) + s_{spatial}(i, j)$$

where $s_{reflectivity}(i, j)$ and $s_{spatial}(i, j)$ are the reflectivity affinity and spatial affinity between the $i^{th}$ and $j^{th}$ super-pixels respectively. The reflectivity affinity is defined as,

$$s_{reflectivity}(i, j) = -\|r_i - r_j\|$$

where $r_i$ and $r_j$ represent the average reflectivities of the $i^{th}$ and $j^{th}$ super-pixel. The spatial affinity is defined as,

$$s_{spatial}(i, j) = \sqrt{((i_x - j_x)^2 + (i_y - j_y)^2)}$$

where $i_x, j_x$ and $i_y, j_y$ represent the coordinates of the two super-pixels in the image.

### B. AFFINITY PROPAGATION SUPER-PIXEL CLUSTERING (APSC)

Then we use the calculated reflectivity-spatial affinity $s(i, j)$ and propagation algorithm to locate the potential exemplars (or clustering centers) from $N$ super-pixels, via iterative responsibility $r(i, j)$ and availability $a(i, j)$ [15].

The $N$ super-pixels are considered as nodes in a network and $e_{i, j} = 1, \ldots, N$ is a set of binary hidden variables, where $e_{i, j} = 1$ indicates that the $j^{th}$ super-pixel is chosen as a clustering center and $e_{i, j} = 0$ indicates that it is not a clustering center. $h_{i, j} = 1, \ldots, N$ is a set of $N^2$ binary hidden variables, where $h_{i, j} = 1$ indicates that the $j^{th}$ superpixel has chosen the $i^{th}$ pixel as its clustering center. Affinity propagation tries to find the most representative samples that maximize the sum of affinities of nodes to their exemplars. This optimization can be implemented via the update of responsibility $r(i, j)$ and availability $a(i, j)$. The availabilities $a(i, j)$ are initialized as zero, and the responsibility $r(i, j)$ are updated by [15],

\[
r(i, j) \leftarrow s(i, j) - \max_{j' \neq j} \{a(i, j') + s(i, j')\}
\]

Then the availabilities $a(i, j)$ are updated by:

\[
a(i, j) \leftarrow \min\{0, r(i, j) + \sum_{i' \neq i} \max\{0, r(i', j)\}\}. \quad i \neq j
\]

\[
a(i, j) \leftarrow \sum_{i' \neq i} \max\{0, r(i', j)\}, \quad i = j
\]

This procedure will terminate until the exemplars are stably determined. The APSC algorithm has some advantages over the traditional clustering algorithms: 1) It is free of parameter and can automatically determine the number of clusters; 2) It can involve many-sides affinity between nodes to find more representative clustering centers; 3) The affinity update is of low complexity.

### C. MULTISTAGE CLUSTERING

After APSC, denote the number of clusters as $V$, and denote the intensity of the centroid of the $n^{th}$ super-pixel in the $v^{th}$ cluster as $I_v^n$. Then the average value in each cluster is calculated as $AI_v^n(v = 1, 2, \ldots, V)$. Then we perform a $K$-means clustering on $AI_v^n$ to cluster the super-pixels into three groups: strictly changed class, strictly unchanged class and fuzzy class. According to $AI_v^n$, the class that has the highest $AI_v^n$ value is considered as the strictly changed class. The class that has the lowest $AI_v^n$ value is considered as the strictly unchanged class. The other class is considered as fuzzy class.

In our work, we select the "confident" pixels from the strictly unchanged and strictly changed classes, to serve as the labeled samples. On the other hand, we also select the "confident" pixels from the fuzzy class, as the unlabeled samples. If a pixel and its neighbors (a squared $b \times b$ neighboring window centered around the pixel) belong to the same class, we choose it as the "confident" pixel. The "confident" pixels in the strictly unchanged class, strictly changed class and fuzzy class are denoted as $\{x_i^1 \}$, $\{x_i^2 \}$ and $\{x_i^3 \}$ respectively. In our work we choose $b = \sqrt{num / 3}$, where $num$ is the number of pixels in the super-pixel. Fig.2 illustrates the procedure of this multistage clustering.

![FIGURE 2. Multistage clustering for determining initial training samples](image-url)
III. EXTREME SELF-PACED LEARNING MACHINE (ESLM)
Semi-supervised classifier can use a handful of labeled samples and large amount of unlabeled samples to enhance the performance of classification [16-19][31]. In this section we first formulate an affinity regularizer to implement the semi-supervised extreme learning. Then an incremental extreme self-paced learning machine is constructed to gradually identify changed pixels.

