# Comprehensive review of hyperspectral image compression algorithms

**Yaman Dua,* Vinod Kumar, and Ravi Shankar Singh**
Indian Institute of Technology (BHU), Department of Computer Science and Engineering,
Varanasi, Uttar Pradesh, India

**Abstract.** Rapid advancement in the development of hyperspectral image analysis techniques has led to specialized hyperspectral missions. It results in the bulk transmission of hyperspectral images from sensors to analysis centers and finally to data centers. Storage of these large size images is a critical issue that is handled by compression techniques. This survey focuses on different hyperspectral image compression algorithms that have been classified into two broad categories based on eight internal and six external parameters. In addition, we identified research challenges and suggested future scope for each technique. The detailed classification used in this paper can categorize other compression algorithms and may help in selecting research objectives. © *2020 Society of Photo-Optical Instrumentation Engineers (SPIE)* [DOI: 10.1117/1.OE.59.9.090902]

**Keywords:** hyperspectral image compression; machine learning; deep learning; tensor decomposition; lossy compression; lossless compression.

Paper 20200575V received May 14, 2020; accepted for publication Sep. 10, 2020; published online Sep. 29, 2020.

## 1 Introduction

Hyperspectral (HS) imaging is an essential concept in remote sensing due to its ability to store information in detail. It has been a topic of keen interest among researchers, in recent years, as it finds its application in target detection, classification, anomaly detection, and spectral unmixing.[1] Hyperspectral image (HSI) sensors collect data in contiguous bands of wavelength ranging from 400 to 2500 nm, beyond the visible range of human vision. Each band has the same number of pixels and a fixed spectral resolution dependent on the capability of sensors. Each pixel has some spatial resolution that defines an area of the surface covered in a pixel. It collects reflectance value of an area for different wavelengths in different bands, forming a data cube that is beneficial in many applications. For instance, it is used in military operations to find and follow the progress of troops.[2] The agricultural sector uses it for quality monitoring, disease control, classification of crops, and improving production.[3] In the manufacturing industry, it helps in finding the fault detection[4] and in the space industry, it is used in the movement of celestial bodies.[5] In the case of remote sensing, it is applied to examine the Earth's surface, classifying minerals, tracking and tracing of natural calamities in the form of floods, drought, etc.

### 1.1 *Motivation*

Along with benefits, HSIs have some limitations that give rise to the concept of compression. The need for HSI compression in remote sensing can be stated as:

- The size of the HSI acquired by the sensors is in hundreds and thousands of megabytes. For example, airborne visible/infrared imaging spectrometer (AVIRIS) sensor captures 224 spectral bands with $614 \times 500$ pixels in each band, where each pixel takes 16 bits. Size of an image from such a sensor is $224 \times 614 \times 500 \times 16 = 131.17$ MB,[6] hence, storage of this large size data is an issue.

*Address all correspondence to Yaman Dua, yamandua.rs.cse18@iitbhu.ac.in

- Limited transmission channel bandwidth: HSIs have to be transmitted from one place to other, and large size of data requires high bandwidth, which is a costly resource in remote sensing applications.
- Limited data–transmission time: At sensors, HSIs are captured very frequently that needs processing at a high rate, which is very complex at capturing device. These images, if not transmitted in limited time, can result in information and calibration loss at data centers.

HSI[7] compression is a technique through which the size of HSI can be reduced without loss in image quality beyond the desired level. It is one of the essential steps of HSI processing technique, which is included in every space mission, as it reduces the cost of bandwidth and storage equipment. In lossless mode, compression reduces the size by storing the same information with a small number of bits, by two methods using different representations, and removing existing redundancy. High redundancy helps compression algorithms to achieve a high compression ratio (CR). Basically, statistical redundancy and psychovisual redundancy are two broad categories of redundancies in digital images. While former one plays significant role in HSI, the latter one is of no prime importance due to its limitation of impact in visible range only. Statistical redundancies occur due to near similar intensity of pixels in neighborhood except at the locations where illumination changes. It can be classified into interpixel redundancy and coding redundancy. There exist three types of interpixel redundancy in an HSI: (i) spatial redundancy: It arises due to intraband dependency that exists in spatial domain, (ii) spectral redundancy: It occurs due to dependency among pixels of different bands at the same spatial location, and (iii) temporal redundancy: It arises when HSI of the same location is taken at different times, dependency in temporal domain (for corresponding spectral and spatial pixels) results in temporal redundancy. These redundancies are decorrelated in compression algorithms, and thus data size is reduced. Original data can be reconstructed using decompression, which is usually the reverse process of compression.

A systematic overview of HSI compression is provided in this paper. Algorithms proposed in the existing literature are divided into different categories based on essential factors and compared along with their future research directions. The overall objective of this survey can be summarized as:

- A clear concept of HSI compression techniques.
- Comparison of algorithms within the scope of this article based on application, implementation, strategy, and location.
- Categorization of techniques based on the architecture of algorithms.
- Some research challenges and future scope in terms of HSI compression techniques.

The remainder of this paper is organized as follows. Section 2 describes the categorization of HSI compression algorithms based on the architecture of algorithm and various parameters. In addition, a detailed analysis of the architecture of algorithms, their advantages, and disadvantages are given. Discussion and open challenges are provided in Sec. 3, and the review ends with concluding remarks in Sec. 4.

## 2 Categorization of Hyperspectral Image Compression Algorithms

HSI compression is a broad domain that can be classified into various categories. In this review, a method of categorization is adopted. We classified the algorithms into three different ways, i.e., classification based on various parameters, set of metrics used to evaluate a particular algorithm, and methodology of algorithms. Its details have been provided in subsequent sections.

### 2.1 *Categorization Based on Methodology*

HSI compression techniques are categorized by the methodology it adopts. There are various methods to compress an image, each way having its advantages as well as limitations. We categorized algorithms into eight broad categories namely transform-based, prediction-based, vector quantization (VQ)-based, compressive sensing-based, tensor decomposition-based, sparse representation-based, multitemporal-based, and learning-based algorithms. Figure 1 shows different compression techniques classified on the basis of its methodology and various

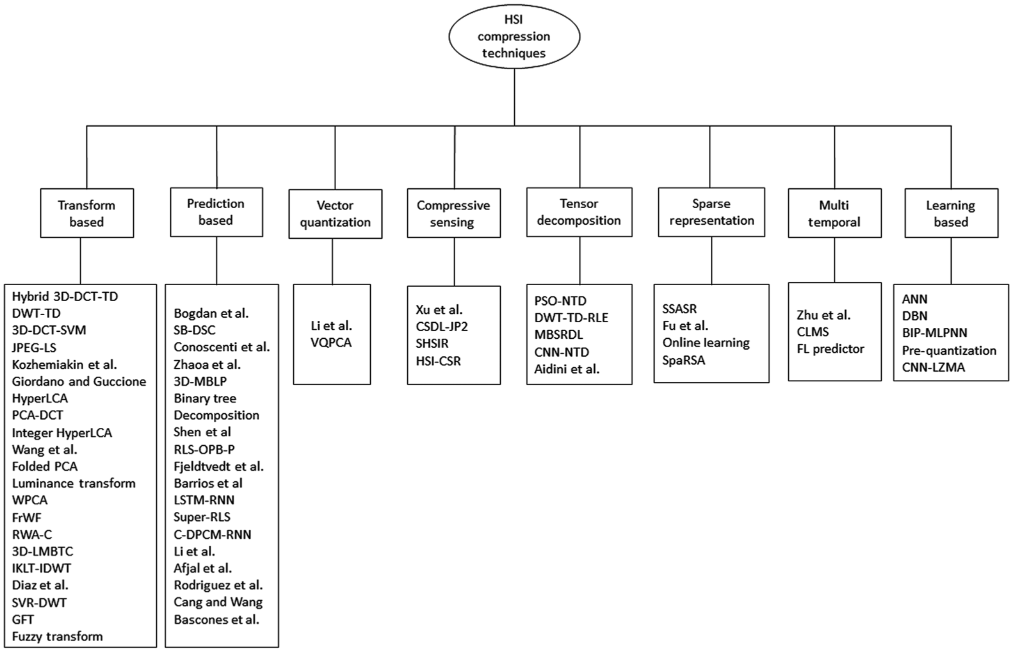

**Fig. 1** Classification of compression techniques.

algorithms in each category. Each method is discussed in detail along with their advantages, limitations, state-of-the-art algorithms, and research challenges in following sections.

### 2.1.1 *Transform algorithms*

*Overview.* Transform-based technique is the most popular two-dimensional (2-D) image compression technique that has been extended to three-dimensional (3-D) or HSI compression. It is known as transform-based technique as it transforms the pixels values into the frequency domain by applying transformation function to all the three dimensions of image. There are some great transformation techniques such as discrete cosine transform (DCT), discrete Fourier transform, discrete wavelet transform (DWT), and Karhunen–Loeve transform (KLT) that are used in image compression. It can remove both spectral and spatial correlation depending on the domain on which it is applied. The technique can be applied in combination with nearly all other methods such as prediction-based, VQ, tucker-based, compressive sensing, and in learning-based algorithms. Some state-of-the-art algorithms in this field are 3D-DWT,[8] 2D-KLT,[9] 3D-set partitioning embedded block (SPECK),[10] and 3D-low memory block tree coding (LMBTC).[11]

*Technique.* Compression using the transform-based method follows some steps that may vary for different algorithms but can be generalized as in Fig. 2. Forward transform applies a transformation function (cosine, wavelet, or Fourier) to one of the spatial- or spectral-domain or both, then performs decorrelation, and generates coefficients. Following which, quantization is completed; this removes factors that are close to zero. In the last step, encoding techniques are applied to the quantized coefficients to generate bit-streams. It could be transmitted or stored with a reduced number of bits per pixel to save space (in storage) and bandwidth (in transmission).

