Tensorized Feature Spaces for Feature Explosion

Ravdeep S. Pasricha*, Pravallika Devineni†, Evangelos E. Papalexakis*, Ramakrishnan Kannan†
*Department of Computer Science and Engineering, University of California Riverside, Riverside, CA, USA.
rpasr001@ucr.edu, epapalex@cs.ucr.edu
†Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN, USA.
{devinenip, kannanr}@ornl.gov

***Abstract*—In this paper[1], we present a novel framework that uses tensor factorization to generate richer feature spaces for pixel classification in hyperspectral images. In particular, we assess the performance of different tensor rank decomposition methods as compared to the traditional kernel-based approaches for the hyperspectral image classification problem. We propose ORION , which takes as input a hyperspectral image tensor and a rank and outputs an enhanced feature space from the factor matrices of the decomposed tensor. Our method is a feature explosion technique that inherently maps low dimensional input space in $\mathbb{R}^K$ to high dimensional space in $\mathbb{R}^R$, where $R \gg K$, say in the order of 1000x, like a kernel. We show how the proposed method exploits the multi-linear structure of hyperspectral three dimensional tensor. We demonstrate the effectiveness of our method with experiments on three publicly available hyperspectral datasets with labeled pixels and compare their classification performance against traditional linear and non-linear supervised learning methods such as SVM with Linear, Polynomial, RBF kernels, and the Multi-Layer Perceptron model. Finally, we explore the relationship between the rank of the tensor decomposition and the classification accuracy using several hyperspectral datasets with ground truth.**

***Index Terms*—Tensor, Tensor Decomposition, Hyperspectral Imaging.**

I. INTRODUCTION

Hyperspectral imaging techniques capture images of objects or materials with hundreds of spectral bands at each pixel [1]. A particularly important use of these techniques is in capturing images of land area on the earth's surface from above using an aircraft or a satellite fitted with sensors. Since objects under observation reflect different wavelengths of the spectral band, each pixel has a large number of features corresponding to the spectral bands. These features have most popularly been used to accomplish two tasks – identify the class of each given pixel, a classification task [2], [3], [4] or, see what that pixel is made of, an unmixing task [5], [6]. Bioucas *et al.* [7] present a survey of problems often encountered in analyzing hyperspectral remote sensing data. In this paper, we focus on the pixel classification task, where labelled data for some of the pixels is available.

Hyperspectral images (HSI) can be considered as a set of images stacked together like a 3D cube. For hyperspectral images with high spatial resolution, the pixel classification task assumes that each pixel is 'pure', i.e. it corresponds to a single class. In contrast with the classification task, the unmixing task assumes that each pixel may be composed of multiple materials or 'endmembers' [7]. The main challenges involved in hyperspectral pixel classification are the large number of spectral bands, leading to high dimensionality and the limited availability of labelled data. Previous work considered the feature space generated using kernel functions for HSI classification [2], [8], [3]. These works treat data as a matrix where each row is a pixel in multi-spectral bands. However, there are three challenges in these kernel spaces (a) choice of kernel and its parameters. For example, tuning the parameter γ in Radial Basis Function (RBF) kernel is non-trivial and impacts the performance of the classifier, (b) the number of features generated by the kernel methods is dependent on the number of pixels, i.e. the kernel function $K(\mathbf{X}, \mathbf{X}) \rightarrow \mathbf{F}$ takes the k spectral bands for xy pixels as $\mathbf{X} \in \mathbb{R}^{xy \times k}$ will yield a feature matrix $\mathbf{F} \in \mathbb{R}^{xy \times xy}$, and (c) these kernel spaces assume that the pixels are independent and identically distributed (IID) samples and ignore the spatial correlation that exists between the pixels.

In this paper we address these challenges by exploring a new feature explosion method called ORION that uses tensor completion to generate a richer feature space by exploiting the multi-dimensional nature of data. ORION allows relaxing the dimension of the obtained feature space $\mathbf{F} \in \mathbb{R}^{xy \times R}$ instead of fixed dimension xy from kernel methods. While we demonstrate the usefulness of ORION for HSI classification, we would like to emphasize that it can be applied to a broader range of problems.

Our contributions in this work are as follows:

- **Tensorized Feature Space:** We introduce a new feature space based on factors generated using tensor factorization[2]. This works better or on par with traditional state-of-the-art classification methods. To the best of our knowledge, this is the first work that presents a formal study of feature space explosion with defined number of

[1]This manuscript has been authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan) This research used resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC05-00OR22725.

