







# Onboard Hyperspectral Image Compression Using Compressed Sensing and Deep Learning

Saurabh Kumar<sup>(✉)</sup>, Subhasis Chaudhuri, Biplab Banerjee,  
and Feroz Ali

Indian Institute of Technology Bombay, Mumbai 400076, MH, India  
{saurabhkm,sc,bbanerjee,feroz}@iitb.ac.in  
<http://www.ee.iitb.ac.in>

**Abstract.** We propose a real-time onboard compression scheme for hyperspectral datacube which consists of a very low complexity encoder and a deep learning based parallel decoder architecture for fast decompression. The encoder creates a set of coded snapshots from a given datacube using a measurement code matrix. The decoder decompresses the coded snapshots by using a sparse recovery algorithm. We solve this sparse recovery problem using a deep neural network for fast reconstruction. We present experimental results which demonstrate that our technique performs very well in terms of quality of reconstruction and in terms of computational requirements compared to other transform based techniques with some tradeoff in PSNR. The proposed technique also enables faster inference in compressed domain, suitable for on-board requirements.

**Keywords:** Fast Hyperspectral imaging · Fast on-board data compression · Compressed sensing · Deep learning

## 1 Introduction

Hyperspectral imaging captures images of a scene at multiple closely spaced wavelengths. Since each band in a datacube is just the same scene imaged at a slightly shifted wavelength, there is a large amount of redundancy in these data cubes. Although this gives us much more information than other modalities for data analysis and inference, in many cases these tasks can also be performed even with a good approximation of the datacube at hand, justifying a need and feasibility of hyperspectral data compression. We would like to compress a hyperspectral datacube on board a remote device which might be a satellite/UAV or even on a ground station for subsequent distribution. Although compression is a computationally intensive task to do onboard, more so for drones and UAVs, if it can be done with a fast and very low complexity lossy algorithm we can save on both transmission time along with onboard power and storage requirements.

Hyperspectral sensors deployed on UAVs have become popular for remote sensing tasks lately. Telmo et al. present a detailed review of various such sensors used for Agriculture and forestry application in [1]. We consider hyperspectral datacubes for demonstration purpose in this work but we wish to emphasise the proposed compression technique can be generalized to any spectral data with multiple bands. The presented method is also applicable to videos but due to motion present between the frames, the effectiveness is comparatively reduced in this case.

There exists a significant technical literature proposing to solve this problem, many inspired from the classical compression standards including both lossless and lossy compression techniques. A review of such technique can be found in [5] which include 3D-SPECK, 3D-SPIHT, and 3D-tarp to name a few. Also, the DWT based JPEG2000 standard has been used widely for this [13].

We propose a lossy compression scheme in this paper and there have been various lossy schemes proposed in the literature. Tang et al. propose two schemes: one based on three-dimensional wavelet coding [15] and the other based on three-dimensional set partitioning [14]. Du et al. first decorrelate the data spectrally with principal component analysis and then use discrete wavelet transform [3]. Wang et al. use independent component analysis [17] and Green et al. use a concept called maximum noise transform [8] to achieve the same. These techniques are quite successful but due to their use of various transforms, they are comparatively slow and not suitable for onboard compression requirements. Apart from this, the unwanted radiometric distortions induced by them at lower bit per pixel regime impair the inference and learning tasks that are performed on the datacubes.

The main contribution of this paper is to provide a computationally very efficient compression technique for HS datacubes at a low bit per pixel regime while keeping the image features intact. The proposed method is shown to provide us with an excellent PSNR and structural similarity than the JPEG2000 at comparable bit per pixel ratios. We also present a compression quality comparison on how the standard machine learning algorithms are affected by these compression techniques and show that the proposed technique outperforms them in all comparisons. We wish to highlight that the proposed method is based on in-place computation and is also very memory efficient and therefore is better suited for onboard compression unlike existing methods.