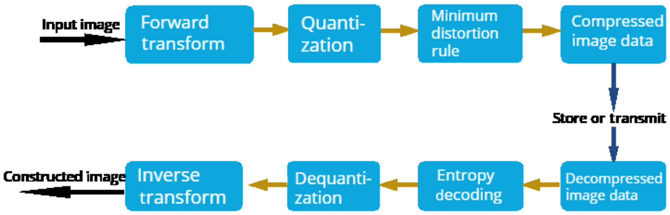

**Fig. 2** Steps of transform-based algorithm.

Some transform-based his compression algorithms in the scope of this article are discussed below. Karami et al.[12] proposed a transform-based technique in which 3D-DCT is applied hisHSI. It converts the raw pixels into frequency-domain coefficients using a cosine and inverse cosine transformation function on all the three dimensions of the image. High and low energy coefficients are separated out of which low energy coefficients are dropped using quantization. Sparse Tucker decomposition (TD) is then applied to modified coefficients to generate a compressed image. The reverse process is followed in decompression that generates original image with some loss due to irreversible quantization process. Karami et al.[8] proposed another transformation technique named 3D-DWT-TD on the HSI, which uses DWT to transform spatial-domain pixels into frequency domain using wavelet function on all the three dimensions of HSI. Four submatrices are generated containing edge-based information, horizontal information, and vertical and approximation information in each submatrix. TD is then applied on all these four matrices separately followed by generation of mode matrices. Entropy coder is then used to code the core tensor and original image is reconstructed using reverse process.

Transformation-based technique has also been applied with machine learning techniques such as support vector machine (SVM)[13] that has the following steps. Cubical HSI is first divided into small frames to reduce the complexity, and 3D-DCT is applied at the compression end on subimages. It is followed by a 3-D zig-zag quantizer that removes the unnecessary coefficients. SVM regression is used on the leftover coefficients to generate support vectors and weights, which are then encoded by entropy coder. Töreyn et al. proposed a hybrid algorithm named as joint photographic experts group-lossless (JPEG-LS)[14] in which one-dimensional (1-D) integer wavelet transform is applied on spectral bands. It gives a residual image that is encoded by Golomb-rice encoding. Decompression is used on the bit-streams that reconstruct the original image without any loss. The performance of the proposed method is comparatively better than JPEG. Kozhemiakin et al.[15] proposed a compression method based on 3-D AGU coder that calculates cross-correlation factor for images in different channels. Frequency coefficients are obtained from 3-D DCT where quantization step is set proportional to noise standard deviation. AGU coder is applied at the last level of compression. Giordano and Guccione[16] proposed a combination of clustering and transformation for compression of HSI implemented on graphical processing unit (GPU). It is a region of interest (ROI)-based compression method that clusters the input image into five application-specific classes with an assumption that reflectance value of pixels is preloaded into memory. Labeling of a block is done according to a rule as "ROI" and others as "not-ROI." Then, principal component analysis (PCA) is applied to reduce the spectral redundancy, and PCs with variance 99.9% are retained. The labeled image is also processed by 2-D DWT for spatial redundancy. Another use of machine learning technique in combination with DCT was proposed in PCA-DCT,[17] where PCA is applied to find the feature vector, similarities, and dissimilarity in the form of residual image. Subsequently, DCT is applied to compress image. To further improve the compression performance, Mei et al.[18] proposed a hybrid algorithm named folded-PCA in combination with JPEG2000. In this technique, the covariance matrix is calculated by folding the spectral vector into a matrix, and eigenvectors are used to obtain principal components that can represent features of the entire image. JPEG2000 is then applied to the reduced image to compress it further. The algorithm was again extended in weighted principal component analysis (WPCA)[19] where an adaptive cosine estimator algorithm was applied for target detection. Then, PCA is applied to HSI after converting it into 2-D matrix but with some modification in the weight matrix. Mean pixel matrix and covariance matrix are calculated by giving more weight to pixels around the detected target.

Wang et al.[10] proposed a joint decoder method where the 3-D wavelet transform to find high- and low-frequency regions. Turbo channel coding is applied for encoding high-frequency region (represents antierror) and 3D-set partitioning in hierarchical trees (SPIHT) for low-frequency region (image energy). The decoder uses low-frequency information to predict high-frequency coefficients. It also creates side information that is jointly decoded. Guerra et al.[20] proposed a lossy compression algorithm named HyperLCA using transformation function to achieve better CR at the cost of reasonable computational complexity. It has three steps namely spectral transformation function to remove spectral redundancy followed by preprocessing stage and lossless encoding as its last step of compression. Most different pixels are selected in preprocessing stage, which can be coded independent of any spatial alignment. Golomb rice coding is used to preserve this information until the image has been finally decompressed. Integer HyperLCA[21] was later proposed as an extension to the original algorithm that improves the performance further by dividing parameter's floating point values into an integer part and a decimal part. Luminance[22] transform proposed a transformation function considering an assumption that intensity of light falling on various bands of the same HSI is almost equal. The authors used luminance transform to reduce the difference of brightness and contrast between spectral bands at the same spatial location. DCT was then applied on the resultant image to minimize the spatial correlation in HSI. The results obtained were better than applying only DCT to raw image. An extension[23] of wavelet-based transformation was proposed by Khan et al. The method used 1-D convolution to decompose the image temporally and fractional wavelet filter (FrWF) transform to remove spectral and spatial correlations, and the coefficients are then quantized. The factors are grouped as significant and nonsignificant using dyadic wavelet transform. Then, it employs 2-D SPIHT to encode these coefficients using a tree-based orientation that can represent insignificant coefficients using a single value. A lossless compression algorithm, regression wavelet analysis-clustered (RWA-C),[24] was proposed by Ahanonu et al. that used cluster analysis to divide the image into $N$ clusters. Wavelet transformation is applied on these clusters to decorrelate spectral information and obtain wavelet coefficients. Linear regression is used on spectral coefficients within a cluster, and significant factors are found through least square regression. Memory requirement of techniques is an important issue that is taken care by 3D-LMBTC[11] to encode a higher bit plane to lower bit plane using wavelet coefficients. A block is matched to every other block to find the significance that is encoded in further steps.

An integer-based hybrid transformation method[25] ILKT-IDWT has the following steps. The input HSI is first converted into multiple 1-D vectors that are clustered and tiled using eigen matrix decomposition. Invertible integer KLT map is then applied to the spectral matrix, followed by an integer DWT to spatially decorrelate the image data. Three different wavelet-based coding are proposed to be implemented on the decorrelated image, they are spatial-oriented tree wavelet (STW), wavelet difference reduction (WDR), and adaptively scanned wavelet difference reduction. In the first method, coefficients are ordered from high magnitude to lower magnitude in pyramid structured tree, making it a complicated process. Second method discovers three categories of arrays using an iterative approach that divides threshold value by 2 each time. Third method applies an adaptive way of scanning among different arrays. Computational complexity of HyperLCA method was reduced in the method given by Díaz et al.[26] The proposed approach utilizes the parallelism in HyperLCA algorithm and performs execution on the Nvidia Jetson TK1 and TX2 GPU. Three models of parallel implementation are proposed to accelerate compression. First model executes transformation on GPU while performing all the steps of HyperLCA sequentially. In the second model, coding and transform are executed by different central processing unit (CPU) processes. Third model implemented the transform using three threads of GPU and each code block on different CPU process. Support vector regression (SVR)-based compression was implemented in SVR-DWT,[27] where 3-D DWT is applied on the input HSI followed by SVR on normalized coefficients to identify the support vectors and weights for spectral information. Weights are quantified using floating point quantizer and encoded by entropy coding technique. Spatial data are separately encoded by lossless differential pulse code modulation (DPCM) to preserve the low-frequency image details. Decompression stage is exact reverse of compression stages. Another transformation function named graph Fourier transform (GFT)[28] is used to decorrelate the HSI in spectral domain. Laplacian matrix is used to obtain the transformation vectors uniquely for each signal. Impact of GFT on correlation is calculated to assess the quantization value for Gaussian Laplacian vectors that are selected depending on the amount of loss permitted by the application. Fuzzy logic[29] has also been used in compression by Monica and Widipaminto. The proposed method modifies the fuzzy transform that finds a correspondence between a set of $n$-dimensional vectors and continuous functions with the help of membership functions. Existing Perfilieva's fuzzy transform uses sinusoidal membership function that is modified to pseudoexponential function. The input image is first broken down into frames of size $(8 \times 8)$ and pixels are normalized into [0, 1]. Each frame is then transformed into $n$-dimensional matrix using fuzzy transform. The advantages, limitations, and future directions of each algorithm are listed in Table 1.