A. EXTREME LEARNING MACHINE (ELM)
Extreme Learning Machine (ELM) is a single-hidden layer feed-forward neural network [20]. Denote the number of hidden layer as \(k\), and denote the nonlinear activation function of the hidden layer as \(g(\cdot)\). Then the output of the hidden layer for the inputs can be written as a matrix,

\[
H = \begin{bmatrix}
g(w_1 \cdot x + b_1) & \ldots & g(w_1 \cdot x + b_M) \\
g(w_2 \cdot x + b_1) & \ldots & g(w_2 \cdot x + b_M) \\
\vdots & \ddots & \vdots \\
g(w_M \cdot x + b_1) & \ldots & g(w_M \cdot x + b_M)
\end{bmatrix}
\]

(8)

where \(x_i\) is the \(i^{th}\) training samples and \(w_i = [w_{i1}, w_{i2}, \ldots, w_{id}]\) is the random weight vector connecting the \(i^{th}\) hidden neuron with input neurons, \(b_i\) is the bias of the \(i^{th}\) hidden neuron. \(K\) and \(Q\) are the number of hidden neurons and samples respectively. Here a continuous sigmoid function \(g(\cdot)\) is adopted in the hidden layer, according to the universal approximation capability of feed-forward neural networks [32]. The activation function in the output layer is linear, so the outputs of the network is \(Y = y_1, y_2, \ldots, y_{M}\), with the output of the \(j^{th}\) neuron being

\[
y_j = \beta_j^T H
\]

(9)

where \(\beta_j = [\beta_{j1}, \beta_{j2}, \ldots, \beta_{jm}]^T (j = 1, 2, \ldots, m)\) is the weight vector connecting the hidden neurons with the \(j^{th}\) output neuron. \(m\) is the number of output neurons. In the classification, \(m\) is the length of the label. Denoting the weight matrix as \(B = [\beta_1, \beta_2, \ldots, \beta_m] \in \mathbb{R}^{m\times Q}\), we solve \(B\) by minimizing the errors of training samples and the norm of weights,

\[
\min_{B} \left\| B^T H - T \right\|^2 + \left\| B \right\|^2
\]

(10)

where \(T = [t_1, t_2, \ldots, t_Q] \in \mathbb{R}^{d\times Q}\) is the desired output matrix, and \(t_i\) is the corresponding output of \(x_i\). Therefore \(B\) can be analytically determined by

\[
B = (H H^T + \lambda I)^{-1} H^T T
\]

(11)

where \(I\) is an identity matrix.

B. AFFINITY REGULARIZED SEMI-SUPERVISED EXTREME LEARNING
In our work, we use the confident pixels set \(\{x_l^c\}\) and strictly unchanged pixels set \(\{x_i^u\}\) as the initial labeled samples \(L_0 = \{(x_i^c, t_i^c) | x_i^c, t_i^c \in \mathbb{R}^d, j \in (c, u), t_i^c \in \{0, 1\}\}\) , where \(t_i^c\) is a 2-d binary vector indicating the label of \(x_i^c\) \(\{t_i^c | t_i^c = [1, 0]^T\}\) when \(j = c\); \(t_i^c = [0, 1]^T\) when \(j = u\); \(i = 1, 2, \ldots, l (l = |L_0|)\) is the sum of the number of strictly unchanged and strictly changed pixels). The fuzzy pixels \(\{x_i^f\}\) are taken as the initial unlabeled samples \(U_0 = \{x_i^f | x_i^f \in \mathbb{R}^d\}\) , where \(i = 1, 2, \ldots, u = |U_0|\) is the sum of the number of fuzzy pixels.

Based on ELM, we develop an Affinity Regularized Semi-supervised Extreme Learning Machine (AR-SELM), based on a manifold regularizer. A local consistency assumption is cast on both the labeled and unlabeled samples, which indicates that samples with large affinity should have similar labels [21]. Consequently, a new affinity regularizer is defined as follows,

\[
R = \sum_{i \in L_0} s(i, j) \left\| Y_i - Y_j \right\|^2 / 2
\]

(12)

where \(Y_i, Y_j\) are the corresponding outputs of two samples \(X_i, X_j\) respectively. Because the items in (12) are positive, this regularizer constrains that if \(X_i\) and \(X_j\) are similar to each other, then the predictions \(Y_i, Y_j\) should be similar as well. Moreover, we reformulate (12) as

\[
R = \sum_{i \in L_0} s(i, j) \left\| Y_i - Y_j \right\|^2 - 2 \left\langle Y_i, Y_j \right\rangle / 2
\]

\[
= \frac{1}{2} \left( T_r(Y_i^{T} - D^{T} W Y_i) + T_r(Y_j^{T} - D^{T} W Y_j) \right) - T_r(Y_i^{T} Y_j^{T} W)
\]

(13)

where \(D \in \mathbb{R}^{d \times (d+1)}\) is a diagonal matrix with the element \(D_{ii} = \sum_{j \neq i} s(i, j)\) .Then the graph Laplacian matrix \(L = D - W\) is calculated.

