Anomaly Detection Based on Sigmoid Metric and Object Area Filtering in Hyperspectral Images

Hamid Esmaeili Najafabadi, *Member, IEEE,* Zhenkai Zhang, *Member, IEEE,* Mahdi Yousefan, Amirshayan Nasirimajd

***Abstract*—This paper outlines a new approach to detect anomalies in hyperspectral images based on peripheral pixels. The proposed methodology contains two main steps. First, a new distance score is introduced based on the sigmoid function and root mean square error (RMSE). We estimate how likely the target pixel is an anomaly by averaging the new metric over its neighboring window. Second, a state-of-the-art method is applied to eliminate unacceptable objects according to their size. In this light, the objects whose size is out of an acceptable interval are removed. Comprehensive experimental evaluations have been conducted to confirm that the proposed method significantly outperforms several recent algorithms in accuracy and computational time.**

***Index Terms*—Anomaly detection, hyperspectral image, object area filtering**

I. Introduction

HYPERSPECTRAL remote sensing (HRS) is a burgeoning, multidisciplinary field with wide applications. In the HRS, each pixel represents a vector of spectral responses across the electromagnetic spectrum [1, 2]. Various research disciplines benefit from these sensors such as geology, hydrology, urban planning, geography, cadastral mapping, cartography, and military [3]. Mainly, two types of applications are prevalent: anomaly classification and anomaly detection [4]. In the latter, the goal is to recognize a distinction that differentiates uncommon observations, such as human-made objects. Here, there is a significant difference between the spectral signatures of anomalies and the natural background materials [5]. Different objects may be considered an anomaly, such as species in agriculture and ecology, rare animals in geology, ships in a sea background, or planes in an airport [6, 7]. Traditionally, anomaly detection is divided into supervised and unsupervised, where unsupervised algorithms often employ statical techniques. For example, Reed-Xiaoli (RX) is an unsupervised algorithm for hyperspectral imaging (HSI) based on a sample correlation matrix to detect anomalies using both a full hyperspectral image, known as global RX (GRX), or a single local sliding window centered around each image pixel, known as local RX (LRX) [8, 9]. There are many improved methods based on RX, such as The Gaussian Mixture Model Method (GMMM) [10]. The GMMM uses a combination of multivariate Gaussian distributions to model the background statistical information. Reference [11] presents a method of anomaly detection for HSI via global and local joint modeling of the background. Another approach to anomaly detection for HSI is the local joint subspace process and support vector machine (SVM) in three consecutive steps: 1) a local joint subspace process, 2) improving detection using SVM, and 3) extracting final anomalies [12]. Another study uses background joint sparse representation (BJSR) for HSI anomaly detection [13]. Reference [14] is developed as a method for HSI anomaly detection using a background-anomaly component projection and separation optimized filter (BASO).

Lately, tensor decomposition has been proposed for anomaly detection [15]. Consecutively, another method is proposed based on a local filtering operation, called attribute and edge-preserving (AED) [16]. AED with multiscale attribute and edge-preserving filters is also presented in [17]. Reference [18] introduces random-selection-based anomaly detection (RSAD). First, it selects a proper subset of background pixels. Then, it uses blocked adaptive computationally efficient outlier nominators to detect anomalies. Finally, a fusion procedure is employed to avoid contamination of the background statistics.

Comparably, deep learning is also gaining significant attention in HSI. For instance, the convolutional neural networks (CNN) are applied for HSI anomaly classification, and detection in [19]. Reference [20] is a novel example of their application to anomaly detection in HSI. However, there are still significant challenges for CNNs in processing high-dimensional information contained in multidimensional data cubes, although they show excellent performance on various visual tasks [21, 22]. In particular, they need vastly time-consuming training and a training dataset. Furthermore, they might not apply to data gathered from a different sensor, i.e., the network might not adapt to a new domain.

Here, we propose an anomaly detection strategy based on pixel pairs root mean square error (PPRMSE) and object area filtering (OAF). The main contributions are as follows:

- First, a dual structure window is used through the picture to compute the similarity of the central pixel in the window with peripheral pixels by applying root mean

Hamid Esmaeili Najafabadi is currently a PostDoc associate at the Department of Electrical and Computer Engineering, University of Calgary, Calgary, AB T2N 1N4, Canada. Email: hamid.esmaeili@gmail.com. Phone: +1(587) 322-3311.

Zhenkai Zhang is currently an associate professor at the Department of electronics and information, Jiangsu University of Science and Technology, Zhenjiang 212003, China, Email: zhangzhenkai@just.edu.cn.

Mahdi Yousefan is currently with the Department of Electrical and Computer Engineering, University of Yazd, Yazd, Iran. Email: m.yousefan1998@gmail.com. Phone: +989391541146.

Amirshayan Nasirimajd is currently with the Department of Control and Computer Engineering, Polytechnic University of Turin, Turin, Italy. Email: sshayan1997@gmail.com. Phone: +393792880711.