**Table 1** Transform-based HSI compression techniques.

| Algorithms | Advantages | Limitations | Future research directions |
|---|---|---|---|
| Hybrid 3D-DCT and TD[12] | Achieves compression objectives as 85% of coefficients are discarded. | Parameters are manually selected. | Develop some method to select the new reduced dimensions automatically. |
| DWT-TD[8] | Better pixel-based results for CA. | The computational load of TD is very high. | Reduce the load by reducing computations of the core tensor. |
| 3D-DCT-SVM[13] | Comparatively better performance. | Implementation complexity not considered. | Different types of nonlinear kernels can be applied to SVM. Optimization algorithms can further optimize SVM. |
| JPEG-LS[14] | Dual-mode of encoding, like for flat-region.<br>RLE is performed and Golomb rice coding elsewhere. | Overall CR depends heavily on the data. | Improve decorrelation performance, then apply 2-D compression to optimize. |
| Kozhemia-kin et al.[15] | Percentage of zeros obtained after quantization of DCT coefficients can predict improvement of CR due to combination of channels into a group. | Applied on multispectral image dataset considering very few parameters. | This technique can be applied in machine learning to check whether to combine channels or not for other algorithms. |
| Giordano and Guccione[16] | Clustering is done onboard. Superior performance in ROI-based compression. | Creation of a database of reflectance is not considered. | Segmentation can be included as preprocessing step. |
| | | Amount of energy consumed on-board is an essential factor, which is not discussed. | Automated process for selection of many bits in ROI part. |
| HyperLCA[20] | The target CR can be fixed in advance. High error resilience. | Lossy compression algorithm, and minute details are lost. | Exploiting parallelism in the proposed algorithm. |
| PCA-DCT[17] | Target extraction not required.<br>Comparably high CR. | Selection of the number of features is not considered. | Use of different transformations technique for spatial decorrelation. |
| Integer HyperLCA[21] | Sensor and data independent.<br>Hardware friendly method | Minute details are lost. | Integer-point operation technique could be used in other algorithms. |
| Wang et al.[10] | The scalable and flexible method as low-frequency components are transmitted first in a narrow channel. | Two approaches used for coding different channels. | Hardware implementation of the proposed algorithm. |
| Folded PCA[18] | Best compression and classification achieved when the number of principal components (PCs) = 40 and $h = 10$. | Not suitable for many applications. | Parallel implementation of the algorithm. |
| Luminance transform[22] | Better energy compaction. | Results are not compared with any other algorithms. | Use the same technique of single transform in other compression methods. |
| WPCA[19] | Better target detection performance especially at low BR. | Poor performance at high BR.<br>Target should be identified in the beginning. | Target-based compression can be used in other applications. |
| | | Weight matrix generation by two different methods is a complex process. | Generalizing this technique using region-based instead of target based. |

**Table 1** (*Continued*).

| Algorithms | Advantages | Limitations | Future research directions |
|---|---|---|---|
| FrWF[23] | Energy efficient encoder due to utilization of lower energy in coding than transmission.<br>Uses less complicated fractional filter that reduces the need of large memory in compression.<br>Only three image lines have to be stored during wavelet transformation. | Information stored in less significant coefficients can never be retrieved, which can be scattered over spatial domain. | The utilization of 3-D SPIHT can further reduce the size of compressed image.<br>Efficient coding of zero tree obtained after applying SPIHT. |
| RWA-C[24] | Performed for a wide range of number of clusters. Clustering improves compression performance. | Integer Cohen, Daubechies, and Feauveau (CDF) 5/3 DWT is used, which fails to give an optimal solution. | CDF 9/7 wavelet can be used to improve performance. |
| 3D LMBTC[11] | Can be implemented in HS sensors. Needs only 12 kilobytes (kb) fixed memory.<br>Coding time is reduced. | Ineffective increase in computation efficiency compared to wavelet block tree coding | Use with other transform coding techniques. |
| IKLT-IDWT[25] | Provides ideal energy compaction in IKLT transform. STW provides best results when implemented with hybrid transform.<br>Different wavelet functions are evaluated for IDWT. | Evaluated only on cubical portion of imageNot fit for on-board and real-time compression. | Identification of scope of parallel implementation to reduce the compression time of STW coding that gives the optimum results. |
| Diaz et al.[26] | Consumes low power providing real-time compression. | High implementation complexity. | Hardware-based accelerators can be used to operate the proposed models on satellite. |
| | Spatial alignment between blocks of pixels are not needed, each can be independently compressed. Developed specially for smart farming application but can be extrapolated to other remote sensing fields. | Not fit for onboard compression | Parallelism model can be used to accelerate some lossless compression algorithm. |
| SVR-DWT[27] | Works fine for small training samples.<br>Compressed data gives better classification performance than original data to support vector-based coding. | Time taken for high resolution is more due to the use of SVR. Compression of spatial information separately is an overhead. | Implementation of wavelet packet transform for better compression performance. |
| GFT[28] | Competitive performance for both low and high bitrate setting.<br>Performance can be varied by selecting different number of transformed coefficients. | Spatial correlation is not considered during compression.<br>Gaussian Laplacian vectors are not selected using any specified method. | Implementation of 2-D transform separately for spatial decorrelation.<br>Use of context-aware algorithm for vector selection to improve performance. |
| Fuzzy transform[29] | Improved performance due to preprocessing and membership function.<br>Solution becomes easy as it simplifies to simple algebra using fuzzy modeling rules. | Parameter selection is based on a hit and trial method that may vary for different images. | Reduce the compression time of the algorithm.<br>Compare the performance with membership functions other than pseudoexponential function. |

*Research challenges and future directions.* This technique can be applied to both data-center and onboard compression due to fast calculations. It has several advantages such as error-tolerance mechanism, high compression performance, flexible in using rate-control mechanism, and global optimum solution. Disadvantages[30] of transform-based compression include high computation time as it performs a large number of computations such as multiplication, transpose, and the inverse of matrix. Optimal performance can be obtained at low bit-rates (BR) only. It[31] destroys the inherent structure of HSI and gives rise to high-order dependency since it considers the image as a matrix.

The computation time of transform-based technique can be reduced by exploiting parallelism in the algorithms and applying it on high-performance computing (HPC) architecture, which can improve its performance. In multitemporal HSI, time domain is also available along with spectral and spatial domains that give rise to temporal correlation. Four-dimensional (4-D) transform-based techniques need to be developed to address such issues. Existing algorithms can be implemented using the latest transformation function with the objective of performance improvement.

### 2.1.2 *Prediction algorithms*

*Overview.* It is an alternative to transform-based algorithms with technical and implementation benefits. In this technique, the value of a pixel is predicted after applying some mathematical functions to the previous pixels. It is developed especially for 3-D images, exploits correlation in both spatial and spectral directions, and removes them. Prediction[32] in HSIs is mainly applied on spectral-domain with the help of a filter after spatial decorrelation gets completed. Mostly used filter functions to calculate weight matrices are recursive least square (RLS) and least mean squares (LMS) filter. Prediction-based algorithm is also used in conjunction with other algorithms to improve performance. Some state-of-the-art algorithms are 3D-DPCM,[33] superpixel-based segmentation-CRLS,[34] RLS-adaptice length prediction,[35] and LMS-APL.[36]

*Technique.* Prediction-based technique is easy to implement on HSIs and can be easily explained by Fig. 3. The first step removes correlation in the spatial domain for all bands, followed by a prediction of pixels of $p$'th band by performing some mathematical operations on pixels of $p$-1 bands. A weight matrix is used for this purpose, generated by a filter function depending on the algorithm. Residuals are calculated by subtracting the original image from the predicted image, which is encoded by entropy or Golomb coder.[37]

Methodology used by some prediction-based algorithms is described below. Table 2 lists the algorithms with their advantages, limitations, and some future directions.

The standard developed by working group of multi/hyperspectral data compression, consultative committee for space data systems (CCSDS), for HSI compression techniques to be used in space missions, named as, CCSDS-123.0-B is based on the predictive compression technique. Issue 1 of the method was introduced in 2012 that focused on the lossless compression of images captured by multiple satellites. Significant limitation of the standard was considerable compression time taken during the process that led to the development of various modified techniques, some of them are discussed here. An enhancement in CCSDS standard algorithm was proposed by Conoscenti et al.[38] by introducing three application-specific extensions such as constant signal-to-noise ratio (SNR), rate control, and hybrid coding. Low energy areas with tremendous noise have been removed from the prediction process by keeping an upper bound

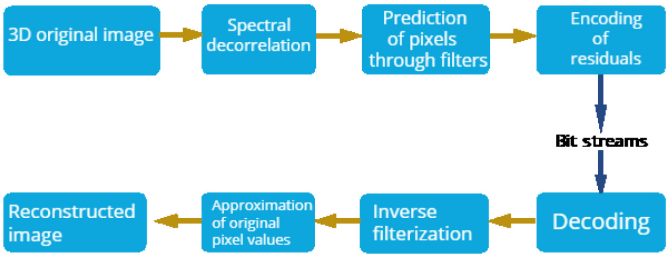

**Fig. 3** Steps of prediction-based algorithm.

**Table 2** Prediction-based HSI compression techniques.

| Algorithms | Advantages | Limitations | Future research directions |
|---|---|---|---|
| Bogdan et al.[40] | A better approach to comparison. | FPGA calculates integers, conversion from floating point to integer point is difficult. | The procedure could be followed for other algorithms. |
| SB-DSC[41] | Decreased encoder complexity for some of the blocks. | Complex implementation. | Implement calculation of MaxAE on hardware, as it consumes most of the time. |
| Conoscenti et al.[38] | Application friendly algorithm.<br>Allows constant SNR compression. User can control the rate. | In some cases, the hybrid encoder provides worse results. | Implementation of the algorithm on hardware.<br>Shared/distributed memory implementation. |
| Zhaoa et al.[39] | Local smoothing and noise effect has been reduced. | Fractal encoding is used which reduces performance at low bitrates. | Include DWT to reduce loss. |
| 3D MBLP[42] | Scalable to new sensors data after configuration. | Nonoptimal solution high complexity. | A new preprocessing step (reordering of the band according to correlation) to be included. |
| Binary tree-based decomposition[43] | Use of tree data structure help to code less occurring pixels with less number of bits. | Tested only on context-based entropy coding. | Implementation of proposed BTBD technique on a better predictor. |
| Shen et al.[44] | Very close performance, even if the entire image is considered ROI.<br>Shorter bit-streams due to separate coding of boundary and other pixels. | Predefined ROI in test images. | Complexity evaluation of the algorithm.<br>Development of some automatic process to select ROI. |
| RLS-OPB-P[35] | Improved computing performance.<br>Multi-GPU concept achieves high speed-up and improves the complexity. | Compression performance not improved significantly. | Further increment in the speed-up by considering host-to-device I/O communication in GPU. |
| Fjeldtvedt et al.[45] | Specifies hidden data dependency.<br>Maximum throughput (both in Msamples/s and Mb/s), least power consumption. | Requires external memory for implementation. | Implementation of the same algorithm on more general platforms such as Virtex and sensor maximum. |
| Barrios et al.[46] | Platform independent. | Poor results when implemented on "Mentor CatapultC." | Implementation of CCSDS 123.0-B-1 lossy and near-lossless versions. |
| LSTM-RNN[47] | First attempt to model temporal correlation dependencies in filtering weights.<br>Achieves minimum RMSE for two datasets. | Only 30% of pixels are used in training each LSTM which may affect performance. | Evaluation of the impact of the reduced error on the performance of predictive compression.<br>Implementation on GPU and parallel architecture. |
| Super RLS[34] | Parallel implementation with 12 parallel workers and changing vector length. | ROI selection is a manual and complicated task. | Implementation on GPU and its critical evaluation. |