[2]We use decomposition and factorization interchangeably throughout the paper

features R, unlike kernel methods that fix the length of the dimension for the number of available samples.

- **Experimental Evaluation:** We demonstrate the effectiveness of our proposed method ORION by evaluating it on publicly available hyperspectral datasets and compare it against traditional state-of-the-art baselines, linear and nonlinear supervised learning methods like Linear, Polynomial and RBF Support Vector Machines, and Multi-Layer Perceptrons.

Reproducibility: To encourage transparency and reproducibility of the experiments, we make our implementation of ORION and baselines publicly available[3]. All datasets used in the experiment are publicly available at [9].

The remainder of this paper is structured as follows, In section II, we formulate the problem more formally in terms of tensor and tensor factorization. Section III describes our proposed method in detail. We demonstrate the effectiveness of our method in section IV. In section V, we present the related work in tensors and hyperspectral field and, finally we conclude our paper in section VI.

II. PROBLEM FORMULATION

In this section, we present some preliminary definitions required for setting up the problem and define our problem. Table I describes the symbols used and their descriptions.

A. Preliminary Definitions

Tensor: Tensors are multi-dimensional arrays and are used to model multi-aspect data. Each dimension of a tensor is called a *mode*. For example, a 1-mode tensor is a vector, and a 2-mode tensor is a matrix. A 3-mode tensor is represented by $\underline{\mathbf{X}}$, where $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$.

For example, the hyperspectral image of a land area consisting of several spectral bands can be modeled as a 3-mode tensor $\underline{\mathbf{X}}$, where the first two modes I and J represent the height and width of the image and the third mode K represents the spectral bands.

Fibers: The row and column vectors are fibers in a matrix. Given an n-mode tensor, we obtain fibers by fixing the $n-1$ indices. For example, in a 3-mode tensor $\underline{\mathbf{X}}$ with indices I, J, K, $\underline{\mathbf{X}}(:, J, K), \underline{\mathbf{X}}(I, :, K)$ and $\underline{\mathbf{X}}(I, J, :)$ are considered mode-1, mode-2 and mode-3 fibers respectively.

Kronecker Product The Kronecker product of two matrices $\mathbf{A} \in \mathbb{R}^{I \times J}$ and $\mathbf{B} \in \mathbb{R}^{M \times N}$ is given by $\mathbf{A} \otimes \mathbf{B} \in \mathbb{R}^{IM \times JN}$

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} \dots a_{1J}\mathbf{B} \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} \dots a_{2J}\mathbf{B} \\ \vdots & \vdots \quad \ddots \\ a_{I1}\mathbf{B} & a_{I2}\mathbf{B} \dots a_{IJ}\mathbf{B} \end{bmatrix}$$

where a_{ij} refers to elements in matrix $\mathbf{A}$.

Khatri-Rao Product Khatri-Rao product of two matrices $\mathbf{A} \in \mathbb{R}^{I \times R}$ and $\mathbf{B} \in \mathbb{R}^{J \times R}$ is the column-wise Kronecker product given by $\mathbf{A} \odot \mathbf{B} \in \mathbb{R}^{IJ \times R}$.

$$\mathbf{A} \odot \mathbf{B} = [\mathbf{a_1} \otimes \mathbf{b_1} \quad \mathbf{a_2} \otimes \mathbf{b_2} \quad \dots \quad \mathbf{a_R} \otimes \mathbf{b_R}]$$

[3]https://github.com/ravdeep003/ORION

where $[\mathbf{a_1}, \mathbf{a_2} \dots, \mathbf{a_R}]$ and $[\mathbf{b_1}, \mathbf{b_2} \dots, \mathbf{b_R}]$ are columns of $\mathbf{A}$ and $\mathbf{B}$ respectively.

Canonical Polyadic Decomposition: Simply referred to as CP or CANDECOMP/PARAFAC [10], [11], it is one of the most commonly used tensor decompositions. The CP decomposition of a 3-mode tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$ for a particular rank R is given by sum of R rank-one tensors:

$$\underline{\mathbf{X}} \approx \sum_{r=1}^{R} \mathbf{A}(:, r) \circ \mathbf{B}(:, r) \circ \mathbf{C}(:, r)$$

where $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are factor matrices of size $I \times R$, $J \times R$ and $K \times R$ respectively and $\circ$ represents the three way outer product. In case of an n-mode tensor, it represents an n-way outer product.