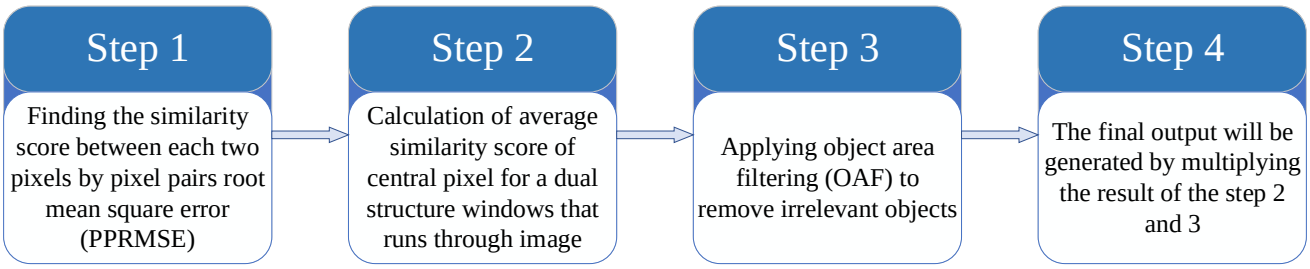

Fig. 1: Flowchart of the proposed approach steps

square error on each pair of pixels, and then the average of similarity in the window is calculated.

- Next, object area filtering is applied to get a binary image with omitted undesirable objects regarding their area size. Finally, the final result is calculated by multiplying the PPRMSE result and OAF output. This approach outperforms current methods in terms of accuracy and time.

This paper is organized into four Sections. In Section II, the details of the newly-developed method are explained. This section contains different subsections, each presenting a critical step of the proposed method. In Section III, a practical dataset is applied to evaluate the proposed method. Then, the result is compared with the most recent state-of-the-art algorithms. Finally, the conclusion is presented in Section IV.

II. Proposed Approach

A hyperspectral image is a data cube that contains both the spatial and spectral information of a sample. In the hyperspectral cube, the first two dimensions (x and y axes) are spatial, while the third dimension (z axis) is wavelength or spectral. The proposed method uses a pixel-wise process. That is, a pixel is used in the form of $1\times1\times L$, where L is the number of spectral bands. Principally, the proposed approach can be divided into four steps given in Fig. 1:

1) calculating the similarity between the target pixel and its neighboring pixels in a dual structure window using pixel pairs root mean square error (PPRMSE),
2) calculating the average similarity between the central pixel in the window and its neighbors,
3) applying object area filtering to eliminate unacceptable objects,
4) generating the final result by multiplying the results of steps (2) and (3).

Each step is further detailed in the following sections:

A. Similarity Calculation of the Current Pixel

In the HSI, each pixel in the form of $1 \times 1 \times C$ can be considered as a discrete signal. The similarity of two pixels in a hyperspectral dataset can simply be calculated through signal cross-correlation, or equivalently Euclidean distance. In this paper, the pixel pair root mean square error (PPRMSE) is used to find the similarity score of two pixels. Consider the two pixels $\mathbf{d} = [d_1, ..., d_L]^T$ and $\mathbf{u} = [u_1, ..., u_L]^T$. Then, we have

$$PPRMSE = ||\mathbf{d} - \mathbf{u}||_2 = \sqrt{\frac{1}{L}\Sigma_{i=1}^{L}\Big(d_i - u_i\Big)^2}. \quad (1)$$

where d_i and u_i refer to value of i-th index of current and neighboring pixels, and $||.||_2$ denotes Euclidean distance, respectively. Then, the PPRMSE value is given to a sigmoid function to map it into the $[0, 1]$-interval, where the sigmoid function is defined by:

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \quad (2)$$

Strictly speaking, we define a new metric on $\mathbb{R}^L$, named as sigmoid metric or distance. In this light, the distance between two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^L$ is defined as

$$d_\sigma(\mathbf{x}, \mathbf{y}) := \sigma(||\mathbf{x} - \mathbf{y}||_2). \quad (3)$$

We observe that $(\mathbb{R}^L, d_\sigma(.))$ is indeed a metric space through following Lemma.

Lemma II.1. The function $d_\sigma(.,.)$ defined in (3) is a metric for the set $\mathbb{R}^L$.

Proof. Three proposition should be verified for all $\mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathbb{R}^L$: 1) $d_\sigma(\mathbf{x}, \mathbf{y}) = 0 \iff \mathbf{x} = \mathbf{y}$, 2) $d_\sigma(\mathbf{x}, \mathbf{y}) = d_\sigma(\mathbf{y}, \mathbf{x})$, and 3) $d_\sigma(\mathbf{x}, \mathbf{y}) \leq d_\sigma(\mathbf{x}, \mathbf{z}) + d_\sigma(\mathbf{z}, \mathbf{y})$. The first two are straightforward. We prove the last by noting that $\sigma(x)$ is an increasing concave function for $x \geq 0$. Because of concavitity it is also sub-additive, i.e., $\sigma(x+y) \leq \sigma(x)+\sigma(y), \forall x, y > 0$. Therefore, we have

$$\begin{aligned} d_\sigma(\mathbf{x}, \mathbf{y}) &= \sigma(||\mathbf{x} - \mathbf{y}||_2) \\ &\leq \sigma(||\mathbf{x} - \mathbf{z}||_2 + ||\mathbf{z} - \mathbf{y}||_2) \\ &\leq \sigma(||\mathbf{x} - \mathbf{z}||_2) + \sigma(||\mathbf{z} - \mathbf{y}||_2) \\ &= d_\sigma(\mathbf{x}, \mathbf{z}) + d_\sigma(\mathbf{z}, \mathbf{y}) \end{aligned} \quad (4)$$

where we also used sub-additivity for $||.||_2$. □

The objects in a picture are determined by employing the newly-defined metric between the two pixels. In particular, two main operations are performed here. First, a dual window structure, given Fig. 2, is run through the picture. Then, the sigma distance of central pixel, denoted by D, with neighbors are calculated using (3).

