Tensor Dictionary Learning with representation quantization for Remote Sensing Observation Compression

Anastasia Aidini[*,†], Grigorios Tsagkatakis[†], and Panagiotis Tsakalides[*,†]

[*]Computer Science Department
University of Crete
Voutes, Crete, Greece
aidini,tsakalid@csd.uoc.gr

[†]Institute of Computer Science
FORTH
Voutes, Crete, Greece
greg@ics.forth.gr

Abstract

Nowadays, multidimensional data structures, known as tensors, are widely used in many applications like earth observation from remote sensing image sequences. However, the increasing spatial, spectral and temporal resolution of the acquired images, introduces considerable challenges in terms of data storage and transfer, making critical the necessity of an efficient compression system for high dimensional data. In this paper, we propose a tensor-based compression algorithm that retains the structure of the data and achieves a high compression ratio. Specifically, our method learns a dictionary of specially structured tensors using the Alternating Direction Method of Multipliers, as well as a symbol encoding dictionary. During run-time, a quantized and encoded sparse vector of coefficients is transmitted, instead of the whole multidimensional signal. Experimental results on real satellite image sequences demonstrate the efficacy of our method compared to a state-of-the-art compression method.

Introduction

In the last decades, remotely sensed images have been widely used in various Earth Observation related applications, including forest monitoring, disaster evaluation, and land cover estimation among others. From an imaging system point of view however, the increasing spatial, spectral and temporal resolutions of the acquired images lead to new challenges due to the moderate increase in space-to-ground transmission bandwidth. As a result, there is an urgent need for efficient compression algorithms capable of reducing the storage and transmission requirements to ground-based stations.

High dimensional observations like time series of images from different modalities represent a class of observations that although are described by multiple dimensions, they are typically characterized by a significantly smaller number of degrees of freedom. This is due to the fact that image time-series from different modalities usually exhibit high redundancies across space, time and modality. As a consequence, data compression can significantly reduce the size of multi-modal image time series without significant loss in quality, exploiting the inherent structure of the data.

From a data structure point of view, image sequences collected from airborne and spaceborne sensors, are naturally encoded by means of high dimensional data structures, known as tensors [1, 2].Tensors are multidimensional arrays that provide a concrete way to represent multimedia data, with entries indexed by several variables. For instance, a time series of images encoding information across multiple spectral bands can be modeled as a 4D object with two spatial, one spectral and one temporal variable.

In this paper, we propose an efficient compression algorithm that uses a novel tensor dictionary learning method based on the CANDECOMP/PARAFAC (CP) decomposition [3] of the tensor data. Specifically, the proposed method involves a training process, in which a dictionary of specially structured tensors is estimated using an Alternating Direction Method of Multipliers (ADMM) [4] approach. Furthermore, a symbol encoding dictionary is also learned from training samples. Given the learned models, a new instance is first presented by a set of sparse coefficients corresponding to linear combination of the elements of the learned dictionary. To further increase the compression rate, the derived coefficients are subsequently quantized and encoded in order to be transmitted, significantly reducing the number of bits required to represent the useful information of the data.

The key novelties of this work are: (i) We proposed an end-to-end compression algorithm, that includes both quantization and encoding of high-dimensional data, making it directly applicable in real-world applications; (ii) We present a novel tensor dictionary learning method for the compression of high dimensional data, by learning a tensor dictionary that is used to represent every new sample given a set of previous samples; (iii) We report the performance of the proposed method on 3D remote sensing observations.

Related Work

To efficiently compress high dimensional data, the correlations among all variables must be simultaneously removed. Typical compression algorithms for multispectral images apply wavelet transform [5, 6] or Principal Components Analysis (PCA) [7] for introducing spectral decorrelation, followed by JPEG2000 [8], which is among the best-performing algorithms for 2D still image compression. Although valid, this approach is not capable of naturally exploiting spatio-spectral correlations, while it cannot handle the temporal aspect of time-series.