Matricization: A tensor can be unfolded or flattened into one of its modes to form a matrix. For example, a 3-mode tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$ can be matricized in three ways: $\mathbf{X_1} \in \mathbb{R}^{I \times JK}$, $\mathbf{X_2} \in \mathbb{R}^{J \times IK}$ and $\mathbf{X_3} \in \mathbb{R}^{K \times IJ}$, where $\mathbf{X_n}$ represents matricization in n^{th}-mode.

Tensor Completion: Tensor completion is the task of predicting missing values in a tensor using tensor factorization. Tensor factorization strives to capture the underlying hidden structure even with the case of missing values [12], [13].

TABLE I
SYMBOLS USED IN THE PAPER

Symbols	Description
$\underline{\mathbf{X}}, \mathbf{X}, \mathbf{x}, x$	Tensor, matrix, column vector, scalar
$\mathbb{R}$	Set of Real Numbers
$\circ$	Outer product
$\underline{\mathbf{X}}(I, J, :)$	Spanning all elements in the 3rd-mode of $\underline{\mathbf{X}}$
$\otimes$	Kronecker product
$\odot$	Khatri-Rao product

We refer interested reader to [14], [15], [16], which present detailed surveys on tensors, tensor decompositions and their applications. This work employs MATLAB format of indexing.

B. Problem Definition

This work explores a new feature space using tensor completion and to show its effectiveness, we apply it to hyperspectral pixel classification. Previous literature related to HSI employed methods like feature reduction and kernel methods [8], [3]. In our work, we exploit the multi-linear structure of the hyperspectral image tensor using tensor decomposition to generate a richer space, where the pixels are linearly separable.

More formally we define our problem as follows:

> Given a three dimensional hyperspectral image tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$, a label matrix $\mathbf{Y} \in \mathbb{R}^{I \times J}$ and rank R, generate a feature space for a classifier such that pixels in the images are classified into one of the given classes.

III. PROPOSED METHOD: ORION

In this section, we introduce our method ORION and present the intuition behind it. ORION takes as input a three dimensional tensor $\underline{\mathbf{X}}$ and a tensor rank R and generates a feature

Algorithm 1 ORION

Input: A 3-mode tensor $\underline{\mathbf{X}}$, a label matrix $\mathbf{Y}$, rank r and testSize

Output: A vector of predicted classes

1: Extract pixel indices $[I, J]$ of all the non-zero classes.
2: Split $[I, J]$ into training and testing data in a stratified fashion.
3: Create tensor $\underline{\mathbf{P}}$ of ones with same dimensions as $\underline{\mathbf{X}}$
4: $\underline{\mathbf{P}}\ [testI, testJ, :] = 0$
{% Tensor completion problem}
5: $\mathbf{A}, \mathbf{B}, \mathbf{C}, \boldsymbol{\lambda} = CP\_WOPT(\underline{\mathbf{X}}, \underline{\mathbf{P}}, Rank)$ [12]
6: $\mathbf{data} = (A \odot B) * diag(\boldsymbol{\lambda})$
7: Using indices in Step 2, split $\mathbf{data}$ into training and testing data
8: Train a linear SVM using training data with 5-fold cross validation
9: Run the model against testing data
10: **return** model predictions

space using tensor factorization. The general idea behind the proposed method lies in mapping the input data to some high dimensional space corresponding to the rank decomposition of the tensor.

A. The ORION algorithm

Algorithm 1 presents the steps involved in ORION , applied to the hyperspectal pixel classification problem. Consider a 3-mode tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$, where I and J represent the resolution of the image and K represents the number of spectral bands. For a given test size, we select pixels with non-zero classes using stratified sampling as specified in line 2 of the algorithm 1. We do this in order to ensure that the training and testing data has the same percentage of representation from each class. We mark all spectral values of test pixels as zero, i.e. all the third mode fibers of the test data points are marked as zero, treating the problem of filling missing values as a tensor completion task. We employ the tensor completion algorithm CP-WOPT (Weighted Optimization) [12], implemented in [17] to predict the missing values. This produces three factor matrices **A**, **B** and **C**. The first two matrices **A** and **B** correspond to the two modes of the image are used to generate a new feature space.

$$\mathbf{data} = \mathbf{A} \odot \mathbf{B}$$

where $\odot$ represents Khatri-Rao product. To scale up the values, we multiply $\mathbf{data}$ with diagonalized $\boldsymbol{\lambda}$ matrix as shown in line 6 in algorithm 1. We use initial training and testing indices to generate training and testing data respectively, removing all the data points with class value as 0. Using the newly created feature space, we now train a linear Support Vector Machine (SVM) with 5-fold cross validation.