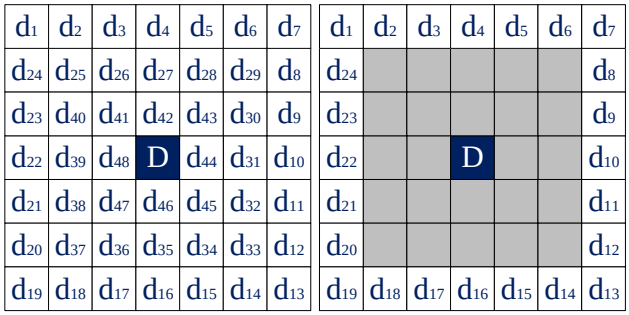

Fig. 2: The left depicts a $(w_{in}, w_{out}) = (5, 7)$ window while right shows a $(1, 7)$ window, d_1 to d_N are the peripheral pixels which their similarity with D is of interest

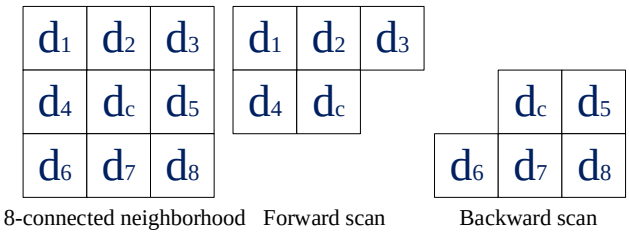

Fig. 3: The masks and the neighborhood of pixel d_c used in OAF algorithm

B. Mean Distance Score

In this step, we compute the average sigmoid metric between the central pixel of the window, D, and the pixels between the two square windows, known as peripheral pixels. In this regard, the notation (w_{in}, w_{out}) specifies a windowing structure with odd edge lengths w_{in} and w_{out} for the inner and outer windows, respectively. For example, $(3, 9)$ indicates a square of length 3 surrounded by a square of length 9. The number of D peripheral pixels is calculated as below:

$$N = (w_{out}^2 - w_{in}^2). \tag{5}$$

Therefore, the distance score between each peripheral pixel and generic pixel D is calculated and collected into the vector $s_E = [s_1, s_2, ..., s_N]^T$. The final score is generated by averaging these scores over peripheral pixels:

$$M(D) = \frac{1}{N}\Sigma_{i=1}^{N} s_i. \tag{6}$$

This method is applied for each pixel in the image by using the same window. In this regard, $M(D)$ can be defined as the membership of the pixel, D, to its corresponding object, which is a real number ranging between zero and one. This number has a higher value in the central pixels of an object than other pixels. Here, M denotes the resulting image, known as the membership map.

C. Object Area Filtering

The main goal of this step is to find connected components and then calculate their areas. For this purpose, a binary image is applied instead of a membership map. As a result, a binary map of objects is generated by applying thresholding on the membership map. Then, a predefined threshold, T, is compared to $M(D)$. The differentiate between $M(D)$ and T defines which pixel is an anomaly pixel. The pixel D is more likely to be an anomaly pixel; if $M(D)$ is higher than T; otherwise, it is a background pixel. Next, non-anomaly objects are omitted by object area filtering (OAF). This method is a two-pass algorithm inspired by the connected component filtering given in [23]. The 8-connected neighborhood is depicted in Fig. 3, backward and forward scan masks are used in this method.

Algorithm 1: Forward Scan Masking

Input: A binary image, M

Result: Connected components of the image given by $labels$

Initialize $labels$ to a all zero matrix with the same dimensions as M;

Initialize uf as an instance of $UFarray$;

for *(x,y) in M* **do**

- **if** *M(x,y) is 1* **then**
  - do nothing
- **if** *$y > 0$ and M(x,y-1) is 0* **then**
  - $labels[(x, y)] = labels[(x, y-1)]$
- **if** *$x+1 <$ width and $y > 0$ and M(x+1,y-1) is 0* **then**
  - $c = labels[(x+1, y-1)]$
  - $labels[x, y] = c$
  - **if** *$x > 0$ and M(x-1,y-1) is 0* **then**
    - $a = labels[(x-1, y-1)]$
    - $uf.union(c, a)$
  - **if** *$x > 0$ and M(x-1,y) is 0* **then**
    - $d = labels[(x-1, y)]$
    - $uf.union(c, d)$
- **if** *$x > 0$ and $y > 0$ and M(x-1,y-1) is 0* **then**
  - $labels[x, y] = labels[(x-1, y-1)]$
- **if** *$x > 0$ and M(x-1,y)* **then**
  - $labels[x, y] = labels[(x-1, y)]$
- **else**
  - $labels[x, y] = uf.makeLabel()$

