STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **12**, March 2024, pp 530–546.
Published online in International Academic Press (www.IAPress.org)

An Effective Randomized Algorithm for Hyperspectral Image Feature Extraction

Jinhong Feng , Rui Yan, Gaohang Yu*, and Zhongming Chen

Department of Mathematics, School of Science, Hangzhou Dianzi University, Hangzhou 310018, China

Abstract Analyzing the spectral and spatial characteristics of Hyperspectral Imaging (HSI) in a three-dimensional space is a challenging task. Recently, there have been developments in 3D feature extraction methods based on tensor decomposition, which allow for the effective utilization of both global and local information in HSI. These methods also explore the inherent low-rank properties of HSI through tensor decomposition. In this paper, we propose a new approach called variable randomized T-product decomposition (Vrt-SVD), which is a variation of Tensor Singular Spectral Analysis. The goal of this approach is to improve the efficiency of tensor methods for feature extraction and reduce artifacts of image processing. By using a randomized algorithm based on the variable t-SVD, we are able to capture both global and local spatial and spectral information in HSI efficiently, which enables us to explore its low-rank characteristics. To evaluate the effectiveness of the extracted features, we use a Support Vector Machine (SVM) classifier to assess the accuracy of image classification. By conducting numerous numerical experiments, we provide strong evidence to show that the proposed method outperforms several advanced feature extraction techniques.

Keywords Hyperspectral images(HSI); randomized algorithm; variable T-product ; 3D feature extraction

AMS 2020 subject classifications 15A18, 15A69, 68U10, 68W20

DOI: 10.19139/soic-2310-5070-1980

1. Introduction

Hyperspectral images offer comprehensive spectral information across numerous spectral channels (also referred to as dimensions or bands). This augmented dimensionality enables a substantial enhancement in the data's information content[15][10]. Consequently, it becomes feasible to distinguish between various classes of interest that possess slightly varied spectral characteristics. Conventional analytical techniques commonly used for grayscale images frequently prove inadequate when applied to hyperspectral images. HSI data are often affected by uncontrollable external factors such as environmental noise, as well as the redundancy and overlap of information in hyperspectral data due to the strong correlation of neighboring bands, or the difficulty of obtaining sufficient training samples, which will lead to unsatisfactory classification performance. Therefore, it becomes crucial to identify an effective feature extraction method that can enhance the distinguishability between various categories in HSI classification[24][5]. In the classification of HSI, the methods for extracting features mainly consist of spectral feature extraction[15], spatial feature extraction[1][17], 3D spatial-spectral feature extraction[11][8][5], and feature extraction based on image segmentation[29][30][12]. Spectral feature extraction methods are based on the spectral data of each pixel, and they effectively extract the relevant data based on certain rules. Spatial feature extraction primarily focuses on the spatial characteristics of HSI and is typically used in conjunction with spectral feature extraction. 3D spatial-spectral feature extraction involves directly extracting the three-dimensional data

*Correspondence to: Gaohang Yu (Email: maghyu@163.com).

ISSN 2310-5070 (online) ISSN 2311-004X (print)

of HSI to maximize the preservation of the original data's information. Image segmentation theory is applied to segment HSI based on the similarity between pixels, and then categorize them.

Scholars have proposed various feature extraction techniques in recent years. These techniques include Principal Component Analysis (PCA)[16], which focuses solely on spectral information and disregards the potential contribution of spatial data. PCA has advantages such as model determinism, simple algorithms, and ease of understanding and processing. However, it neglects the spatial distribution information of hyperspectral data and fails to reveal the internal structure of the data. Additionally, it is prone to overlooking important information when characterizing fine substances by disregarding the latter principal components. As a result, there has been significant attention given to the concept of integrating spatial and spectral information for feature extraction in order to enhance classification performance. However, these methods often process spatial and spectral information independently, overlooking the interdependence of spectral continuity and spatial similarity within HSI data. To overcome this limitation, approaches for extracting the spatial characteristics of three-dimensional spectral data have also been introduced.

In 2015, a two-dimensional extension of generalized spectral analysis (2DSSA) was proposed by Zabalza et al. [28] to effectively extract spatial information. The application of 2DSSA to HSI involved decomposing each band image into different trends, oscillations, and noises. By using trends and selected oscillations as features, the reconstructed signal demonstrated strong noise suppression ability and robustness, making it effective for data classification. However, when applying SSA for feature extraction in HSI, the conventional pixel-based 1DSSA failed to produce satisfactory results. Similarly, the two-dimensional SSA based on band images was not feasible. To address this issue, Fu et al. [4] proposed a novel method called 1.5DSSA in 2020 for spectral spatial feature extraction in HSI. This method utilized pixels from small windows as spatial information. For each sequentially acquired pixel, similar pixels were located within the pixel-centered window to form an extended trajectory matrix for feature extraction. In HSI, most feature extraction and data classification methods rely on calibration data sets. However, using calibration data sets not only requires additional work but also leads to information loss of deleted bands. To overcome these challenges, Ma et al. [19] proposed a new framework in 2021 for spatial spectral feature extraction, known as multiscale two-dimensional singular spectrum analysis and principal component analysis. This framework aimed to achieve robust feature extraction and data classification for HSI. Firstly, multiscale 2DSSA was applied to the multi-scale spatial features of each spectral band of HSI to extract and determine the changing trend within the window. Then, the extracted trend signals of various scales were used as input features for principal component analysis in the spectral domain. This step aimed to reduce dimensionality and extract spatial spectral features. The spatial spectral features of each scale were classified separately and then fused at the decision-making level to improve efficiency.

The traditional approaches to singular spectral analysis in the spectral domain and spatial domain suffer from certain drawbacks, including being affected by window size, requiring significant computational resources for large windows, and not being able to capture spatial features of joint spectral analysis. To address these limitations, a new method called super-pixel adaptive SSA was introduced by Sun et al. [25]. This method utilizes local spatial information from HSI to perform adaptive SSA on super-pixels. By extracting local features, especially in HSI, the method improves the ability to characterize objects in the image.

In 2022, Fu et.al.[6] proposed a method for extracting features and classifying spectral-spatial hyperspectral images. The method combines multiscale two-dimensional singular spectral analysis fusion with Principal Component Analysis and Segmented-PCA(SPCA). This approach involves separately reducing the spectral dimension using PCA and SPCA. Then, multiscale two-dimensional SSA is used to extract spatial features at different scales from the images after dimension reduction by SPCA. PCA is further applied to reduce the dimensions, and the resulting multiscale spatial features are combined with the global spectral features obtained through PCA to form multiscale spectral-spatial features. In 2022, Fu et.al.[7] also proposed an enhanced two-dimensional singular spectrum analysis (E2DSSA) method for extracting spatial background and structural information from selected frequency bands. This method aims to mitigate the impact of intra-class variability and spatial domain noise . In 2023, Fu et.al.[5] introduced a Tensor Singular Spectrum Analysis (TensorSSA) method for extracting global and low-rank features from HSI. TensorSSA utilizes an adaptive embedding operation to construct the trajectory tensor, which includes the entire HSI. This operation takes advantage of spatial similarity

while preserving and enhancing the low rank of the original tensor. The trajectory tensor captures both global and local spatial and spectral information, but significantly increases the size compared to the original HSI. The increase in size depends on the adaptive embedding parameter, which affects the computational time for subsequent t-SVD decomposition [13, 14]. Inspired by TensorSSA, our work aims to address this challenge by introducing randomized algorithms and zero padding techniques to reduce computation time and improve accuracy.

The use of adaptive embedding in t-SVD leads to a larger tensor, which increases the computational time required for decomposition. Additionally, TensorSSA, a t-SVD-based tensor decomposition method, may require zero padding to achieve accurate results in signal and image processing. In our approach, we also employ zero padding to ensure accuracy, but this results in an expansion of the trajectory tensor size. As a result, additional time and resources are needed for t-SVD, making deterministic t-SVD less practical for large-scale datasets. To mitigate these challenges, we introduce randomized methods[3, 9, 23, 26, 27] to reduce time costs, taking into account the low-rank nature of the trajectory tensor. However, simple randomized methods may not strike the optimal balance between time and accuracy. So we use the rt-SVD algorithm with Krylov subspace iteration to enhance feature extraction accuracy. In this paper, we present a novel method called variable T-product randomized algorithm(Vrt-SVD), which is a variation of Tensor Singular Spectral Analysis. This method is designed to extract global and low-rank features from Hyperspectral Images (HSI) efficiently. Specifically, we first create the trajectory tensor that spans the entire HSI using an adaptive embedding operation. Then, we employ Vrt-SVD to obtain global and local spatial and spectral information, enabling the exploration of low-rank features. We evaluate the effectiveness of feature extraction by measuring image classification accuracy using a Support Vector Machine (SVM) classifier. The experimental results validate the effectiveness of our proposed algorithm.

To summarize, this paper presents two main contributions. Firstly, we propose Vrt-SVD, a novel method for extracting 3D features from hyperspectral images (HSI) using TensorSSA. This method aims to capture the overall spectral-spatial correlation in HSI. Through experiments on four publicly available datasets, we demonstrate the superiority of Vrt-SVD, even when dealing with limited training samples. Secondly, we develop a technique that combines Vrt-SVD with a trajectory tensor. This is achieved by utilizing variable T-product and Krylov subspace iteration. This approach allows for the joint characterization of global low-rank features within the HSI. The proposed algorithm efficiently solves singular values in the variable Fourier domain and provides the best low-rank approximation of the trajectory tensor through truncation.