Denote the output of the hidden layer for the inputs \(t_i \) as \(H_i\). By modifying the objection function of ELM in (10), the objective function of AR-SELM can be written as:

\[
\min_{B_0} \left\| B_0^T H_0 - Y \right\|^2 + \left\| B_0 \right\|^2 + \lambda \left( Y_i^{T} L Y_i \right)
\]

(14)

The analytical solution to equation (14) can be calculated to determine the weights of extreme learning machine,

\[
B_0 = (H_0^T H_0 + \lambda I)^{-1} H_0^T Y
\]

(15)

As soon as the weights are calculated, the unknown samples can be predicted as changed or unchanged pixels from \(Y = B_0^T H_0\).

C. SELF-PACED LEARNING
In our work, an incremental extreme self-paced learning machine is developed to gradually predict the unknown...
samples. After the multistage clustering, some samples are extracted as the initial labeled and unlabeled samples. The affinity regularized semi-supervised classifier is then extended to the self-paced mode, which can automatically update the decision boundary by involving the confident pixels into the training. The labeled and unlabeled data come chunk by chunk, and the weights are incrementally learned. So the pixels in the image are gradually predicted as the changed or unchanged pixels.

Suppose at time $t$, the training dataset $S_t = L_t \cup U_t$ has been learned to solve $B_t, H_t$ is the hidden matrix of $S_t$, $H_t$ is the hidden matrix of $L_t$, $H_t$ is the hidden matrix of $U_t$. The unlabeled samples are predicted by their outputs $Y_t = B_t^T H_t$, where $H_t$ is the hidden matrix of unlabeled samples in $S_t$. The samples whose predicted labels are have been learned to solve $B_t, H_t$ is the hidden matrix of $S_t$. $H_t$ is the hidden matrix of $L_t$, $H_t$ is the hidden matrix of $U_t$. The unlabeled samples are predicted by their outputs $Y_t = B_t^T H_t$, where $H_t$ is the hidden matrix of unlabeled samples in $S_t$. The samples whose predicted labels are confident are selected to be the new received chunk $S_{t+1} = \{x_j\}_{j=1}^k$ by sequentially $\max \max(Y_t)$.

At time $t+1$, the new received chunk $S_{t+1}$ can be classified as two groups: $k/2$ labeled samples (denoted as $S'_{t+1}$) and $k/2$ unlabeled samples (denoted as $S''_{t+1}$). Denote the hidden matrix respect to the new chunk as $H_{t+1}$. The hidden matrix and target matrix of $S'_{t+1}$ are denoted as $H_t$ and $Y_t$, respectively. The update training dataset is denoted as $S_{t+1} = L_{t+1} \cup U_{t+1}$, where $L_{t+1} = L_t \cup S'_{t+1}$, $U_{t+1} = U_t \cup S''_{t+1}$. The output matrix of the hidden layer for all the received data and the labeled data has been received up to time $t+1$,

$$
H_{t+1} = \begin{bmatrix} H_t \\ H_t \end{bmatrix}, \quad H_{t+1} = \begin{bmatrix} H_t \\ H_t \end{bmatrix}
$$

According to (15), the new weights of extreme self-paced learning machine can be determined by,

$$B_{t+1} = \begin{bmatrix} I & H_t^T H_t + \lambda H_t^T L_t H_t \end{bmatrix} H_t^T Y_t.
$$

For simplicity, we denote $A_{t+1}$ as,

$$A_t = \begin{bmatrix} I + H_t^T H_t + \lambda H_t^T L_t H_t \end{bmatrix} H_t^T Y_t.
$$