To address these challenges, a limited number of tensor-based approaches for the compression of multidimensional data have been presented that are able to retain the structure of the data. Specifically, a nonnegative CP decomposition algorithm has been proposed in [9] for compression of hyperspectral image time series where the input data is represented by a few arrays with reduced dimensions. A compression algorithm based on the CP decomposition is introduced in [10], in which a data tensor is represented by a small number of rank-1 tensors, while in [11] a method based on the Tucker decomposition, where the original tensor is compressed into a core tensor and a factor matrix with compressed dimensionality, is presented. However, the above models require either the transmission of all rank-1 tensors with their coefficients, or the core tensor and the factor matrices. These requirements are in stark contrast to the proposed approach that only requires the transmission of the sparse coefficients. The proposed method can be regarded as an improved version of the one described in [12], where in this case a tensor dictionary learning algorithm is introduced for learning the rank-1 tensors used in the representation of new samples.

Although multidimensional dictionary learning algorithms [13] have been successfully employed for dynamic tomographic reconstruction [14, 15], video completion [16] and multispectral image denoising [16, 17], they have not been considered for compression. Specifically, an extension of the K-SVD [18] algorithm, called as K-CPD, is proposed in [19], using CP decomposition with a tensor dictionary that consists of rank-1 tensors and a Multilinear Orthogonal Matching Pursuit (MOMP) algorithm for the calculation of the sparse representation of the tensor signal. Furthermore, several dictionary learning algorithms have been developed based on the Tucker model [20–23], in which a dictionary for each mode is learned and the sparseness constraint is applied to the core tensor. A low-rank regularization term in order to learn a dictionary as a sum of few Kronecker products of smaller subdictionaries is also introduced in [24].

Tensor Preliminaries

An N-way or Nth-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is defined as a multidimensional array, whereby the order of a tensor is the number of its dimensions. A fundamental property of tensors is the tensor rank, a generalization of matrix rank. Specifically, considering that a rank-1 N-th order tensor is the outer product of N vectors with elements the product of the corresponding vector elements, the rank of a tensor is defined as the minimum number of rank-1 tensors needed to produce the original tensor. Unfortunately, there is no straightforward algorithm to determine the rank of a given tensor; in fact, the problem is NP-hard [25]. However, in most applications, one is really interested in fitting a model that has the essential or meaningful number of components, which is much less than the actual rank of the tensor that we observe, due to noise and sensor imperfections [2].

The CANDECOMP/PARAFAC (CP) decomposition represents an Nth-order tensor $\mathcal{X}$ as a linear combination of rank-1 tensors, i.e.,

$$\mathcal{X} = \sum_{r=1}^{R} \lambda_r \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \cdots \mathbf{a}_r^{(N)} = [\![\boldsymbol{\lambda}; \mathbf{A}^{(1)}, ..., \mathbf{A}^{(N)}]\!], \tag{1}$$

where R is a positive integer, $\mathbf{A}^{(n)} = [\mathbf{a}_1^{(n)}\ \mathbf{a}_2^{(n)}\ \cdots\ \mathbf{a}_R^{(n)}]$ are the factor matrices, $\boldsymbol{\lambda} \in \mathbb{R}^R$ and $\mathbf{a}_r^{(n)} \in \mathbb{R}^{I_n}$, for $r = 1, ..., R$ and $n = 1, ..., N$.

A common framework for tensor computations is to turn the tensor into a matrix, which is called matricization or unfolding of the tensor [26] and allows the use of algorithms designed on matrices. Specifically, the mode-n matricization of $\mathcal{X}$ is denoted as $\text{unfold}_n(\mathcal{X}) = X_{(n)} \in \mathbb{R}^{I_n \times \prod_{i \neq n} I_i}$ and corresponds to a matrix with columns being the vectors obtained by fixing all indices of $\mathcal{X}$ except the n-th index. However, the structure of the data is not preserved in this way and the high dimensional relationships, e.g., across neighbouring pixels or time instances, are lost.

Proposed Compression Method

The proposed compression method is composed of two parts, the training and the runtime phase. During training, available data is utilized for developing the appropriate tensor representation model which is subsequently utilized for compression of new samples. [1]

Training process

Given a set of training samples, each of them being an N-th order tensor $\mathcal{X}^j \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, we can produce an $(N+1)$-th order training tensor $\mathcal{X} = (\mathcal{X}^1, \mathcal{X}^2, ..., \mathcal{X}^T) \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N \times T}$, where T is the number of training samples.