B. Intuition Behind ORION

The idea behind ORION is to map the input space to higher dimensional space by exploiting multi-linear structure of the tensor. Consider a 3-mode tensor, $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$. The CP decomposition with rank R of $\underline{\mathbf{X}}$ yields three factor matrices **A**, **B** and **C** of size $I \times R$, $J \times R$ & $K \times R$ respectively. The Khatri-Rao product of matrices **A** and **B** generates the new data space in $\mathbb{R}^{IJ \times R}$. Whereas unfolding of tensor $\underline{\mathbf{X}}$ in third mode would generate data space in $\mathbb{R}^{IJ \times K}$. Since $K \ll IJ$, the matrix rank is bounded by K. However, the upper bound on tensor rank for which CP can still uniquely identify the components within the tensor is $\min(IJ, JK, KI)$ [16], which is considerably larger. Thus, by using a large-enough rank, by virtue of CP's uniqueness [16], we are able to extract a feature space that is more expressive than simple unfolding of the features (or any spectral method in that unfolded matrix).

IV. EXPERIMENTAL EVALUATION

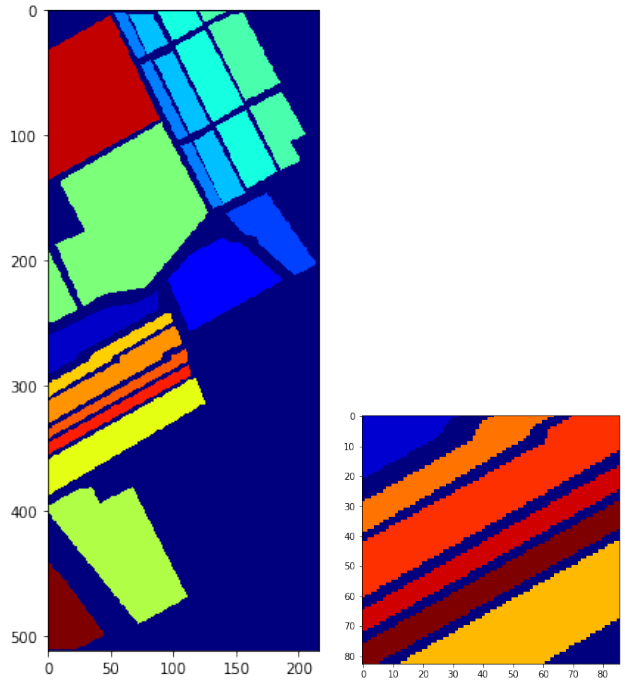
In this section, we describe our experimental setup and present our results. We use Tensor Toolbox [17] in Matlab for our tensor completion task, CP-WOPT [12] is implemented in this toolbox. For classification algorithms we use Python Scikit-Learn [18] and tensorly [19] for tensor operations. In the interest of reproducibility, the implementation of our algorithm and baselines used is publicly available[4].

A. Datasets

To evaluate our method, we use the following publicly available datasets [9].

- **Indian Pines:** This dataset was acquired using the AVIRIS[5] sensor [20] and consists of 145×145 pixels and 200 spectral bands. This dataset consist of 10249 labelled pixels spanning over 16. There is high class imbalance in this dataset.
- **University of Pavia:** This dataset was collected using ROSIS sensor over Pavia in Northern Italy. The original image resolution of the dataset is 610×610 but most of the image didn't contain any information so the image resolution is reduced to 610×340 over 103 spectral bands. This dataset consist of 42776 labelled pixels spanning over 9 classes
- **Salinas:** This dataset was gathered using AVIRIS sensor over Salinas Valley, California. The dataset consists is of 512×217 resolution over 204 spectral bands. It has 54219 non-zero pixels, labelled with 16 classes. Figure 1(a) shows the ground truth of Salinas dataset.
- **Salinas-A**: This dataset represents a subscene in the Salinas dataset. It consists of 86×83 pixels and 6 classes. Number of nonzero pixels in this dataset are 5348. Figure 1(b) presents the ground truth of Salinas-A dataset.