## 2 Proposed Compression Approach

In this paper, we propose a very low complexity hyperspectral datacube compression technique. Due to this feature, the proposed technique primarily excels for onboard compression requirements. Our technique is inspired from imaging methodology proposed by Hitomi et al. [9] for increasing the frame rate of a normal off-the-shelf camera by using a liquid crystal on silicon to compressively acquire video coded snapshots. A much higher frame rate video data volume is reconstructed from these snapshots using a sparse recovery algorithm.

We use two major concepts in this work to achieve the proposed compression. Compression is done by employing coded aperture imaging on a given hyperspectral datacube. This is what gives us a very light and low complexity compression routine. Decompression from this kind of sparse acquisition of data is typically done using the sparse recovery algorithms like orthogonal matching pursuit or iterative hard thresholding. These techniques are, however, very slow. We in this paper propose a deep neural network based sparse recovery algorithm to reconstruct the datacube from the compressively coded snapshots. This does not require a dictionary learning for reconstruction and speeds up our reconstruction algorithm by many folds once the training has been done.

## 2.1 Coded Sampling

Consider a hyperspectral datacube with  $P$  spectral bands of spatial size  $M \times N$  pixels. Let  $E(x, y, z)$  denote this hyperspectral volume. We code  $T$  such consecutive and non-overlapping bands into a single coded snapshot, given by,

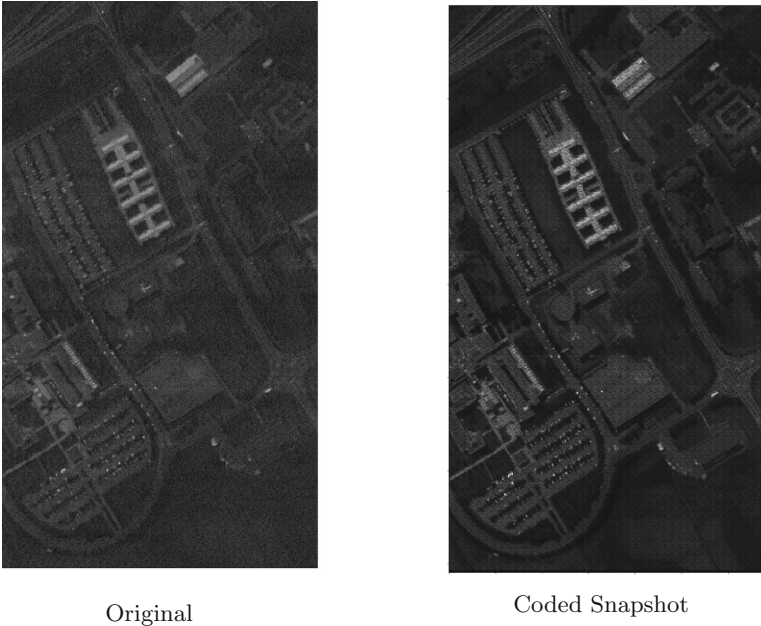
$$I(x, y) = \sum_{z=1}^T C(x, y, z)E(x, y, z), \quad (1)$$

where  $C(x, y, z)$  denotes an element of the random code matrix, generated to obtain random pixel sampling of the given hyperspectral volume. The code matrix is constructed such that the number of ones and the number of zeros in it are equal. We can thus write Eq. 1 as a matrix equation,  $\mathbf{I} = \mathbf{C}\mathbf{E}$ . Here  $\mathbf{I}$  (coded snapshot) and  $\mathbf{E}$  (hyperspectral volume) are matrices of sizes  $M \times N$  and  $M \times N \times T$ , respectively. Note that Eq. 1 allows one to perform in-place computation. The coded snapshot of  $T$  bands can be progressively transmitted as the coded bitstream. Since the random code matrix block can be generated with random generation seed, one can just transmit this seed with the coded bit stream to construct the code matrix on the receiver end. Further if onboard computation permits, the coded snapshots, thus obtained can be further encoded via JPEG2000 or an equivalent encoder. Figure 1 shows one of the coded snapshots made from 16 bands of the Pavia University datacube. Note the similarity of the coded snapshot with the original bands.