The graph Laplacian matrix at time $t+1$ can be expressed as

$$L_{t+1} = \begin{bmatrix} L_t + D_{t+1} - S_{t+1} \\ -S_{t+1} & D_{t+1} + L_t \end{bmatrix}
$$

where $L$ and $L_t$ are the graph Laplacian matrices with respect to the dataset $S_t$ and $S_{t+1}$, respectively. $S_{t+1}$ is affinity matrix from the dataset $S_t$ to $S_{t+1}$, $S_{t+1}$ is affinity matrix from the dataset $S_{t+1}$ to $S_t$. $D_{t+1}$ and $D_{t+1}$ are the diagonal matrices whose main diagonal elements are row sums of $S_{t+1}$ and $S_{t+1}$, respectively. According to the relationship between time $t$ and $t+1$, $A_{t+1}$ can be written as

$$A_{t+1} = A_t + H_t^T H_t + \lambda H_t^T L_t H_t.
$$

Then we calculate,

$$A_{t+1} = \left( A_t + H_t^T H_t + \lambda H_t^T L_t H_t \right) H_t^T F_t
$$

$$= F_t - \lambda F_t H_t \left( I + \lambda L_t H_t^T F_t H_t \right) L_t H_t F_t
$$

where $F_t = A_t + H_t^T H_t = A_t - A_t H_t \left( I + \lambda L_t H_t A_t^T H_t \right) H_t A_t^T H_t$.

If we denote $K_t$ as

$$K_t = I - A_t H_t \left( I + \lambda L_t H_t A_t^T H_t \right) H_t A_t^T H_t
$$

Substituting $K_t$ into $F_t$, we can get $F_t = K_t A_t^T H_t$ and $A_{t+1} = J A_t^T H_t$, where $J = K_t - \lambda K_t A_t H_t \left( I + \lambda L_t H_t K_t A_t^T H_t \right) I L_t H_t K_t$. Thus we can obtain the weights of the AR-SELMA,

$$B_{t+1} = \begin{bmatrix} A_t^T H_t Y_t \\ J B + J A_t^T H_t Y_t \end{bmatrix}
$$

D. ESLM based on-orbit change detection

In the on-orbit change detection, the working process can be divided into a self-paced learning phase and then a prediction phase. The procedure of ESLM based change detection is described in Algorithm 1.

Algorithm 1: ESLM based change detection algorithm

Input: Two SAR images

Output: Changes of the two images

Initialization phase: The samples after multistage clustering are used as the initial training samples: $S_0 = L_0 \cup U_0$

1) Generate the hidden matrix $H_0$, $Y_0$ for $S_0$;
2) Calculate the graph Laplacian matrix $L_0$ for $S_0$;
3) Randomly initialize the parameter of ELM;
4) Calculate $B_0$ from (15);

Self-paced learning phase: Set $t=0$ and repeat the following steps,

1) Formulate the data chunk $S_t$ from $B_0$;
2) Calculate the mapping matrix $H_t$ for the newest chunk;
3) Calculate the Laplacian matrix $L_t$ with respect to the input of dataset $S_t$;
4) Calculate the mapping matrix $H_t$ for the newest labeled samples, calculate $K_t, J_t, B_t$ and $A_{t+1}$;
5) Set $t=t+1$, go to step 1);

Until the variation of $\|B_t - B_{t-1}\| < \epsilon$

Predicting phase: Use the learned $B_t$ to evaluate the output of the unknown samples to locate the changed pixels.

IV. EXPERIMENTAL RESULTS

To investigate the performance of the proposed method, in this section some experiments are taken on five real multi-temporal SAR images, including the Bern dataset acquired by the ERS-2 SAR, the Ottawa dataset acquired by a RADARSAT SAR, the Yellow River dataset acquired by Radarsat-2, and two single-look SAR images pairs on the Wuhan dataset acquired by PALSAR. The first three datasets have the ground-truth and the last two datasets have not the ground-truth.
A. DATASETS AND COMPARATIVE METHODS

1) Dataset 1: The first dataset covers a region near the city of Bern, Switzerland, in April and May 1999. The size of the two SAR images is 301×301, with the resolution being 30m, which are shown in Fig.3(a) and Fig.3(b) [3]. The images are collected by the SAR on the European Remote Sensing 2 satellite. The radar works in C-band and has VV polarization. The ground-truth is shown in Fig.3(c).

![FIGURE 3. SAR images of Bern. (a) Image in 04, 1999. (b) Image in 05, 1999. (c) Ground-truth.](image)

2) Dataset 2: The second dataset covers a region over the city of Ottawa, in May and August 1997. The size of the two SAR images is 290×350, with the resolution being 10m, which are shown in Fig.4(a) and Fig.4(b). The images are collected by the SAR on the RADARSAT. The radar works in C-band and has HH polarization. The ground-truth is shown in Fig.4(c).

