SAND2017-3184C

Supervised non-negative tensor factorization for automatic hyperspectral feature extraction and target discrimination

Dylan Anderson[a], Aleksander Bapst[a], Joshua Coon[a], Aaron Pung[a], and Michael Kudenov[b]

[a]Sandia National Laboratories, PO Box 5800, Albuquerque, USA
[b]North Carolina State University, 437 Monteith, Raleigh, USA

ABSTRACT

Hyperspectral imaging provides a highly discriminative and powerful signature for target detection and discrimination. Recent literature has shown that considering additional target characteristics, such as spatial or temporal profiles, simultaneously with spectral content can greatly increase classifier performance. Considering these additional characteristics in a traditional discriminative algorithm requires a feature extraction step be performed first. An example of such a pipeline is computing a filter bank response to extract spatial features followed by a support vector machine (SVM) to discriminate between targets. This decoupling between feature extraction and target discrimination yields features that are suboptimal for discrimination, reducing performance. This performance reduction is especially pronounced when the number of features or available data is limited. In this paper, we propose the use of Supervised Nonnegative Tensor Factorization (SNTF) to jointly perform feature extraction and target discrimination over hyperspectral data products. SNTF learns a tensor factorization and a classification boundary from labeled training data simultaneously. This ensures that the features learned via tensor factorization are optimal for both summarizing the input data and separating the targets of interest. Practical considerations for applying SNTF to hyperspectral data are presented, and results from this framework are compared to decoupled feature extraction/target discrimination pipelines.

Keywords: hyperspectral, tensor factorization, discriminative, classification, dimensionality reduction

1. INTRODUCTION

Hyperspectral imaging (HSI) contains hundreds of contiguous and narrow spectral bands, providing a highly discriminative signature for different materials and targets. This rich discriminative information has made hyperspectral data highly valuable in a broad range of applications. Examples include precision agriculture,[1] food safety,[2] target signature detection,[3] and disaster management.[4] Traditionally, these applications are addressed via a pixel-wise classification to assign land-cover, target type, material, etc. Recent advances have continually improved spatial and spectral resolution, and facilitated collection of additional modalities such as temporal or polarimetric.[5] Considering these modalities jointly with spectral information can improve classification performance.[6]

Performing classification directly in hyperspectral data space is subject to the curse of dimensionality and prone to overfitting, with the amount of data required scaling exponentially with the number of spectral bands. To remedy this, dimensionality reduction techniques are employed prior to classification. Linear dimensionality reduction techniques, such as those based on matrix factorization, have enjoyed great success within hyperspectral applications; these techniques are well aligned to spectral mixing that occurs on a detector focal plane. Performing dimensionality reduction prior to classification has a fundamental limitation: class information is not used to extract spectral features. The subspace is learned to represent the data with minimal error, but the subspace that best represents the data may be poorly suited for discriminating between classes. For instance, if background clutter accounts for the majority of spectral variation then the subspace will be dominated by background representation.

Further author information: (Send correspondence to Dylan Anderson)
Dylan Anderson: E-mail: dzander@sandia.gov, Telephone: 1 505 844 7013

In this paper, we introduce the Supervised Non-Negative Tensor Factorization (SNTF), a model that performs linear spectral unmixing and classification simultaneously. SNTF learns a linear subspace specifically targeted to the classification task via the Fisher discriminant criterion, boosting the discriminative power of each spectral feature. This improves classification performance, particularly when the number of mixing components is limited. Furthermore, since SNTF factorizes tensors it naturally extends to consider additional modalities such as spatial, temporal, or polarimetric. Finally, the projection and classification mechanism of SNTF mimics integration of data into multi-spectral bands, allowing it to be used as an application-specific sensor design tool. This paper proceeds as follows. In Section 2, we introduce notation and review matrix and tensor factorization models. In Section 3, we introduce the SNTF model and multiplicative update learning algorithm. In Section 4 we demonstrate SNTF experimentally followed by conclusions in Section 5.

2. BACKGROUND

2.1 Notation

In this section, we introduce the notation used throughout this paper. We adopt the notation and conventions of Ref 7. These notations are summarized in Table 1. A *tensor* is a multidimensional array. The *order* of a tensor is its number of dimensions. Tensors represent generalizations of vectors and matrices; matrices are second-order tensors, vectors are first-order tensors, and scalars are zero-order tensors. Tensors of order three or greater are called higher order tensors. We denote scalars as lowercase letters, e.g. m, vectors as boldface lowercase letters, e.g. $\boldsymbol{m}$, matrices as boldface uppercase letters, e.g. $\boldsymbol{M}$, and higher-order tensors as boldface script letters, e.g. $\boldsymbol{\mathcal{M}}$.

We denote the ith element of vector $\boldsymbol{m}$ as m_i, the (i,j) element of matrix $\boldsymbol{M}$ as m_{ij}, and the (i,j,k) element of a third-order tensor $\boldsymbol{\mathcal{M}}$ as m_{ijk}, or more succinctly $m_{\boldsymbol{i}}$, where $\boldsymbol{i}$ denotes the tuple (i,j,k). Indices range from 1 to their capital, e.g. $i = 1,\ldots,I$. The nth element in a sequence is denoted by a superscript in parentheses, e.g., $\boldsymbol{M}^{(n)}$ denotes the nth matrix in a sequence of N matrices. A *fiber* is defined by fixing every index of a tensor but one. Columns and rows are mode-1 and mode-2 fibers of matrices, respectively. Fibers are always assumed to be oriented as column vectors.

We denote hyperspectral-data as an N^{th}-order tensor, $\boldsymbol{\mathcal{X}} \in \mathbb{R}_+^{I_1\times\ldots\times I_N}$. For a pixel-wise hyper-spectral image, we have $\boldsymbol{X} \in \mathbb{R}_+^{B\times IJ}$, where the image has size $I\times J$ and B spectral bands. Spatial information can be included, such as local-neighborhoods or extended morphological attribute profiles, resulting in a higher-order tensor.[6] For convenience, we denote the spectral mode as b. The class labels corresponding to $\boldsymbol{\mathcal{X}} \in \mathbb{R}_+^{I_1\times\ldots\times I_N}$, are denoted by the tensor $\boldsymbol{\mathcal{Y}} \in \mathbb{Z}^{I_1\times\ldots\times I_{b-1}\times I_{b+1}\times\ldots\times I_N}$. For a pixel-wise hyperspectral image, this corresponds to a vector of class labels over the pixels, given by $\boldsymbol{y} \in \mathbb{Z}^{IJ}$. We now introduce several algebraic operations used in this paper.