To compress a multidimensional sample, we learn a tensor dictionary $\mathcal{D} \in \mathbb{R}^{I_1 \times \cdots \times I_N \times K}$ that consists of K rank-1 tensors $\mathcal{D}^{(k)} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$, $k = 1, .., K$, by solving the following optimization problem

$$\begin{aligned} &\min_{\mathcal{D}, \mathbf{A}} \tfrac{1}{2} \|\mathcal{X} - \mathcal{D} \times_{N+1} \mathbf{A}\|_F^2 \\ &\text{subject to } \|\mathbf{A}(t,:)\|_0 \leq \lambda, \ \forall t = 1, .., T \end{aligned}, \tag{2}$$

where the constraint $\|\mathbf{A}(t,:)\|_0$ promotes sparsity on the coefficients of each sample by keeping a number of non-zero elements that depends on the sparsity parameter λ and zeros out others. To solve this problem, we introduce a novel tensor dictionary learning algorithm that is described below, using the Alternating Direction Method of Multipliers (ADMM).

Despite the tensor dictionary, we also obtain a symbol encoding dictionary in order to represent the compressed data in a binary format for transmission or storage. Specifically, the coding dictionary maps a binary number to a discrete set of symbols S, using fewer bits for the most common symbols. To populate the symbols of this dictionary, we split the range of values of the sparse coefficients $\mathbf{A}$ obtained from the proposed tensor dictionary learning method, into $2^b - 1$ equal partitions, where b is a given number of bits. Then, the set of symbols S is defined to be the boundaries of those partitions.

Compression and Decompression

In the subsequent compression process, each new sample $\mathcal{X}' \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ can be written as a linear combination of the atoms of the dictionary with sparse coefficients

$$\mathcal{X}' = \mathcal{D} \times_{N+1} \mathbf{a}, \tag{3}$$

where $\mathbf{a} \in \mathbb{R}^K$ with $\|\mathbf{a}\|_0 \leq \lambda$, by solving the problem in (2) using the learned dictionary. To achieve this, we apply the ADMM method described below, skipping the dictionary update. Subsequently, we quantize the obtained sparse vector of coefficients $\mathbf{a} = (a_1, .., a_K)$ to b bits. Specifically, we use a uniform quantizer $\mathcal{Q} : \mathbb{R} \to S$, assuming that the set of symbols $S = \{q_1, q_2, ..., q_{2^b}\}$ are the quantization levels and $c_s = \frac{q_s + q_{s+1}}{2}$, $s = 1, ..., 2^b - 1$ are the quantization boundaries. Then, we encode the quantized vector of coefficients $\mathbf{a}_q = [\mathcal{Q}(a_1), \mathcal{Q}(a_2), .., \mathcal{Q}(a_K)]$ in order to be transmitted, using Huffman coding and the learned encoding dictionary. Therefore, we

[1] The code of the proposed compression method is available at https://github.com/splicsforth/TensorDictionaryLearningWithRepresentationQuantization.

transmit only K numbers with a few non-zero elements, instead of the whole tensor sample $\mathcal{X}'$, achieving a high compression ratio.

In order to decompress the received data $\mathbf{a}_q$, we only need to synthesize the dictionary $\mathcal{D}$ obtained from the training process, with the decoded and quantized sparse coefficients $\mathbf{a}_q$. Therefore, the reconstructed sample $\mathcal{X}'$ will be

$$\mathcal{X}' \approx \mathcal{D} \times_{N+1} \mathbf{a}_q, \tag{4}$$

which is an efficient approximation of the test sample $\mathcal{X}'$, as will be demonstrated in the experimental results Section.

Proposed Tensor Dictionary Learning Method

Given the training samples in the tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N \times T}$, we need to learn the tensor dictionary $\mathcal{D} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N \times K}$ and the sparse coefficients $\mathbf{A} \in \mathbb{R}^{T \times K}$, by solving the minimization problem in (2), or equivalently the problem

$$\min_{\mathcal{D},\mathbf{A}} \frac{1}{2} \|\mathcal{X} - \mathcal{D} \times_{N+1} \mathbf{A}\|_F^2 + \lambda \sum_{t=1}^{T} \|\mathbf{A}(t,:)\|_0. \tag{5}$$