2. Preliminary

2.1. Variable T-product

Definition 1 *(Variable product of two p-dimensional vectors)[21]* For any $\mathbf{a}, \mathbf{b} \in R^p$ or C^p, then

$$(\mathbf{a} \odot_v \mathbf{b})(k) = \sum \mathbf{a}(i)\mathbf{b}(j) : i + j - k - 1 = 0 \quad mod(v), i, j = 1, \cdots, p,$$

where $k = 1, \cdots, p$. We call $\mathbf{a} \odot_v \mathbf{b}$ the variable product of $\mathbf{a}$ and $\mathbf{b}$.

Definition 2 *(Variable T-product of two third-order tensors in the real number field)[21]*

The variable T-product of $\mathcal{A} = (\mathbf{a}_{il}) \in R^{m\times q\times p}, \mathcal{B} = (\mathbf{b}_{lj}) \in R^{q\times n\times p}$ is defined as $\mathcal{C} = \mathcal{A} *_v \mathcal{B}$, and

$$\mathbf{c}_{ij} = \sum_{l=1}^{q} \mathbf{a}_{il} \odot_v \mathbf{b}_{lj}.$$

If $v = p$, then the variable T-product $*_v$ degenerates to the T-product $*$.

Definition 3 *(Zero-Padding Discrete Fourier Transform)[21]*

Let T be a Zero-Padding Discrete Fourier Transform matrix (ZDFT), and

$$T=\begin{bmatrix} 1 & 1 & \cdots & 1 & 1 \\ 1 & w_v & \cdots & w_v^{p-2} & w_v^{p-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & w_v^{v-2} & \cdots & w_v^{(v-2)(p-2)} & w_v^{(v-2)(p-1)} \\ 1 & w_v^{v-1} & \cdots & w_v^{(v-1)(p-2)} & w_v^{(v-1)(p-2)} \end{bmatrix},$$

where $w_v = e^{-\frac{2\pi i}{v}}$, and $i=\sqrt{-1}$ is the imaginary part unit, then T consists of the first p columns of the $v\times v$ discrete Fourier transform DFT matrix F_v, hence T is column full rank p, and $T^HT = vI_p$, where I_p is the $p\times p$ identity matrix. For any $\mathbf{a}\in C^p$, Zero-Padding the vector $\mathbf{a}$ yields $\widetilde{\mathbf{a}}=(\mathbf{a};0)\in C^v$, then $T\mathbf{a}=F_v\widetilde{\mathbf{a}}$. As a result, T is said to be a ZDFT matrix and the variable T-product can be interpreted by a ZDFT matrix.

Theorem 1[21] Assume $\mathcal{A}=(\mathbf{a}_{il})\in R^{m\times q\times p}, \mathcal{B}=(\mathbf{b}_{lj})\in R^{q\times n\times p}$, Let $\mathcal{A}_0=(\mathbf{a}_{il})\in R^{m\times q\times v}, \mathcal{B}_0=(\mathbf{b}_{lj})\in R^{q\times n\times v}$ be the Zero-Padding correspondence tensor of $\mathcal{A}$ and $\mathcal{B}$. For any $k=p+1,\cdots,v$, there are zero-positive slices of$\mathcal{A}_0(:,:,k)$ and $\mathcal{B}_0(:,:,k)$ and

$$\mathcal{A}*_v\mathcal{B}=(\mathcal{A}_0*\mathcal{B}_0)(:,:,[1:p]),$$

where $*$ is T-product.

Definition 4 *(Conjugate transpose of tensor)[13]* Given a tensor$\mathcal{A}\in R^{m\times q\times p}$, make conjugate transpositions of all the forward slices of $\mathcal{A}$. Then reverse the order of the forward slices from the 2nd to the pth transposition to obtain its conjugate transposition $\mathcal{A}^\top\in R^{q\times m\times p}$.

Definition 5 *(Identity tensor [13])*

The tensor $\mathcal{J}\in R^{i\times i\times p}$ where the first frontal slice is a Identity matrix of $i\times i$ and the other forward slices are all zero matrices is called the Identity tensor.

Definition 6 *(f-diagonal tensor [13])*

A tensor is said to be f-diagonal if all its frontal slices are diagonal matrices.

Definition 7 *(orthogonal tensor [13])*

A tensor$\mathcal{A}\in R^{m\times q\times p}$ is an orthogonal tensor if

$$\mathcal{A}^\top*_v\mathcal{A}=\mathcal{A}*_v\mathcal{A}^\top=\mathcal{J},$$

Definition 8 *t-SVD based on variable T-product*[21]

the tensor $\mathcal{A}=\in R^{m\times q\times p}$ whose t-SVD based on variable T-product is

$$\mathcal{A}=\mathcal{U}*_v\mathcal{S}*_v\mathcal{V}^\top,$$

where $\mathcal{S}\in R^{m\times q\times p}$ is f-diagonal tensor, $\mathcal{U}\in R^{m\times m\times p}$ and $\mathcal{V}\in R^{n\times n\times p}$ is orthogonal tensor, $*_v$ is variable T-product.

Definition 9 *(Variable Tensor tuble Rank [21])*

Let $\hat{\mathcal{C}}=fft_{p2v}(\mathcal{C},[],3)$ denote the tensor $\mathcal{C}\in R^{m\times n\times p}$ as ZDFT along mode 3, and $\mathcal{C}=ifft_{v2p}(\hat{\mathcal{C}},[],3)$ denote the tensor $\hat{\mathcal{C}}$ as inverse zero-additive Discrete Fourier Transform (IZDFT) along mode 3, and write

$$\hat{\mathcal{C}}=diag(\hat{\mathcal{C}}^{(1)},\hat{\mathcal{C}}^{(2)},\cdots,\hat{\mathcal{C}}^{(v)})\in\mathcal{C}^{mv\times mv},$$

denotes the block diagonal array, then the variable tubel rank of the tensor $\mathcal{C}$ is defined as

$$Rank_v(\mathcal{C})=max\{Rank(\hat{\mathcal{C}}^{(1)}),Rank(\hat{\mathcal{C}}^{(2)}),\cdots,Rank(\hat{\mathcal{C}}^{(v)})\},$$

clearly, when $v=p$, $Rank_v(\mathcal{C})$ degenerates to a tuble rank $Rank_t(\mathcal{C})$.

The main symbols used in this paper are shown in Table 1.

Table 1. Main notations.

Symbols	Notation
x	scalar
$\mathbf{x}$	vector
X	matrix
$\mathcal{X}$	tensor
$X^\top$	transpose of X
$X^\dagger$	pseudo-inverse of X
$\odot_v$	variable product.
$*_v$	variable T-product

2.2. Randomized block Krylov Iteration

In situations where achieving the highest level of accuracy is crucial, simple randomized methods can be insufficient, despite their ability to reduce computational time for singular value decomposition in low-rank data. In order to address this issue, subspace iteration methods have been developed to improve accuracy. One such method is the block Krylov iteration technique, introduced by Musco et al.[20], which is particularly effective for data with small singular value gaps. This method has been shown to be robust to noise and outperforms standard power iteration techniques in experimental. Qiu et al.[22] applied the block Krylov iteration to the Tucker decomposition and confirmed its resilience to noise, highlighting its advantages in enhancing the accuracy of data analysis .

Algorithm 1 Randomized block Krylov Iteration(rBKI)[20]

Input: matrix $A \in R^{m\times n}$, target rank r, oversampling parameter p.
Output: U, S, V

1: Create random Gaussian matrix $\Omega \in R^{n\times(r+p)}$.
2: Construct Krylov space $K = [A^\top A\Omega, (A^\top A)^2\Omega, \cdots, (A^\top A)^q\Omega]$.
3: Calculate $[Q, \sim] = qr(K, 0)$.
4: Calculate $Y = AQ$.
5: Set $(U, S, V) = svd(Y, 0)$.
6: $U = U(:, 1:r).S = S(1:r, 1:r), V = QV(:, 1:r)$.

2.3. TensorSSA

In order to improve the effectiveness of HSI 3D feature extraction and reduce computational cost, Fu et.al.[5] designed a new 3D feature extraction framework based on the T-product, called TensorSSA. It includes four steps: (1) adaptive embedding; (2) t-SVD decomposition; (3) low rank representation; (4) reprojection. In TensorSSA, the input HSI data is represented by a 3D tensor $\mathcal{X}$, where t, h and b represent the width, height, and band number, respectively. The overall architecture of the TensorSSA method is shown in Figure 1.

In order to construct the TensorSSA model, it is essential to effectively integrate both spatial and spectral information. The utilization of spatial information plays a crucial role in this process. Many spatial methods, such as 2DSSA, typically use fixed rectangular windows for extracting features. However, this approach may not be suitable for all objects in hyperspectral images, especially those with irregular shapes and varying sizes. Spatial self-similarity is a common characteristic in hyperspectral imagery. In response to this, Fu et.al.[5] proposes the use of an adaptive embedding window that aligns with spatial self-similarity, which is in line with the principles of TensorSSA. The adaptive window enables the extraction of features that can accommodate the irregular shapes and varying sizes of objects in hyperspectral data. The comparison between 2DSSA and TensorSSA is illustrated in Figure 2.