## 2.2 Sparse Reconstruction

Now given a coded snapshot we can solve a standard sparse recovery problem at the receiver to reconstruct the hyperspectral volume  $\mathbf{E}$  by solving a standard sparse recovery problem assuming we also are given a learned dictionary  $\mathbf{D}$ . In this approach, we model the hyperspectral volume  $\mathbf{E} = \mathbf{C}\mathbf{D}\alpha$ , as a sparse, linear combination of the atoms (patches) from the learned dictionary. We could then use orthogonal matching pursuit (OMP) [16] or iterative hard thresholding to solve an  $l_0$  minimization problem of the following form

$$\hat{\alpha} = \arg \min_{\alpha} \|\alpha\|_0 \text{ subject to } \|\mathbf{C}\mathbf{D}\alpha - \mathbf{I}\|_2^2 < \epsilon. \quad (2)$$



**Fig. 1.** A coded snapshot of 16 bands of Pavia University dataset along with the original band for reference.

Here  $\alpha$  is the sparse weights of a linear combination of dictionary atoms and  $\epsilon$  is the acceptable tolerance. However, solving Eq. 2 with OMP is computationally very demanding, and cannot as yet, be performed in real-time at the receiver. Hence we opt for a deep learning based solution.

### 2.3 Coded Snapshot-Deep Learning Based Compression Pipeline

A given hyperspectral datacube is divided into volume patches of size  $m \times m \times T$ , where  $m$  is the spatial dimension of such a patch and  $T$  is the spectral dimension. We perform coded sampling on these patches and obtain the coded snapshots which may or may not be further JPEG2000 compressed. The coded snapshots are retrieved from the compressed data stream at decoder by JPEG2000 decompression and then decoded using a multilayer perceptron decoder. The deep learning based sparse recovery has been shown to be better [10] than the iterative techniques like OMP, LASSO and also the recent Gaussian Mixture model based approach [18].

We consider a multilayer perceptron (MLP) architecture as proposed in [11] [10] to solve the above sparse recovery problem in Eq. 2 and learn a non-linear function  $f(\cdot)$ . The function  $f(\cdot)$  maps the coded snapshot patch  $\mathbf{I}_i \in \mathbb{R}^{m^2}$  to a hyperspectral volume  $\mathbf{E}_i \in \mathbb{R}^{m^2 T}$ . Each of the  $K$  hidden layers of this multilayer perceptron can be defined as

$$H_k(\mathbf{I}_i) = \sigma(w_i \mathbf{I}_i + \mathbf{b}_i), \quad (3)$$

where  $\mathbf{b}_i$  is the bias vector and  $w_i$  is the weight matrix. The weight matrix  $w_1 \in \mathbb{R}^{m^2 \times m^2 T}$  connects the input layer to the first hidden layer and rest  $w_{K-2} \in \mathbb{R}^{m^2 T \times m^2 T}$  connect the intermediate adjacent hidden layers. The last hidden layer connects to the output layer via  $w_0 \in \mathbb{R}^{m^2 T \times m^2 T}$  and  $b_0$ . We use the rectified linear unit (ReLU) as our non-linear function  $\sigma(\cdot)$  defined as  $\sigma(x) = \max(0, x)$ . We train the proposed MLP by learning all the weights and biases by a back propagation algorithm minimizing the quadratic error between the set of coded measurements and corresponding hyperspectral volume patches. The loss function is the mean squared error (MSE) which is given by

$$L(\theta) = \frac{1}{N} \sum_{i=1}^N \|f(I_i, \theta) - E_i\|_2^2, \quad (4)$$

where  $\theta$  is the common parameter place holder for the MLP weights and  $N$  is the total number of HS volume patches. We use this MSE as our objective to maximize the PSNR which is directly related to this quantity.

### 3 Inference with Compressed Data

Spectral datacubes have nearly no motion across the bands and just intensity variation. This property leads to our coded snapshots being visually very similar to the original spectral bands. There is some distortion due to the coded sampling process but the major image features are still quite recognisable. This visual similarity between the original bands and the coded snapshots lets us use various inference methods directly on them. This leads to faster inference tasks as it can be performed directly on the compressed data even before decompression with slight loss in accuracy. In the next section we show evaluation of a few inference tasks like classification and clustering directly on our coded snapshots and compare them to when this is done on the original datacube.