To simplify the solution, we introduce the auxiliary variable $\mathbf{G} \in \mathbb{R}^{T \times K}$, such that $G = A$, in order to impose the sparsity constraint on it. Therefore, we can reformulate the minimization problem in (5) as

$$\begin{gathered} \min\nolimits_{\mathcal{D},\mathbf{A},\mathbf{G}} \tfrac{1}{2} \|\mathcal{X} - \mathcal{D} \times_{N+1} \mathbf{A}\|_F^2 + \lambda \sum_{t=1}^{T} \|\mathbf{G}(t,:)\|_0 \\ \text{subject to } \mathbf{G} = \mathbf{A} \end{gathered} \tag{6}$$

and we can apply the Alternating Direction Method of Multipliers (ADMM) to solve it.

In more detail, the augmented Lagrangian function for the problem (6) is given by

$$\mathcal{L}(\mathbf{A}, \mathbf{G}, \mathcal{D}, \mathbf{Y}) = \frac{1}{2} \|\mathcal{X} - \mathcal{D} \times_{N+1} \mathbf{A}\|_F^2 + \lambda \sum_{t=1}^{T} \|\mathbf{G}(t,:)\|_0 + < \mathbf{Y}, \mathbf{G} - \mathbf{A} > + \frac{p}{2} \|\mathbf{G} - \mathbf{A}\|_F^2, \tag{7}$$

where $\mathbf{Y} \in \mathbb{R}^{T \times K}$ stands for the Lagrange multiplier matrix, while $p > 0$ denotes the step size parameter. Following the general algorithmic strategy of the ADMM scheme, we seek for the stationary point solving iteratively for each one of the variables while keeping the others fixed. As a result, we create the following sequence of update rules at each iteration l, until a maximum number of iterations is reached or the decrease in the objective function between consecutive iterations is smaller than a predefined threshold.

- For minimizing the augmented Lagrangian function with respect to the sparse coding matrix $\mathbf{A}$, we solve the individual sparse coding problem

$$\mathbf{A}^* = \text{argmin}_{\mathbf{A}} \mathcal{L}. \tag{8}$$

  Setting, $\nabla_{\mathbf{A}} \mathcal{L} = 0$, the coefficients are updated as

$$\mathbf{A} = (\mathbf{X}_{(N+1)} \cdot \mathbf{D}_{(N+1)}^T + \mathbf{Y} + p \cdot \mathbf{G}) \cdot (\mathbf{D}_{(N+1)} \cdot \mathbf{D}_{(N+1)}^T + p \cdot \mathbf{I})^{-1}, \tag{9}$$

  where $\mathbf{I}$ is the identity matrix with dimensions $K \times K$.

- Similarly, we set $\nabla_{\mathbf{G}} \mathcal{L} = 0$ to update $\mathbf{G}$ as

$$\mathbf{G} = H_\lambda(\mathbf{A} - \frac{\mathbf{Y}}{p}), \quad \text{where} \quad H_\lambda(x) = \begin{cases} x, & |x| > \lambda \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

  is the hard-thresholding operator that keeps a number of non-zero elements depending on the parameter $\lambda > 0$ and zeros out others.

- For fixed sparse coefficients $\mathbf{A}$ and $\mathbf{G}$, the tensor dictionary $\mathcal{D}$ can be obtained by solving the individual problem

$$\mathcal{D}^* = \text{argmin}_{\mathcal{D}} \mathcal{L}. \tag{11}$$

  Setting, $\nabla_{\mathcal{D}} \mathcal{L} = 0$, we update

$$\mathcal{D}^{(l)} = \mathcal{D}^{(l-1)} + \mathcal{X} \times_{N+1} \mathbf{A}^{-1} \tag{12}$$

  at iteration l and we normalize the dictionary $\mathcal{D}^{(l)}$ in order to have values in the interval $[0, 1]$.

- Finally, the Lagrangian multiplier matrix is updated as

$$\mathbf{Y}^{(l)} = \mathbf{Y}^{(l-1)} + p \cdot (\mathbf{G} - \mathbf{A}) \tag{13}$$

  at each iteration l. In our setup, we set $p = 0.6$.

Note that in our method, we are able to directly solve (5) rather than solving it with respect to each N-th order tensor sample $\mathcal{X}^j$. Thus, our sparse coding process runs much faster than the other related solutions.