**Table 2** (*Continued*).

| Algorithms | Advantages | Limitations | Future research directions |
|---|---|---|---|
| C-DPCM-RNN[33] | Three different network structures have been evaluated. | Running time and complexity are not considered.<br>Ineffective for the calibrated image because of similar corresponding spectral lines. | Spectral clustering specific to HS image can be exploited.<br>Train deeper networks to improve accuracy and rate. |
| Li et al.[48] | Complexity reduced by parallel processing.<br>Three different techniques of parallel implementation proposed. | Complex implementation. | FPGA implementation following the same procedure.<br>Optimization of parameters. |
| Afjal et al.[49] | Decreases the computational complexity involved in selection of optimal bands for different predictive compression.<br>Proposed three band reordering techniques.<br>Maximum weighted tree had been constructed using some heuristics that improved the compression performance. | Heuristic work only for the images from similar sensors.<br>Trial and error method is used to obtain the number of segments. | Scaling the heuristic values for sensors with different characteristics.<br>Evaluation of heuristics in other prediction-based approaches. |
| Rodriguez et al.[50] | Data parallelism is obtained on FPGA.<br>Onboard compression architecture utilizing less energy than state-of-the-art method.<br>Flexible for different number of accelerators. | Inherits the drawbacks of lossless CCSDS 123 algorithm such as propagation of error for large size packet. | Can be evaluated for prediction algorithms with better performance than CCSDS. |
| Cang and Wang[51] | Application of compressive sensing technique improves the compression performance.<br>Selection of second band as reference reduces the complexity of the algorithm. | No standard technique for selection of number of groups.<br>Iterative prediction is time consuming process. | Can be combined with state-of-the-art prediction-based compression. |
| Bascones et al.[52] | Achieves 50% of theoretical efficiency.<br>Claimed to be the fastest solution compared to other FPGA architecture. | Consumes more power than previous implementation but lower than GPU. | Reduction of critical path of pipeline.<br>Evaluation by increasing the block size of image. |

on the relative error. Rate control algorithm proposed in the article is a low complexity method that gives the user a control over accepted loss. Finally, a hybrid encoder has been proposed to improve the coding performance. Zhaoa et al.[39] proposed an approach to predict the pixels of HSI. Input image is partitioned into a group of bands (GOB) using segmentation techniques, and intraband prediction is applied to the first band in each GOB to remove the spatial correlation. Rest of the bands are passed with fractal encoding that performs interband prediction using the local search algorithm. Fractal parameters and residual error thus generated are transformed and quantized further using DCT to further remove redundancy in spatial axes. The coefficients are further processed with entropy coder to generate bitstreams. Bogdan et al.[40] evaluated the performance of CCSDS 121 predictor by implementation on various hardware accelerators field programmable gate array (FPGA). Its results can be used to design a requirement-based FPGA for different satellites. Skip block-based distributed source coding (SB-DSC)[41] technique has been used that uses multiple encoders to code different blocks of an image after calculation of absolute error of the image. DWT is applied on the raw input 3-D image to separate the low- and high-frequency pixels, which can then be separately coded based on 3-D SPIHT and transmitted through different channels using Turbo channel coding. Pixels blocks with mean absolute error less than three and maximum absolute error less than four are skipped from coding.

3-D multiband linear predictor (MBLP)[42] algorithm is proposed to identify redundancy in third dimension from the information of already predicted bands using the two algorithms namely 2-D linearized median predictor and MBLP. Residuals are then coded using PAQ8 algorithm in conjunction with arithmetic coding. A compression method based on the binary tree data structure is proposed by Shahriyar et al.[43] that decomposed HSI into similar sized blocks stored in binary tree. Entire block is coded by arithmetic coding to reduce the size of compressed data. Shen et al.[44] proposed a method that removed redundancy in the spatial domain by chaos small-world algorithm. Spectral decorrelation is done with the help of RLS filter in which boundary pixels and normal pixels are processed with a different technique. Multivariate Gaussian distributions encoding is used to generate bit-stream. RLS filter is also used in RLS-optimal prediction band (OPB)-P[35] to predict the pixel values with varying number of bands. The selection of optimal number of bands, to be used for all the bands, is calculated from the spectral signature of first band. The method has been implemented on GPU to reduce the compression time and optimize the intermediary operations. Fjeldtvedt et al.[45] proposed a hardware implementation of CCSDS-123 standard, which performs local sum, local difference, and directional difference beforehand. The dot product of weight matrix and already calculated central difference is explicitly performed on hardware. Residual mapping, encoding, and packing are done in the last stage of compression to reduce the overall complexity of hardware. Barrios et al.[46] proposed a different implementation of lossless CCSDS algorithm on various high-level synthesis tools. The results of each have been compared with existing algorithms and suggestions are provided. Super RLS[34] method includes some steps such as intraband encoding, superpixel segmentation, vectorization, RLS prediction, and entropy encoding. In the first step, spatial correlation is removed from the input image by subtracting the arithmetic mean of neighborhood pixels from each pixel in each band. Then, a segmentation algorithm is proposed to partition the image into small regions based on some similarity, and leading pixels in each area are called superpixels. These superpixels represent the entire block to which they belong, and vectorization technique is used to generate supervoxel for each area. RLS prediction is used to create residuals for each voxel that is entropy coded at the compression end.

Prediction based on data dependency is proposed in long short-term memory-recurrent neural network (LSTM-RNN)[47] algorithm. Since weight of the filters used in prediction is dependent on the previous weights, this method trains a network to learn the time series data obtained in the form of weights. Then, prediction is applied on the input image and context-based conditional average prediction is used to reduce first-order entropy. Following which, adaptive filtering is employed that uses gradient descent algorithm to minimize the residuals. C-DPCM-RNN[33] is another technique that uses neural network to predict the pixels of spectral bands after sufficient training. In the algorithm, different predictors are used to predicting each spectral line. First band is directly encoded and transmitted, from second band to $N$'th band. C-DPCM is used to generate the residuals. $(N + 1)$'th band onward, a trained deep neural network is used, where $N$ is the prediction order selected after obtaining the training accuracy. Li et al.[48] proposed a faster implementation of prediction-based compression C-DPCM on GPU. The algorithm proceeds by clustering of spectral lines into $M$ classes using $k$-means algorithm. Calculation of prediction coefficients for each class using traditional DPCM on GPU, followed by encoding of the residual image. RWA-C[24] already discussed in transformation-based technique can be classified in this category. Afjal et al.[49] evaluated the effect of band reordering on context-based adaptive lossless image coder and lossless CCSDS prediction-based compression algorithm. Band reordering is the technique that rearranges the band sequence in the image to be compressed to code good predictor bands first that will affect the prediction of later bands. Three different approaches used to find optimal band reordering are proposed that are based on various heuristics. Band reordering based on consecutive continuity breakdown heuristics (BRCCBH) obtains the highest correlated bands first and rest bands are arranged in decreasing order of correlation. Band reordering based on weighted-correlation heuristic assigns some weight values to current bands in the reordered list and uses weighted correlation factor initialized either by first band or maximum correlated pair. Based on segmented of bands (BRSB) divides the bands in set of multiple segments using average correlation value, which is followed by BRCCBH for band reordering. Rodriguez et al.[50] proposed another technique for hardware acceleration of lossless CCSDS 123 algorithm. It uses dynamic and partial reconfiguration-based architecture that manages HyLoC, a low complexity compressor core for fast and real-time compression. The number of cores can be modified according to the requirements of application that make it a ready to use hardware platform. Prediction-based reconstruction is proposed by Cang and Wang[51] in the article utilizing compressed sensing and interspectral reconstruction. Similar bands are grouped by correlation factor, and a standard band is selected in each group (second band is generally chosen due to band correlation). Gaussian matrix is used to sparsely represent the standard reference band. General bands are predicted from the reference band itself iteratively by reducing the error until considerable lossy image is obtained. A low complexity predictive lossy compression algorithm is implemented on space qualified hardware accelerator Virtex-5 by Bascones et al.[52] The algorithm is highly pipelined with minimum use of FPGA and multiple steps working together.