[4]https://github.com/ravdeep003/ORION
[5]https://aviris.jpl.nasa.gov/



(a) Salinas Full. (b) Salinas Subscene.

Fig. 1. Ground truth of Salinas and Salinas-A HSI datasets.

B. Baseline Methods

We evaluate our proposed method ORION against traditional linear and non-linear methods like the Kernel SVM and Multi-Layer Perceptron, and employ grid search to tune the hyperparameters in these methods.

1) Support Vector Machines (SVMs): Support vector machines (SVMs) [21] are supervised learning methods that are powerful classifiers, specially when combined with kernels. SVMs discriminate between between data points by finding a hyperplane which maximizes the margin. Given a dataset (x_i, y_i), where x_i are data points and y_i are labels,

$$\min_{\mathbf{w},b,\xi} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_i \xi_i \tag{1}$$

subject to $y_i(\mathbf{w}^T\phi(x_i) + b) \geq 1 - \xi_i$ and $\xi_i \geq 0$ for all $i = 1 \ldots n$

where $\mathbf{w}$ is a normal to the hyperplane separating the data points, C is a penalty term for misclassification, ξ is slack variables, ϕ is mapping from input space to kernel space and b is the bias or the offset for the hyperplane. Equation 1 is the primal form and it has dual form:

$$\min_{\alpha_1,\ldots\alpha_n} \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j K_{ij} - \sum_i^n \alpha_i \tag{2}$$

such that $0 \leq \alpha_i \leq C$ and $\sum_i^n \alpha_i y_i = 0$

where α_i are Lagrange multipliers and K_{ij} is inner product of data points in kernel space. We use the following kernels in our experiments:

- Linear Kernel: We tune the parameter **C** in the objective function. We do a grid search from 10^{-3} to 10^3 in multiples of 10.

$$K(x_i, x_j) = {x_i}^T x_j$$

- Polynomial Kernel: We tune the parameters **C** and **degree** of the polynomial. **C** is tuned in the range 10^{-3} to 10^3 and **degree** is tuned in the range from 2 to 5. We perform a grid search to tune these parameters.

$$K(x_i, x_j) = ({x_i}^T x_j + 1)^d$$

where d is the degree of the polynomial.

- RBF Kernel: We tune the parameters **C** and γ. C is tuned from the same range as before and γ is tuned from 10^{-3} to 10 in multiples of 10.

$$K(x_i, x_j) = \exp\left(-\gamma||x_i - x_j||^2\right)$$

2) Multi-Layer Perceptron (MLP): We use a Multi Layer Perceptron model (also known as artificial neural network) as one of our baselines. Various parameters involved in training the model were tuned using grid search. These parameters include hidden layer sizes $[(50, 100, 50), (100, 100, 100), (150, 100, 150)]$, alpha - $L2$ regularization term $[10^{-4}, 10^{-3}, 10^{-2}]$, initial learning rate $[10^{-4}, 10^{-3}, 10^{-2}]$ and learning rate (constant or adaptive). We used ReLu (rectified linear unit) activation function, Adam optimization solver, and an iteration count of 500.

All of the models were trained with 5-fold cross validation on training data and using weighted $F1$ as scoring function for 10 different runs. We chose the weighted $F1$ as our scoring function since the datasets are multi-class and have imbalanced classes. To evaluate the performance of different models, we use overall accuracy and $F1$ score of the classifier against test data. We also employ one against one scheme for multi-class classifiers.

C. Results

We performed an $80 - 20$ stratified split on the datasets, where 80% of the data was used for training and the rest 20% for testing, and ran experiments using ORION and baseline methods. Table II presents the results of that experiment. In case of Indian Pines and Salinas, we see that ORION with rank 1000 and 2000 performs better than the baselines. In case of University of Pavia, ORION with 2000 rank performs better than all baselines, with the Multi-Layer Perceptron being a close second. We discuss the results of Salinas-A in subsection IV-D.