Experimental Results

The efficacy of the proposed compression algorithm is evaluated over time series of satellite derived observations and more specifically time series of normalized difference vegetation index (NDVI) [27]. Unlike other types of image time series, this class of measurements is not characterized by typical "motion" but rather by slowly evolving phenomena. In our experiments, we used times series of size $200 \times 200 \times 7$, where the last dimension indicates the number of days, using 50 samples for the training process. The recovery performance is measured in terms of the Normalized Mean

Square Error (NMSE) which is defined as $\text{NMSE} = \frac{\|\mathcal{Y}-\mathcal{Y}'\|_2^2}{\|\mathcal{Y}\|_2^2}$, where $\mathcal{Y}$ and $\mathcal{Y}'$ are the original and the reconstructed signal, respectively.

An important parameter of the proposed method is the number of atoms of the dictionary learned in the training process. Figure 1 reports on the reconstruction quality for several time series and different number of atoms of the dictionary, using 80% sparsity level and 8 bits of quantization. The most robust behavior is observed for a middle value of the number of atoms. However, the improvement is negligible, as the coefficients are sparse enough in each case.

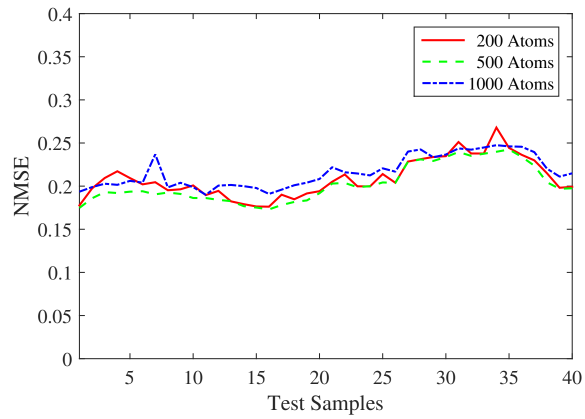

Figure 1: Reconstruction quality for each test sample and different number of atoms of the dictionary, using 80% sparsity level and 8 bits of quantization.

Another parameter that was examined is the effect of the sparsity on the vector of coefficients that are transmitted during run-time. Figure 2 presents the NMSE for different sparsity levels, using a dictionary with 500 atoms and 8 bits of quantization. Notably, the reconstruction is more efficiently for high sparsity level. In addition, higher sparsity of the coefficients leads to lower bits needed for transmission.

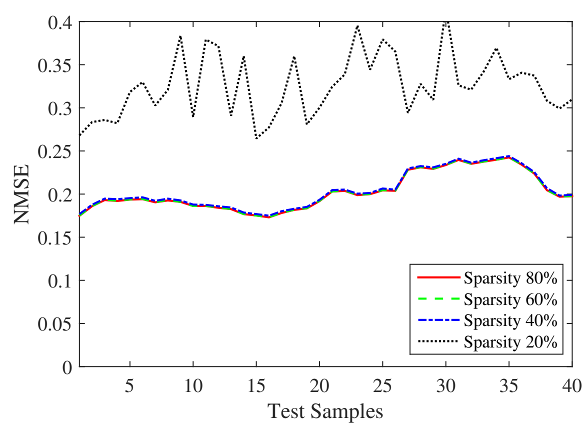

Figure 2: Reconstruction quality for each test sample and different sparsity levels, using a dictionary with 500 atoms and 8 bits of quantization.

Our study also emphasizes on the impact of the number of bits used for the quantization of the coefficient vector, when a dictionary of 500 atoms is learned using 80% sparsity level. The results are reported in Table 1 for several testing samples and number of quantization bits, indicating that more bits lead to better performance for our method. However, the reconstruction quality is decreasing over time.

Table 1: Reconstruction error for different samples as a function of quantization bit number.