Utilizing an adaptive embedding window is evidently more versatile compared to the fixed window approach employed by 2DSSA, which makes it better suited to adapt to the various spatial structures found in HSI.

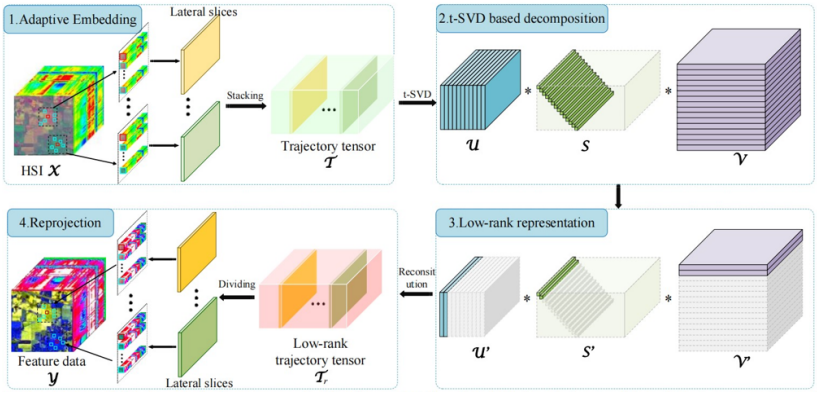


Figure 1. Overall framework of TensorSSA method[5]

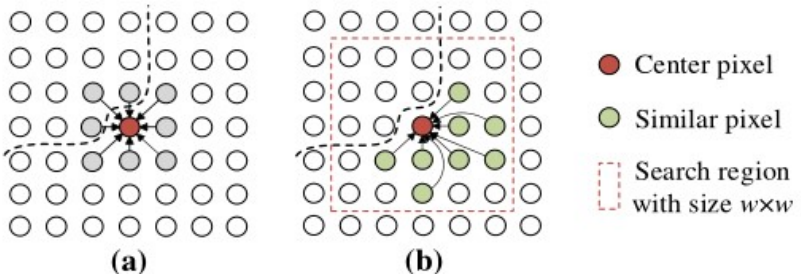


Figure 2. Embedding comparison of 2DSSA(a) and TensorSSA(b)[5]

Specifically, for a specific pixel $x_i \in R^{b\times 1}(i = 1, 2, \cdots, t \times h)$ to be processed, a search area of $w \times w$ is set as the center, and $l - 1$ pixels with a high similarity to it are found in the search area (l is set to any size smaller than $w \times w$), where similarity is measured by the normalized Euclidean distance, so the matrix $N_i \in R^{l\times b}$ is obtained, which is taken as the lateral slice of the trajectory matrix $\mathcal{J} \in R^{l\times th\times b}$, and the obtained matrix has the following characteristics: (1) It contains the spectral and spatial information of the entire HSI; (2) The forward slice is a quasi-Hankel matrix; (3) Because of the high correlation between the pixel similarity in mode 1 and the spectrum in mode 3, the trajectory tensor $\mathcal{J}$ has the property of low rank.

After obtaining the trajectory tensor by adaptive embedding, TensorSSA adopts a deterministic SVD decomposition in the DFT domain as follows.

Algorithm 2 Truncated t-SVD

Input: tensor $\mathcal{J} \in \mathbb{R}^{l\times th\times b}$, truncation parameter $Rank_k$.
Output: $\mathcal{U}', \mathcal{S}', \mathcal{V}'$

1: $\bar{\mathcal{J}} = fft(\mathcal{J}, [], 3)$.
2: **for** $i = 1, 2, \cdots, b$ **do**
3: $\quad[\bar{\mathcal{U}}(:,:,i), \bar{\mathcal{S}}(:,:,i), \bar{\mathcal{V}}(:,:,i)] = svd(\bar{J}(:,:,i),' econ')$.
4: **end for**
5: $\mathcal{S} = ifft(\bar{\mathcal{S}}, [], 3), \mathcal{V} = ifft(\bar{\mathcal{V}}, [], 3), \mathcal{U} = ifft(\bar{\mathcal{U}}, [], 3)$.
6: $\mathcal{S}' = \mathcal{S}(1 : Rank_k, 1 : Rank_k, :), \mathcal{V}' = \mathcal{V}(:, 1 : Rank_k, :), \mathcal{U}' = \mathcal{U}(:, 1 : Rank_k, :)$.

The t-SVD Algorithm 2 adopted by TensorSSA is a tensor decomposition method based on discrete Fourier transform (DFT), and it may not be enough to directly apply DFT in signal processing and image processing. Therefore, zero padding [18] and [2] is needed to obtain accurate results, otherwise artifacts may occur. In our

method, to obtain more accurate results, we need to add zero to the trajectory tensor, which will lead to a larger size of the third module. The algorithm is summarized in Algorithm 3:

Algorithm 3 Variable truncated t-SVD (Vt-SVD)

Input: tensor $\mathcal{J} \in \mathbb{R}^{l\times th\times b}$, zero padding parameter v, truncation parameter $Rank_v$.
Output: $\mathcal{U}', \mathcal{S}', \mathcal{V}'$

$\bar{\mathcal{J}} = fft_{p2v}(\mathcal{J}, [], 3)$.
for $i = 1, 2, \cdots, v$ **do**
  $[\bar{\mathcal{U}}(:,:,i), \bar{\mathcal{S}}(:,:,i), \bar{\mathcal{V}}(:,:,i)] = svd(\bar{J}(:,:,i),' econ')$.
end for
$\mathcal{S} = ifft_{v2p}(\bar{\mathcal{S}}, [], 3), \mathcal{V} = ifft_{v2p}(\bar{\mathcal{V}}, [], 3), \mathcal{U} = ifft_{v2p}(\bar{\mathcal{U}}, [], 3)$.
$\mathcal{S}' = \mathcal{S}(1 : Rank_v, 1 : Rank_v, :), \mathcal{V}' = \mathcal{V}(:, 1 : Rank_v, :), \mathcal{U}' = \mathcal{U}(:, 1 : Rank_v, :)$.

To compare the zero-padding transformations, we give the randomized Algorithm 4 for the T-product in the DFT domain, the algorithm will be given in the next section.

Algorithm 4 t-SVD with Randomized block krylov iteration(rBKI-SVD)

Input: tensor $\mathcal{J} \in \mathbb{R}^{l\times th\times b}$, iteration q, oversampling parameter p, rank r, $k = r + p$.
Output: $\mathcal{U}', \mathcal{S}', \mathcal{V}'$

$\bar{\mathcal{J}} = fft(\mathcal{J}, [], 3)$.
create random Gaussian tensor $\Omega = zeros(th, k, n3), \Omega(:,:,1) = randn(th, k), \Omega_V = fft(\Omega, [], 3)$.
for $i = 1, 2, \cdots, b$ **do**
  $K_i = [\bar{\mathcal{J}}(:,:,i)'\bar{\mathcal{J}}(:,:,i)\Omega_V(:,:,i), \ldots, (\bar{\mathcal{J}}(:,:,i)'\bar{\mathcal{J}}(:,:,i))^q\Omega_V(:,:,i)] \in R^{th\times k}$.
  $(Q_i, \sim) = qr(K_i, 0) \in R^{th\times k}$.
  $M_i = \bar{\mathcal{J}}(:,:,i)Q_i \in R^{l\times k}$.
  $[\bar{\mathcal{U}}(:,:,i), \bar{\mathcal{S}}(:,:,i), \bar{\mathcal{V}}(:,:,i)] = svd(M_i, 0), \bar{U} \in R^{l\times k}, \bar{S} \in R^{k\times k}, \bar{V} \in R^{k\times k}$.
  $\bar{\mathcal{V}}(:,:,i) = Q_i\bar{\mathcal{V}}(:,:,i)$.
end for
$\mathcal{S} = ifft(\bar{\mathcal{S}}, [], 3), \mathcal{V} = ifft(\bar{\mathcal{V}}, [], 3), \mathcal{U} = ifft(\bar{\mathcal{U}}, [], 3)$.
$\mathcal{S}' = \mathcal{S}(1 : Rank_v, 1 : Rank_v, :), \mathcal{V}' = \mathcal{V}(:, 1 : Rank_v, :), \mathcal{U}' = \mathcal{U}(:, 1 : Rank_v, :)$.