In this approach, a union-find data structure (UFDS) is employed after the initial scan to record identical information between provisional labels. Here, we implement the UFDS through a $makeLabel$ and $union$ functions. In this light, a new label is created by the $makeLabel$ function for each new node. Then, every two nodes with equal labels are unified by the $union$ function. A UFDS can be seen hypothetically as a rooted tree, where each node of the tree is a temporary label, and each edge represents the equivalence between two labels. The two passes of UFDS are detailed as below:

1) Forward Scan Masking: This phase gets a binary version of the image as input and gives connected components as its output. The objects in the detected image, and d_c is the current pixel with d_1, d_2, d_3 and d_4 as its neighbors. Two types of pixels exist: 1 and 0. During the scanning, we ignore zero pixels that are part of the background. Any pixel that lies

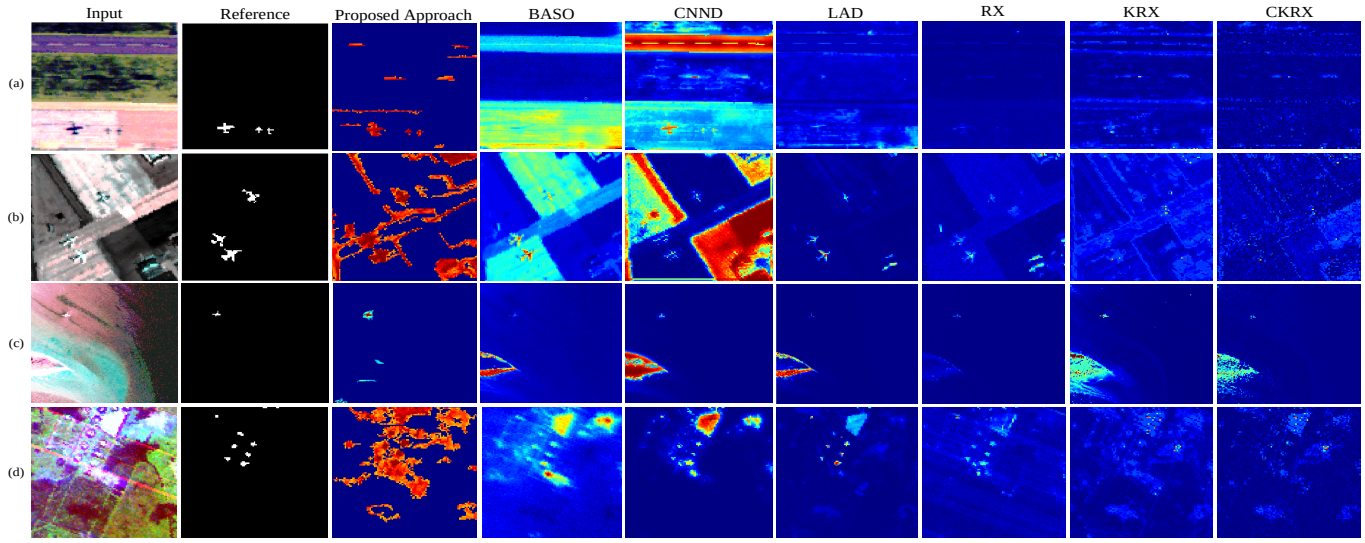

Fig. 4: Here are the test data images. The first column contains color composites images, and the second column shows ground truth, and the rest of the columns show the result of different methods

outside the image's bounds is one by default. As a result, six states exist during forward scan masking (FSM):

1. If pixel d_c is equal to one, then it is not a component. As a result, we ignore it.
2. If pixel d_2 is part of the image and equal to zero, d_1, d_3 and d_4 are all member of the same component as neighbors. Therefore, we assign d_2's label to d_c.
3. If pixel d_3 is part of the image and equal to zero, and d_2 is one of its neighbor, but not d_1 and d_4, then:
   - If pixel d_1 is part of the image and equal to zero, there is a connection between d_1 and d_3 through d_c. As a result, the $union$ is considered for their sets.
   - If pixel d_4 is part of the image and equal to zero, there is a connection between d_4 and d_3 through d_c. As a result, the $union$ is considered for their sets.
4. If pixel d_1 is part of the image and equal to zero, and already it is clear that d_2 and d_3 are one, d_4 and d_1 are neighbors, so both of them own the same label. Therefore, d_1's label will be assigned to d_c.
5. If pixel d_4 is part of the image and equal to zero, and already it is clear that d_1, d_2 and d_3 are one. Therefore, d_4's label will be assigned to d_c.
6. Otherwise, any neighboring pixels is one. As result d_c pixel is a new component.
   To simplify implementation, we bring the involved steps in FSM Algorithm 1:

2) Component Size Classification: Here, the labels matrix produced by the previous part is used to calculate the size of each component. Then, the connected components are classified based on their area. Acceptable anomaly area range (AARR) is defined as the interval where the anomalies' area lies. Each object with a size out of this range of anomalies is removed. In this light, the anomalies are filtered, and a binary map, denoted by Q, is produced. The pseudo-code for this pass is given in Algorithm 2.