One of the challenges in HSI classification is that the amount of labelled data available is limited. To demonstrate this, we run all experiments using 30-70 stratified split, where 30% is training data and 70% is testing data. For the most part, our results remain similar. For Indian Pines and Salinas datasets, ORION with 1000 and 2000 rank provides better overall classification accuracy. In case of University of Pavia, two best performing methods are SVM with RBF kernel and MLP, but ORION with rank 2000 performs at par with the

TABLE II
CLASSIFICATION ACCURACY OF ALL THE METHODS FOR 80-20 SPLIT

	Indian Pines	Pavia University	Salinas-A	Salinas
Linear SVM	0.8708± 0.0035	0.9176 ± 0.0017	**0.9986 ± 0.0016**	0.9339 ± 0.0014
Polynomial SVM	0.8979± 0.0054	0.9481 ± 0.0015	0.9978 ± 0.0015	0.9463 ± 0.0014
RBF SVM	0.9178± 0.0050	0.9622 ± 0.0020	**0.9985 ± 0.0017**	0.9620 ± 0.0024
MLP	0.9182± 0.0057	**0.9635 ± 0.0041**	0.9982 ± 0.0010	0.9629 ± 0.0045
ORION -1000	**0.9916± 0.0022**	0.9502 ± 0.0032	0.9690 ± 0.0067	**0.9927 ± 0.0010**
ORION -2000	**0.9949± 0.0022**	**0.9828 ± 0.0030**	0.9680 ± 0.0063	**0.9954 ± 0.0006**

TABLE III
CLASSIFICATION ACCURACY OF ALL THE METHODS FOR 30-70 SPLIT

	Indian Pines	Pavia University	Salinas-A	Salinas
Linear SVM	0.8371 ± 0.0034	0.9134 ± 0.0015	**0.9965 ± 0.0010**	0.9322 ± 0.0007
Polynomial SVM	0.8511 ± 0.0042	0.9367 ± 0.0010	0.9941 ± 0.0017	0.9406 ± 0.0009
RBF SVM	0.8739 ± 0.0041	**0.9546 ± 0.0007**	**0.9966 ± 0.0011**	0.9515 ± 0.0012
MLP	0.8693 ± 0.0098	**0.9556 ± 0.0029**	0.9931 ± 0.0029	0.9475 ± 0.0041
ORION -1000	**0.9725 ± 0.0032**	0.9119 ± 0.0015	0.8607 ± 0.0146	**0.9662 ± 0.0013**
ORION -2000	**0.9806 ± 0.0031**	0.9544 ± 0.0021	0.8982 ± 0.0073	**0.9832 ± 0.0013**

TABLE IV
MEAN F1-SCORE OF ALL THE METHODS FOR 80-20 SPLIT OVER 10 RUNS

	Indian Pines	Pavia University	Salinas-A	Salinas
Linear SVM	0.8700 ± 0.0036	0.9162 ± 0.0019	**0.9986 ± 0.0016**	0.9326 ± 0.0016
Polynomial SVM	0.8977 ± 0.0054	0.9477 ± 0.0016	0.9978 ± 0.0015	0.9454 ± 0.0014
RBF SVM	0.9175 ± 0.0050	0.9620 ± 0.0020	**0.9985 ± 0.0017**	0.9620 ± 0.0024
MLP	0.9180 ± 0.0056	**0.9634 ± 0.0040**	0.9982 ± 0.0010	0.9628 ± 0.0045
ORION -1000	**0.9915 ± 0.0022**	0.9484 ± 0.0038	0.9687 ± 0.0068	**0.9927 ± 0.0010**
ORION -2000	**0.9949 ± 0.0022**	**0.9823 ± 0.0032**	0.9675 ± 0.0066	**0.9954 ± 0.0006**

TABLE V
MEAN F1-SCORE OF ALL THE METHODS FOR 30-70 SPLIT OVER 10 RUNS

	Indian Pines	Pavia University	Salinas-A	Salinas
Linear SVM	0.8358 ± 0.0033	0.9118 ± 0.0016	**0.9965 ± 0.0010**	0.9310 ± 0.0006
Polynomial SVM	0.8503 ± 0.0042	0.9361 ± 0.0010	0.9941 ± 0.0017	0.9396 ± 0.0009
RBF SVM	0.8734 ± 0.0041	**0.9544 ± 0.0007**	**0.9966 ± 0.0011**	0.9512 ± 0.0012
MLP	0.8690 ± 0.0095	**0.9555 ± 0.0029**	0.9931 ± 0.0029	0.9469 ± 0.0041
ORION -1000	**0.9725 ± 0.0032**	0.9068 ± 0.0018	0.8583 ± 0.0151	**0.9661 ± 0.0013**
ORION -2000	**0.9804 ± 0.0031**	0.9528 ± 0.0025	0.8961 ± 0.0074	**0.9832 ± 0.0012**

baseline and has similar classification accuracy. This depicts that ORION is effective even with limited labelled data. For both of the scenarios (80-20 and 30-70 split), tables IV and V report mean $F1$ scores of our method and baselines. They follow the same trend as the overall accuracy.