*Research challenges and future directions.* Prediction-based compression has several benefits over transform-compression such as low-complexity, better performance for average, and large BR.[30] It supports optimum global results, as an entire image is used at once by the algorithm for prediction. Near-lossless compression can be achieved by prediction compression with the help of a quantizer.[53] This technique has many drawbacks such as low performance, poor fault tolerance, and error propagation, and images are processed only after conversion of 3-D matrix into 2-D matrix that too on the small neighborhood. Performance of existing algorithms can be improved by the development of hybrid algorithms, i.e., combining two or more techniques, including new filters and considering an optimal number of prediction bands. Error mapping with residuals and selection of learning parameters with compressed sensing technique can significantly increase the performance. Modifications in prediction-based HSI compression can lead to an optimal solution for all applications.

### 2.1.3 *Vector quantization*

*Overview.* VQ is a data compression technique that takes 3-D HSI data cube as input and returns a compressed image. Two significant steps of VQ are training (codebook generation) and coding (code vector matching).[54,55] Quantization is mostly used along with transform-based or learning-based techniques as it uses a training algorithm to generate an optimal codebook. Algorithms based on VQ have high complexity, so the principal objective of the method is to develop an efficient algorithm that has fast execution.[56] State-of-the-art algorithms in the field are vector quantization principal component analysis (VQPCA)[57] and online learning dictionary.[58]

*Technique.* Compression using VQ can be divided into three phases:[59] the first phase that generates codebook is called the design phase. The second phase is the encoding phase in which HSI is taken as input and converted to blocks and then to $n$-$D$ vectors. Followed by a search algorithm that is used to find an optimal vector in codebook with minimum distortion and its index is sent to the receiver. Encoding phase can be better understood by Fig. 4. After this third and last phase starts called the decoding phase, the index at the receiver is searched in codebook already present at the decoder end, and code-vectors are regenerated to reconstruct entire image.

A state-of-the-art VQ method was proposed by Li et al.[60] in which input pixels are clustered using correlation vector (CV). Least square residual is then used to predict the spectral bands of each cluster. The residuals are encoded using the concept of VQ with minimal side information. The algorithm has been implemented and evaluated for most of the applications. Báscones et al.[57] proposed a algorithm (VQPCA) based on the concepts of wavelet transform, VQ, dimensionality reduction techniques such as PCA, and standard spatial compression technique JPEG2000. The raw pixels are decorrelated using VQ, which are then passed by PCA to obtain the important components in a few bands. JPEG2000 is applied to the bands with maximum

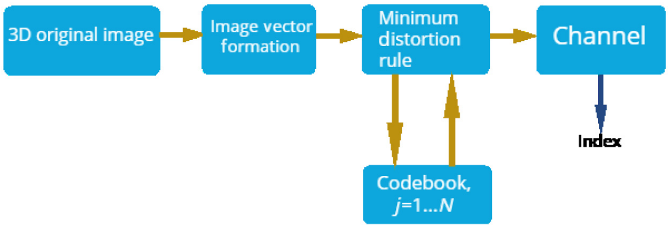

**Fig. 4** Encoding phase of vector-quantization technique.

**Table 3** VQ-based HSI compression techniques.

| Algorithms | Advantages | Limitations | Future research directions |
|---|---|---|---|
| Li et al.[60] | Evaluated for CA and anomaly detection techniques. | The inappropriate process to generalize the number of clusters. | Optimize the process of generation of CV. |
| VQPCA[57] | Speedup of 12% to 16% in execution time. | Parameters are optimized according to a particular configuration. | Nonlinear dimensionality reduction techniques can be applied. |
|  | SNR and CR improve consistently. |  | Optimization of parameters for different applications. |

information and the resultant is transmitted or stored after entropy coding. It results in a lossy image compression technique, which can be extended to near-lossless one after slight modification.

Table 3 presents their advantages, limitations, and future directions.

*Research challenges and future directions.* Some advantages of the technique are near-lossless compression algorithm and better compression performance. It also has some challenges associated with it like the requirement of substantial resources for codebook generation, more processing time[61] to convert a large number of pixels into vectors. Due to seasonal change and atmospheric effects, a single codebook cannot meet the demand of onboard compression and generation of various small codebooks is costly.

### 2.1.4 *Compressive sensing*

*Overview.* The technique is famous for on-board compression algorithms as it shifts the computation complexity of encoder to the decoder. It is used in real-time compression as it senses a small chunk of data, compresses it, transmits the compressed data to the receiver, and then accepts another piece. State-of-the-art algorithms for compressed sensing are sparsification of HSI[62] and reconstruction (SHSIR), reweighted Laplace prior-based HCS (RLPHCS), OMP, and structured sparsity (SSHBCS) that show better performance for small BR. The main objective of compressive sensing is to reduce memory usage during computation.[63] It can also be used as the hardware-based or traditional[64] software-based.[65]

*Technique.* Some algorithms based on compressive sensing are listed in Table 4 for better comparison. Compressive sensing algorithms have the concept of different encoding and decoding algorithms, which can be described using three steps. HSI signals are sensed at the encoder, and very few samples (sensing matrix) of this 3-D image are converted to a 2-D matrix of dimension $N \times B$, where $N =$ number of pixels and $B =$ number of bands. This matrix is converted to a 2-D matrix of much smaller dimension by applying different algorithms. This small matrix is encoded and transmitted through a channel. Then, next part of the same image is sensed, and the

**Table 4** Compressive sensing-based HSI compression techniques.

| Algorithms | Advantages | Limitations | Future research directions |
|---|---|---|---|
| Xu et al.[66] | Low encoder complexity.<br>Prevents error propagation in spatial axis. | Rate distortion not optimal. | Hardware implementation by compressing blocks at the same time due to their independent nature. |
| Gunasheela and Prasantha[62] | Low memory requirements.<br>Low complexity at the encoder side. | Decompression complexity is very high. | Parallel implementation at encoder site. |
| CSDL_JP2[67] | Low power requirements.<br>High operation time.<br>Low complexity at the encoder. | The complex sparse recovery algorithm. | More efficient decompression technique should be incorporated.<br>Parallel implementation of the decompression algorithm. |
| SHSIR[62] | Performance is calculated for different sampling rates and noise level. | Compression time is high. | Optimize for images with a large number of end members. |
| HSI-CSR[68] | Minimization problem is handled by an ADMM technique.<br>Utilizes the concepts of nonlocal tensor sparsity and low rank property over spatial and spectral domain of HSI.<br>Claimed a better reconstruction algorithm based on compressive sensing technique.<br>Provided better results for noise suppression, classification accuracy, and quantitative assessments. | Only considers decompression or reconstruction stage, so cannot be used for the images compressed using different techniques. | Can be implemented for image compression after evaluating the suitable compression technique. |

process is repeated until the entire image is sent to the decoder. The decoder reconstructs all the samples together and thus has high complexity.

An existing technique proposed by Xu et al.[66] divided the input HSI into blocks, with each block having its reasonable bit rate. Multiple linear regression is applied to obtain side information for each one. Optimal quantization step size is assigned, which can help in efficient decompression separately. CSDL-JP2[67] is another state-of-the-art compressive sensing technique in which matrix of measurement code is used to generate a database for coded snapshots. Real-time compression is done by deciding on encoder from snapshot database. A deep neural network is used by the sparse recovery algorithm to regenerate the original image. Gunasheela and Prasantha+[62] proposed SHSIR method with the following steps. Image is first represented in 2-D matrix of size $(P \times B)$ dimension, where $P =$ number of pixels per band and $B =$ number of spectral bands. Then, compressive sensing is applied in spectral axis. Linear mixing model is used to approximate the resultant matrix. Parameters of which are optimized by Bragman iteration. This method was extended by generating spectral vectors for each spatial pixel. The compressed image using SHSIR[62] is modeled by the linear operator, and convex optimization is used with compressive sensing technique to improve the performance. his-CSR[68] is a method that can reconstruct the original values of pixel by sensing a small part of it. The algorithm consists of two stages namely sensing and reconstruction. A random matrix is used to obtain the measurements, which is combined with the parameters and multipliers to get the initial image. Blocking technique is then used to group the tensor cube, its output is $k$-NN classified followed by stacking.

It is followed by reconstruction step where nonlocal similarity and low rank approximation is utilized to regain the original image.

*Research challenges and future directions.* This technique has many benefits as low encoder complexity, smaller memory requirements, low bandwidth for transmission, and better performance. Contrary to this, there are many challenges in the technique such as expensive decompression, identification of the sensing matrix as it should follow isometric, and full rank property.[69] Reconstruction hisHSI[70] is a very complex process involving spectral unmixing and convex optimization. Its use in real-time compression can be a great achievement if bulky computation devices can reduce decoder complexity. Another future dimension opens up due to the fact that most of the applications require synchronized rate of encoder and decoder that has not yet been considered in any article.

### 2.1.5 *Tensor decomposition algorithms*

*Overview.* Tensor decomposition is one of the latest techniques for image compression that gives high performance compared to traditional methods. Tensor could be considered as an $n$-dimensional matrix, which can be decomposed very easily. In this technique, HSI is stored into 3-D tensor ($Y$) and one of the TD techniques[70] is applied to decompose the 3-D tensor ($Y$) into lower dimension tensors ($X$). Decomposed tensor is then encoded and transmitted through the channel. Some state-of-the-art algorithms of the techniques non-negative tucker decomposition (NTD)[8]-DWT, convolution neural network (CNN)[31]-NTD, and NTD-DCT[12] have shown excellent results.

*Technique.* The technique is mostly applied along with some techniques such as transform-based, learning-based, or prediction-based. The steps of the algorithm are described in Fig. 5.