D. Generating Final result

The final result is generated by multiplying the membership map and the output of OAF:

$$output = Q \times M. \tag{7}$$

Here, the multiplication is pixel-wise and the output is the final result of our proposed method.

III. Numerical Experimentation

The proposed approach is evaluated in terms of process time and accuracy. It is compared with six well-known algorithms. The benchmark algorithms are CNND [20], BASO [14], LAD [24], RX [8], KRX [9], and CKRX [25]. All mentioned benchmarks are grouped into recent approaches to hyperspectral image anomaly detection and established traditional benchmarks. A personal Core-i5 computer with 8 Gbytes RAM is used for all simulations. Matlab 2016 is used for BASO, LAD, RX, LRX, and CKRX, which is their original language. The Keras of the TensorFlow libraries in Python is used for CNND implementation. The parameters for all algorithms are selected as the corresponding paper suggested to insure that all algorithms work on their best capacity. The Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) is used [26] as the dataset for evaluation.

A. Test Data Sets

Four AVIRIS dataset spectrometer images are used for testing purposes. The size of all images is 100×100 except for image (c), which is 150×150. The AVIRIS uses a second-generation remote sensing instrument to gather its data. An opto-mechanical line arrays of detectors captures a 550-pixel swath in 224-bands ranging from 0.4 to 2.4 microns. The used images are depicted in Fig. 4 and briefly explained bellow:

- Image (a): an image captured from Gulfport airport scene with 3.4 m resolution.

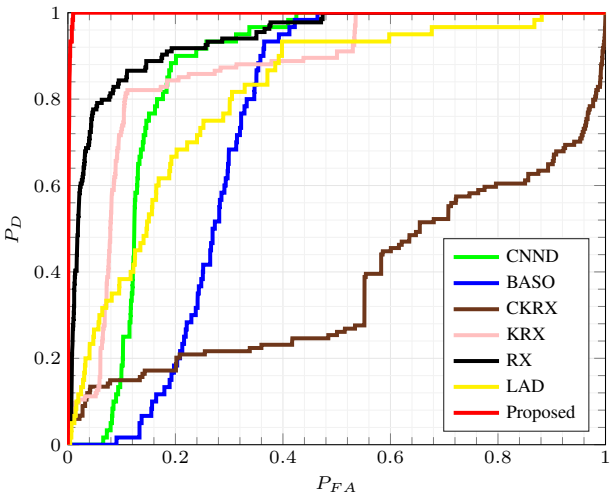

Fig. 5: ROC of proposed algorithm and different benchmarks when operating on image (a)

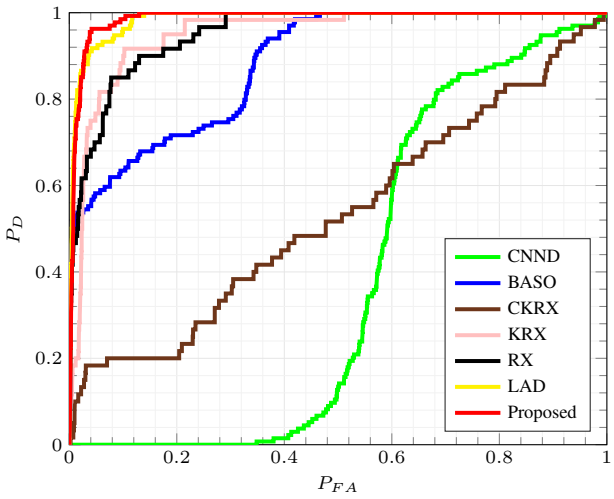

Fig. 6: ROC of proposed algorithm and different benchmarks when operating on image (b)

- Image (b): image captured from San Diego airport scene with unknown resolution.
- Image (c): image captured from Cat Island beach scene with 17.2m resolution.
- Image (d): image captured from Texas Coast urban scene with 17.2m resolution.

B. Parameter Tuning

During prediction, three main parameters are involved that should be carefully tuned: window sizes, a threshold for constructing the binary image, and the AARR. In this regard, the following remarks should be noted:

- Two type of window shapes are used in this method: the single and dual. In the single window, the distance score of the center pixel and its neighbors is calculated using sigmoid metric. This score is only calculated between the center pixel and the pixels between the inner and exterior windows in the dual window. It is worth noting that different sizes of windows bring different performances for the proposed method.
- A good binary image for the OAF requires the best possible threshold value. This value is vital for this method because the binary image has a huge influence on the final result.

Algorithm 2: Component Size Classification

Input: $labels$

Result: Q, a matrix with the same size as original image, where each pixel represents the size of the component it blongs to

Initialize Q to an all zero matrix with same size as $labels$;

for *label in labels* **do**
  $ComponentSize(label) = 0$

for *x,y in labels* **do**
  $ComponentSize(labels[(x,y)]) = ComponentSize(labels[(x,y)]) + 1$

for *x,y in labels* **do**
  if *ComponentSize(labels[(x,y)]) is in AARR* **then**
    $Q(x,y) = 1$
  else
    $Q(x,y) = 0$

C. Detection Performance

The receiver operating characteristic (ROC) [27] and area under the curve (AUC) [28] are used to evaluate and illustrate the performance. In this regard, the experimental results of the proposed and benchmark strategies are compared by ROC. Moreover, AUC is used to summarize the performance as a single number, where better performance corresponds to larger value. The ROC figures reveal that the proposed method outperforms other methods in terms of accuracy as depicted in Figs. 5, to 8 for images (a) to (d), respectively.