We explore the effect of rank on the overall accuracy for Indian Pines and Salinas-A datasets. Figure 2 shows the plot of mean accuracy vs. rank for Indian Pines dataset. We observed that as the rank increases, classification accuracy improves as well until a certain point, where the change in rank does not provide any significant improvements and the accuracy stabilizes.

D. Discussion about Salinas-A and Salinas

The Salinas-A (Figure 1(b)) dataset has 5348 labelled pixels with 6 classes, and is a subscene of the full Salinas (Figure 1)(a) dataset which has 54129 labelled pixels and 16 classes. For Salinas-A, all baselines outperform ORION with rank 1000 and 2000, however, in the case of Salinas, ORION with same ranks outperforms all the baselines. This trend is similar in both 80-20 and 30-70 splits. Upon visual inspection, Salinas-A appears linearly separable whereas Salinas is not. We conjecture that Salinas appears to have more concrete and uniform blocks, potentially better trilinear structure that is exploited via CP decomposition. Judging the trilinearity of a dataset is a difficult problem and while there exist heuristics for this [22], we reserve further investigation for our future work.

V. RELATED WORK

Hyperspectral image classification takes as input a set of observations and assigns a unique label to each pixel [6]. Supervised linear methods in HSI classification are prone to the curse of dimensionality due to the lack of large number of training samples [23]. Support vector machines (SVM) have been employed to deal with this phenomenon [8]. SVMs allow classification of data points in a higher dimensional space using a nonlinear transformation.

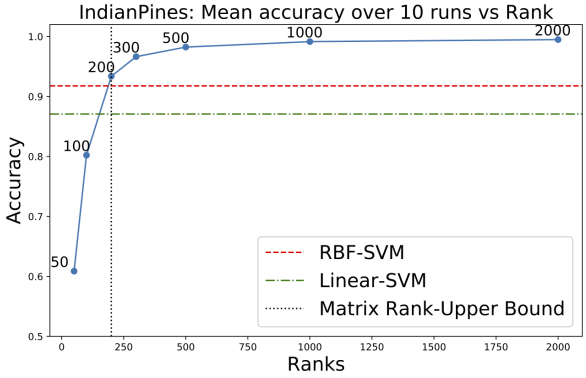


Fig. 2. Mean accuracy vs. Rank of the tensor observed over 10 runs. As rank increases, the classification accuracy increases and stabilizes after a certain point.

Tensor methods like CP decomposition [4], [24] have been used to represent high-order feature data in low-dimensional space with good accuracy. [25] presented a deep learning-based classification method that hierarchically constructs high-level features automatically. In particular, their model exploits a convolutional neural network (CNN) to encode the spectral and spatial information of pixels and a multilayer perceptron to conduct the classification task. [26] use recurrent neural network (RNN) to characterize the sequential property of a hyperspectral pixel vector for the classification task. Our proposed method ORION employs factors obtained from tensor factorization to generate a feature space that maps the given feature space to a higher-dimensional space in order to improve classification accuracy.

VI. CONCLUSIONS

In this paper, we propose a novel hyperspectral pixel classification model that employs tensor factorization to generate a new feature space. Specifically, our proposed method ORION exploits the multi-linear structure of the tensor to find a richer space. To showcase the effectiveness of our method, We conducted experiments using publicly available hyperspectral image datasets where we compared ORION against baselines methods like Kernel SVMs and Multi-layer Perceptron. Our proposed method is able to provide significantly higher accuracy in majority of the cases, even with limited training samples.

ACKNOWLEDGEMENTS

This research was supported in part by an appointment to the Oak Ridge National Laboratory ASTRO Program funded through ORNL's Laboratory Directed Research and Development(LDRD), sponsored by the U.S. Department of Energy and administered by the Oak Ridge Institute for Science and Education. This research used resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC05-00OR22725. Research was partially supported by the National Science Foundation CDS&E Grant no. OAC-1808591.

REFERENCES

[1] A. F. Goetz, G. Vane, J. E. Solomon, and B. N. Rock, "Imaging spectrometry for earth remote sensing," *science*, vol. 228, no. 4704, pp. 1147–1153, 1985.

