A TENSOR DECOMPOSITION METHOD FOR UNSUPERVISED FEATURE LEARNING ON SATELLITE IMAGERY

Golnoosh Dehghanpoor, Michael Frachetti, Brendan Juba

Washington University in St. Louis
g.dehghanpoor@wustl.edu, frachetti@wustl.edu, bjuba@wustl.edu

ABSTRACT

We introduce a tensor factorization approach to unsupervised feature learning of hyper-spectral imagery, and demonstrate its effectiveness on land type classification of publicly available datasets. The results show that this approach can produce state of the art accuracy, compared to other methods for feature learning in the classification task.

Index Terms— Tensor decomposition, feature learning, hyper-spectral imagery

1. INTRODUCTION

Hyper-spectral images, i.e. images with a high density of samples in the electromagnetic spectrum, are powerful devices for identifying ground objects according to their unique spectral signature. These images are used in a variety of applications including environmental sciences [1] and land-cover mapping. But as the number of spectral dimensions is in the range of hundreds, a feature learning step is needed to make processing them tractable. To this end, many algorithms have been proposed. At a high level, these algorithms can be classified as supervised and unsupervised feature learning methods. In the supervised approach, labeled samples are used to devise features that can then distinguish the classes. However given that hand-labeling hyper-spectral data is a laborious task, we will focus on Unsupervised Feature Learning (UFL) methods, which don't require prior knowledge of the structures present in images or training data. In the unsupervised approach only the raw data is given to the algorithm and from that alone the algorithm extracts features that represent data well. The UFL approaches include PCA based approaches [2],[3], various discriminant analysis based approaches [4], manifold learning based approaches [5], [6] and auto encoder based approaches [7],[8]. Existing approaches use local optimization methods which can get stuck in local minima. In this paper we introduce a tensor decmposition framework for formulating feature learning of hyper-spectral imagery.

Tensor-decomposition methods have been employed outside the scope of satellite image processing. Frequently, the use of tensor representations focuses on de-noising and compression, rather than feature learning for unsupervised classification (i.e., clustering). [9] used non-negative tensor decomposition for dimensionality reduction of images, and demonstrated its use for unsupervised face recognition. [10] used the tensor decomposition method for obtaining image descriptors. [11] introduced the TenSR Framework for sparse representation of multidimensional signals, based on Tucker factorization, which has the added degree of freedom of a "core" tensor transforming the components.

Overview of our contributions: In this paper we propose a novel tensor-decomposition based formulation for the task of UFL. To the best of our knowledge, this is the first tensor-decomposition formulation of unsupervised feature learning for satellite imagery classification. Inspired by the dictionary learning approach in this setting [12], we use an overcomplete factorization in which we encourage sparsity with respect to the patch representations. This has been used previously in the context of Tucker decompositions [13] but not, to our knowledge, in the context of decompositions into linear combinations of rank-1 tensors ("CP decomposition"). We compare the performance of our method with a state-of-the-art method using a 3d auto-encoder [8], and show that our new method achieves competitive performance. To evaluate robustness, we also consider cross-training, where we train features on one dataset and test it on another. We show that given the training dataset is rich enough, the results are satisfactory and compare it with another approach[14].

2. PRELIMINARIES AND NOTATION

Here we introduce our notation for tensors and their decomposition. Tensors or multi-way arrays are functions of three or more indices $(i, j, k, \ldots)$. They are the generalization of matrices. One difference between tensors and matrices is in their decomposition. Matrix decomposition doesn't produce unique results unless it has strict constraints such as orthogonality, but tensor decomposition is unique under mild constraints. [15] provides a thorough review of tensors and their decomposition.

We use the following notation. The size of a matrix with n rows and m columns is denoted by (n, m), and $A(:, m)$ is the mth column of A. Vectorization of a matrix along the columns is the process of concatenating all its columns into a long vector, and similarly for row vectorization.