Furthermore, we evaluate the proposed method in various scenarios. They include varing window sizes and presence or absence of the OAF. First, the impact of different window sizes on the AUC, when OAF is present, is reported in Table I for all images. Table II reports the same parameters when OAF is absent. Bold text indicates the best accuracy in each Table. According to Table III(a), the proposed approach shows the highest AUC numbers across all images. The results in Table III(b) confirms that the proposed method consumes less time compared to some of the state-of-the-art benchmarks, such as CNND, KRX, and CKRX.

IV. CONCLUSION

We introduced a new detector algorithm for hyperspectral data anomaly detection. The chief idea of this paper is based on a newly-introduced distance score of each pixel with respect to their peripheral pixels. This score is created by feeding the pixel pairs root mean square error (PPRMSE) into a sigmoid function. We then applied object area filter (OAF)

TABLE I: AUC results of the Proposed method for images (a), (b), (c), and (d) with applying different window sizes and AARR intervals

(a)

(w_{in}, w_{out})	Time (sec)	AUC	Threshold	AARR (pixels)
(1, 3)	5.64	0.9709	0.50	(70, +∞)
(1, 5)	16.62	0.9804	0.70	(25, +∞)
(1, 7)	30.78	0.9835	0.30	(90, +∞)
(1, 9)	49.21	**0.9869**	0.70	(40, +∞)
(3, 5)	11.57	0.9802	0.60	(25, +∞)
(3, 7)	25.74	0.9838	0.70	(20, +∞)
(3, 9)	45.22	0.9853	0.70	(20, +∞)
(5, 7)	16.37	0.9805	0.70	(15, +∞)
(5, 9)	43.79	0.9836	0.60	(30, +∞)
(7, 9)	22.35	0.9800	0.72	(30, +∞)

(b)

(w_{in}, w_{out})	Time (sec)	AUC	Threshold	AARR (pixels)
(1, 3)	6.69	0.9956	0.60	(7, 70)
(1, 5)	22.07	0.9961	0.70	(4, 80)
(1, 7)	40.38	0.9967	0.70	(3, 70)
(1, 9)	56.00	**0.9982**	0.75	(11, 80)
(3, 5)	12.71	0.9955	0.70	(5, 110)
(3, 7)	28.81	0.9966	0.75	(10, 80)
(3, 9)	45.33	0.9977	0.75	(13, 80)
(5, 7)	17.31	0.9975	0.76	(13, 80)
(5, 9)	38.10	0.9964	0.76	(11, 80)
(7, 9)	23.50	0.9938	0.80	(9, 80)

(c)

(w_{in}, w_{out})	Time (sec)	AUC	Threshold	AARR (pixels)
(1, 3)	12.46	0.9995	0.30	(1, 50)
(1, 5)	36.73	0.9997	0.30	(15, +∞)
(1, 7)	74.38	**0.9998**	0.30	(20, +∞)
(1, 9)	126.60	**0.9998**	0.35	(20, +∞)
(3, 5)	22.36	0.9995	0.30	(20, +∞)
(3, 7)	55.68	**0.9998**	0.35	(15, +∞)
(3, 9)	97.93	**0.9998**	0.33	(15, +∞)
(5, 7)	37.34	**0.9998**	0.35	(15, +∞)
(5, 9)	77.79	**0.9998**	0.40	(15, +∞)
(7, 9)	49.61	**0.9998**	0.40	(15, +∞)

(d)

(w_{in}, w_{out})	Time (sec)	AUC	Threshold	AARR (pixels)
(1, 3)	6.73	0.9892	0.60	(15, 200)
(1, 5)	17.44	0.9940	0.60	(15, 50)
(1, 7)	33.55	0.9967	0.68	(15, 50)
(1, 9)	54.08	**0.9979**	0.68	(10, 50)
(3, 5)	12.14	0.9949	0.68	(15, 50)
(3, 7)	27.47	0.9967	0.70	(15, 50)
(3, 9)	48.01	0.9975	0.70	(15, 50)
(5, 7)	17.82	0.9965	0.70	(15, 50)
(5, 9)	41.16	0.9970	0.70	(15, 50)
(7, 9)	23.18	0.9970	0.75	(10, 50)

TABLE II: AUC results of the Proposed method for images (a), (b), (c), and (d) with applying different window sizes and without using OAF

(a)

(w_{in}, w_{out})	AUC
(1, 3)	0.9630
(1, 5)	0.9771
(1, 7)	0.9825
(1, 9)	**0.9845**
(3, 5)	0.9788
(3, 7)	0.9827
(3, 9)	0.9840
(5, 7)	0.9798
(5, 9)	0.9816
(7, 9)	0.9766

(b)

(w_{in}, w_{out})	AUC
(1, 3)	0.9875
(1, 5)	**0.9903**
(1, 7)	0.9903
(1, 9)	0.9896
(3, 5)	0.9840
(3, 7)	0.9858
(3, 9)	0.9859
(5, 7)	0.9806
(5, 9)	0.9824
(7, 9)	0.9793