## 4 Experimental Results

The proposed compression algorithm for low bit rate compression was tested on aerial hyperspectral datasets. Pavia University dataset<sup>1</sup> is used in our experiments since the ground truth is available for this dataset. However, the same trained neural network was used to decompress all test datasets. We present qualitative and quantitative evaluation which were performed on the calibrated Pavia University dataset. We also use the same dataset for evaluation and comparison of feature preservation performance of various compression schemes. Final datacube size used is  $610 \times 340 \times 96$ , after removal of noisy bands. There are 10 classes available in the ground truth with widely varying sample counts

<sup>1</sup> [http://www.ehu.es/ccwintco/index.php/Hyperspectral\\_Remote\\_Sensing\\_Scenes](http://www.ehu.es/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes).

across classes. This work uses Python programming language as a base and the PyTorch framework for implementation of the decompression using multilayer perceptron. We wish to add here that the Pavia Dataset considered for uniformity across experiments, we have also run our experiments on uncalibrated dataset, which can be found in supplementary material.

#### 4.1 Training the Multi-layer Perceptron

We train our neural network on the hyperspectral data of natural scenes [4] provided by the University of Manchester. Each of our test datacubes is divided into volume patches of size  $m = 8$  and  $T = 16$ . These datacubes are unrelated to the test set. We use these volume patches to obtain corresponding coded snapshot patches by multiplying them with a measurement code matrix. The network was trained for 7 hidden layers and  $4 \times 10^6$  iterations with a mini-batch size of 200. The input features were normalized to zero mean and unit standard deviation. The weights of each layer were uniformly distributed in  $(\frac{-1}{\sqrt{s}}, \frac{1}{\sqrt{s}})$ , where  $s$  is the size of the previous layer. The optimizer used is stochastic gradient descent (SGD) with a learning rate of 0.01 and a momentum of 0.9.

#### 4.2 Performance Evaluation

**Qualitative Comparison.** Figure 2 presents a few reconstruction results of the Pavia University datacube for comparing visual quality for different compression techniques. It may be observed that JPEG2000 compressed images incur smoothed out edges, while the proposed technique preserves the edges and details comparatively better and gives a much better reconstruction much closer to the original image band. The textures can be seen to be comparatively better preserved in the reconstruction by the proposed method, and compares well with the PCA based spectral decorrelation. Figure 3 presents the difference images of PCA+JP2 reconstruction and CSDL reconstruction with respect to the original HS band. We highlight here that the proposed technique does not involve any kind of spectral transform or decorrelation as it would require additional computation.

**Quantitative Comparison.** The rate-distortion curves for three compression techniques applied to the Pavia university dataset are presented in Fig. 4. It can be easily observed from the plots that the proposed technique is better in terms of PSNR and SSIM compared to JPEG2000 over the entire low bit per pixel regime. This implies that for a given bit per pixel ratio we are able to squeeze out a higher quality image when using the proposed technique over the JPEG2000 compression. It can be seen that a PSNR improvement of around 4–5 dB and SSIM improvement of over 40–60% can be achieved by using the proposed technique over JPEG2000. However the PCA transform based spectral decorrelation does provide a better PSNR measure, although the SSIM figures are nearly identical. This also indicates that the proposed technique is able to