Here we work with three-way tensors, and (I, J, K) is the size of the tensor T along each dimension. The Canonical Polyadic Decomposition (CPD) of T with c components extracts from it three matrices with sizes (I, c), (J, c), (K, c), each of which are the concatenation of c vectors of appropriate size and satisfy the following equation:

$$T(i, j, :) = \sum_c I(i, c)J(j, c)K(:, c)$$

3. TENSOR DECOMPOSITION FEATURE LEARNING

We now describe our new method for learning features using tensor decomposition.

1) Create tensor: The input to the algorithm is a set of randomly sampled hyper-spectral patches with a user-defined size. We define the variables p as the size of the dimension of each patch, b as the number of spectral bands, and s as the total number of patches. Each band of the patch is vectorized to get vectors of size p^2. (We flatten

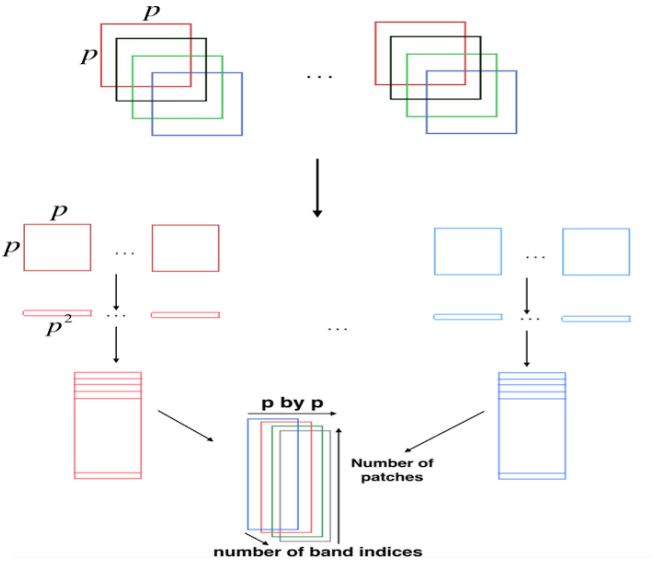

Fig. 1. The process of building a tensor out of randomly sampled multi-spectral patches. Colors represent the different spectral bands. Patches of each band are flattened and concatenated into a tensor.

the patch as we do not wish to factorize the spatial dimensions—we would like to be able to capture arbitrary intensity patterns in a patch with our filters.) Then a matrix of size (p^2, s) is created by concatenating all the vectors as rows of the matrix. The tensor is then constructed by adding each band as the third index. These processes are shown in Figure 1.

2) Factorize tensor: We now have a (p^2, s, b) tensor. To interpret the results of a tensor factorization with c components, consider the element $(i, j, :)$, which is the vectorization of all spectral bands of the ith pixel in the jth sampled patch. The following equation holds between the original tensor and its factorization, where T is the original tensor and T_p, T_b and T_s are the factors along each original dimension:

$$T(i, j, :) = \sum_c T_p(i, c)T_s(j, c)T_b(:, c)$$

From the above equation, we can see that the elements of $T_b(:, c)$ can be interpreted as a decomposition of the original spectrum of bands (in T) into c indices for each spectrum (can be thought of as learning spectral indices present in the image). With this insight, we perform non-negative CPD factorization on this tensor with c factors and minimizing the l_1-norm along all dimensions. This style of decomposition with sparse factors was considered previously for Tucker decompositions by [13]. We note that although the CPD is unique for the mimimum number of components c, in general we use a non-minimal value of c. We use the sparsity-encouraging l_1-minimization constraint since we want to encourage the different classes to be represented by distinct spectral indices, and the sparsity constraint discourages the construction of common patterns that can be used across classes. In particular, when we extract features using these patterns, the decompositions into patterns will be unique if they are sufficiently sparse, in spite of the possible overcompleteness with respect to this dimension.