(c)

(w_{in}, w_{out})	AUC
(1, 3)	**0.9916**
(1, 5)	0.9905
(1, 7)	0.9910
(1, 9)	0.9911
(3, 5)	0.9881
(3, 7)	0.9897
(3, 9)	0.9903
(5, 7)	0.9901
(5, 9)	0.9908
(7, 9)	0.9903

(d)

(w_{in}, w_{out})	AUC
(1, 3)	0.9841
(1, 5)	0.9936
(1, 7)	0.9960
(1, 9)	**0.9973**
(3, 5)	0.9940
(3, 7)	0.9963
(3, 9)	0.9972
(5, 7)	0.9959
(5, 9)	0.9967
(7, 9)	0.9963

TABLE III: Assessment of AUC (a) accuracy and (b) computation time between the proposed and benchmark methods

(a)

Image	Proposed Approach	BASO	CNND	LAD	RX	KRX	CKRX
(a)	0.9869	0.7292	0.8546	0.8076	0.9526	0.9516	0.5310
(b)	0.9982	0.8840	0.3849	0.9855	0.9403	0.8681	0.3722
(c)	0.9998	0.9589	0.9571	0.9203	0.9807	0.9293	0.8168
(d)	0.9979	0.9376	0.9521	0.9853	0.9907	0.9282	0.8180

(b)

Image	Proposed Approach	BASO	CNND	LAD	RX	KRX	CKRX
(a)	49.21	16	1505	0.19	0.26	2007.2	247.9
(b)	56.00	18.3	107	0.21	0.22	1739.6	275.6
(c)	37.34	73	1203	0.39	0.66	7854	465.6
(d)	54.08	32	807	0.19	0.26	2080	264.2

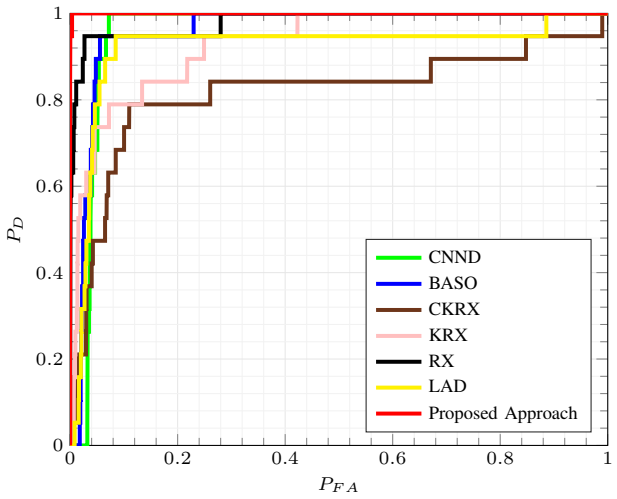

Fig. 7: ROC of proposed algorithm and different benchmarks when operating on image (c)

to find connected components and calculated their areas to remove objects with an unacceptable defined range. Extensive empirical simulations are used to confirm that the proposed method is both time-efficient and highly accurate in detecting anomalies in hyperspectral images compared to state-of-the-art approaches.

References

[1] M. Eismann and S. (Society), *Hyperspectral Remote Sensing*, ser. Press Monographs. SPIE, 2012.

[2] J. Liang, J. Zhou, X. Bai, and Y. Qian, "Salient object detection in hyperspectral imagery," in *2013 IEEE International Conference on Image Processing*. IEEE, 2013, pp. 2393–2397.

[3] S. M. Schweizer and J. M. F. Moura, "Hyperspectral imagery: clutter adaptation in anomaly detection," *IEEE Transactions on Information Theory*, vol. 46, no. 5, pp. 1855–1871, 2000.

[4] D. Manolakis and G. Shaw, "Detection algorithms for hyperspectral imaging applications," *IEEE signal processing magazine*, vol. 19, no. 1, pp. 29–43, 2002.

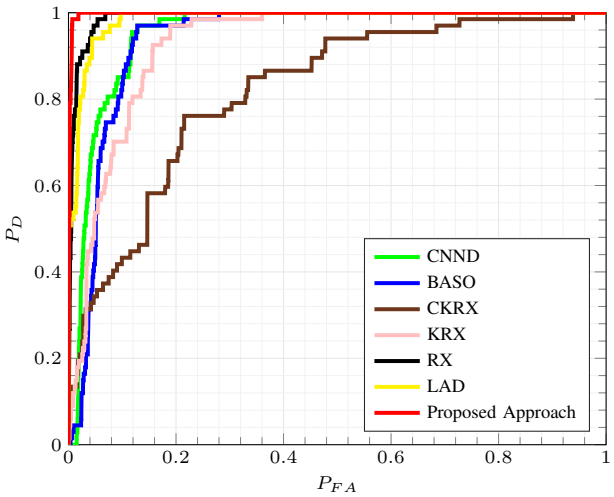

Fig. 8: ROC of proposed algorithm and different benchmarks when operating on image (d)

[5] W. Li and Q. Du, "Collaborative representation for hyperspectral anomaly detection," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 3, pp. 1463–1474, 2015.