1. **Matricization:** The mode-n matricization of a tensor $\boldsymbol{\mathcal{X}}$ is denoted by $\boldsymbol{X}_{(n)}$ and arranges the mode-n fibers into columns of a matrix. Formally,[7] tensor element $(i_1, i_2, \ldots, i_N)$ maps to matrix element (i_n, j), where
$$j = 1 + \sum_{k=1,k\neq n}^{N} (i_k - 1)J_k \quad \text{with} \quad J_k = \prod_{m=1,m\neq n}^{k-1} I_m.$$

2. **Outer Product:** The outer product of the tensors $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1\times I_2\times\ldots\times I_N}$ and $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{J_1\times J_2\times\ldots\times J_M}$ is given by
$$\boldsymbol{\mathcal{Z}} = \boldsymbol{\mathcal{X}} \circ \boldsymbol{\mathcal{Y}} \in \mathbb{R}^{I_1\times I_2\times\ldots\times I_N\times J_1\times J_2\times\ldots\times J_M}$$
where
$$z_{i_1,i_2,\ldots i_N,j_1,j_2,\ldots j_M} = x_{i_1,i_2,\ldots i_N} y_{j_1,j_2,\ldots j_M}$$

3. **Element-Wise Product:** The element-wise multiplication (division) to two equally-sized tensors $\boldsymbol{\mathcal{A}} \circledast \boldsymbol{\mathcal{B}}$ $(\boldsymbol{\mathcal{A}} \oslash \boldsymbol{\mathcal{B}})$ produces a tensor $\boldsymbol{\mathcal{C}}$ of the same size with elements $c_{\boldsymbol{i}} = a_{\boldsymbol{i}} b_{\boldsymbol{i}}$ $(c_{\boldsymbol{i}} = a_{\boldsymbol{i}}/b_{\boldsymbol{i}})$ for all $\boldsymbol{i}$.

4. **Kronecker Product:** For two vectors $\boldsymbol{a} \in \mathbb{R}^J$ and $\boldsymbol{b} \in \mathbb{R}^T$, the Kronecker product (denoted $\otimes$) is given by

$$\boldsymbol{a} \otimes \boldsymbol{b} = \begin{bmatrix} a_1\boldsymbol{b} \\ a_2\boldsymbol{b} \\ \vdots \\ a_J\boldsymbol{b} \end{bmatrix} \in \mathbb{R}^{JT}.$$

5. **Khatri-Rao Product:** For two matrices $\boldsymbol{A} \in \mathbb{R}^{I\times J}$ and $\boldsymbol{B} \in \mathbb{R}^{T\times J}$ with the same number of columns J, their Khatri-Rao product (denoted $\odot$) is given by

$$\boldsymbol{A} \odot \boldsymbol{B} = \left[\boldsymbol{a}_1 \otimes \boldsymbol{b}_1, \boldsymbol{a}_2 \otimes \boldsymbol{b}_2, \ldots, \boldsymbol{a}_J \otimes \boldsymbol{b}_J\right] \in \mathbb{R}^{IT\times J}.$$

6. **Mode-n Product:** The mode-n product between a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1\times\ldots\times I_N}$ and matrix $\boldsymbol{A} \in \mathbb{R}^{J_n\times I_n}$ (denoted $\times_n$) is given by

$$\boldsymbol{\mathcal{Y}} = \boldsymbol{\mathcal{X}} \times_n \boldsymbol{A} \in \mathbb{R}^{I_1\times\ldots\times I_{n-1}\times J_n\times I_{n+1}\times\ldots\times I_N}$$
$$y_{i_1,\ldots i_{n-1},j_n,i_{n+1},\ldots i_N} = \sum_{i_n} x_{i_1,\ldots i_N} a_{i_n,j_n}$$

Table 1: Notations used in this paper.

Symbol	Definition
α, λ	scalar
$\boldsymbol{m}$	vector
$\boldsymbol{M}$	matrix
$\boldsymbol{\mathcal{M}}$	tensor
$m_{\boldsymbol{i}}$	tensor element at index $\boldsymbol{i}$
$\boldsymbol{\mathcal{X}}$	hyperspectral data
$\boldsymbol{\mathcal{Y}}$	class labels
b	spectral mode
$\boldsymbol{M}_{(n)}$	mode-n matricization
$\circledast$	element-wise multiplication
$\oslash$	element-wise division
$\circ$	outer product
$\otimes$	Kronecker product
$\odot$	Khatri-Rao product
$\times_n$	mode-n product

2.2 Spectral Unmixing as Matrix Decomposition

Hyperspectral signatures collected in natural environments are invariably a mix of multiple components, regardless of spatial resolution. Higher order effects, such as atmospheric scattering, can introduce mixing effects into otherwise pure pixels. The narrow, contiguous spectral bands typically greatly outnumber the mixture components, allowing the unmixing problem to be cast as the solution of an over-determined linear system. This system can be solved via matrix factorization (MF) techniques. Standard MF finds two matrices whose product approximates the original data $\boldsymbol{X} \approx \boldsymbol{W}\boldsymbol{H}$. Many techniques for MF exist, but singular value decomposition (SVD, or, equivalently, principal component analysis, PCA) and Non-negative matrix factorization (NMF) are the most common. NMF and its variants have been widely used for spectral unmixing. By imposing non-negativity constraints the learned matrices are readily interpretable: $\boldsymbol{W}$ represents the spectral mixing components present in the scene and $\boldsymbol{H}$ represents the abundances of each component present in a pixel. This is shown in Figure 1.

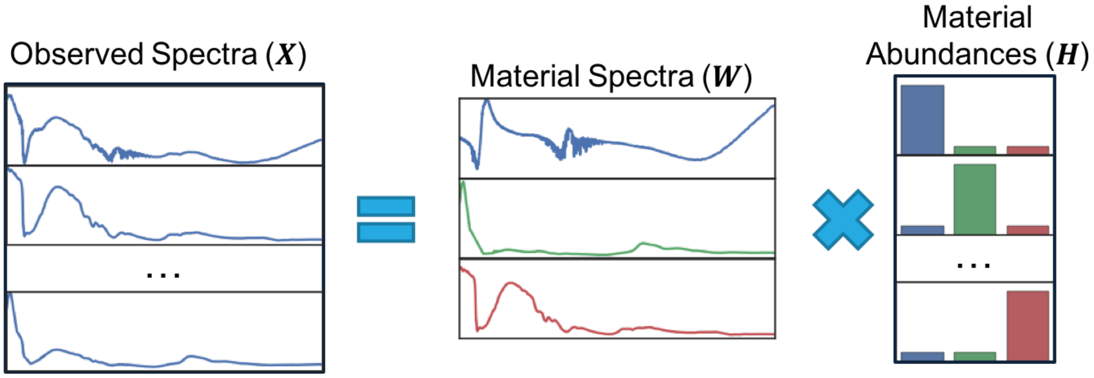
Observed Spectra ($\boldsymbol{X}$)
Material Spectra ($\boldsymbol{W}$)
Material Abundances ($\boldsymbol{H}$)
...
...