3) Compute new features: After the CPD factorization, we take the c components along the band dimension (b) as spectral indices. Now we define a system of equations that takes a new image patch and decomposes it. We define a $(p^2 b, c)$ dictionary using the tensor decomposition. Take C_i to be the ith factor of T_b. We construct p^2 patterns from this one factor, according to the following formula:

$$C_{i,j} = C_i T_p(j, i) \quad j = 1, \ldots, p^2$$

Now we have p^2 texture patterns, corresponding the neighboring pixel's spectral indices. With this procedure we obtain p^2 vectors each of size b, which we then concatenate into a $p^2 b$ dimension column vector. Concatenating all these column vectors in a matrix, gives us a $(p^2 b, c)$-size dictionary matrix. Denote this dictionary by D. Given a new hyper-spectral patch, we vectorize the patch and concatenate these vectors to form a new $p^2 b$ vector V_b, analogous to our transformation of our filters earlier. Now we have the system of constraints:

$$Dx = V_b \quad \text{s.t.} \quad x \geq 0$$

where x is a c by 1 column vector. This is a linear program, which can be solved by standard methods. In particular, if the solution is sufficiently sparse, it is known that it is unique [16]. Solving this system gives the decomposition of the new hyper-spectral patch over the texture patterns. We drop all but the top n entries of x to obtain the values of our n new features for the patch. The extended feature space as described above is then used in a standard classification method. In our experiments, we used the SVM for supervised classification.

4. EXPERIMENTS

To evaluate the method, we compared its performance against a state of the art method ([8]), which uses 3D convolutional auto encoders for feature learning on three publicly available hyper-spectral datasets. The Indian pines and Salinas datasets acquired by the AVIRIS sensor, the Pavia University and Pavia center datasets acquired by the ROSIS sensor were used. For all the experiments, the patch size was set to 8, 75 components were extracted from the tensor and all were used in the dictionary. 2000 randomly sampled patches were used to construct the tensor. The sensitivity of Average Accuracy(AA) and Overall Accuracy(OA) to component number, patch size and number of random samples for tensor construction are shown in figures 2, 3 and 4. The classification was done with SVM using 10-fold cross-validation. 10 percent of the data was used for training and the rest for testing. (MATLAB's implementation of SVM was used). To evaluate the robustness of the learned features, we also used the learned dictionary from one dataset to learn a classifier on a different dataset. These experiments were only conducted on datasets that were acquired by the same sensor and in order to match the spectral signature of different datasets, only the common bands were used and the rest were discarded.

4.1. Results

Tables 1 to 3 show the result of TD versus the 3D convolutional auto-encoder method on the datasets. Table 1 shows the regular experiment settings on Indian pines, Pavia university, Pavia center and Salinas datasets. Tables 2 and 3 show the results of cross dictionary usage in testing for datasets.

The results show that the tensor decomposition method provides more accurate results in terms of general and overall accuracy. However when comparing the class-wise accuracies of the Indian pines data, we can see that in two classes its accuracy is significantly lower. This can be explained by the number of pixels belonging to some classes (e.g class 9) compared to the others. As in this method

Class	3DCAE 1	TD 1	3DCAE 2	TD 2	3DCAE 3	TD 3
1	90.48	86.04	100.00	99.83	95.21	98.33
2	92.49	96.04	99.29	99.91	96.06	99.98
3	90.37	95.06	97.13	100	91.32	85.18
4	86.90	91.24	97.91	99.04	98.28	95.63
5	94.25	91.99	98.26	99.17	95.55	99.42
6	97.07	97.87	99.98	99.61	95.30	99.60
7	91.26	100	99.64	99.38	95.14	92.26
8	97.79	99.76	91.58	94.05	91.38	93.27
9	75.91	66.66	99.28	99.96	99.96	93.38
10	87.34	94.52	96.65	98.75		
11	90.24	98.25	97.74	99.69		
12	95.76	92.92	98.84	99.94		
13	97.49	100	99.26	99.51		
14	96.03	98.77	97.49	98.35		
15	90.48	90.62	87.85	84.44		
16	98.82	96.42	98.34	99.38		
AA(%)	92.04	93.69±1.68	97.45	98.94±0.02	95.36	95.90±0.11
OA(%)	92.35	96.51±0.31	95.81	97.92±0.10	95.39	97.87±0.19

Table 1. Overall and average accuracy and class accuracies of one run and mean var for 1: Indian pines, 2: Salinas Valley, 3: Pavia University.