[6] C.-I. Chang and S.-S. Chiang, "Anomaly detection and classification for hyperspectral imagery," *IEEE transactions on geoscience and remote sensing*, vol. 40, no. 6, pp. 1314–1325, 2002.

[7] D. W. Stein, S. G. Beaven, L. E. Hoff, E. M. Winter, A. P. Schaum, and A. D. Stocker, "Anomaly detection from hyperspectral imagery," *IEEE signal processing magazine*, vol. 19, no. 1, pp. 58–69, 2002.

[8] I. S. Reed and X. Yu, "Adaptive multiple-band cfar detection of an optical pattern with unknown spectral distribution," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 38, no. 10, pp. 1760–1770, 1990.

[9] J. M. Molero, E. M. Garzón, I. García, and A. Plaza, "Analysis and optimizations of global and local versions of the rx algorithm for anomaly detection in hyperspectral data," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 6, no. 2, pp. 801–814, 2013.

[10] D. W. J. Stein, S. G. Beaven, L. E. Hoff, E. M. Winter, A. P. Schaum, and A. D. Stocker, "Anomaly detection from hyperspectral imagery," *IEEE Signal Processing Magazine*, vol. 19, no. 1, pp. 58–69, 2002.

[11] Z. Wu, W. Zhu, J. Chanussot, Y. Xu, and S. Osher, "Hyperspectral anomaly detection via global and local joint modeling of background," *IEEE Transactions on Signal Processing*, vol. 67, no. 14, pp. 3858–3869, 2019.

[12] P. Xiang, H. Zhou, H. Li, S. Song, W. Tan, J. Song, and L. Gu, "Hyperspectral anomaly detection by local joint subspace process and support vector machine," *International Journal of Remote Sensing*, vol. 41, no. 10, pp. 3798–3819, 2020.

[13] J. Li, H. Zhang, L. Zhang, and L. Ma, "Hyperspectral anomaly detection by the use of background joint sparse

representation," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 6, pp. 2523–2533, 2015.

[14] S. Chang, B. Du, and L. Zhang, "Baso: A background-anomaly component projection and separation optimized filter for anomaly detection in hyperspectral images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 7, pp. 3747–3761, 2018.

[15] X. Zhang, G. Wen, and W. Dai, "A tensor decomposition-based anomaly detection algorithm for hyperspectral image," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 10, pp. 5801–5820, 2016.

[16] X. Kang, X. Zhang, S. Li, K. Li, J. Li, and J. A. Benediktsson, "Hyperspectral anomaly detection with attribute and edge-preserving filters," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 10, pp. 5600–5611, 2017.

[17] S. Li, K. Zhang, Q. Hao, P. Duan, and X. Kang, "Hyperspectral anomaly detection with multiscale attribute and edge-preserving filters," *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 10, pp. 1605–1609, 2018.

[18] B. Du and L. Zhang, "Random-selection-based anomaly detector for hyperspectral imagery," *IEEE Transactions on Geoscience and Remote sensing*, vol. 49, no. 5, pp. 1578–1589, 2010.

[19] Y. Li, W. Xie, and H. Li, "Hyperspectral image reconstruction by deep convolutional neural network for classification," *Pattern Recognition*, vol. 63, pp. 371–383, 2017.

[20] W. Li, G. Wu, and Q. Du, "Transferred deep learning for anomaly detection in hyperspectral imagery," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 5, pp. 597–601, 2017.

[21] M. Paoletti, J. Haut, J. Plaza, and A. Plaza, "A new deep convolutional neural network for fast hyperspectral image classification," *ISPRS journal of photogrammetry and remote sensing*, vol. 145, pp. 120–147, 2018.

[22] W. Hu, Y. Huang, L. Wei, F. Zhang, and H. Li, "Deep convolutional neural networks for hyperspectral image classification," *Journal of Sensors*, vol. 2015, 2015.

[23] K. Wu, E. Otoo, and K. Suzuki, "Optimizing two-pass connected-component labeling algorithms," *Pattern Analysis and Applications*, vol. 12, no. 2, pp. 117–135, 2009.

[24] F. Verdoja and M. Grangetto, "Graph laplacian for image anomaly detection," *Machine Vision and Applications*, vol. 31, no. 1, p. 11, 2020.

[25] J. Zhou, C. Kwan, B. Ayhan, and M. T. Eismann, "A novel cluster kernel rx algorithm for anomaly and change detection using hyperspectral images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 11, pp. 6497–6504, 2016.

[26] G. Vane, M. Chrisp, H. Enmark, S. Macenka *et al.*, "Airborne visible/infrared imaging spectrometer (aviris): an advanced tool for earth remote sensing," *rsfr*, vol. 2, pp. 751–757, 1984.

[27] A. P. Yonelinas, "Receiver-operating characteristics in recognition memory: evidence for a dual-process model." *Journal of Experimental Psychology: Learning, Memory, and Cognition*, vol. 20, no. 6, p. 1341, 1994.

[28] I. Pollack and R. Hsieh, "Sampling variability of the area under the roc-curve and of d'e." *Psychological Bulletin*, vol. 71, no. 3, p. 161, 1969.