[2] G. Camps-Valls, L. Gomez-Chova, J. Muñoz-Marí, J. Vila-Francés, and J. Calpe-Maravilla, "Composite kernels for hyperspectral image classification," *IEEE geoscience and remote sensing letters*, vol. 3, no. 1, pp. 93–97, 2006.

[3] Y. Chen, N. M. Nasrabadi, and T. D. Tran, "Hyperspectral image classification via kernel sparse representation," *IEEE Transactions on Geoscience and Remote sensing*, vol. 51, no. 1, pp. 217–231, 2012.

[4] K. Makantasis, A. D. Doulamis, N. D. Doulamis, and A. Nikitakis, "Tensor-based classification models for hyperspectral data analysis," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 12, pp. 6884–6898, 2018.

[5] N. Keshava and J. F. Mustard, "Spectral unmixing," *IEEE signal processing magazine*, vol. 19, no. 1, pp. 44–57, 2002.

[6] J. M. Bioucas-Dias, A. Plaza, N. Dobigeon, M. Parente, Q. Du, P. Gader, and J. Chanussot, "Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches," *IEEE journal of selected topics in applied earth observations and remote sensing*, vol. 5, no. 2, pp. 354–379, 2012.

[7] J. M. Bioucas-Dias, A. Plaza, G. Camps-Valls, P. Scheunders, N. Nasrabadi, and J. Chanussot, "Hyperspectral remote sensing data analysis and future challenges," *IEEE Geoscience and remote sensing magazine*, vol. 1, no. 2, pp. 6–36, 2013.

[8] G. Camps-Valls and L. Bruzzone, "Kernel-based methods for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 6, pp. 1351–1362, 2005.

[9] G. de Inteligencia Computacional, "Hyperspectral remote sensing scenes," http://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes. [Online]. Available: http://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes

[10] J. D. Carroll and J.-J. Chang, "Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition," *Psychometrika*, vol. 35, no. 3, pp. 283–319, 1970.

[11] R. A. Harshman *et al.*, "Foundations of the PARAFAC procedure: models and conditions for an explanatory multimodal factor analysis," *UCLA Working Papers in Phonetics*, vol. 16, pp. 1–84, 1970.

[12] E. Acar, D. M. Dunlavy, T. G. Kolda, and M. Mørup, "Scalable tensor factorizations for incomplete data," *Chemometrics and Intelligent Laboratory Systems*, vol. 106, no. 1, pp. 41–56, 2011.

[13] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 208–220, 2012.

[14] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

[15] E. E. Papalexakis, C. Faloutsos, and N. D. Sidiropoulos, "Tensors for data mining and data fusion: Models, applications, and scalable algorithms," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 8, no. 2, p. 16, 2017.

[16] N. D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos, "Tensor decomposition for signal processing and machine learning," *IEEE Transactions on Signal Processing*, vol. 65, no. 13, pp. 3551–3582, 2017.

[17] B. W. Bader, T. G. Kolda *et al.*, "Matlab tensor toolbox version 2.6," Available online, February 2015. [Online]. Available: http://www.sandia.gov/~tgkolda/TensorToolbox/

[18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[19] J. Kossaifi, Y. Panagakis, A. Anandkumar, and M. Pantic, "Tensorly: Tensor learning in python," *Journal of Machine Learning Research (JMLR)*, vol. 20, no. 26, 2019.

[20] M. F. Baumgardner, L. L. Biehl, and D. A. Landgrebe, "220 band aviris hyperspectral image data set: June 12, 1992 indian pine test site 3," Sep 2015. [Online]. Available: https://purr.purdue.edu/publications/1947/1

[21] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.
[22] R. Bro and H. A. Kiers, "A new efficient method for determining the number of components in parafac models," *Journal of Chemometrics: A Journal of the Chemometrics Society*, vol. 17, no. 5, pp. 274–286, 2003.
[23] G. Hughes, "On the mean accuracy of statistical pattern recognizers," *IEEE transactions on information theory*, vol. 14, no. 1, pp. 55–63, 1968.
[24] M. Jouni, M. Dalla Mura, and P. Comon, "Hyperspectral image classification using tensor cp decomposition," in *IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2019, pp. 1164–1167.
[25] K. Makantasis, K. Karantzalos, A. Doulamis, and N. Doulamis, "Deep supervised learning for hyperspectral data classification through convolutional neural networks," in *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, 2015, pp. 4959–4962.
[26] L. Mou, P. Ghamisi, and X. X. Zhu, "Deep recurrent neural networks for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 7, pp. 3639–3655, 2017.