Original



JPEG2000

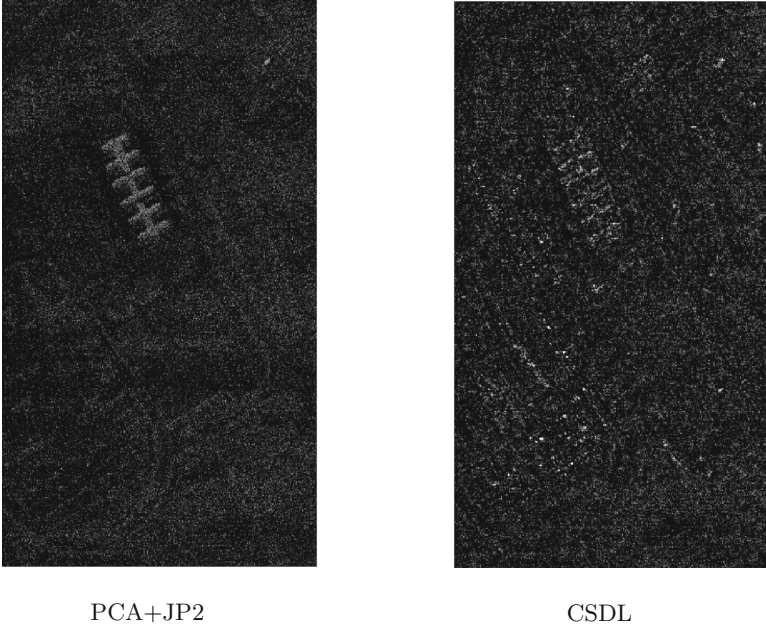


PCA+JPEG2000



CSDL

**Fig. 2.** A reconstructed spectral band from Pavia University datacube using JPEG2000, PCA based spectral decorrelation and CSDL compression techniques at a bpp of 0.3.



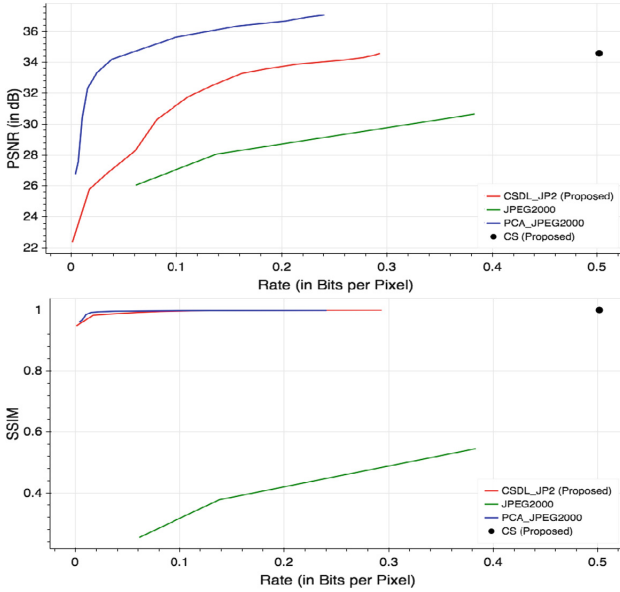
**Fig. 3.** Difference images of the PCA+JP2 reconstructed and CSDL reconstruction with the original HS band.

exploit the spectral redundancy present in the datacube and not just the spatial redundancy.

**Feature Preservation.** Hyperspectral datacubes due to their huge size and a large number of bands typically are employed for inference and classification based tasks to extract useful information from them. We present a comparison of how the performance of a machine learning algorithm is affected by these compression schemes. We show the effect of compression on both supervised classification and on clustering techniques on calibrated Pavia dataset as the corresponding ground truth are available for them. In case of supervised classification, we show comparison using three commonly used classifiers, namely, the decision tree classifier [6], the nearest neighbour classifier and the multilayer perceptron classifier [2]. We use a 70:30 train-test split for each of the classifiers,  $K = 15$  for the KNN and 3 layers for the MLP.

Our models for each of these classifiers are trained on the original uncompressed datacube along with its ground truth (performance shown under raw data in Table 1). The pixel values of various bands after decompression were taken to be features and the corresponding ground truth values were taken as the labels. The classification of reconstructed datacubes was done using the same model trained on the original datacube. This comparison is presented in Table 1,





**Fig. 4.** PSNR and SSIM vs BPP plots for the Pavia University datacube using JPEG2000, PCA based spectral decorrelation and CSDL compression techniques. The black dot corresponds to CSDL alone, without subsequent JPEG2000.

which demonstrates how the proposed technique is better than both JPEG2000 and PCA based spectral decorrelation.