Genetic algorithm-based compression technique has been proposed by Karami et al.[71] named particle swarm optimization (PSO) NTD. The algorithm follows the procedure in which NTD is applied to the original image, which is then combined with the linear mixing model. It generates a smaller tensor with reduced dimensions and three factor matrices that is a temporary decomposition. The optimization problem is solved by applying genetic algorithm (GA) with various parameters and multiple mutations are performed to obtain a final optimized solution. The objective is to minimize the root mean square difference between multiple matrices of input and decompressed image. Rajan and Murugesan proposed a hybrid algorithm DWT-TD-ALS-RLE[72] based on adaptive least square (ALS) and run length encoder (RLE). In the method, 2D-DWT is initially applied on each band across spectral domain to remove redundancy. Coefficients of each band are combined to generate a 3-D tensor, which undergoes TD along with ALS to obtain reduced tensor with minimum error. It is followed by RLE to generate bitstream that can be transmitted or stored in comparatively less bandwidth or memory. The original image can be reconstructed during decompression. Dictionary learning technique is used in tensor decomposition in multidimensional block-sparse representation and dictionary learning (MBSRDL).[73] his is initially represented as a 3-D tensor and three dictionaries are trained using the sophisticated dictionary learning algorithm. Both spatial and spectral domains are compressed separately using TD. Tensor decomposition has also been used along with deep learning technique in CNN-NTD,[31] where CNN-based transform is proposed to transform large-scale

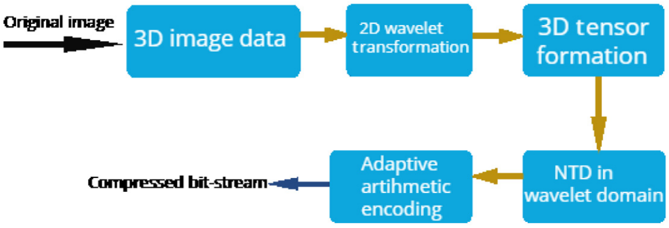

**Fig. 5** Tensor decomposition compression technique.

spectral tensor into small-scale. Then, NTD is applied to further reduce the dimensionality of small scale tensor obtained in first step. Resultant tensor has been transformed into frequency domain using 3-D DCT to remove spatial and spectral correlation. Entropy encoding is used to generate bit streams from the high energy coefficients. Aidini et al.[74] proposed another tensor-based compression in which compressed image is quantized and transmitted to Earth, where it is decompressed and analyzed for processing. Tensor recovery algorithm proposed in this article is an extension of quantized matrix recovery that obtains the original dimensions of the image. It is then passed to super-resolution algorithm that uses coupled dictionary learning to regain the original pixel values. The problem of identification of matrix pertaining to a particular signal is solved using alternative direction multiplier method (ADMM). A CNN network is proposed to learn spatial features from high-resolution images that obtain remarkable results. Critical analysis and future scope of the tensor-based technique are listed in Table 5.

*Research challenges and future directions.* These algorithms have high compression performance with reduced run-time but it suffers from many limitations. High computation complexity, manual parameter updating procedure, data dependency, etc. are some of the challenges in the method. The future scope in this technique is to utilize the parallelism existing in the algorithm for parallel implementation. It can also be extended to automate the process of selection of dimensions to compress image at a particular rate. Also, more hybrid algorithms combining tensor-based compression and other such techniques can be developed.

**Table 5** Tensor decomposition-based HSI compression techniques.

| Algorithms | Advantages | Limitations | Future research directions |
|---|---|---|---|
| PSO-NTD[71] | Application-oriented compression | Fast sub-NTD requires many end-members, so it cannot be applied to radiance data. | Parallel implementation of the algorithm. |
| DWT-TD(ALS)-RLE[72] | Less complexity with the use of ALS. | High memory consumption and the processing time. | Development of RLE compression algorithm. |
| | Different PSNR value for different bands of HSI. | | Simplifications of tensor calculations. |
| MBSRDL[73] | Retains the structural features of the image to a more considerable extent.<br>Fast computation speed under scarce resources. | Not suitable for high sampling rats. | Apply different TD methods to improve efficiency. |
| CNN-NTD[31] | Low-complexity even though it uses CNN and NTD. | Implementation of the proposed NTD is a difficult task. | Parallel implementation of the learning algorithm to reduce complexity. |
| | | | Combining the Distributed source coding scheme with CNN. |
| | | | Training and constructing more complex CNN for compression. |
| Aidini et al.[74] | Proposed method addresses the problem of multilevel quantization in classification.<br>Overcomes the limitations of compression to improve the classification accuracy. | Application-oriented compression and reconstruction. | Method can be generalized for applications other than classification. |
| | Can reconstruct the original real values of pixels. | | |

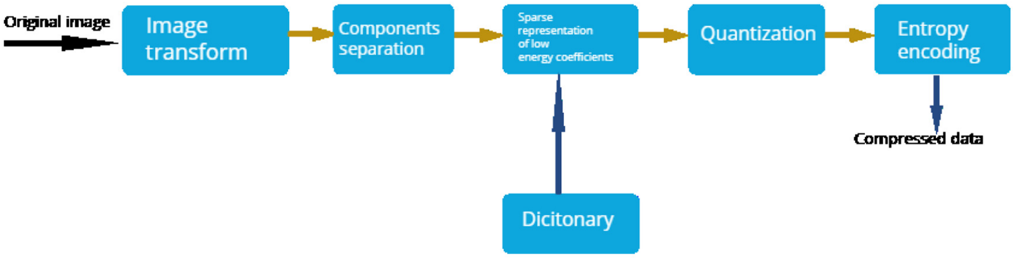

**Fig. 6** Sparse-based technique.

### 2.1.6 *Sparse representation algorithms*

*Overview.* The technique compresshisHSI using very few values from a range of pixel values using some quantization method and values near to zero are dropped. It helps to reduce the use of storage and bandwidth by coding only a small set of values. It is mostly used in the classificatihisof HSI as features could be separated by a distinct boundary when sparse representation is used. The technique helps in the ROI-based compression when combined with learning-based compression. Some state-of-the-art algorithms in the technique are compressive-projection principal component analysis,[75] GIST, SpaRSA,[76] spectral–spatial adaptive sparse representation (SSASR),[77] and TwIST.[76]

*Technique.* Sparse coding is used by multiple algorithms in different style,and a generalized and in-depth description of the technique is given in Fig. 6. First step of the algorithm is vectorization in which pixels with different features are mapped and converted to vectors. Next step is sparse coding where these vectors are converted to sparse vectors, which are then encoded into bit-streams. Algorithms based on sparse representation within the scope of this article are described below. Table 6 provides their advantages, limitations, and future research directions. SSASR method was proposed in 2017 for transformation of spectral signatures of pixels to sparse coefficients, most of which are zero. Superpixels are obtained from the image to divide a large size image into multiple blocks of small size, which are converted into vectors of equal size. Adaptive sparse coding is then applied to generate sparse coefficients. These pixels are quantified by discrete quantization, which are then encoded by Huffman coding to generate the bitstream. Jifara et al.[77] proposed a method based on the spectral curve, which is unique for different materials. Spectral curve is described by sparse dictionary that gets updated using the concept of online learning. It is a lossy compression technique that uses the proximity-based optimization technique. Online dictionary learning[58] reduces the time and cost associated with large size of dictionary coding and transmission, and it learns iteratively by selecting one item

**Table 6** Sparse representation-based HSI compression techniques.

| Algorithms | Advantages | Limitations | Future research directions |
|---|---|---|---|
| SSASR[77] | Optimal execution time.<br>SNR depends on many superpixels. | Inadpative for different images. | The process to select many superpixels is needed in the algorithm. |
| Fu et al.[78] | Loss in spectral information is minimal, compared to other algorithms. | Not suitable for small BR. | Quantization parameter should be changed to obtain better results at small BR. |
| Online learning[58] | Spectral clustering method.<br>Better performance than other algorithms. | Only compression performance is considered. | Evaluation of complexity of the algorithm.<br>Reduce the existing complexity. |
| SpaRSA[76] | Application-oriented performance evaluation. | Fails for anomaly detection at 0.1 bps rate. | Parallel implementation of the algorithm. |

from training set. An advantage offered by this method was sparse representation of the spectral curve of blocks of pixels. SpaRSA and GIST proximity-based optimization algorithms have shown the optimum results for the purpose of anomaly detection.

Fu et al.[78] clustered original pixels into general-pixels represented by simultaneous sparse coding, which gives only nonzero coefficients. Coefficients are quantized using some threshold value that is a bit rate deciding factor. Quantizer gives the control to user to decide the quality of the reconstructed image by modifying the bit rate. Quantized coefficients are further compressed by DPCM filter and converted to binary bitstream by Huffman coding. Another state-of-the-art algorithm for sparse representation algorithm is SpaRSA[76] that has the following steps. One element from the training set is taken at a time to update the dictionary of pixels. Sparse representation is used to store the coefficients with a loss function of optimization algorithm. Dictionary update and dictionary learning are the two algorithms used to minimize the loss function for application-specific compression. CSDL-JP2[67] categorized under compressive sensing technique is also an example of sparse representation algorithm that has very high computational complexity.

### 2.1.7 *Multitemporal compression algorithms*

*Overview.* Multitemporal HSI is a set of HSIs[79] collected from the same location at a different time. A new temporal (or time) domain gets added to the original 3-D image matrix, forming a 4-D matrix. It can be thought of as video compression, but the concept of video and multitemporal is entirely different. Compression of the 4-D image[80] is called multitemporal compression or 4-D compression, which is very important for military operations, disaster management, prevention from calamities, space observation, etc.

*Technique.* 4-D HS image with details is given in Fig. 7, where $(x, y)$ and $z$ represent spatial and spectral domains, respectively, and $T$ represents time domain. All four parameters are variable in a 4-D image.