Figure 1: Spectral mixing formulated as matrix factorization.

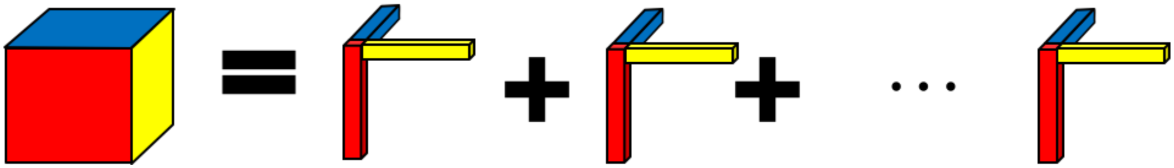

Figure 2: The CP tensor decomposition model. CP represents the tensor as a summation of rank one tensors.

2.3 Tensor Decomposition

Tensor factorization (decomposition) extends the concepts of matrix factorization and leverages multi-way structural information that is lost when data modes are collapsed to apply matrix factorization.[8,9] Hyperspectral data is frequently collected with modes in addition to spectral such as spatial or temporal. By exploiting the multi-linear structure of data, the underlying mixing components typically become more enhanced and better exposed.[9] For example, consider a series of hyperspectral images collected over the same scene at different times of day. Adjacent temporal collections should be highly correlated, but by matricizing the data into a spectral-by-samples matrix the temporal covariance information is lost.

The Canonical Decomposition (CANDECOMP, or simply CP – also known as PARAFAC) model is a common tensor decomposition that can be viewed as a multi-linear generalization of the singular value decomposition.[7] The CP model approximates an N^{th}-order tensor $\boldsymbol{\mathcal{X}}$ as a sum of K rank-one tensors, which can be expressed as:

$$\boldsymbol{\mathcal{X}} = \sum_{k=1}^{K} \boldsymbol{a}_k^{(1)} \circ \ldots \circ \boldsymbol{a}_k^{(N)} = [\![\boldsymbol{A}^{(1)}, \ldots, \boldsymbol{A}^{(N)}]\!]. \tag{1}$$

This is shown in Figure 2. To fit the factors, $\boldsymbol{A}^{(n)}$, a minimization over a divergence between $\boldsymbol{\mathcal{X}}$ and $[\![\boldsymbol{A}^{(1)}, \ldots, \boldsymbol{A}^{(N)}]\!]$ is performed. Details of the CP model, along with other common tensor decompositions can be found in Ref 7.

As in matrix factorization, a natural modification to the CP model is to impose non-negativity constraints which leads to physically meaningful factors: $\boldsymbol{A}^{(b)} \in \mathbb{R}_{+}^{B \times K}$ represents the K dominant spectra present in $\boldsymbol{\mathcal{X}}$. We denote the CP model with non-negativity constraints as Non-negative Tensor Factorization (NTF). A common strategy for learning NTF is to extend the well-known multiplicative update rule (MUR) for Non-negative Matrix Factorization.[10] The MUR cycles over each of the N factor matrices, updating one while holding the rest constant. Since factors are updated by multiplication with a non-negative update, initializing factors with non-negative values ensures they remain non-negative.

The most common functions to minimize are Frobenius-norm of the reconstruction error and generalized Kullback-Leibler (KL) I-Divergence. The generalized KL divergence represents the distance between two distributions, and for NTF is given by

$$D_{KL}\left(\boldsymbol{\mathcal{X}} \parallel [\![\boldsymbol{A}^{(1)}, \ldots, \boldsymbol{A}^{(N)}]\!] \right) = \sum_{\boldsymbol{i}} \left(x_{\boldsymbol{i}} \ln \frac{x_{\boldsymbol{i}}}{[\![\boldsymbol{A}^{(1)}, \ldots, \boldsymbol{A}^{(N)}]\!]_{\boldsymbol{i}}} - x_{\boldsymbol{i}} + [\![\boldsymbol{A}^{(1)}, \ldots, \boldsymbol{A}^{(N)}]\!]_{\boldsymbol{i}} \right). \tag{2}$$

The NTF model for KL divergence is fit by the N minimizations

$$\left\{ \min_{\boldsymbol{A}^{(n)}>0} D_{KL}\left(\boldsymbol{\mathcal{X}} \parallel \llbracket \boldsymbol{A}^{(1)}, \ldots, \boldsymbol{A}^{(N)} \rrbracket \right) \right\}_{n=1}^{N}. \tag{3}$$

The Karush-Kuhn-Tucker (KKT) optimality conditions for the I-divergence are[9]

$$\boldsymbol{A}^{(n)} \geq 0, \tag{4}$$

$$\nabla_{\boldsymbol{A}^{(n)}} D_{KL} \geq 0, \tag{5}$$

$$\boldsymbol{A}^{(n)} \circledast \nabla_{\boldsymbol{A}^{(n)}} D_{KL} \geq 0, \quad \forall n. \tag{6}$$

The gradient of Eq. (2) with respect to a factor matrix is given by:

$$\nabla_{\boldsymbol{A}^{(n)}} D_{KL} = \mathbf{1}\mathbf{1}^T \boldsymbol{A}^{\odot -n} - \left(\boldsymbol{X}_{(n)} \oslash \hat{\boldsymbol{X}}_{(n)} \right) \boldsymbol{A}^{\odot -n}, \tag{7}$$

where $\boldsymbol{A}^{\odot -n} = \left[\boldsymbol{A}^{(N)} \odot \ldots \odot \boldsymbol{A}^{(n+1)} \odot \boldsymbol{A}^{(n-1)} \odot \ldots \odot \boldsymbol{A}^{(1)} \right]$ and $\hat{\boldsymbol{X}}_{(n)} = \llbracket \boldsymbol{A}^{(1)}, \ldots, \boldsymbol{A}^{(N)} \rrbracket_{(n)}$. Substituting Eq. (7) into Eq. (6) gives the MUR:

$$\boldsymbol{A}^{(n)} \leftarrow \boldsymbol{A}^{(n)} \circledast \left\{ \left[\left(\boldsymbol{X}_{(n)} \oslash \hat{\boldsymbol{X}}_{(n)} \right) \boldsymbol{A}^{\odot -n} \right] \oslash \left(\mathbf{1}\mathbf{1}^T \boldsymbol{A}^{\odot -n} \right) \right\} \tag{8}$$

which can be simplified further by neglecting the scaling diagonal matrix. The algorithm with this simplification is given in Algorithm 1. For a survey on non-negative tensor decompositions and learning algorithms, see Ref 9.