![FIGURE 4. SAR images of Ottawa. (a) Image in 05, 1997. (b) Image in 08, 1997. (c) Ground-truth.](image)

3) Dataset 3: The third dataset covers a region over the Yellow River estuary in China in June 2008 and June 2009. The size of the two SAR images is 257×289, with the resolution being 3m, which are shown in Fig.5(a) and Fig.5(b). The images are collected by the SAR on the Radarsat-2. The radar works in C-band and has HH polarization. The ground-truth is shown in Fig.5(c). In addition, speckle noise on the image Fig.5(a) is much more than that of Fig.5(b), because Fig.5(a) is single-look image and Fig.5(a) is four-look.

![FIGURE 5. SAR images of the Yellow River Estuary. (a) Image in 06, 2008. (b) Image in 06, 2009. (c) Ground-truth.](image)

4) Dataset 4 and 5: The last two datasets cover the region of Wuhan Province in China. The original two SAR images are obtained by PALSAR in June 2006 and in March 2009. The radar works in L-band and has HH polarization. The resolution of the two images is 10m, and the images are shown in Fig.5(a) and Fig.5(b) respectively. The range and size of the two SAR images are 40km×70km and 500×500 respectively. The Wuhan dataset is single-look, so the two images have more speckle noises than the first four datasets. Two obviously changed areas are selected in original SAR images to verify the performance of the proposed algorithm.

In order to verify the effectiveness of ESLM, five related and state-of-the-art algorithms are used for a comparison, including the Mathematical Morphology and K-means clustering (MMK) [30], Deformable Dictionary Learning (DDL) [26], Neighborhood-based Ratio (NR) [27], Local Restricted Convolutional Network (LRCN) [28] and Contractive Autoencoder (CA) [29]. Both the visual results and numerical results are demonstrated, along with the consumed time of these methods. All the experiments are taken on Intel® Core™ i3-3210, CPU @2.10GHz 4.0GB Windows 10 systems, Matlab 2014a.

B. EVALUATION CRITERION AND PARAMETER SELECTION

In order to evaluate the performance of various kinds of algorithms, some evaluation criterions are used, including Missed Alarms (MA: the number of undetected pixels in the changed region), False Alarms (FA: the number of changed pixels that are wrongly detected as unchanged pixels), Overall Error (OE) and Kappa Coefficient (KC) [22]. Moreover, in this section we make an analysis on the parameter on the first three datasets.

![FIGURE 6. Performances of ESLM with the variation of parameter \( \lambda \). (a) Bern dataset. (b) Ottawa dataset. (c) Yellow dataset. (d) Kappa coefficient.](image)
\( N_c \) is the number of changed pixels in the ground-truth and \( N_u \) is the number of unchanged pixels in the ground-truth. Then some evaluation metrics are calculated [23]: (1) false alarm rate: \( P_{fa} = FA / N_u \). (2) missed alarm rate: \( P_{ma} = MA / N_c \). (3) total errors rate: \( P_{oe} = OE / (N_c + N_u) \).

In our method, there is a single parameter \( \lambda \), which determines the weight of affinity regularizer in the semi-supervised extreme learning machine. In order to choose an appropriate \( \lambda \), we vary its value from 0 to 1, and the results of the first three datasets are shown in Fig.6. From the variation of MA, FA and OE with the parameter \( \lambda \) in Fig.6(a)(b)(c), we can observe that for the datasets, these values first increase and then decrease. When \( \lambda \) takes the value in the range of [0.08, 0.15], the three metrics can achieve relatively lower values, which indicates more accurate change detection accuracy. Fig.6(d) shows the variation of KC with the parameter \( \lambda \) for the first three datasets, and form it we can observe that KC also takes a relatively larger value in this range for the three datasets. So in the following experiments, we set \( \lambda \) as 0.1.

C. EXPERIMENTAL RESULTS ON THE FIRST THREE DATASETS

In this section we first investigate the performance of the proposed APSC algorithm. For the first three datasets, the labeled pixels, fuzzy pixels and unlabeled pixels extracted by APSC are shown in Fig.7-9. Fig.7(a), Fig.8(a) and Fig.9(a) show the strictly changed pixels on the Bern, Ottawa and Yellow datasets respectively. When compared them with the ground-truth of the three datasets, we can observe that most of the detected labeled pixels are correct, which validate the effectiveness of APSC.

Moreover, from Fig.7(a), Fig.8(a) and Fig.9(a) we can see that the detected strictly changed pixels are locally connected. That is, APSC can effectively reduce the influence of speckle noises existed in the original images. Fig.7(b), Fig.8(b) and Fig.9(b) show the fuzzy pixels detected by APSC on the Bern, Ottawa and Yellow datasets respectively. From them we can observe that the fuzzy pixels prone to be uniformly distributed in the whole image, which contain both changed and unchanged pixels. This is consistent with the uncertainty of fuzzy pixels. Fig.7(c), Fig.8(c) and Fig.9(c) show the strictly unchanged pixels by APSC on the Bern, Ottawa and Yellow datasets respectively. From them we can observe that most of them are unchanged pixels, and this is especially obvious in Fig.9(c). However, the detected strictly unchanged pixels not only contain pixels with low reflectivity but also include some speckle noises, which will bring some misclassification to ESLM.