Number of Bits	NMSE			
	1st Sample	10th Sample	25th Sample	35th Sample
4	0.2915	0.3226	0.3510	0.3936
6	0.1816	0.1941	0.2134	0.2510
8	0.1748	0.1860	0.2044	0.2425

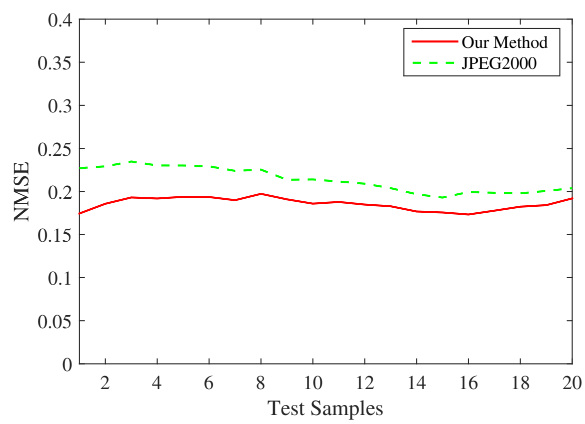

Figure 3: Reconstruction quality for several test samples, using 0.06 bpppb with our compression method and JPEG2000.

Table 2: Reconstruction quality for different number of bpppb.

NMSE	bpppb			
	0.20	0.16	0.08	0.03
Our Method	0.1754	0.1751	0.1748	0.1777
JPEG2000	0.1838	0.1929	0.2169	0.2493

Finally, we compared our algorithm with the state-of-the-art compression algorithm, namely JPEG2000, applied without any decorrelation in the third dimension. Specifically, Figure 3 presents the NMSE for several time series, using a dictionary with 800 atoms, 10% sparsity level and 0.06 bits per pixel per band (bpppb), where a band is a time instant in our case. The results indicate that our method can provide a better approximation of the original data compared to JPEG2000 using such a high compression ratio. This can also be observed from the results presented in Table 2, where the efficacy of both methods is examined on different compression rates. As it was expected, the more the bpppb, the better the performance of JPEG2000. However, our method can equally efficient reconstruct the signal, even for a higher compression ratio, using the appropriate parameters.

Conclusions

In this work, we presented a novel tensor dictionary learning method that can be used to efficiently compress high dimensional data. Our approach can be extended to arbitrary high dimensions, retaining the structure of the data. It is also directly applicable in real-world applications since it includes quantization and encoding of the compressed data. Experimental results on real satellite image time series indicate the efficient approximation of the compressed images, even for high compression ratios.

Acknowledgments

This research work was supported by the Hellenic Foundation for Research and Innovation (HFRI) and the General Secretariat for Research and Technology (GSRT), under HFRI PhD Fellowship grant no. 1509 and HFRI Faculty grant no. 1725.

References

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

[2] Nicholas D Sidiropoulos, Lieven De Lathauwer, Xiao Fu, Kejun Huang, Evangelos E Papalexakis, and Christos Faloutsos, "Tensor decomposition for signal processing and machine learning," *IEEE Trans. on Signal Processing*, vol. 65, no. 13, pp. 3551–3582, 2017.

[3] J Douglas Carroll and Jih-Jie 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.

[4] Yuling Jiao, Qinian Jin, Xiliang Lu, and Weijie Wang, "Alternating direction method of multipliers for linear inverse problems," *SIAM Journal on Numerical Analysis*, vol. 54, no. 4, pp. 2114–2137, 2016.

[5] Michael Weeks and Magdy A Bayoumi, "Three-dimensional discrete wavelet transform architectures," *IEEE Trans. on Signal Processing*, vol. 50, no. 8, pp. 2050–2063, 2002.

[6] Amar Aggoun, "Compression of 3d integral images using 3d wavelet transform," *Journal of Display Technology*, vol. 7, no. 11, pp. 586–592, 2011.

[7] Qian Du and James E Fowler, "Hyperspectral image compression using jpeg2000 and principal component analysis," *IEEE Geoscience and Remote sensing letters*, vol. 4, no. 2, pp. 201–205, 2007.

[8] Charilaos Christopoulos, Athanassios Skodras, and Touradj Ebrahimi, "The jpeg2000 still image coding system: an overview," *IEEE transactions on consumer electronics*, vol. 46, no. 4, pp. 1103–1127, 2000.

[9] Miguel Angel Veganzones, Jérémy E Cohen, Rodrigo Cabral Farias, Jocelyn Chanussot, and Pierre Comon, "Compression-based nonnegative tensor cp decomposition of hyperspectral big data," .