Multitemporal compression[81] is obtained by enhancing the 3-D prediction-based technique to 4-D prediction based for temporal decorrelation. Lossless compression is expected in 4-D images as these are processed and used by automated programs running on the computer. Methodology of some state-of-the-art algorithms is presented below. Table 7 provides their advantages, limitations, and future research directions. Zhu et al.[79] proposed a compression algorithm applicable on temporal HSIs using the concept of change detection. A reference image is selected among the multitemporal images that should be present at the decoder end. Matrix operations are used to detect change in temporal domain of images with respect to the reference image. It suggested three techniques for efficient compression of 4-D images starting from likelihood ratio of the detected to spectral concatenation and independent approach. Two temporal HSIs are concatenated on the spectral axis forming a single HSI with twice the number of bands as original in the spectral concatenation method. Independent approach ignores the reference image at all during spatial decorrelation. In all the three techniques, PCA, SubPCA, and

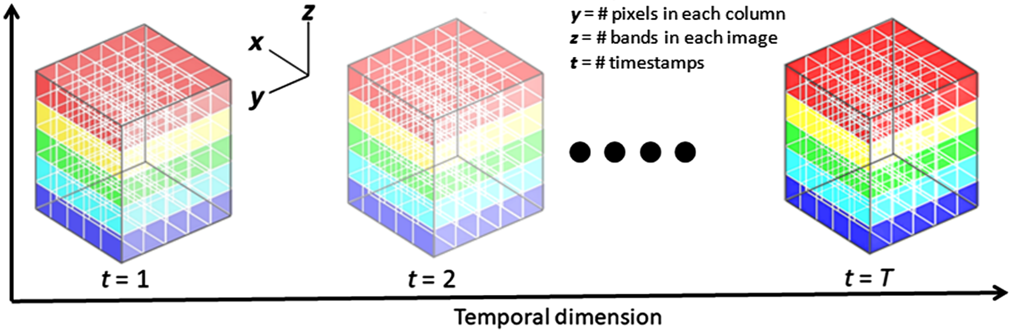

**Fig. 7** Multitemporal HSI.

**Table 7** Multitemporal-based HSI compression techniques.

| Algorithms | Advantages | Limitations | Future research directions |
|---|---|---|---|
| Zhu et al.[79] | Change removal approach preserves temporal changes with high fidelity. | PCA and sub-PCA are used which have high complexity. | Different spectral transforms techniques can be applied. |
| CLMS[36] | Better prediction in terms of lower bit rates. | Performance improvement, due to many bands and small BR, decays fast. | Hardware implementation of the algorithm. |
| Fast lossless 4D predictor[81] | Exploits temporal redundancy.<br>A method of data collection by SOC700 HS camera. | Very less publicly available data. | Extendable for real-time compression. |

DWT spectral transforms are applied on the images to improve the compression performance by reducing the number of bands and spatial data to be coded.

Shen et al.[36] proposed an adaptive learning-based compression of multitemporal HSIs. It used correntropy least mean square (CLMS) algorithm for prediction of pixels on the basis of already predicted spectral and temporal information. The performance of the method is comparatively high due to the presence of temporal correlation. It ensures lossless compression by coding the prediction error using Golomb rice coding and arithmetic coding. The method was further improved with the help of fast-lossless-4D predictor[81] in which pixels are predicted by using a linear combination of neighboring pixels in all four dimensions. By subtracting data with their local mean in each band and then for each pixel in a band in raster scan order. Taking its residual and finally updating the weight parameter. Residuals are encoded by entropy coder at the compression end and the decompression follows an exact reverse of the compression steps.

*Research challenges and future directions.* Implementation of 4-D compression algorithms is quite easy as traditional 3-D algorithms are extended to the fourth dimension. Complexity that includes running time and computation resources is very high for these algorithms. Future scope of the technique is to develop hybrid algorithms combining prediction-and learning-based techniques. Performance of transform-based technique can be evaluated in the temporal domain.

### 2.1.8 *Learning-based algorithms*

*Overview.* It is one of the most popular techniques as it involves machine learning and deep learning in compression. The method has always been studied along with prediction-based technique as it also predicts pixel values. However, it has widely known features that learn and update parameters automatically. It[82] is used with all other techniques after little modifications and achieves better performance. Some acclaimed machine learning algorithms applicable in the technique are SVM,[13] artificial neural network (ANN),[83] backpropagation network,[84] CNN,[31] independent component analysis (ICA)/PCA,[85] and clustering algorithms.[16]

*Technique.*

- Comprehensive analysis of the algorithms is done by describing their methodologies. ANN-based algorithm was proposed for compression of HSI by Masalmah et al.[83] that follows the given steps. The original image was first divided into sub-blocks, and corresponding column vectors are arranged into matrices. A neural network is proposed with input, hidden, and output layers having neurons corresponding to the size of sub-blocks. Output matrices of the hidden layer and output layer are calculated after simulation of weight and biases. Input layer contains the original image whereas the output layer contains decompressed image. A hidden layer with a smaller number of neurons contains compressed image. Postprocessing is done to obtain the decompressed image. Jiang et al.[86] proposed a deep belief network (DBN) for estimation of parameters of Golomb rice coding algorithm used to compress HSI. DNN is used to select the best coding parameters for compression by treating it as a pattern classification problem. DBN is trained, and then the real estimation of parameter starts. At last, Golomb rice coding is used to encode the image. In 2019, another method based on neural network was proposed in block-based interband predictor[84] using multilayer propagation neural network (BIP-MLPNN). Each band is converted into a matrix of dimension (256 × 256), with values mapped from [0, 1] after preprocessing. Input the image modeled by hidden and output layers using tansig and purelin as transfer function, respectively. Modify weight, bias, to minimize the error and finally encode the bias, weights, and residuals. Decoder network is the same as encoder followed by normalization and reverse mapping. C-DPCM-RNN[33] and LSTM-RNN[47] discussed in prediction method is an example of use of neural network in compression.

- The use of CNN in compression of HSI can be observed in the algorithm named prequantization.[87] The input image is compressed by a lossy predictor that uses the CCSDS standard along with quantization of raw pixel data before prediction. The residuals thus generated are coded by entropy coding. At the decompression end, CNN is used to reconstruct the original image with some induced loss. Network is trained with some data different from the original image, with the predefined constant learning rate. The performance of the decompression can be significantly improved along with the decrease in time due to pretrained network. CNN-NTD[31] algorithm discussed in tensor decomposition method can also be categorized under learning-based compression due to the use of deep learning (CNN) in its first step. An application-oriented compression has been proposed by Sujitha et al.[88] that used the concepts of Lempel Ziv Markov chain algorithm (LZMA) coder to generate the bitstreams. CNN-LZMA is the algorithm that learns to generate the compact representation of the raw 3-D image. Reduced dimensions are subsequently coded with LZMA, an enhancement of Lempel Ziv Welch (LZW) coder. Reconstruction is obtained by LZMA decoder followed by residual learning to train the CNN at decoder side, which reverses downsampling to regain the original image. 3D-DCT-SVM,[13] PCA-DCT,[17] folded PCA,[18] WPCA,[19] and Giordano and Guccione[16] discussed in transformation method is an example of use of machine learning in compression. Similarly, VQPCA[57] from VQ method and CSDL_JP2[67] from compressive sensing can also be categorized under learning-based compression.

Table 8 provides the advantages, limitations, and future research directions of these algorithms. Figure 8 shows general steps of learning-based algorithm using CNN and some transformation function to compress an his. In the first step, CNN is applied to compress 3-D data cube representihisHSI. Then, transform-based algorithm does domain transformation of a smaller 3-D cube and then coefficients are encoded. In backward CNN, original image is reconstructed by applying CNN again with minimal error.

*Research challenges and future directions.* The method sustains high complexity of machine learning and deep learning algorithms. It also requires more resources but can be easily implemented in hardware and other HPC architecture. The method can be improved by implementing the fundamentals of deep learning and advanced machine learning in compression and developing more hybrid algorithms with automated processes.

## 2.2 *Categorization Based on Various Parameters*

In this category, the algorithm is classified based on six parameters such as the loss associated with compression, the platform where compression is performed, ROI capability, application of compression, strategy to start compression process, and implementation environment. These parameters are selected for categorization since the process directly or indirectly depends on it.

### 2.2.1 *Based on the output of the compression algorithm*

It determines the quality of the image obtained after compression. The output of the algorithm is the most important factor that depends on task to be performed on the compressed image. Some

**Table 8** Learning-based HSI compression techniques.

| Algorithms | Advantages | Limitations | Future research directions |
|---|---|---|---|
| ANN[83] | Different training techniques are considered with their performance. | Effect on quality due to unmixing of HSI. | Spectral decorrelation can be applied before this algorithm. |
| DBN[86] | Simulation is performed on synthesized data.<br>No assumption (such as geometrical distribution of data) is considered. | Computational complexity is high. | Implementation of modified DBN as proposed in the article.<br>Estimation of correlation in generated features. |
| BIP-MLPNN[84] | Residuals are not coded using entropy encoder. | Poor performance than CCSDS in low variance data. | Use of the deep neural network to improve the accuracy of the predictive model. |
| Prequantization[87] | Faster than the lossy version of CCSDS standard. | Train and test dataset are strictly disjoint. | Applying the same algorithms to different sensors data such as ultraspectral sensors. |
| CNN-LZMA[88] | Real-time data generated at a speed of 10 to 20 Mbps can be encoded in real time by LZMA.<br>Claimed to be executed in scarcity of resources. | Large size dictionary reduces the performance.<br>Focuses on the structural information existing in the image. | Execution on real-time hardware and applications other than industrial IoT.<br>Modification of hyperparameters of the CNN model. |

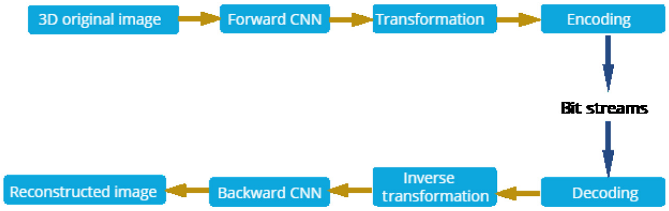

**Fig. 8** Steps of CNN-based compression technique.

pixels lose their original value during the process and some error is induced in them. Quality of resultant image is inversely related to the induced error. There are three methods of compression based on the quality of the reconstructed image.