When the three groups of pixels are identified, then ESLM is incrementally trained and detect changes. Consequently we compare the detection results of the five comparative methods and ESLM. The change detection results of the first three datasets are shown in Fig.10~Fig.12 respectively. Fig.10(a)-(f) show the detected change maps of the first dataset, by MMK [30], DDL [26], NR [27], LRCN [28], CA [29] and ESLM. From them we can see that there are little difference among the
detection results of different, since there are little speckle noises existed in the original images. MMK [30], LRCN [28], CA [29] and ESLM present similar results, and there are some wrongly detected pixels in the result of DDL [26] and NR [27]. Compared with the unsupervised methods, ESLM not only provides more refined edges but also smaller noises when compared with the. Moreover, ESLM can present comparable results with the supervised methods that explore a lot of labeled pixels for training, including LRCN [28] and CA [29]. However, different with supervised approaches, ESLM need not manually label large number of changed and unchanged pixels. By using the self-paced learning, ESLM can automatically label change pixels in an incremental manner. Besides it, ESLM also explore fuzzy pixels to enhance the detection accuracy, and the parameters of ESLM are analytically determined, which can also achieve rapid detection of changes.

Fig.11(a)-(f) show the detected change maps of the second dataset by MMK [30], DDL [26], NR [27], LRCN [28], CA [29] and ESLM respectively. From Fig.12(f) we can see that our method can present comparable results with other supervised methods. However, ESLM consumes less time than the supervised methods, such as LRCN [28] and CA [29]. Among the six methods, MMK present very homogeneous detection results. Because the ground-truth of the third dataset is also homogeneous, the change map generated by MMK [30] are more accord with the ground-truth when compared with other methods.

### TABLE I

<table>
<thead>
<tr>
<th>Methods</th>
<th>FA</th>
<th>MAP</th>
<th>P_{FA}</th>
<th>P_{MAP}</th>
<th>OE</th>
<th>P_{OE}</th>
<th>KC</th>
<th>T(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMK [30]</td>
<td>116</td>
<td>190</td>
<td>16.50</td>
<td>306</td>
<td>0.33</td>
<td>0.874</td>
<td>104.5</td>
<td></td>
</tr>
<tr>
<td>DDL [26]</td>
<td>141</td>
<td>199</td>
<td>17.20</td>
<td>340</td>
<td>0.37</td>
<td>0.743</td>
<td>64.3</td>
<td></td>
</tr>
<tr>
<td>NR [27]</td>
<td>127</td>
<td>402</td>
<td>34.80</td>
<td>529</td>
<td>0.58</td>
<td>0.760</td>
<td>20.3</td>
<td></td>
</tr>
<tr>
<td>LRCN [28]</td>
<td>110</td>
<td>194</td>
<td>16.80</td>
<td>304</td>
<td>0.33</td>
<td>0.895</td>
<td>248.1</td>
<td></td>
</tr>
<tr>
<td>CA [29]</td>
<td>109</td>
<td>185</td>
<td>16.02</td>
<td>294</td>
<td>0.32</td>
<td>0.901</td>
<td>132.7</td>
<td></td>
</tr>
<tr>
<td>ESLM</td>
<td>107</td>
<td>187</td>
<td>16.19</td>
<td>294</td>
<td>0.32</td>
<td>0.908</td>
<td>21.6</td>
<td></td>
</tr>
</tbody>
</table>