3. Proposed algorithm

3.1. T-SVD decomposition based on variable T-product randomized algorithm

The trajectory tensor $\mathcal{J}$ derived by adaptive embedding maintains and enhances the low-rank nature of the original tensor. However, due to the expansion of the third mode and the l-fold increase in the size of the trajectory tensor in comparison to the original HSI, it becomes necessary to decompose the trajectory tensor using randomized t-SVD based on a variable T-product. This is performed to extract the intrinsic characteristics of the HSI. The key steps in this process are as follows. Let $\bar{\mathcal{J}} = fft_{p2v}(\mathcal{J}, [], 3)$ be the transformation tensor of $\mathcal{J}$, where $fft_{p2v}(\mathcal{J}, [], 3)$ represents the ZDFT along the third dimension of the tensor. The generated random Gaussian tensor is also subjected to ZDFT along the third dimension. Second, in the variable discrete Fourier domain, similar to the matrix situation, a Krylov subspace K is constructed corresponding to the forward slice of each transformed tensor, and the K is subjected to economical QR decomposition to obtain Q, then the row information of each forward slice is extracted after the trajectory tensor transformation, which is recorded as M_i, that is,

$$M_i = \bar{\mathcal{J}}(:,:,i)Q_i \in R^{l\times k}.$$

SVD is then performed,

$$[\bar{\mathcal{U}}(:,:,i),\bar{\mathcal{S}}(:,:,i),\bar{\mathcal{V}}(:,:,i)] = svd(M_i,0),\bar{U}\in R^{l\times k},\bar{S}\in R^{k\times k},\bar{V}\in R^{k\times k}.$$

The corresponding $\mathcal{U}(:,:,i),\mathcal{S}(:,:,i),\mathcal{V}(:,:,i)$ is taken as each frontal slice of $\bar{\mathcal{U}},\bar{\mathcal{S}},\bar{\mathcal{V}}$, and finally, the IZDFT is proceed,

$$\mathcal{S} = ifft_{v2p}(\bar{\mathcal{S}},[],3),\mathcal{V} = ifft_{v2p}(\bar{\mathcal{V}},[],3),\mathcal{U} = ifft_{v2p}(\bar{\mathcal{U}},[],3).$$

3.2. Low rank representation

In order to characterize the low rank of the tensor more effectively, the characteristic tensors $\mathcal{U},\mathcal{S},\mathcal{V}$ obtained by t-SVD based on variable T-product randomized algorithm are truncated to obtain the best low rank approximation of the original trajectory tensor, and thus the ideal tensor variable tubal rank is defined:

$$\begin{aligned} Rank_v(J) &\le min\{th,l\};\\ \mathcal{S}' &= \mathcal{S}(1:Rank_v,1:Rank_v,:);\\ \mathcal{V}' &= \mathcal{V}(:,1:Rank_v,:);\\ \mathcal{U}' &= \mathcal{U}(:,1:Rank_v,:); \end{aligned}$$

Thus a new trajectory tensor is obtained:

$$J_{Rank_v} = \mathcal{U}' *_v \mathcal{S}' *_v \mathcal{V}'^{\top} \in R^{l\times th\times b},$$

where $*_v$ is a variable T-product, and tensor J_{Rank_v} is a low rank tensor, which can be regarded as the rank $Rank_v$ approximation of the original trajectory tensor J, and $Rank_v$ determines the amount of information used for tensor reconstruction. The proposed algorithm is summarized in Algorithm 5, and the computational complexity is summarized in Table 2.

Algorithm 5 Variable t-SVD with Randomized block krylov iteration(Vrt-SVD)

Input: tensor $\mathcal{J}\in\mathbb{R}^{l\times th\times b}$, iteration q, oversampling parameter p, rank r, $k=r+p$, zero padding parameter v.
Output: $\mathcal{U}',\mathcal{S}',\mathcal{V}'$

$\bar{\mathcal{J}} = fft_{p2v}(\mathcal{J},[],3)$.
Create random Gaussian tensor $\Omega = zeros(th,k,n3),\Omega(:,:,1) = randn(th,k),\Omega_V = fft_{p2v}(\Omega,[],3)$.
for $i=1,2,\cdots,v$ **do**
  $K_i = [\bar{\mathcal{J}}(:,:,i)'\bar{\mathcal{J}}(:,:,i)\Omega_V(:,:,i),\ldots,(\bar{\mathcal{J}}(:,:,i)'\bar{\mathcal{J}}(:,:,i))^q\Omega_V(:,:,i)]\in R^{th\times k}$.
  $(Q_i,\sim) = qr(K_i,0)\in R^{th\times k}$.
  $M_i = \bar{\mathcal{J}}(:,:,i)Q_i\in R^{l\times k}$.
  $[\bar{\mathcal{U}}(:,:,i),\bar{\mathcal{S}}(:,:,i),\bar{\mathcal{V}}(:,:,i)] = svd(M_i,0),\bar{U}\in R^{l\times k},\bar{S}\in R^{k\times k},\bar{V}\in R^{k\times k}$.
  $\bar{\mathcal{V}}(:,:,i) = Q_i\bar{\mathcal{V}}(:,:,i)$.
end for
$\mathcal{S} = ifft_{v2p}(\bar{\mathcal{S}},[],3),\mathcal{V} = ifft_{v2p}(\bar{\mathcal{V}},[],3),\mathcal{U} = ifft_{v2p}(\bar{\mathcal{U}},[],3)$.
$\mathcal{S}' = \mathcal{S}'(1:Rank_v,1:Rank_v,:),\mathcal{V}' = \mathcal{V}(:,1:Rank_v,:),\mathcal{U}' = \mathcal{U}(:,1:Rank_v,:)$.

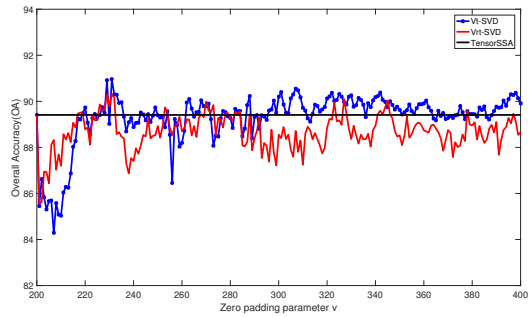
Table 2. Computational complexity of methods t-SVD, Vt-SVD, rBKI-SVD and Vrt-SVD.

Method	Computational Complexity
t-SVD	$\mathcal{O}(l^2\times th\times b)$
Vt-SVD	$\mathcal{O}(l^2\times th\times v)$
rBKI-SVD	$\mathcal{O}(q^2\times k^2\times th\times b)$
Vrt-SVD	$\mathcal{O}(q^2\times k^2\times th\times v)$

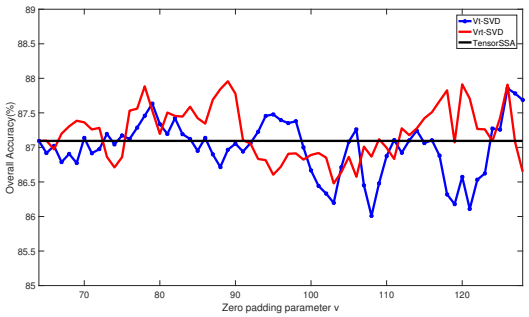
4. Numerical experiment

In this section, we evaluate the effectiveness of the proposed method on four public datasets[†]: the Indian Pines(IP), Pavia University(PU), and MUUFL Gulfport(MG), Salinas-A(SA). Their dimensions are $145 \times 145 \times 200$, $610 \times 340 \times 103$, $325 \times 337 \times 64$, and $86 \times 86 \times 224$, respectively. The parameters were set as follows. For the IP and SA datasets, we used a search area w of 11×11 and l taken as 49. For the PU dataset, we took 5×5 for w and 9 for l. For the MG dataset, we used two different sets of parameter values - one with $w = 7 \times 7$ and $l = 25$, and the other with $w = 5 \times 5$ and l = 9.

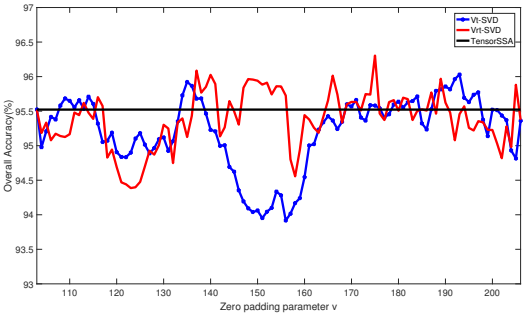
The implementation of the experiment was carried out using Matlab R2016a on a laptop running Windows 10. The laptop was equipped with an AMD Ryzen 7 5800H 3.2GHz CPU and 16GB of RAM.



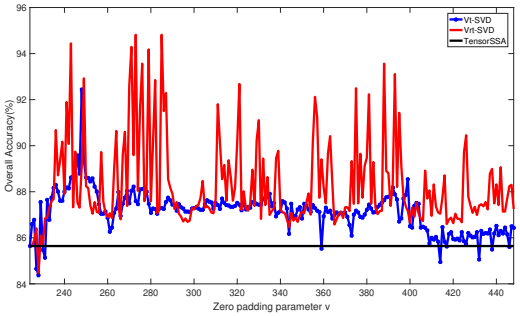
(a) $IP, 2\%$



(b) $MG, 1\%$



(c) $PU, 1\%$



(d) $SA, 0.01\%$

Figure 3. These figures show the overall accuracy obtained for TensorSSA and Vt-SVD and Vrt-SVD with progressively increasing zero-padding parameter v. Figure (a) shows the overall precision with a sampling rate of 2% in the IP data set, Figure (b) shows the overall precision with a sampling rate of 1% in the MG data set, Figure (c) shows the overall precision with a sampling rate of 1% in the PU data set, and Figure (d) shows the overall accuracy with a sampling rate of 0.01% in the SA data set.