[10] Leyuan Fang, Nanjun He, and Hui Lin, “Cp tensor-based compression of hyperspectral images,” *JOSA A*, vol. 34, no. 2, pp. 252–258, 2017.

[11] Lefei Zhang, Liangpei Zhang, Dacheng Tao, Xin Huang, and Bo Du, “Compression of hyperspectral remote sensing images by tensor approach,” *Neurocomputing*, vol. 147, pp. 358–363, 2015.

[12] Anastasia Aidini, Grigorios Tsagkatakis, and Panagiotis Tsakalides, “Compression of high-dimensional multispectral image time series using tensor decomposition learning,” in *Proc. European Signal Processing Conference (EUSIPCO)*, 2019.

[13] Florian Roemer, Giovanni Del Galdo, and Martin Haardt, “Tensor-based algorithms for learning multidimensional separable dictionaries,” in *IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2014, pp. 3963–3967.

[14] Shengqi Tan, Yanbo Zhang, Ge Wang, Xuanqin Mou, Guohua Cao, Zhifang Wu, and Hengyong Yu, “Tensor-based dictionary learning for dynamic tomographic reconstruction,” *Physics in Medicine & Biology*, vol. 60, no. 7, pp. 2803, 2015.

[15] Yanbo Zhang, Xuanqin Mou, Ge Wang, and Hengyong Yu, “Tensor-based dictionary learning for spectral ct reconstruction,” *IEEE transactions on medical imaging*, vol. 36, no. 1, pp. 142–154, 2016.

[16] Zemin Zhang and Shuchin Aeron, “Denoising and completion of 3d data via multidimensional dictionary learning,” *arXiv preprint arXiv:1512.09227*, 2015.

[17] Yi Peng, Deyu Meng, Zongben Xu, Chenqiang Gao, Yi Yang, and Biao Zhang, “Decomposable nonlocal tensor dictionary learning for multispectral image denoising,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2949–2956.

[18] Michal Aharon, Michael Elad, and Alfred Bruckstein, “K-svd: An algorithm for designing overcomplete dictionaries for sparse representation,” *IEEE Transactions on signal processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

[19] Guifang Duan, Hongcui Wang, Zhenyu Liu, Junping Deng, and Yen-Wei Chen, “K-cpd: Learning of overcomplete dictionaries for tensor sparse coding,” in *International Conference on Pattern Recognition*. IEEE, 2012, pp. 493–496.

[20] Cesar F Caiafa and Andrzej Cichocki, “Computing sparse representations of multidimensional signals using kronecker bases,” *Neural computation*, vol. 25, no. 1, pp. 186–220, 2013.

[21] Syed Zubair and Wenwu Wang, “Tensor dictionary learning with sparse tucker decomposition,” in *Int. Conference on Digital Signal Processing*. IEEE, 2013, pp. 1–6.

[22] Mohsen Ghassemi, Zahra Shakeri, Anand D Sarwate, and Waheed U Bajwa, “Stark: Structured dictionary learning through rank-one tensor recovery,” in *Int. Workshop on Computational Advances in Multi-Sensor Adaptive Processing*. IEEE, 2017.

[23] Zahra Shakeri, Waheed U Bajwa, and Anand D Sarwate, “Minimax lower bounds on dictionary learning for tensor data,” *IEEE Transactions on Information Theory*, vol. 64, no. 4, pp. 2706–2726, 2018.

[24] Cassio Fraga Dantas, Michele N Da Costa, and Renato da Rocha Lopes, “Learning dictionaries as a sum of kronecker products,” *IEEE Signal Processing Letters*, vol. 24, no. 5, pp. 559–563, 2017.

[25] Johan Håstad, “Tensor rank is np-complete,” *Journal of Algorithms*, vol. 11, no. 4, pp. 644–654, 1990.

[26] Tamara Gibson Kolda, “Multilinear operators for higher-order decompositions.,” Tech. Rep., Sandia National Laboratories, 2006.

[27] Ross S Lunetta, Joseph F Knight, Jayantha Ediriwickrema, John G Lyon, and L Dorsey Worthy, “Land-cover change detection using multi-temporal modis ndvi data,” *Remote sensing of environment*, vol. 105, no. 2, pp. 142–154, 2006.