Algorithm 1 KL-NTF

```
Input: Input data $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$, number of components $K$
Output: N factor matrices, $\boldsymbol{A}^{(n)} \in \mathbb{R}_+^{I_N \times K}$
repeat
    $\hat{\boldsymbol{X}}_{(n)} = \llbracket \boldsymbol{A}^{(1)}, \ldots, \boldsymbol{A}^{(N)} \rrbracket_{(n)}$
    for $n = 1$ to $N$ do
        $\boldsymbol{A}^{(n)} \leftarrow \boldsymbol{A}^{(n)} \circledast \left[ \left( \boldsymbol{X}_{(n)} \oslash \hat{\boldsymbol{X}}_{(n)} \right) \boldsymbol{A}^{\odot -n} \right]$
        if $n \neq N$ then
            $\boldsymbol{A}^{(n)} = \boldsymbol{A}^{(n)} \mathrm{diag} \mathbf{1}^T {\boldsymbol{A}^{(n)}}^{-1}$
        end if
    end for
until stopping criterion met
```

3. SUPERVISED NON-NEGATIVE TENSOR FACTORIZATION

In this section, we derive the Supervised Non-negative Tensor Factorization (SNTF) model. This model incorporates auxiliary class information into the tensor factorization process by means of the linear Fisher discriminant criterion.[11,12] By including class information, the spectral factors are learned by balancing representing the input data and linearly separating the classes. The model is built from the NTF model described in Section 2.3, and uses a multiplicative update scheme for learning.

3.1 Regularized NTF

Before introducing the SNTF model, we first revisit the NTF model from Section 2.3. In practice, we have found that regularizing the band factors $\boldsymbol{A}^{(b)}$ improves both classification performance and factor interpretability when applied to hyperspectral data. The multiplicative update rule allows for straightforward inclusion of regularization terms. Specifically, we add smoothing and decorrelation terms to the KL-divergence objective.

The smoothing term penalizes large values of the finite difference approximation of the spectral bands. The smoothing term and its gradient are given by

$$\frac{\alpha_{sm}}{2}\|\boldsymbol{L}\boldsymbol{A}^{(b)}\|_F^2, \quad \text{where } \boldsymbol{L} = \begin{bmatrix} -1 & 2 & -1 & & & 0 \\ & -1 & 2 & -1 & & \\ \vdots & & \ddots & \ddots & \ddots & \\ 0 & & & -1 & 2 & -1 \end{bmatrix} \tag{9}$$

$$\nabla_{\boldsymbol{A}^{(n)}} = \alpha_{sm}\boldsymbol{L}^T\boldsymbol{L}\boldsymbol{A}^{(b)}. \tag{10}$$

The decorrelation term penalizes correlation between spectral factors, encouraging them to represent different spectral regions. The term and its gradient are given by

$$\alpha_{cr}Tr(\boldsymbol{A}^{(b)^T}\boldsymbol{1}_{B\times B}\boldsymbol{A}^{(b)}), \tag{11}$$

$$\nabla_{\boldsymbol{A}^{(n)}} = \alpha_{cr}\boldsymbol{A}^{(b)}\boldsymbol{1}_{K\times K}. \tag{12}$$

For a derivation of how Eq. (11) penalizes correlation between factors, see Ref. 9. The optimization problem for $\boldsymbol{A}^{(b)}$ becomes

$$\min_{\boldsymbol{A}^{(b)}>0}\left[D_{KL}\left(\boldsymbol{\mathcal{X}} \parallel [\![\boldsymbol{A}^{(1)},\dots,\boldsymbol{A}^{(N)}]\!]\right) + \frac{\alpha_{sm}}{2}\|\boldsymbol{L}\boldsymbol{A}^{(b)}\|_F^2 + \alpha_{cr}Tr(\boldsymbol{A}^{(b)^T}\boldsymbol{1}_{K\times K}\boldsymbol{A}^{(b)})\right], \tag{13}$$

and the corresponding multiplicative update is

$$\boldsymbol{A}^{(b)} \leftarrow \boldsymbol{A}^{(b)} \circledast \frac{\left(\boldsymbol{X}_{(n)} \oslash \hat{\boldsymbol{X}}_{(n)}\right)\boldsymbol{A}^{\odot -n}}{\boldsymbol{1}\boldsymbol{1}^T\boldsymbol{A}^{\odot -n} + \alpha_{sm}\boldsymbol{L}^T\boldsymbol{L}\boldsymbol{A}^{(b)} + \alpha_{cr}\boldsymbol{A}^{(b)}\boldsymbol{1}_{K\times K}}. \tag{14}$$

3.2 Model Formulation

Following Refs. 12 and 13, the class information in $\boldsymbol{\mathcal{Y}}$ is captured via Fisher's criterion, which expects samples of the same class to be close together and samples of different classes to be far apart in the low dimensional subspace. Contrary to Ref. 14, SNTF does not apply Fisher's criterion directly to the tensor subspace. Rather, it transforms the original data into a linear combination of spectral bands as follows:

$$\left[\boldsymbol{\mathcal{X}} \times_b \boldsymbol{A}^{(b)}\right]_{(b)} = \boldsymbol{A}^{(b)^T}\boldsymbol{X}_{(b)}. \tag{15}$$

This projective sub-space allows for class-discriminative information encoded during model fitting to be extended to unlabeled data at test time. In contrast to the direct tensor subspace, which is an additive basis, the mode-b product is analogous to applying optical filters. This simulates a multi-spectral system tuned to the classification problem. In this way, SNTF can be used as a sensor design tool for spectral band design. Imposing Fisher's criterion on this subspace yields the pair of optimization problems[12, 13]

$$\min_{\boldsymbol{A}^{(b)}} Tr(\boldsymbol{A}^{(b)^T}\boldsymbol{S}_w\boldsymbol{A}^{(b)}) \tag{16}$$

$$\max_{\boldsymbol{A}^{(b)}} Tr(\boldsymbol{A}^{(b)^T}\boldsymbol{S}_b\boldsymbol{A}^{(b)}), \tag{17}$$

where $\boldsymbol{S}_w$ denotes the within-class scatter (distance between samples of the same class), and $\boldsymbol{S}_b$ denotes the between class scatter (distance between different classes). This is shown graphically in Figure 3. The scatter matrices are given by

$$\boldsymbol{S}_w = \sum_c\sum_{j=1}^{n_c}\left(\boldsymbol{x}_{(b)j} - \bar{\boldsymbol{x}}_{(b)c}\right)\left(\boldsymbol{x}_{(b)j} - \bar{\boldsymbol{x}}_{(b)c}\right)^T \tag{18}$$

$$\boldsymbol{S}_b = \sum_c n_c\left(\bar{\boldsymbol{x}}_{(b)} - \bar{\boldsymbol{x}}_{(b)c}\right)\left(\bar{\boldsymbol{x}}_{(b)} - \bar{\boldsymbol{x}}_{(b)c}\right)^T \tag{19}$$