### TABLE II

<table>
<thead>
<tr>
<th>Methods</th>
<th>FA</th>
<th>MAP</th>
<th>P_{FA}</th>
<th>P_{MAP}</th>
<th>OE</th>
<th>P_{OE}</th>
<th>KC</th>
<th>T(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMK [30]</td>
<td>1349</td>
<td>1330</td>
<td>1308</td>
<td>2679</td>
<td>2.63</td>
<td>0.903</td>
<td>115.2</td>
<td></td>
</tr>
<tr>
<td>DDL [26]</td>
<td>1366</td>
<td>760</td>
<td>4.73</td>
<td>2126</td>
<td>2.09</td>
<td>0.922</td>
<td>67.1</td>
<td></td>
</tr>
<tr>
<td>NR [27]</td>
<td>2148</td>
<td>2191</td>
<td>13.65</td>
<td>4339</td>
<td>4.27</td>
<td>0.839</td>
<td>29.4</td>
<td></td>
</tr>
<tr>
<td>LRCN [28]</td>
<td>1107</td>
<td>1258</td>
<td>7.84</td>
<td>2365</td>
<td>2.33</td>
<td>0.926</td>
<td>253.8</td>
<td></td>
</tr>
<tr>
<td>CA [29]</td>
<td>832</td>
<td>1301</td>
<td>8.10</td>
<td>2133</td>
<td>2.10</td>
<td>0.911</td>
<td>203.3</td>
<td></td>
</tr>
<tr>
<td>ESLM</td>
<td>787</td>
<td>1200</td>
<td>7.48</td>
<td>1987</td>
<td>1.96</td>
<td>0.927</td>
<td>27.1</td>
<td></td>
</tr>
</tbody>
</table>

### TABLE III

<table>
<thead>
<tr>
<th>Methods</th>
<th>FA</th>
<th>MAP</th>
<th>P_{FA}</th>
<th>P_{MAP}</th>
<th>OE</th>
<th>P_{OE}</th>
<th>KC</th>
<th>T(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMK [30]</td>
<td>2691</td>
<td>1864</td>
<td>13.9</td>
<td>4555</td>
<td>6.13</td>
<td>0.807</td>
<td>139.3</td>
<td></td>
</tr>
<tr>
<td>DDL [26]</td>
<td>3907</td>
<td>2180</td>
<td>16.2</td>
<td>6087</td>
<td>8.19</td>
<td>0.647</td>
<td>49.1</td>
<td></td>
</tr>
<tr>
<td>NR [27]</td>
<td>3858</td>
<td>2002</td>
<td>14.9</td>
<td>5860</td>
<td>7.89</td>
<td>0.663</td>
<td>33.4</td>
<td></td>
</tr>
<tr>
<td>LRCN [28]</td>
<td>2979</td>
<td>2117</td>
<td>15.8</td>
<td>5096</td>
<td>6.86</td>
<td>0.725</td>
<td>217.6</td>
<td></td>
</tr>
<tr>
<td>CA [29]</td>
<td>3023</td>
<td>2292</td>
<td>17.1</td>
<td>5315</td>
<td>7.15</td>
<td>0.754</td>
<td>182.4</td>
<td></td>
</tr>
<tr>
<td>ESLM</td>
<td>2732</td>
<td>1994</td>
<td>14.85</td>
<td>4726</td>
<td>6.36</td>
<td>0.781</td>
<td>35.2</td>
<td></td>
</tr>
</tbody>
</table>

The numerical metric (FA, MAP, OE, P_{FA}, P_{MAP}, P_{OE}, KC) and the consumed time of the six methods are shown in Table I, Table II and Table III respectively. As to these evaluations,
the best results of the six methods are denoted as bold. From the results we can see that in most cases, ESLM has better performance in detecting change details than the other methods. As can be seen from the tables, the proposed ESLM can totally achieve small detection error. Moreover, the proposed method not only has the highest kappa coefficients, but also has acceptable running time.

D. EVALUATION CRITERION AND PARAMETER SELECTION

In order to separately investigate the role of the clustering algorithm and the classifier in our method, in this test we first compare our proposed APSC algorithm with AP algorithm, and then compare the AR-SELM with ELM. The performance of the conventional AP and our proposed APSC algorithms, are compared on the three datasets. Fig.13-15 show the convergence curves of AP and APSC for the three datasets respectively. From Fig.13-15, we can observe that for all the three datasets, APSC required less number of iterations before convergence than that of AP. In other words, APSC has faster clustering speed than AP. Moreover, we can also find that not only the clustering speed of the APSC algorithm is faster than that of AP, but also a smooth convergence, which can be observed in Fig.13(b), Fig.14(b) and Fig.15(b).

Moreover, two methods are used to compare with our proposed method, including the AP-ELM (which uses AP for clustering and ELM without affinity regularizer), AP-ARSELM (which uses AP for clustering and AR-SELM with affinity regularizer). The comparison results of their performance on the first three datasets are shown in Table IV–VI respectively. It can be clearly seen from them that APSC has an improvement over AP about the performance, and our proposed ESLM is superior to the original ELM by adding the spatial and reflectivity affinity between pixels in the classification.