**Table 1.** Effect on classification accuracy (in percentages) of supervised classifiers over three different compression techniques applied to Pavia University datacube at 0.3 bpp.

Classifier	Raw data	JPEG2000	PCA+JPEG2000	CSDL
D-Tree	78.322	57.023	76.396	<b>78.295</b>
K-NN	79.695	79.592	77.971	<b>79.458</b>
MLP	79.533	79.449	79.500	<b>79.512</b>

As a part of another comparison, the test datacubes were classified into clusters without any training to examine how much spectral distortion comes in due to these compression schemes. The K-Means algorithm [12] is used to cluster the datacubes and the results are shown in Fig. 5 for the Pavia University dataset. The number of clusters ( $N = 5$ ) has been kept same in all cases. A quick observation that can be made from this result is that the regular features of the image and region boundaries are considerably distorted in the JPEG2000 reconstructed image bands. The CSDL compressed image band, on the other hand, shows a much lesser distortion. This experiment clearly demonstrates that

the proposed compression technique is able to preserve the clusters intact and induces comparably a lower radiometric distortion when compared to both the JPEG2000 and PCA transform based coding techniques for a given compression ratio. The proposed method retains nearly the same accuracy as the original raw data even when the image is compressed at 0.3 bpp.

**Computational Requirements.** The proposed technique offers a much faster and in-place computation at encoder in comparison to the other two compression algorithms. Table 2 demonstrates that a speedup factor of 20 can be achieved by the CSDL (without further JPEG2000 encoding) technique in comparison to PCA transform spectral decorrelation based compression.

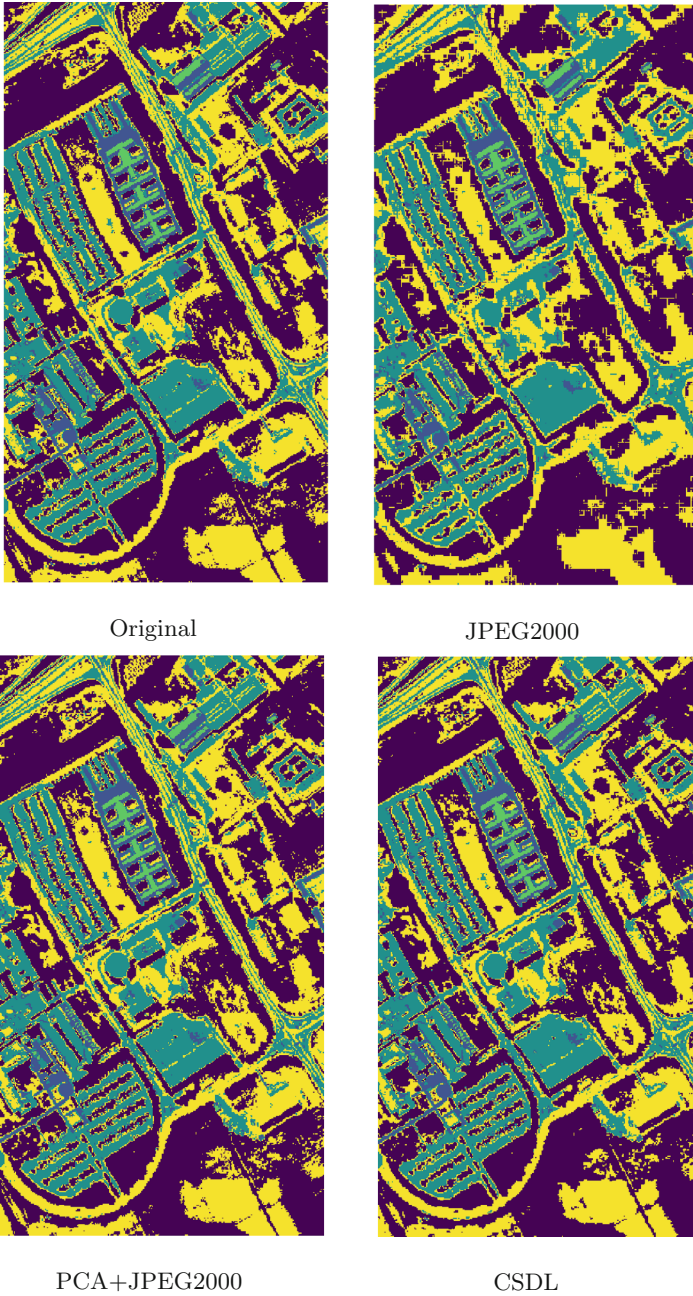
**Table 2.** Computation time (in seconds) for the PCA, PCA+JPEG2000, CS, CS+JPG2000 coding at 0.3 bpp on a 3.5 GHz i7 computer with 16 GB RAM. These are averages of ten runs with a minimal load on CPU i.e. no other processes running.