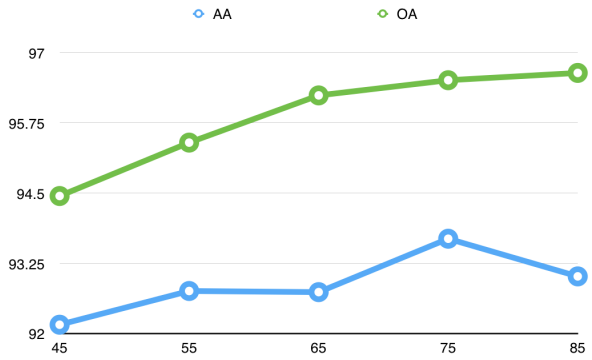

Fig. 2. Sensitivity of AA and OA to component number change in Indian pines dataset

Class	Ind on Sal	Sal on Ind
AA(%)	61.83±6.65	97.31±0.02
OA(%)	69.85±0.49	94.83±0.01

Table 2. Overall and average accuracy and class accuracies of one run for Indian pines trained on Salinas and vice versa.

Class	SCAE-Hyperion	TD
AA(%)	96.56±0.51	94.48±0.09
OA(%)	95.84±0.94	96.86±0.03

Table 3. Overall and average accuracy and class accuracies of one run for Pavia U trained on Pavia center compared with [14]

one large tensor is formed for extracting the dictionary from all the classes simultaneously, smaller classes, whose corresponding portions of the image can't be described well from the elements of larger classes, will suffer lower performance. This need not be the case for every small class, as we see that class number 7 also has few pixels

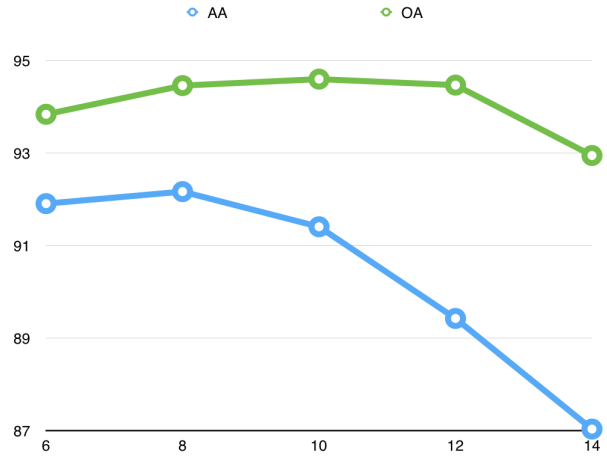

Fig. 3. Sensitivity of AA and OA to patch size change in Indian pines dataset

but the proposed method has a high accuracy for that class, indicating that its spectral-spatial signature can be reasonably reconstructed from the features used for larger classes. A future direction to remedy this problem could be constructing separate tensors for different classes, which would then be a supervised learning variant of this method.

5. CONCLUSION AND DISCUSSION

We proposed a novel tensor decomposition method for unsupervised feature learning. We compared our method against a state of the art method for UFL on publicly available datasets, where it was shown that it achieved competetive accuracy. The drawback of the method is that if the dataset contains classes that have few datapoints but vary widely in terms of their spectral and spatial signatures, their

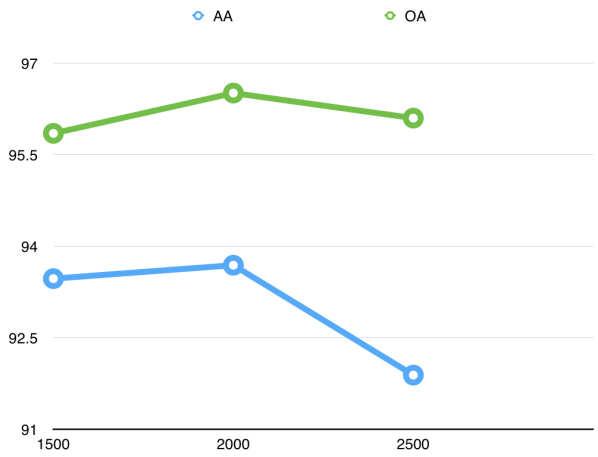

Fig. 4. Sensitivity of AA and OA to number of randomly sampled patches in Indian pines dataset

class accuracy could be low while the overall and average accuracies remain competetive or even higher than the other method. One way to remedy this would be to construct seperate tensors for various classes, which would be a supervised feature learning approach.