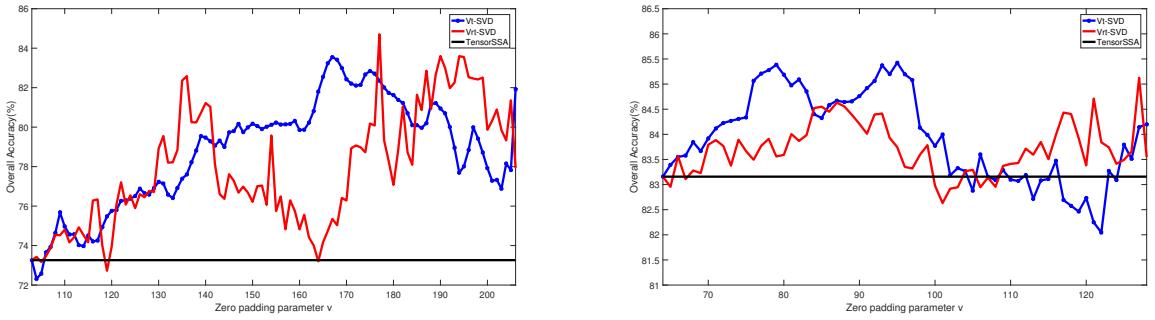
4.1. Find the best zero-padding parameter

In order to determine the optimal value of the zero-padding parameter v for the ZDFT transform, various values of v are tested at different sampling rates. The goal is to identify the value of v that results in the highest precision for each sampling rate. The sampling rates for the IP and MG datasets are $\{1\%, 2\%, 3\%, 4\%, 5\%\}$,

[†] $https://www.ehu.eus/ccwintco/index.php/Hyperspectral\_Remote\_Sensing\_Scenes.$

the sampling rates for the PU dataset are $\{0.5\%, 1\%, 2\%, 3\%, 4\%, 5\%\}$, the sampling rates for the SA dataset are $\{0.01\%, 0.1\%, 0.5\%, 1\%, 2\%\}$, and the rest of the samples were used for testing. Explanation of why the SA data set takes such a low sampling rate: because the SA data set is very small, each piece is 86×86, with few features and few categories, the accuracy increases very slowly when the sampling rate higher than 1%, which is almost completely classified. Figure 3 demonstrates the variability in precision in various datasets when different values of v are used. This experiment aims to evaluate the overall classification accuracy discrepancies between Vt-SVD and Vrt-SVD. The training set employs a constant sampling rate, while the zero-padding parameters are adjusted. The ultimate goal is to determine the most suitable parameter v by comparing it with the baseline method, TensorSSA.

The results of the experiments show that the classification accuracy of the four datasets improves after applying Zero-padding Discrete Fourier Transform (ZDFT), surpassing the accuracy achieved by TensorSSA. This indicates that zero-padding is effective. Specifically, within the SA dataset, most of the zero-padding parameters improve the overall classification accuracy. However, it should be noted that the impact of these parameters varies across datasets. In some cases, certain parameters may even decrease the overall classification accuracy. Furthermore, we observed that at extremely low sampling rates (for example, PU with a sampling rate of 0.1% and MG with a sampling rate of 0.5%), the zero-padding parameter tends to have a more pronounced impact on improving the overall classification accuracy when using the relatively larger adaptive embedding parameter l, as shown in Figure 4. This trend holds for various sampling rates. In summary, properly selecting zero-padding parameters can effectively enhance accuracy. Simultaneously, when comparing the performance of Vt-SVD and Vrt-SVD across the four datasets, it is noteworthy that Vrt-SVD consistently attains the highest overall classification accuracy in three of the datasets: MG, PU and SA, while IP is superior in the case of Vt-SVD, yielding better results.



(a) $MG - 0.5\%$ (b) $PU - 0.1\%$

Figure 4. These figures represent the overall accuracy of PU and MG for different zero padding parameters in extremely low sample rate.

At different sampling rates, the optimal value of v is taken differently, and the optimal value of v at each sampling rate under each data set is shown in the following Table 3, 4 .

	IP					MG				
Sample rate(%)	1	2	3	4	5	1	2	3	4	5
Vt-SVD	233	231	310	306	270	126	80	74	94	94
Vrt-SVD	271	229	323	326	326	89	124	89	124	124

Table 3. The best zero padding parameter v of IP and MG at different sample rate

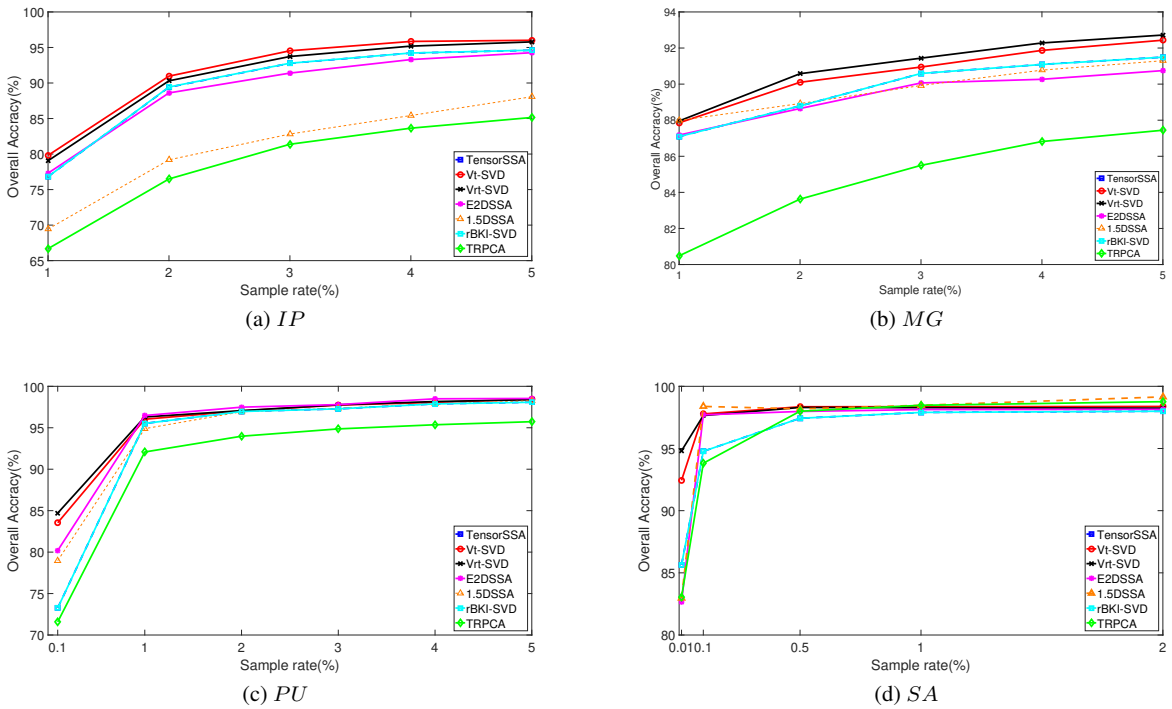
4.2. Classification results

From Table 3, 4, we can know the optimal value of the zero-padding parameter v corresponding to different data sets and different training percentages. Therefore, in the second experiment, the overall accuracy and CPU time

	PU						SA				
Sample rate(%)	0.1	1	2	3	4	5	0.01	0.1	0.5	1	2
Vt-SVD	167	193	106	134	206	206	248	237	255	251	252
Vrt-SVD	177	175	156	172	197	172	273	232	234	234	276

Table 4. The best zero-padding parameter v of PU and SA at different sample rate

of the proposed two algorithms (Vrt-SVD, Vt-SVD) in the classification task are obtained by using the optimal value of the corresponding variable v, and compared with spectral spatial feature extraction method 1.5DSSA, enhanced two-dimensional singular spectrum analysis method (E2DSSA), traditional singular spectral analysis method TRPCA, 3D feature extraction method TensorSSA, and TensorSSA with rBKI-SVD. To ensure a fair comparison of algorithm, we averaged the results from five experiments for each algorithm.



(a) IP (b) MG

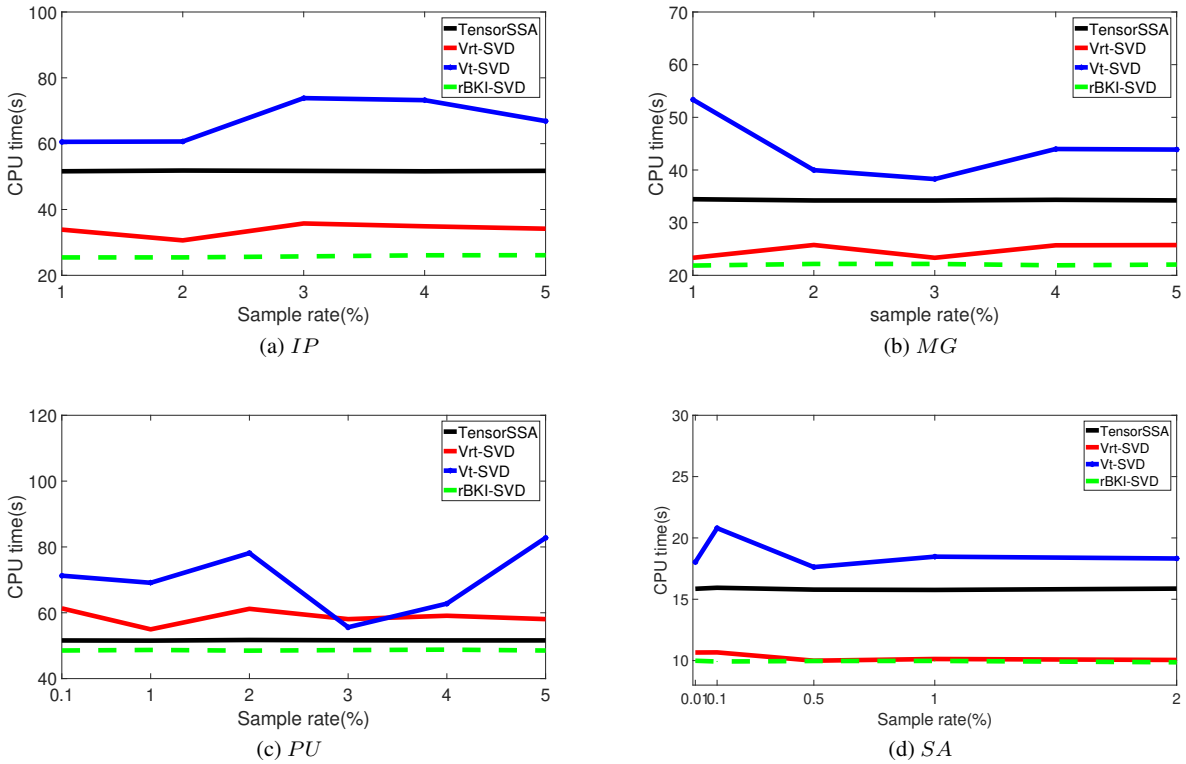
(c) PU (d) SA

Figure 5. These figures represent the overall accuracy of each dataset for different methods and different sample rates.