TABLE IV

<table>
<thead>
<tr>
<th>Methods</th>
<th>FA</th>
<th>P_A</th>
<th>P_M</th>
<th>OE</th>
<th>KC</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP-ELM</td>
<td>370</td>
<td>0.41</td>
<td>250</td>
<td>21.65</td>
<td>620</td>
</tr>
<tr>
<td>AP-ARSELM</td>
<td>189</td>
<td>0.21</td>
<td>130</td>
<td>11.26</td>
<td>319</td>
</tr>
<tr>
<td>ESLM</td>
<td>107</td>
<td>0.12</td>
<td>187</td>
<td>16.19</td>
<td>294</td>
</tr>
</tbody>
</table>

TABLE V

<table>
<thead>
<tr>
<th>Methods</th>
<th>FA</th>
<th>P_A</th>
<th>P_M</th>
<th>OE</th>
<th>KC</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP-ELM</td>
<td>3325</td>
<td>3.28</td>
<td>1094</td>
<td>6.82</td>
<td>4419</td>
</tr>
<tr>
<td>AP-ARSELM</td>
<td>1253</td>
<td>1.23</td>
<td>958</td>
<td>5.97</td>
<td>2211</td>
</tr>
<tr>
<td>ESLM</td>
<td>787</td>
<td>0.78</td>
<td>1200</td>
<td>4.66</td>
<td>1987</td>
</tr>
</tbody>
</table>

TABLE VI

<table>
<thead>
<tr>
<th>Methods</th>
<th>FA</th>
<th>P_A</th>
<th>P_M</th>
<th>OE</th>
<th>KC</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP-ELM</td>
<td>4723</td>
<td>6.36</td>
<td>5492</td>
<td>40.89</td>
<td>10215</td>
</tr>
<tr>
<td>AP-ARSELM</td>
<td>3309</td>
<td>4.46</td>
<td>1549</td>
<td>11.53</td>
<td>4858</td>
</tr>
<tr>
<td>ESLM</td>
<td>2732</td>
<td>3.68</td>
<td>1994</td>
<td>14.85</td>
<td>4726</td>
</tr>
</tbody>
</table>

E. EXPERIMENTAL RESULTS ON THE LAST TWO DATASETS

In this test, we test our proposed method on the last two datasets. Since the Wuhan area dataset have no ground-truth to quantitatively measure the performance of these algorithms, the detection results of six methods are shown in Fig.16 and Fig.17 for visual comparison. Fig.16(a)-(b) and Fig.17(a)-(b) show the original source images, and the log-ratio DMs of the forth and fifth dataset are shown in Fig.16(c) and Fig.17(c) respectively. The main changes in Fig.16 and Fig.17 for visual comparison. Fig.16(a)-(b) include the Erqi Changjiang River Bridge, some ships and buildings.

The detection results of the six methods on the two datasets are shown in Fig.16(c)-(i) and Fig.17(c)-(i), respectively. As shown in Fig.16 (e) and (h), we could observe that DDL and CA produce much more isolate points, which denotes that they are not robust enough to the strong multiplicative noise. However, from Fig.16(i) we can observe that our method can produce less noises and detect more homogeneous changes when compared with other methods. Similarly, the detected changes by ESLM in Fig.17 (i) have less isolated points.
Human intelligence indicates a learning agent interacts with a dynamic environment and updates its action policies to maximize its long-term rewards. Inspired by it, an automatic and robust unsupervised method via multistage clustering and new extreme self-paced learning machine, is proposed for on-orbit change detection of SAR images. The advantage of ESLM is two folds: 1) it can gradually make avail of the most confident samples, so it has low memory cost and low computational complexity, which is desirable for on-orbit processing; 2) it adopts the semi-supervised model to realize an unsupervised classification, so having high automatic degree. The detection accuracy and efficiency of the proposed method are verified on several real SAR image datasets, and the results are promising for on-orbit change detection. However, as we can see in the results of the Yellow River dataset, our algorithm is also influenced by heavy speckle noises. Therefore, improving the performance of the algorithm to deal with very heavy speckle noises will be considered in our future work.

ACKNOWLEDGMENT
This work was supported by the National Natural Science Foundation of China (Nos.61771380, U1730109, 91438103, 61771376, 61703328, 91438201, U1701267, 61703328); the Equipment pre-research project of the 13th Five-Years Plan (Nos. 6140137050206, 414120101026, 6140312010103, 6141A020223, 6141B06160301, 6141B07090102); the Major Research Plan in Shaanxi Province of China (Nos.2017ZDXM-GY-103, 017ZDCXL-GY-03-02), the Foundation of the State Key Laboratory of CEMEE (No.2017K0202B, 2018K0101B), the Science Basis Research Program in Shaanxi Province of China (No.16JK1823, 2017JM6086).

REFERENCES