6. ACKNOWLEDGMENTS

We thank the anonymous reviewers for their valuable feedback. This work was supported by NSF award CCF-1718380.

7. REFERENCES

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

[2] A. Plaza, P. Martinez, J. Plaza, and R. Perez, "Dimensionality reduction and classification of hyperspectral image data using sequences of extended morphological transformations," *IEEE Trans. Geosci. Remote Sens*, vol. 43, no. 3, pp. 466–479, 2005.

[3] Y. Ren, L. Liao, S. J. Maybank, Y. Zhang, and X. Liu, "Hyperspectral image spectral-spatial feature extraction via tensor principal component analysis," *IEEE Geosci. Remote Sens. Lett*, vol. 14, no. 95, pp. 1431–1435, 2017.

[4] Q. Du, "Modified fishers linear discriminant analysis for hyperspectral imagery," *IEEE Geosci. Remote Sens. Lett*, vol. 4, no. 4, pp. 503–507, 2007.

[5] L. Ma, M. M. Crawford, and J. Tian, "Local manifold learning-based k-nearest-neighbor for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens*, vol. 48, no. 11, pp. 4099–4109, 2010.

[6] G. Chen and S.-E. Qian, "Dimensionality reduction of hyperspectral imagery using improved locally linear embedding," *Journal of Applied Remote Sensing*, vol. 1, no. 1, pp. 1–10, 2007.

[7] C. Tao, H. Pan, Y. Li, and Z. Zou, "Unsupervised spectralspatial feature learning with stacked sparse autoencoder for hyperspectral imagery classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 12, pp. 2438–2442, 2015.

[8] Y. Geng Z. Zhang X. Li S. Mei, J. Ji and Q. Du, "Unsupervised spatialspectral feature learning by 3d convolutional autoencoder for hyperspectral classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 9, pp. 6808 – 6820, 2019.

[9] Amnon Shashua and Anat Levin, "Linear image coding for regression and classification using the tensor-rank principle," in *CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*. IEEE, 2001, vol. 1, pp. I–I.

[10] Muhammad Ali and Hassan Foroosh, "Character recognition in natural scene images using rank-1 tensor decomposition.," *Proceedings of the IEEE International Conference on Image Processing*, pp. 2891–2895, 2016.

[11] Na Qi, Yunhui Shi, Xiaoyan Sun, and Baocai Yin, "Tensr: Multi-dimensional tensor sparse representation," *Computer Vision and Pattern recognition*, pp. 5916–5925, 2016.

[12] G. Sheng, W. Yang, T. Xu, and H. Sun, "High-resolution satellite scene classification using a sparse coding based multiple feature combination," *Journal of Remote Sensing*, vol. 33, no. 8, pp. 2395–2412, 2012.

[13] Morten Mørup, Lars Kai Hansen, and Sidse M Arnfred, "Algorithms for sparse nonnegative tucker decompositions," *Neural computation*, vol. 20, no. 8, pp. 2112–2131, 2008.

[14] R. Kemker and C. Kanan, "Self-taught feature learning for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 5, pp. 2693–2705, 2017.

[15] N. Sidiropoulos, L. De Lathauwer, F. Xia, K. Huang, E. Papalexakis, and C. Faloutsos, "Tensor decomposition for signal processing and machine learning," *IEEE TRANSACTIONS ON SIGNAL PROCESSING*, vol. 65, no. 13, pp. 2108–2123, 2017.

[16] Alfred M Bruckstein, Michael Elad, and Michael Zibulevsky, "On the uniqueness of nonnegative sparse solutions to underdetermined systems of equations," *IEEE Transactions on Information Theory*, vol. 54, no. 11, pp. 4813–4820, 2008.