The results shown in Figure 5 clearly demonstrate that increasing the number of training samples leads to improved accuracy for all comparison methods. It is worth noting that the Vrt-SVD and Vt-SVD methods proposed in this study outperform the other three methods in situations where the sample rates are low, such as 1% of PU and IP, as well as 0.01% of SA. Additionally, the performance of the other methods varies across different datasets, whereas our method consistently achieves the excellent performance across all datasets when the training percentage is extremely low.

The accuracy rankings in the IP dataset for all sampling rates are as follows: Vt-SVD, Vrt-SVD, rBKI-SVD, TensorSSA, E2DSSA, 1.5DSSA, TRPCA. Vt-SVD consistently achieves the highest accuracy in this dataset. For the MG dataset, 1.5DSSA achieves the highest accuracy, followed by Vrt-SVD, Vt-SVD, with TRPCA yielding the lowest accuracy. In the PU dataset, the accuracy is most prominent at the 0.1% sampling rate, with Vrt-SVD delivering the highest accuracy, followed by Vt-SVD. However, at other sampling rates, E2DSSA performs better

than both Vrt-SVD and Vt-SVD. Except for TRPCA, the accuracy of other methods becomes quite similar after the 1% sampling rate. In the SA dataset, at the 0.01% sampling rate, Vrt-SVD achieves the highest accuracy, followed by Vt-SVD. However, when the sampling rates increase, the accuracy results are not as impressive as those of E2DSSA and TRPCA. In some cases, our method falls behind other methods, especially at high sampling rates.



(a) IP (b) MG

(c) PU (d) SA

Figure 6. These figures represent the CPU time(s) of each dataset for TensorSSA, rBKI-SVD, Vt-SVD and Vrt-SVD in different sample rates.

Figure 6 presents the CPU time for different methods at various training percentages for the four datasets. From the figure 6, we can see that the time of TensorSSA is constant at each sample rate because both TensorSSA and rBKI-SVD are in the DFT domain and the third dimension is constant . The rBKI-SVD has a somewhat different time because of the randomized method, but it is also almost constant. Vt-SVD and Vrt-SVD both have optimal Zero-padding parameter v at different sample rates. v will determine the size of the third dimension of the tensor decomposition, so the time will be different at each sample rate. It is worth noting that due to zero-padding, both Vt-SVD and Vrt-SVD have longer execution times compared to TensorSSA and TensorSSA-rBKI, respectively. From Figure 6, it is clear that Vrt-SVD demonstrates significantly faster execution times than Vt-SVD and TensorSSA in the IP, MG, and SA datasets. In these cases, TensorSSA is faster than Vt-SVD. This difference can be attributed to the extension of the third dimension after zero-padding, which increases the decomposition time and improves overall accuracy. However, in the PU dataset, Vrt-SVD requires more time than TensorSSA. This is due to the use of a smaller parameter l in constructing the trajectory tensor, resulting in a smaller first dimension for each frontal slice. As a result, there is not a significant time difference between deterministic economic SVD and randomized SVD. Although this diminishes the time advantage of Vrt-SVD, it still outperforms TensorSSA in terms of accuracy.

In order to conduct a thorough quantitative evaluation of the proposed method, we utilized five evaluation metrics: Classification Accuracy (CA), Overall Accuracy (OA), Average Accuracy (AA), Kappa Coefficient

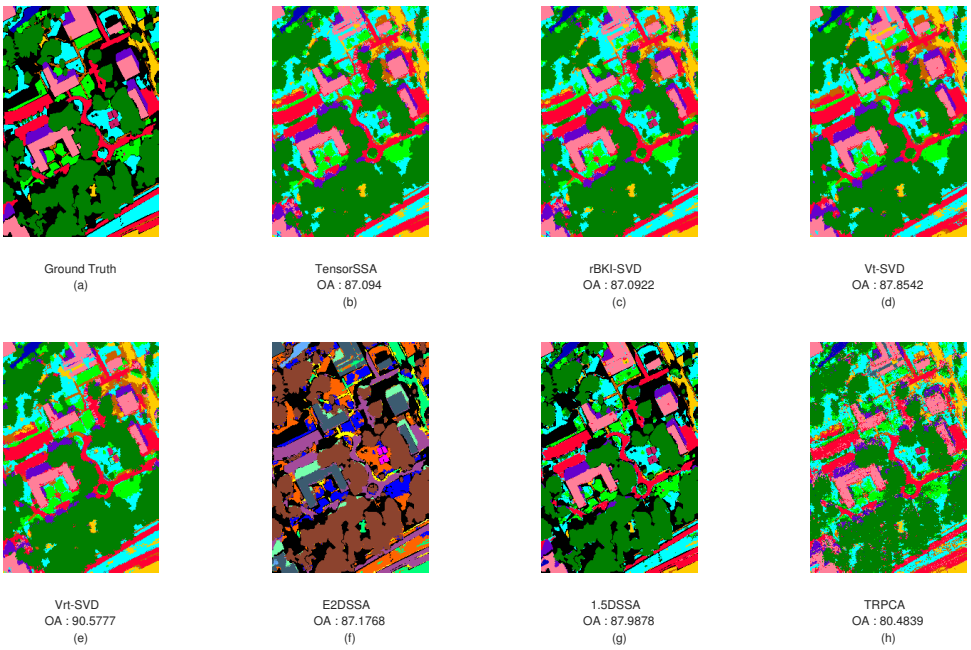


Figure 7. Pseudo-color images of data set MG, from left to right, are GT, classification obtained by methods TensorSSA, rBKI-SVD, Vt-SVD, Vrt-SVD, E2DSSA, 1.5DSSA, TRPCA.

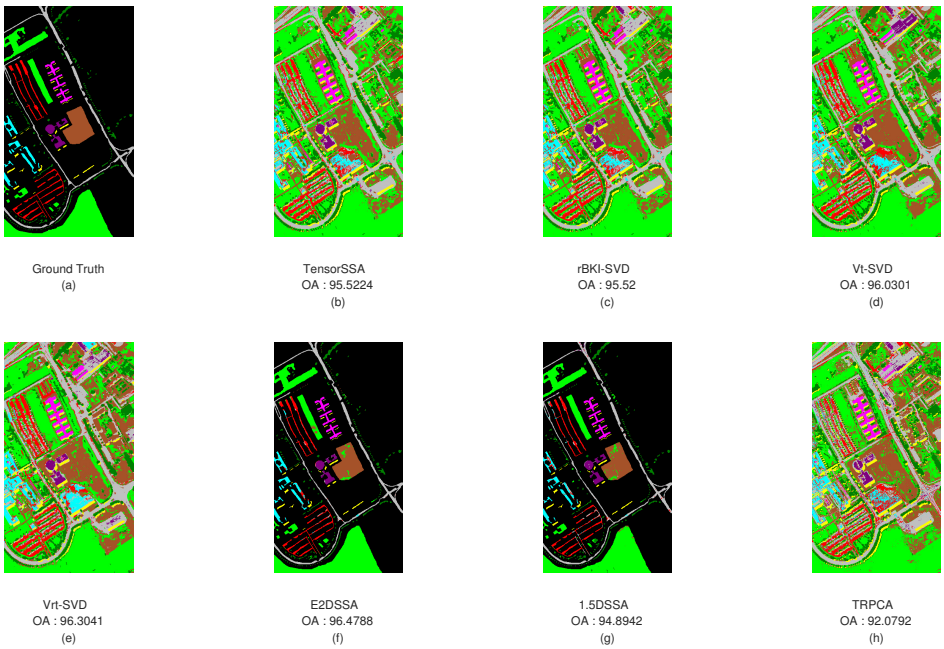


Figure 8. Pseudo-color images of data set PU, from left to right, are GT, classification obtained by methods TensorSSA, rBKI-SVD, Vt-SVD, Vrt-SVD, E2DSSA, 1.5DSSA, TRPCA.