Datacube size	PCA+JPEG2000	PCA	CS+JPEG2000	CS
$610 \times 340 \times 32$	7.051	6.675	<b>0.267</b>	<b>0.016</b>
$610 \times 340 \times 96$	20.329	19.539	<b>0.927</b>	<b>0.065</b>

Table 2 also highlights the advantage of using a CS coding over a PCA transform based spectral decorrelation without the following JPEG2000 compression. The former is over two magnitudes faster than the latter. This would be desirable at compression setups with minimal computational power, battery power and storage, for instance, an onboard platform. We also wish to highlight that the proposed technique offers faster compression than even the recent state of the art Hyperspectral data compression technique by Fu et al. [7]. As per the timing results presented by the authors, their technique requires a computation time comparable to the PCA+JPEG2000 based method.

**Inference with Compressed Data.** We train a fully connected neural net on our coded snapshots for evaluation of classification performance in the compressed domain itself. As in the features preservation section above, each pixel of the coded snapshot datacube is taken as a sample with the spectral axis constituting the features. The comparison is again presented using the three classifiers namely, Decision tree (D-Tree), K nearest neighbours (K-NN) and a multilayer perceptron (MLP). The classification accuracy comparison is shown in Table 3. This task of performing inference in compressed domain is not possible in other transform coding techniques.

It can be observed that with a loss in accuracy of around 1–5% we can perform inference tasks like classification on our data faster. Since we are dealing directly with the coded snapshots, we save processing power as the number of features a classifier needs to take is  $T$  fold less. In the above experiment  $T$  was taken to be 16.



**Fig. 5.** Effect of various compression techniques (at 0.3 bpp) on unsupervised clustering of the Pavia University dataset.

**Table 3.** Effect on classification accuracy (in percentages) of supervised classifiers over raw data and coded snapshots of Pavia University datacube.

Classifier	Raw data	Coded snapshots
D-Tree	78.322	72.570
K-NN	79.695	75.646
MLP	79.533	79.274

## 5 Discussions and Conclusions

UAVs and drones are a major source of hyperspectral data collection for various remote sensing and monitoring applications. Being comparatively more feasible than launching satellites they offer a higher resolution imagery along with a faster acquisition-retrieval-processing cycles. But with a limited power available onboard the acquired data has to be brought down to a receiving station for processing. Although the transform coding based data compression saves the transmission bandwidth and time, the onboard power requirements for it are quite high.

The proposed method of coded snapshots involves no transform coding as part of the first step and can benefit from transform coding on top of it, if computational onboard power availability allows it. This leads to a lower power requirements and hence longer operation times for the UAV deployed. Along with this, the similarity between the coded snapshots and the original datacube bands lets us do inference in compressed domain itself which leads to faster decisions and pre-processing before the data is sent to ground stations. Onboard inference tasks can speed up disaster recovery efforts along with other planning and public warning systems.