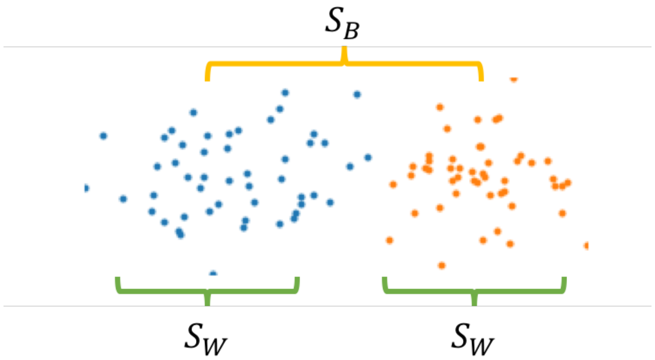

Figure 3: The within class scatter, $\boldsymbol{S}_W$, measures the distance between samples of the same class. The between class scatter, $\boldsymbol{S}_B$, measures the distance between classes. Fisher's criterion minimizes $\boldsymbol{S}_W$ and maximizes $\boldsymbol{S}_B$.

where C is the number of classes, n_c is the number of samples in class c, $\bar{\boldsymbol{x}}_{(b)c}$ denotes the mean-vector for class c and $\bar{\boldsymbol{x}}_{(b)}$ denotes the overall mean-vector. Eqs. (16) and (17) can be combined into a single optimization as

$$\min_{\boldsymbol{A}^{(b)}>0} Tr\left(\boldsymbol{A}^{(b)^T}(\lambda \boldsymbol{S}_w - \boldsymbol{S}_b)\boldsymbol{A}^{(b)}\right) \tag{20}$$

where λ is the largest eigenvalue of $\boldsymbol{S}_w^{-1}\boldsymbol{S}_b$ to guarantee the convexity of Fisher's criterion.[13] Combining Eq. (20) with the constrained objective in Eq. (13) gives the SNTF objective function:

$$\min_{\boldsymbol{A}^{(n)}>0} \Big[D_{KL}\left(\boldsymbol{\mathcal{X}} \parallel [\![\boldsymbol{A}^{(1)}, \dots, \boldsymbol{A}^{(N)}]\!]\right) + \frac{\alpha_{sm}}{2}\|\boldsymbol{L}\boldsymbol{A}^{(b)}\|_F^2 + \alpha_{cr} Tr(\boldsymbol{A}^{(b)^T}\mathbf{1}_{K\times K}\boldsymbol{A}^{(b)}) + \frac{\alpha}{2} Tr\left(\boldsymbol{A}^{(b)^T}(\lambda \boldsymbol{S}_w - \boldsymbol{S}_b)\boldsymbol{A}^{(b)}\right)\Big] \; \forall n, \tag{21}$$

where α controls the importance of Fisher's criterion relative to fitting the input data.

3.3 Learning Algorithm

The SNTF model seeks factor matrices, $\left\{\boldsymbol{A}^{(n)}\right\}_{n=1}^{N}$, that satisfy the optimization given in Eq. (21). Since the additional terms included in SNTF depend only on $\boldsymbol{A}^{(b)}$, the gradient for $\left\{\boldsymbol{A}^{(n)}\right\}_{n\neq b}$ is equal to Eq. (7), and the corresponding updates proceed as in NTF (Eq. (8)). To find the multiplicative update rule for $\boldsymbol{A}^{(b)}$, the gradient of Eq. (21) is

$$\nabla_{\boldsymbol{A}^{(b)}} D \; = \; \mathbf{1}\mathbf{1}^T \boldsymbol{A}^{\odot -n} - \left(\boldsymbol{X}_{(n)} \oslash \hat{\boldsymbol{X}}_{(n)}\right)\boldsymbol{A}^{\odot -n} + \alpha \boldsymbol{A}^{(b)^T}(\lambda \boldsymbol{S}_w - \boldsymbol{S}_b) + \alpha_{sm}\boldsymbol{L}^T\boldsymbol{L}\boldsymbol{A}^{(b)} + \alpha_{cr}\boldsymbol{A}^{(b)}\mathbf{1}_{K\times K}. \tag{22}$$

Any real-valued matrix can be split into the difference of two positive matrices defined by its positive elements minus its negative elements, i.e. $\boldsymbol{S} = [\boldsymbol{S}]_+ - [-\boldsymbol{S}]_+$, where the operator $[\boldsymbol{S}]_+$ keeps only the non-negative entries in $\boldsymbol{S}$. Applying this property to the term corresponding to the gradient of Fisher's criterion gives:

$$\lambda \boldsymbol{S}_w - \boldsymbol{S}_b = [\lambda \boldsymbol{S}_w - \boldsymbol{S}_b]_+ - [\boldsymbol{S}_b - \lambda \boldsymbol{S}_w]_+ \,. \tag{23}$$

Using Eq. (23) and subsituting Eq. (22) into the KKT optimality condition (Eq. (6)) gives the multiplicative update rule for $\boldsymbol{A}^{(b)}$:

$$\boldsymbol{A}^{(b)} \leftarrow \boldsymbol{A}^{(b)} \circledast \frac{\left(\boldsymbol{X}_{(n)} \oslash \hat{\boldsymbol{X}}_{(n)}\right)\boldsymbol{A}^{\odot -n} + \alpha\left[\boldsymbol{S}_b - \lambda \boldsymbol{S}_w\right]_+}{\mathbf{1}\mathbf{1}^T\boldsymbol{A}^{\odot -n} + \alpha\left[\lambda \boldsymbol{S}_w - \boldsymbol{S}_b\right]_+ + \alpha_{sm}\boldsymbol{L}^T\boldsymbol{L}\boldsymbol{A}^{(b)} + \alpha_{cr}\boldsymbol{A}^{(b)}\mathbf{1}_{K\times K}}. \tag{24}$$

Since numerator and denominator consist of summations of strictly non-negative terms, non-negativity of $\boldsymbol{A}^{(b)}$ is preserved if initialized with strictly non-negative elements. The algorithm for SNTF is listed in Algorithm 2.