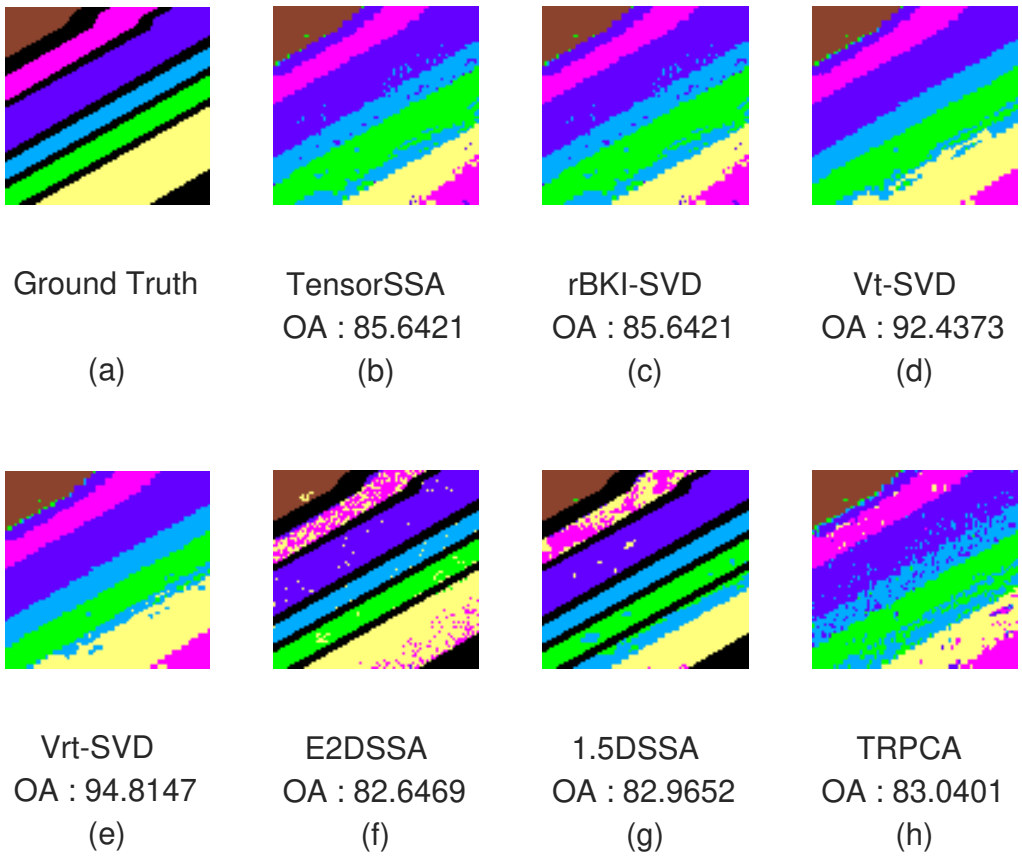


Figure 9. Pseudo-color images of data set SA, from left to right, are GT, classification obtained by methods TensorSSA, rBKI-SVD, Vt-SVD, Vrt-SVD, E2DSSA, 1.5DSSA, TRPCA.

(kappa), and the execution time(CPU time) for each category. To ensure the precision and dependability of our findings, we independently performed each experiment five times, and the average execution times are reported in the experiments. The classification outcomes for different datasets are summarized in Table 5, 6, 7, and 8.

The superior performance of Vrt-SVD and Vt-SVD in terms of accuracy is evident in Table 5, 6, 7, 8. In Table 5, Vt-SVD and Vrt-SVD exhibit the highest OA, AA, and Kappa values, with Vrt-SVD achieving the best OA in half the time compared to Vt-SVD. Our method, compared to TensorSSA, not only achieves higher accuracy but also avoids the blocky nature. Moving on to Table 6, 1.5DSSA outperforms other methods in terms of OA, AA, and Kappa. Vrt-SVD and Vt-SVD follow closely, with only a slight decrease in precision compared to 1.5DSSA. The difference in precision between Vrt-SVD and 1.5DSSA is small, with OA, AA, and Kappa values being lower by 0.03%, 0.81%, and 0.04%, respectively. However, Vrt-SVD takes only one-third the time of 1.5SSA. In Table 7, Vrt-SVD was the best in AA, while E2DSSA achieved the best in OA and Kappa. Vrt-SVD is 0.18% and 0.24% lower than E2DSSA in OA and KAPPA, respectively. Like the Table 6, Vrt-SVD takes only one-half the time of E2DSSA in PU. In Table 8, Vrt-SVD is best in OA, AA, and Kappa. The following qualitative evaluation of the proposed methods, classification maps generated by all methods, pseudo-color images and GT on three datasets are shown in Figure 7, 8, 9, respectively.

As can be seen in Figure 7, 8, 9 , TensorSSA retains the details of ground objects, such as tiny ground objects and roads. Through zero-padding and randomization, our method can get a classification map that is closer to the actual ground object distribution in a shorter time on most data sets, and TensorSSA, 1.5DSSA and E2DSSA cannot reasonably distinguish the boundaries of features, and ignore some smaller features. Our proposed method preserves the details of objects well and can obtain a better distribution map and extract features better. In a word, our method can obtain the classification map that is closest to the actual real distribution.

Table 5. Classification results of data set IP by different methods (2% training percentage)

#	Class	Samples	TensorSAA	Vrt-SVD	Vt-SVD	E2DSSA	1.5DSSA	rBKI-SVD	TRPCA
1	Alfalfa	46	97.78	97.78	97.78	88.89	73.33	97.78	84.44
2	Corn-notill	1428	85.63	86.70	87.85	90.78	64.83	85.63	73.27
3	Corn-mintill	830	91.27	90.41	89.42	80.20	67.28	91.39	65.19
4	Corn	237	70.69	94.83	93.97	60.34	30.17	70.69	28.02
5	Grass-pasture	483	87.53	94.08	88.79	91.12	79.70	87.53	81.82
6	Grass-trees	730	95.94	90.49	96.78	90.49	97.76	95.94	85.03
7	Grass-pasture-mowed	28	96.30	96.30	96.30	96.30	70.37	96.30	81.48
8	Hay-windrowed	478	92.74	95.73	92.09	90.81	91.24	92.74	98.08
9	Oats	20	100.00	94.74	94.74	100.00	78.95	100.00	52.63
10	Soybean-notill	972	82.25	78.15	77.52	85.40	75.11	82.25	71.95
11	Soybean-mintill	2455	91.19	90.56	92.81	89.15	88.69	91.19	84.24
12	Soybean-clean	593	81.93	81.41	85.89	84.85	43.89	81.93	41.65
13	Wheat	205	96.50	98.00	97.00	97.00	99.00	96.50	96.00
14	Woods	1265	97.66	98.22	97.82	95.72	95.24	97.66	93.95
15	Buildings-Grass-Trees-Drives	386	79.89	97.88	95.24	85.19	48.94	80.42	52.38
16	Stone-Steel-Towers	93	98.90	98.90	98.90	100.00	57.14	98.90	29.67
	OA (%)		89.41	90.31	**90.96**	88.60	77.83	89.44	76.50
	AA(%)		90.39	**92.76**	92.68	89.14	72.60	90.43	69.99
	Kappa*100(%)		87.94	88.96	**89.71**	87.00	74.53	87.98	73.03
	CPU times(s)		48.92	32.61	64.99	**12.92**	31.07	25.44	41.03

Table 6. Classification results of data set MG by different methods (1% training percentage)

#	Class	Samples	TensorSAA	Vrt-SVD	Vt-SVD	E2DSSA	1.5DSSA	rBKI-SVD	TRPCA
1	Trees	23246	95.51	96.07	96.46	95.48	95.68	95.50	92.72
2	Grass	4270	76.58	85.71	78.19	77.31	73.60	76.58	59.40
3	MG-surface	6882	79.38	76.96	76.21	81.17	83.44	79.38	73.64
4	Dirt/sand	1826	90.98	84.23	87.55	89.04	83.62	90.98	77.48
5	Road	6687	91.19	90.76	92.67	91.25	89.82	91.19	84.52
6	Water	466	84.60	91.97	84.60	78.52	80.26	84.60	69.85
7	B-shadow	2233	69.46	68.51	71.13	65.97	76.38	69.46	51.40
8	Buildings	6240	85.80	85.28	85.46	84.41	88.38	85.80	75.88
9	Sidewalk	1385	37.78	62.87	64.26	47.63	51.71	37.78	44.86
10	Y-curb	183	12.71	29.28	17.68	1.10	12.15	12.71	17.13
11	Cloth-P	269	81.20	45.49	46.62	83.46	90.98	81.20	46.24
	OA		87.09	87.96	87.85	87.18	**87.99**	87.09	80.48
	AA		73.20	74.28	72.80	72.30	**75.09**	73.20	63.01
	Kappa*100		82.91	84.03	83.86	82.99	**84.07**	82.90	74.04
	times(s)		34.21	23.23	52.45	39.18	74.03	**21.85**	50.45

Table 7. Classification results of data set PU by different methods (1% training percentage)