We presented a computationally very fast and simple encoder for low bit per pixel compression of hyperspectral datacubes. These desirable features of the encoder bring in some computationally undesirable effects due to the need of sparse recovery algorithms for which we presented a computationally faster and parallelizable decoder architecture based on deep neural networks. The proposed technique also offers the capability for progressive transmission of compressed data and saves compression time and a good amount of onboard storage due to in-place computation. The proposed technique offers a low computational complexity at the encoder end which is highly desirable at a drones, UAVs or spacecraft, with limited computational and power capabilities. The deep learning based decoder, however being comparatively complex, can be offloaded to a GPU making it much faster than iterative sparse recovery algorithms. This coding technique also lets us perform inference on the compressed data directly leading to a faster and computationally light on board-decision making, when required.

## References

1. Adão, T., et al.: Hyperspectral imaging: a review on UAV-based sensors, data processing and applications for agriculture and forestry. *Remote Sens.* **9**, 1110 (2017)
2. Benediktsson, J., Kanellopoulos, I.: Classification of multisource and hyperspectral data based on decision fusion. *IEEE Trans. Geosci. Remote Sens.* **37**, 1367–1377 (1999)
3. Du, Q., Fowler, J.: Hyperspectral image compression using JPEG2000 and principal component analysis. *IEEE Geosci. Remote Sens. Lett.* **4**, 201–205 (2007)
4. Foster, D., Amano, K., Nascimento, S., Foster, M.: Frequency of metamerism in natural scenes. *JOSA* **23**, 2359–2372 (2006)
5. Fowler, J., Rucker, J.: Three-dimensional wavelet-based compression of hyperspectral imagery. In: *Hyperspectral Data Exploitation: Theory and Applications*, pp. 379–407 (2007)
6. Friedl, M., Brodley, C.: Decision tree classification of land cover from remotely sensed data. *Remote Sens. Environ.* **61**, 399–409 (1997)
7. Fu, W., Li, S., Fang, L., Benediktsson, J.A.: Adaptive spectral-spatial compression of hyperspectral image with sparse representation. *IEEE Trans. Geosci. Remote Sens.* **55**(2), 671–682 (2017)
8. Green, A., Berman, M., Switzer, P., Craig, M.: A transformation for ordering multispectral data in terms of image quality with implications for noise removal. *IEEE Trans. Geosci. Remote Sens.* **26**, 65–74 (1988)
9. Hitomi, Y., Gu, J., Gupta, M., Mitsunaga, T., Nayar, S.: Video from a single coded exposure photograph using a learned over-complete dictionary. In: *International Conference on Computer Vision*, pp. 287–294. IEEE (2011)
10. Iliadis, M., Spinoulas, L., Katsaggelos, A.: Deep fully-connected networks for video compressive sensing. *Digit. Sig. Process.* **72**, 9–18 (2018)
11. Kulkarni, K., Lohit, S., Turaga, P., Kerviche, R., Ashok, A.: Reconnet: non-iterative reconstruction of images from compressively sensed measurements. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 449–458 (2016)
12. Likas, A., Vlassis, N., Verbeek, J.: The global k-means clustering algorithm. *Pattern Recogn.* **36**, 451–461 (2003)
13. Rucker, J., Fowler, J., Younan, N.: JPEG2000 coding strategies for hyperspectral data. In: *Geoscience and Remote Sensing Symposium*, vol. 1. IEEE (2005)
14. Tang, X., Cho, S., Pearlman, W.: 3D set partitioning coding methods in hyperspectral image compression. In: *International Conference on Image Processing*, vol. 2, p. 239. IEEE (2003)
15. Tang, X., Pearlman, W., Modestino, J.: Hyperspectral image compression using three-dimensional wavelet coding. In: *Image and Video Communications and Processing*, vol. 5022, pp. 1037–1048. International Society for Optics and Photonics (2003)
16. Tropp, J., Gilbert, A.: Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Trans. Inf. Theory* **53**, 4655–4666 (2007)
17. Wang, J., Chang, C.: Independent component analysis-based dimensionality reduction with applications in hyperspectral image analysis. *IEEE Trans. Geosci. Remote Sens.* **44**, 1586–1600 (2006)
18. Yang, J., et al.: Video compressive sensing using Gaussian mixture models. *IEEE Trans. Image Process.* **23**(11), 4863–4878 (2014)