Algorithm 2 SNTF

Input: Input data $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \dots \times I_N}$, number of components K, tuning parameters α, α_{sm}, and α_{cr}
Output: N factor matrices, $\boldsymbol{A}^{(n)} \in \mathbb{R}_{+}^{I_N \times K}$

```
repeat
    $\hat{\boldsymbol{X}}_{(n)} = [\![\boldsymbol{A}^{(1)}, \dots, \boldsymbol{A}^{(N)}]\!]_{(n)}$
    for $n = 1$ to $N$ do
        if n = b then
            Update $\boldsymbol{A}^{(b)}$ according to Eq. 24.
        else
            $\boldsymbol{A}^{(n)} \leftarrow \boldsymbol{A}^{(n)} \circledast \left[\left(\boldsymbol{X}_{(n)} \oslash \hat{\boldsymbol{X}}_{(n)}\right) \boldsymbol{A}^{\odot_{-n}}\right]$
        end if
        if $n \neq N$ then
            $\boldsymbol{A}^{(n)} = \boldsymbol{A}^{(n)} \mathrm{diag} \mathbf{1}^T {\boldsymbol{A}^{(n)}}^{-1}$
        end if
    end for
until stopping criterion met
```

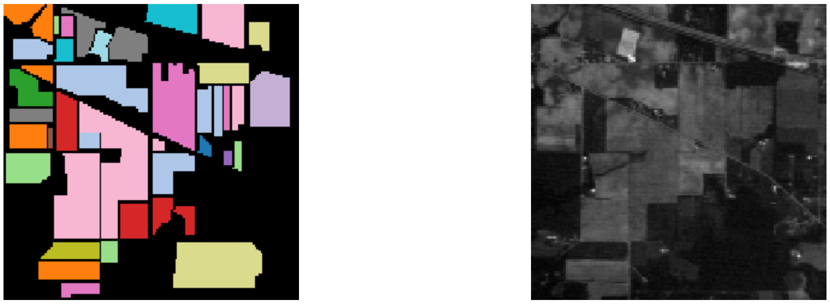

Figure 4: Indian Pines ground truth (left) and 450 nm band (right).

3.4 Class Predictions

In practice, any standard classification algorithm can be applied to the learned subspace given by ${\boldsymbol{A}^{(b)}}^T \boldsymbol{X}_{(b)}$. However, Fisher's criterion corresponds to maximum likelihood classification with multivariate normal distributions for each class.[15] Specifically, ${\boldsymbol{A}^{(b)}}^T \boldsymbol{X}_{(b)}$ is modeled as a Gaussian mixture with tied covariance matrices for each mixture component. To predict the class of unlabeled data, first its mode-b product with $\boldsymbol{A}^{(b)}$ is computed, and then its probability under each class distribution is evaluated. While more sophisticated classifiers can be used, this simple scheme provides probabilistic interpretations of classes and paves the way for open-set classification. This also allows for target detection schemes, and allows for imposing class priors to achieve posterior probabilities to set probability of detection thresholds.

4. EXPERIMENTS

We evaluate SNTF on the Indian Pines hyperspectral image classification dataset. Indian Pines is a 145 × 145 hyperspectral image of Northwestern Indiana. The scene contains two-thirds agriculture, and one-third forest and other native vegetation. The data contains 16 different classes, and 224 spectral bands from 0.4-2.5 μm. We utilize reflectance calibrated data, with 24 water absorption bands discarded. This data set is difficult due to the extreme class imbalance and relatively low number of samples for rare classes. The ground truth and a selected band from this dataset are shown in Figure 4.

4.1 Feature Extraction

In this section, we explore the spectral factors learned by SNTF. We fit NTF and SNTF models, each with a single spectral factor, over two classes from the Indian Pines dataset: "grass-pasture-mowed" and "hay-windrowed".

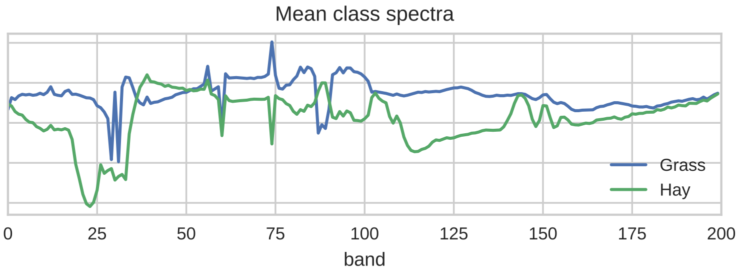

Figure 5: Average spectra for classes “grass-pasture-mowed” and “hay-windrowed” from the Indian Pines dataset.

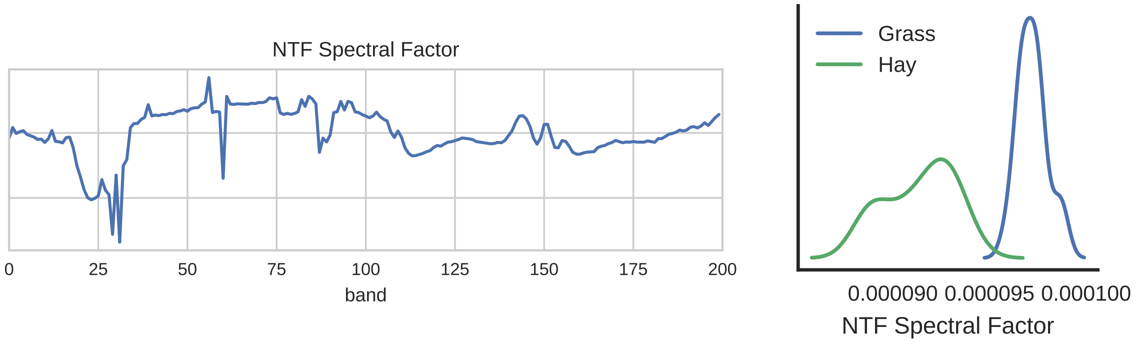

Figure 6: The NTF spectral factor learned for “grass-pasture-mowed” and “hay-windrowed”, and the distribution of these classes when this factor is applied. NTF is focused on fitting the data with minimal error, so it learns a broad spectral feature that does not fully separate the classes.

The average spectra for each of these classes is shown in Figure 5. NTF is focused on fitting the data with minimal error, so the factor it learns is spectrally broad to summarize the data accurately. Class information is not used while learning the factor, so it does not fully separate the two classes. The learned factor and the distributions of the two classes when the factor is applied are shown in Figure 6. On the other hand, we fit SNTF with an arbitrarily high value of $\alpha = 1e6$, so that the model is concerned almost exclusively with separating the classes instead of fitting the data. SNTF keys in on narrow spectral features, such as the absorption region of “hay-windrowed” around band 24, that are distinct between the classes. This yields a factor that neatly separates the classes. The SNTF learned factor and class distributions are shown in Figure 7. This trivial example highlights a unique application of SNTF compared to unsupervised unmixing techniques: it can be used as means for structure discovery. In more complicated datasets, such as those collected under a multitude of illumination or atmosphere conditions, SNTF can be applied to reveal the spectrally distinct structure between classes and across scenarios.