#	Class	Samples	TensorSAA	Vrt-SVD	Vt-SVD	E2DSSA	1.5DSSA	rBKI-SVD	TRPCA
1	Asphalt	6631	96.59	95.55	94.61	99.13	96.36	96.57	88.71
2	Meadows	18649	99.29	99.47	99.37	99.26	98.68	99.29	97.73
3	Gravel	2099	90.47	91.05	91.77	89.75	82.87	90.47	84.65
4	Trees	3064	89.48	90.77	90.01	91.92	88.43	89.48	93.24
5	Painted metal sheets	1345	99.10	99.02	99.10	98.72	99.62	99.10	98.87
6	Bare Soil	5029	90.82	92.53	96.00	92.77	92.15	90.82	88.87
7	Bitumen	1330	94.45	93.69	92.86	93.62	96.88	94.45	78.19
8	Self-Blocking Bricks	3682	87.16	94.10	88.89	91.77	85.35	87.16	77.75
9	Shadows	947	98.51	97.12	97.23	91.68	99.89	98.51	99.68
	OA		95.52	96.30	96.03	**96.48**	94.89	95.52	92.08
	AA		93.99	**94.81**	94.43	94.29	93.36	93.98	89.74
	Kappa*100		94.02	95.07	94.72	**95.31**	93.19	94.02	89.47
	times(s)		56.50	65.89	82.98	114.75	266.00	**48.54**	371.52

Table 8. Classification results of data set SA by different methods (0.01% training percentage)

#	Class	Samples	TensorSAA	Vrt-SVD	Vt-SVD	E2DSSA	1.5DSSA	rBKI-SVD	TRPCA
1	Brocoli_green_weeds_1	391	99.49	99.74	99.74	99.23	99.49	99.49	99.49
2	Corn_senesced_green_weeds	1343	53.80	85.92	74.37	50.22	1.34	53.80	53.06
3	Lettuce_romaine_4wk	616	87.97	88.46	91.54	61.79	86.34	87.97	88.94
4	Lettuce_romaine_5wk	1525	98.29	100.00	100.00	100.00	100.00	98.29	95.41
5	Lettuce_romaine_6wk	674	95.10	99.41	99.55	99.55	99.85	95.10	83.95
6	Lettuce_romaine_7wk	799	98.50	98.50	99.50	97.74	97.74	98.50	96.49
	OA		85.64	**94.81**	92.44	82.65	73.25	85.64	83.04
	AA		88.86	**95.34**	94.12	84.76	80.79	88.86	86.22
	Kappa*100		82.27	**93.53**	90.60	78.52	67.66	82.27	79.04
	times(s)		16.62	10.85	18.52	**5.10**	13.29	8.44	21.61

5. Conclusion

In this paper, we introduce two novel methods, Vt-SVD and Vrt-SVD, which aim to improve the extraction of hyperspectral image (HSI) features by considering both spectral and spatial characteristics. The suggested method employs a randomized block Krylov Iteration algorithm that relies on the variable t-SVD to effectively analyze the low-rank representation of the spatial and spectral information in the HSI at both global and local levels. Compared to the truncated SVD, the randomized block Krylov Iteration algorithm requires less time and produces similar classification outcomes. By employing zero-padding Discrete Fourier Transform, the classification accuracy can be significantly enhanced. Our proposed Vrt-SVD and Vt-SVD approaches yield superior classification results compared to the DFT domain-based TensorSSA. The experimental results on four publicly available datasets demonstrate the effectiveness of the proposed algorithm. Even with a limited number of training samples, this approach successfully extracts HSI features while preserving irregular boundaries and structural shapes in the classified images.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 12071104), and the Natural Science Foundation of Zhejiang Province (No. LY22A010012).

REFERENCES

1. Bhardwaj K, Patra S. An unsupervised technique for optimal feature selection in attribute profiles for spectral-spatial classification of hyperspectral images[J]. ISPRS Journal of Photogrammetry and Remote Sensing, 2018, 138: 139-150.
2. Donciu C, Temneanu M. An alternative method to zero-padded DFT[J]. Measurement, 2015, 70: 14-20.
3. Dong W, Yu G, Qi L, et al. Practical Sketching Algorithms for Low-Rank Tucker Approximation of Large Tensors[J]. Journal of Scientific Computing, 2023, 95(2): 52.
4. Fu H, Sun G, Zabalza J, et al. A novel spectral-spatial singular spectrum analysis technique for near real-time in situ feature extraction in hyperspectral imaging[J]. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 2020, 13: 2214-2225.
5. Fu H, Sun G, Zhang A, et al. Tensor Singular Spectral Analysis for 3D feature extraction in hyperspectral images[J]. IEEE transactions on geoscience and remote sensing, pp. 1-14, 2023, Art no. 5403914, doi: 10.1109/TGRS.2023.3272669.
6. Fu H, Sun G, Ren J, et al. Fusion of PCA and segmented-PCA domain multiscale 2-D-SSA for effective spectral-spatial feature extraction and data classification in hyperspectral imagery[J]. IEEE Transactions on Geoscience and Remote Sensing, 2020, 60: 1-14.
7. Fu H, Zhang A, Sun G, et al. A novel band selection and spatial noise reduction method for hyperspectral image classification[J]. IEEE transactions on geoscience and remote sensing, 2022, 60: 1-13.
8. Ghasemzadeh A, Demirel H. 3D discrete wavelet transform-based feature extraction for hyperspectral face recognition[J]. Iet Biometrics, 2018, 7(1): 49-55.
9. Halko N, Martinsson P G, Tropp J A. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions[J]. SIAM review, 2011, 53(2): 217-288.
10. Huang H, Peng S, Yu G, et al. Hyperspectral image restoration based on color superpixel segmentation[J]. Statistics, Optimization & Information Computing, 2024, 12(1): 267-280.
11. Jia S, Shen L, Zhu J, et al. A 3-D Gabor phase-based coding and matching framework for hyperspectral imagery classification[J]. IEEE transactions on cybernetics, 2017, 48(4): 1176-1188.
12. Jin X, Gu Y. Superpixel-based intrinsic image decomposition of hyperspectral images[J]. IEEE Transactions on Geoscience and Remote Sensing, 2017, 55(8): 4285-4295.
13. Kilmer M E, Martin C D. Factorization strategies for third-order tensors[J]. Linear Algebra and its Applications, 2011, 435(3): 641-658.
14. Kolda T G, Bader B W. Tensor decompositions and applications[J]. SIAM review, 2009, 51(3): 455-500.
15. Li X, Li Z, Qiu H, et al. An overview of hyperspectral image feature extraction, classification methods and the methods based on small samples[J]. Applied Spectroscopy Reviews, 2023, 58(6): 367-400.
16. Lu C, Feng J, Chen Y, et al. Tensor robust principal component analysis with a new tensor nuclear norm[J]. IEEE transactions on pattern analysis and machine intelligence, 2019, 42(4): 925-938.
17. Li F, Wang J, Lan R, et al. Hyperspectral image classification using multi-feature fusion[J]. Optics & Laser Technology, 2019, 110: 176-183.
18. Muquet B, Wang Z, Giannakis G B, et al. Cyclic prefixing or zero padding for wireless multicarrier transmissions?[J]. IEEE Transactions on communications, 2002, 50(12): 2136-2148.
19. Ma P, Ren J, Zhao H, et al. Multiscale 2-D singular spectrum analysis and principal component analysis for spatial–spectral noise-robust feature extraction and classification of hyperspectral images[J]. Ieee Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 2020, 14: 1233-1245.
20. Musco C, Musco C. Randomized block krylov methods for stronger and faster approximate singular value decomposition[J]. Advances in neural information processing systems, 2015, 28.
21. Qi L, Yan R, Luo Z, et al. Variable T-Product and Zero-Padding Tensor Completion with Applications[J]. arXiv preprint arXiv:2305.07837, 2023.
22. Qiu Y, Sun W, Zhou G, et al. Towards Efficient and Accurate Approximation: Tensor Decomposition Based on Randomized Block Krylov Iteration[J]. arXiv preprint arXiv:2211.14828, 2022.
23. Qi L, Yu G. T-singular values and t-sketching for third order tensors[J]. arXiv preprint arXiv:2103.00976, 2021.
24. Rasti B, Hong D, Hang R, et al. Feature extraction for hyperspectral imagery: The evolution from shallow to deep: Overview and toolbox[J]. IEEE Geoscience and Remote Sensing Magazine, 2020, 8(4): 60-88.
25. Sun G, Fu H, Ren J, et al. SpaSSA: Superpixelwise adaptive SSA for unsupervised spatial–spectral feature extraction in hyperspectral image[J]. IEEE Transactions on Cybernetics, 2021, 52(7): 6158-6169.
26. Tropp J A, Yurtsever A, Udell M, et al. Practical sketching algorithms for low-rank matrix approximation[J]. SIAM Journal on Matrix Analysis and Applications, 2017, 38(4): 1454-1485.
27. Wolf A S J W. Low rank tensor decompositions for high dimensional data approximation, recovery and prediction[D]. Technische Universität Berlin, 2019.
28. Zabalza J, Ren J, Zheng J, et al. Novel two-dimensional singular spectrum analysis for effective feature extraction and data classification in hyperspectral imaging[J]. IEEE transactions on geoscience and remote sensing, 2015, 53(8): 4418-4433.
29. Zhang S, Li S, Fu W, et al. Multiscale superpixel-based sparse representation for hyperspectral image classification[J]. Remote Sensing, 2017, 9(2): 139.
30. Zhan T, Lu Z, Wan M, et al. Multiscale superpixel kernel-based low-rank representation for hyperspectral image classification[J]. IEEE Geoscience and Remote Sensing Letters, 2019, 17(9): 1642-1646.