4.2 Classification

Classification performance is evaluated with 25% of data used for training, resampled over 10 trials. Parameters (including rank) for all models were selected via 10 repeated trials, with 25% of each class randomly selected for the training subset in each trial. Parameters that gave the best average accuracy averaged over the 10 trials were selected for comparison. Average accuracy is used as the evaluation metric due to the class imbalance present in Indian Pines. We consider the following spectral-only models:

- **PCA + SVM**: Principal component analysis (PCA) with a radial basis function (RBF) support vector machine (SVM) classifier.
- **NMF + SVM**: Non-negative matrix factorization with an RBF SVM classifier.
- **LDA**: Linear Discriminant Analysis.

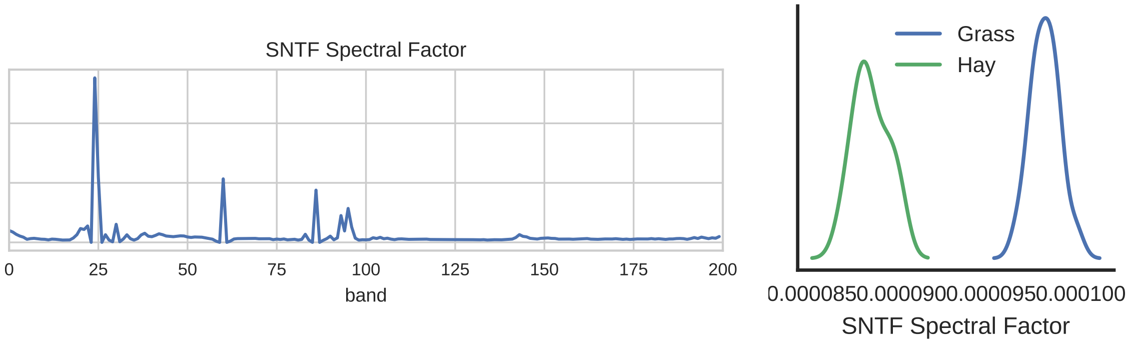

Figure 7: The SNTF spectral factor learned for discriminating between "grass-pasture-mowed" and "hay-windrowed", and the distribution of these classes when this factor is applied. SNTF cleanly separates the two classes and emphasizes narrow spectral features (such as the Hay absorption feature around band 24) that are distinct between the classes.

- **pNTF**: NTF with smoothing and decorrelating regularization terms and a Gaussian mixture model classifier from Eq. (14).
- **SNTF**: Supervised Non-Negative Tensor Factorization.

There are a number of ways to include spatial context information into SNTF, such as local neighborhoods or EMAPs.[6] We omit a spatial-spectral version of SNTF from analysis to directly compare the discriminative power of SNTF against competing linear unmixing models. However, since SNTF factorizes tensors, it can readily extend to additional modes. Considering these modes in additional to spectral is well known in the literature to increase performance.[6,16]

Average accuracies for the Indian Pines dataset are shown in Figure 8. PCA + SVM, pNTF, and SNTF are the top performing models, with comparable accuracies. However, SNTF and pNTF achieve these accuracy levels with factors constrained to be non-negative and a simple linear classifier. The tensor factorization models outperform the other non-negative model, NMF + SVM. Perhaps more interesting is that the added supervisory penalty in SNTF does provide a statistically significant increase in accuracy over the regularized NTF model. This analysis was performed with model parameters including rank that yielded the best overall performance. However, as is explored in Section 4.3, SNTF provides the greatest benefit when the number of spectral features (rank) is limited.

4.3 Practical Considerations

SNTF seeks to perform realistic spectral unmixing while also performing classification. Because it balances fitting the data with separating the classes, it is of greatest benefit when the number of mixing components is limited. The supervisory information biases the learned spectral features to encode additional discriminative information. As the number of spectral features increases, this discriminatory information is captured as part of the inherent data representation. This is illustrated in Figure 9, which shows the classification performance of pNTF and SNTF for a grid of different factorization ranks.

SNTF has four different tuning parameters: rank (number of mixing components), α (weight of Fisher criterion), α_{sm} (weight of smoothing penalty), and α_{cr} (weight of correlation penalty). The rank is often dependent upon the specific application. For instance, if the goal is to design bands for a multi-spectral sensor the rank is limited to the number of bands in the sensor. Likewise, if the goal is compression then the rank is selected to achieve the desired level of compression. This still leaves three independent tuning parameters. Tuning over the full grid of parameters can prove computationally intractable for large datasets. We propose the following scheme for parameter selection:

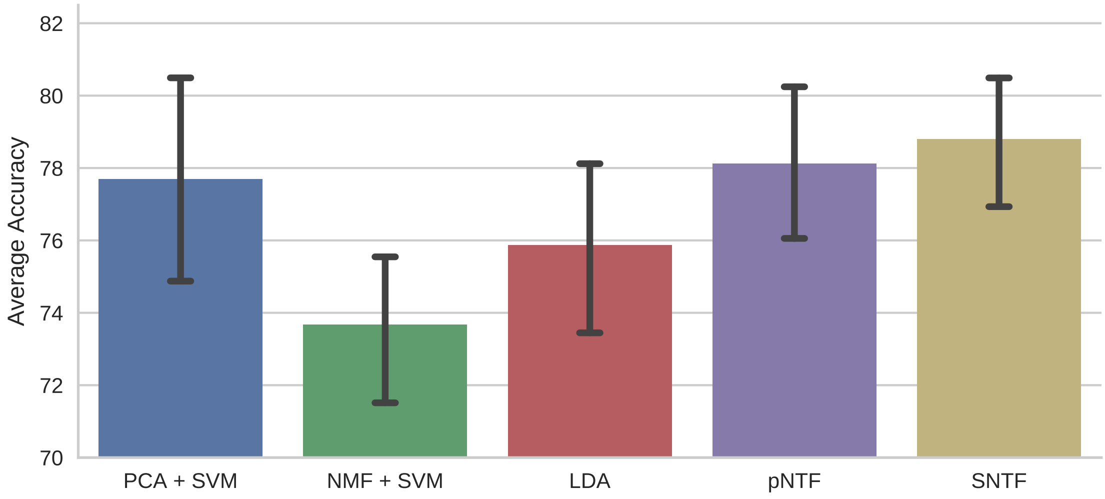

Figure 8: Average classification accuracies for Indian Pines dataset using only spectral information. pNTF and SNTF achieve state-of-the-art performance while also constraining factors to be non-negative.

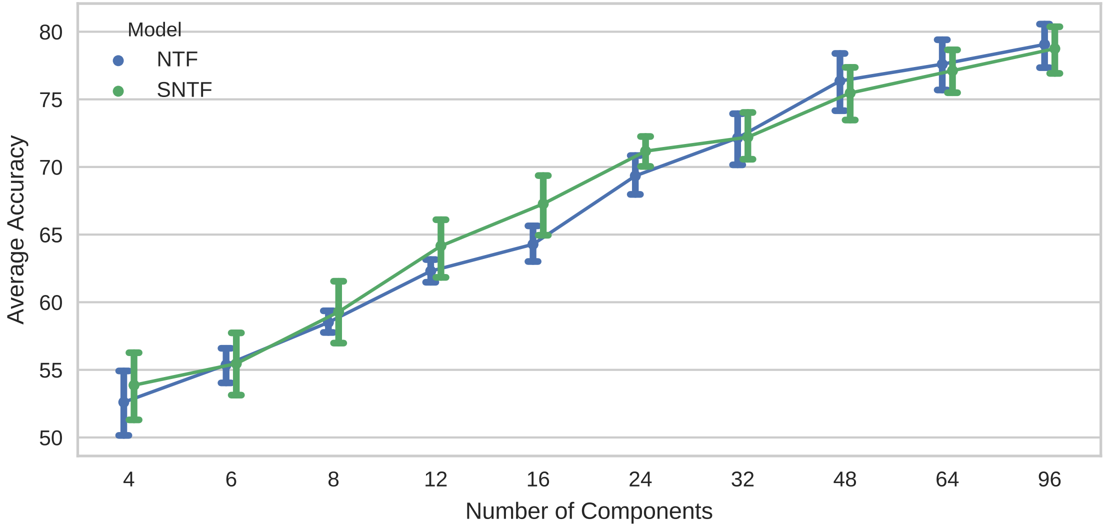

Figure 9: Average Accuracy on Indian Pines dataset as a function of number of mixture components. SNTF provides greatest benefit when the number of components is low. SNTF results are slightly dodged for ease of viewing.

1. Fit an NTF model to the data over a grid of α_{cr} and α_{sm}. Choose the largest values that give within one standard error of the best classification performance.

2. Fit the SNTF model using the previously selected α_{cr} and α_{sm} over a grid of α. Choose the best value.

In our experiments, we have found it important to set α_{cr} and α_{sm} to larger values than the optimum, since SNTF has a tendency to overfit and requires additional regularization. We have also found that SNTF converges significantly faster for greater regularization values.

The SNTF algorithm does not proscribe a fixed initialization for each of the tensor factors, just that they must be non-negative. In practice we have found the Higher-Order Support Vector Decomposition[9] thresholded to be non-negative as an effective form of initialization. Other common techniques for initialization include randomized or NMF based. Prior to applying SNTF, we energy-normalize each sample (integrate over all values in the sample and divide). This preserves spectral shapes but improves invariance to illumination, calibration error, etc. SNTF is not limited to reflectance based data.

5. CONCLUSION

This paper proposes the supervised non-negative tensor factorization, an extension to the NTF model to encode supervisory information into the learned spectral factors. Joint learning of the hyperspectral tensor and supervisory information yields spectral factors that are better suited for downstream classification tasks, boosting performance. The SNTF model includes additional regularization parameters to bias towards physically meaningful hyperspectral features. Furthermore, the unique discriminatory subspace utilized by SNTF allows it to be used a sensor design tool for selecting multi-spectral bands tuned to classification tasks. The multiplicative update rule based on KL-divergence for NTF is modified to include Fisher's discriminant criterion and regularization terms. Experimental results on standard hyperspectral image classification datasets demonstrate interpretable spectral feature extraction and state-of-the-art classification performance. Since the core of SNTF is a tensor factorization, the algorithm readily extends to spatial information, such as local neighborhoods or EMAPs.[6] This allows SNTF to adapt to more advanced collections in the future, such as hyperspectral imagery over a single location with a high revisit rate, or spectral-polarimetric imagers.

REFERENCES

[1] Mulla, D. J., "Twenty five years of remote sensing in precision agriculture: Key advances and remaining knowledge gaps," *Biosystems Engineering* **114**(4), 358 – 371 (2013). Special Issue: Sensing Technologies for Sustainable Agriculture.

[2] Feng, Y.-Z. and Sun, D.-W., "Application of hyperspectral imaging in food safety inspection and control: A review," *Critical Reviews in Food Science and Nutrition* **52**(11), 1039–1058 (2012). PMID: 22823350.

[3] Chang, C.-I. and Du, Q., "Estimation of number of spectrally distinct signal sources in hyperspectral imagery," *IEEE Transactions on Geoscience and Remote Sensing* **42**, 608–619 (March 2004).

[4] Jha, M. N., Levy, J., and Gao, Y., "Advances in remote sensing for oil spill disaster management: State-of-the-art sensors technology for oil spill surveillance," *Sensors* **8**(1), 236–255 (2008).

[5] Hagen, N. and Kudenov, M. W., "Review of snapshot spectral imaging technologies," *Optical Engineering* **52**(9), 090901–090901 (2013).

[6] Fan, L. and Messinger, D. W., "Tensor subspace analysis for spatial-spectral classification of hyperspectral data," (2016).

[7] Kolda, T. G. and Bader, B. W., "Tensor decompositions and applications," *SIAM review* **51**(3), 455–500 (2009).

[8] Mrup, M., "Applications of tensor (multiway array) factorizations and decompositions in data mining," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* **1**(1), 24–40 (2011).

[9] Cichocki, A., Zdunek, R., Phan, A. H., and Amari, S.-i., [*Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*], John Wiley & Sons (2009).

[10] Lee, D. D. and Seung, H. S., "Algorithms for non-negative matrix factorization," in [*Advances in Neural Information Processing Systems 13*], Leen, T. K., Dietterich, T. G., and Tresp, V., eds., 556–562, MIT Press (2001).

[11] Kotsia, I., Zafeiriou, S., and Pitas, I., "A novel discriminant non-negative matrix factorization algorithm with applications to facial image characterization problems," *IEEE Transactions on Information Forensics and Security* **2**, 588–595 (Sept 2007).

[12] Zafeiriou, S., "Discriminant nonnegative tensor factorization algorithms," *IEEE Transactions on Neural Networks* **20**(2), 217–235 (2009).

[13] Guan, N., Zhang, X., Luo, Z., Tao, D., and Yang, X., "Discriminant projective non-negative matrix factorization," *PLOS ONE* **8**, 1–12 (12 2013).

[14] Wu, F., Tan, X., Yang, Y., Tao, D., Tang, S., and Zhuang, Y., "Supervised nonnegative tensor factorization with maximum-margin constraint," in [*Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*], 962–968, AAAI Press (2013).

[15] Hastie, T. and Tibshirani, R., "Discriminant analysis by gaussian mixtures," *Journal of the Royal Statistical Society. Series B (Methodological)* **58**(1), 155–176 (1996).

[16] Chen, Y., Zhao, X., and Jia, X., "Spectral-spatial classification of hyperspectral data based on deep belief network," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **8**, 2381–2392 (June 2015).