*Lossy.* Lossy compression can be defined as the process of compression in which the original image cannot be restored in the reconstructed image. Some information is discarded at the compression end, which cannot be recovered during decompression. It is mainly used when the application is error-tolerant, i.e., specific loss in data has no effect on the output. It results in high compression performance by reducing the size of the compressed image and thus maintains a trade-off between space and precision. Lossy compression is dominated by hardware implementation.[89]

*Lossless.* It consists of the techniques that can precisely reconstruct the original image without any loss of information. Used in applications where even small loss in error is not acceptable such as military operations, global positioning system (GPS) tracking, and target identification. Lossless compression[90] results in reduced compression performance, i.e., CR. In HSI compression techniques, lossless compression is preferred as these images store important information that is used in the analysis, classification, target identification, etc.

*Near-lossless.* The term has been used interchangeably as “controlled-lossy” compression, which means a loss in information can be controlled according to desired compression performance. It can be understood as a fuzzy set between lossy and lossless compression that changes its form according to the application. It can be used in medical imaging, remote sensing, etc. Moreover, very few works have been done in this field, and the area needs more exploration.[91]

### 2.2.2 *Based on the platform where compression is performed*

In remote sensing, location of data acquisition and data processing is generally different, and the availability of resources at both locations differs. Images are acquired through satellites, flights, drones, camera mounted at an altitude, etc., where memory devices and processing units are limited. Sensors, installed in devices, capture signal and immediately transmit it to the receiver, which is mostly at the ground or in space (rare case). In this work, the receiver can be understood as a data center, which has all the resources such as many CPUs, GPUs, and unlimited memory devices. Resource availability affects the performance and execution of the algorithm, and it can be classified into two categories as onboard compression and data-center compression.

*On-board compression.* Compression performed on raw signal/images at the source of acquisition is termed as onboard compression. Satellites or air-borne devices can carry minimal resources that too can be affected by radiation. So these algorithms are devised to perform in resource constraint environment.

*Data center compression.* HSIs are compressed at receiver, where the resources are available in bulk. Algorithms executed in this environment need not suffer from scarcity of resources and thus have better performance.

### 2.2.3 *Based on the region of interest capability*

Performance of an algorithm can be significantly improved if a small part of the same HS image is compressed rather than the entire image as the performance is inversely related to the size of the image. Compression algorithms can be classified into two categories on this basis.

*Region of interest-based compression.* ROI-based compression can be stated as the technique in which part of the image is compressed with high BR and remaining portion with a small BR. A portion of the image containing vital information is identified in the first step of such algorithms. There can be several such parts that can be compressed with the same algorithm but different BR depending on the significance of information stored in it.

*Full image compression.* It is a technique in which the entire image is compressed with the same BR, and no target identification is needed in advance. Performance of these algorithms is not as good as the performance of ROI-based algorithms.

### 2.2.4 *Based on the application*

Compression algorithms can be classified on the basis of application in two categories: transmission and storage. In either case, the output varies and steps too. The algorithm should be developed according to the purpose it has to serve to obtain better results.

*For transmission.* Compression performed to transmit signal at some other location requires a stream of bits along with header and side information. The algorithm developed to generate a stream of bits that should not be used to construct compressed image to preserve time and resources.

*For storage.* The compressed data are stored for future use, which can be reconstructed to the original image when needed. Increased steps in these algorithms qualify it to be classified into a different category.[92]

### 2.2.5 *Based on the strategy of compression*

HSI compression algorithms can be classified into two categories based on how they consider an original source image. This classification helps to identify the nature of the compression algorithm and its steps. Basically, HSI is a 3-D tensor, but some algorithms transform it into a 2-D array and then perform operations. While some of them consider it as 3-D cuboid and directly compress it.[14]

*3-D data cube.* Algorithms falling in this category consider HSI in 3-D cube or cuboid and directly apply steps of compression. Spatial and spectral decorrelation need not be performed separately in this case.

*2-D compression.* Some algorithms can only be applied only on 2-D array, so they first convert 3-D-HSI to a 2-D array. There are two approaches for the conversion, in the first approach, each 2-D band (of size $p$, $q$) is converted to 1-D vector (of size $p \times q$) in raster scan order and each band is appended columnwise. The second approach is to remove correlation in spectral dimension and apply 2-D compression algorithm in each band, considering each band as a separate image.

### 2.2.6 *Based on the implementation environment*

Compression algorithms can be implemented in two environments, i.e., sequential and parallel implementation. These algorithms are categorized on the basis of run-time as sequential implementation has more run-time than its parallel counterpart. All sequential algorithms cannot be implemented in a parallel environment due to the design issue.

*Sequential implementation.* These algorithms do not require any specialized hardware or machine for implementation. They get executed with high run-time on a regular machine.

*Implementation on HPC architecture.* Algorithms having independent blocks (that can be executed concurrently with other blocks) is implemented on HPC architecture with reduced run-time. There are three types of architectures on which algorithms are executed with little modifications.

Shared/distributed memory: More than one threads or processors or CPUs are used in the execution of the algorithm to reduce the time complexity. It uses some software libraries such as open multiprocessing (OpenMP)[93] and message passing interface to get the benefit of shared and distributed memory architecture.[94]

FPGA: FPGA is specialized electronic hardware that is used to run algorithm in less time with improved performance. There are many FPGA devices available,[95] which can be selected based on their performance and need in compression.

GPU: GPU is an electronic circuit that is used to implement compression algorithms after a small modification. It also improves the complexity and speeds up the execution by performing CPU operations at very high speed.[96] It has many cores and inherent characteristics of parallelism, with a considerable overhead of communication between CPU and GPU.

An HSI compression algorithm may come under more than one category according to its features. Table 9 classifies algorithms according to these categories, where some abbreviations are used. Compression performed at the location of acquisition and at the data-center is represented by onboard and DC, respectively. ROI represents the ability of the algorithm to support ROI-based compression. Algorithms considering input image as 3-D data cube or transforming 3-D tensor to 2-D matrix for compression are represented under the strategy. Implementation environment where algorithm is implemented is sequential implementation (seq), shared memory/distributed memory (SM/DM), GPU, and hardware accelerators such as FPGA. This categorization helps us to find out the scope for future research according to exploration level.

**Table 9** Categorization of compression algorithms based on various parameters.

| | Compression type | | | Platform type | | ROI | Application | | Strategy | | Environment | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Algorithms | Lossy | Loss-less | Near lossless | On-board | DC | | Tran-smit | Store | 3-D | 2-D | Seq | SM/DM | GPU | FPGA |
| Hybrid 3D-DCT-TD[12] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ |
| DWT-TD[8] | ✗ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| 3D-DCT-SVM[13] | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ |
| JPEG-LS[14] | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| Kozhemiakin et al.[15] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| Giordano and Guccione[16] | ✓ | ✗ | ✗ | ✗ | ✓ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ |
| HyperLCA[20] | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| PCA-DCT[17] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ |
| Integer HyperLCA[21] | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| Wang et al.[10] | ✗ | ✓ | ✓ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ |
| Folded PCA[18] | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| Luminance transform[22] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ |
| WPCA[19] | ✓ | ✗ | ✗ | ✗ | ✓ | ✓ | ✗ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ |
| FrWF[23] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✗ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ |
| RWA-C[24] | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| D-LMBTC[11] | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |

**Table 9** (*Continued*).

| | Compression type | | | Platform type | | ROI | Application | | Strategy | | Environment | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Algorithms | Lossy | Loss-less | Near lossless | On-board | DC | | Tran-smit | Store | 3-D | 2-D | Seq | SM/DM | GPU | FPGA |
| IKLT-IDWT[25] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| Diaz et al.[26] | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ |
| SVR-DWT[27] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ |
| GFT[28] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ |
| Fuzzy transform[29] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| Bogdan et al.[40] | ✗ | ✓ | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ |
| SB-DSC[41] | ✗ | ✓ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| Conoscenti et al.[38] | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| Zhaoa et al.[39] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| 3D-MBLP[42] | ✗ | ✓ | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| Binary tree-based decomposition[43] | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| Shen et al.[44] | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ |
| RLS-OPB-P[35] | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ | ✗ |
| Fjeldtvedt et al.[45] | ✗ | ✓ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ |
| Barrios et al.[46] | ✗ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ |
| LSTM-RNN[47] | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| SuperRLS[34] | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✗ | ✗ |

**Table 9** (*Continued*).

| | Compression type | | | Platform type | | ROI | Application | | Strategy | | Environment | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Algorithms | Lossy | Loss-less | Near lossless | On-board | DC | | Tran-smit | Store | 3-D | 2-D | Seq | SM/DM | GPU | FPGA |
| C-DPCM-RNN[33] | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| Li et al.[48] | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ |
| Afjal et al.[49] | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| Rodriguez et al.[50] | ✗ | ✓ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ |
| Cang and Wang[51] | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✗ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ |
| Bascones et al.[52] | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ |
| Li et al.[60] | ✓ | ✗ | ✗ | ✗ | ✓ | ✓ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| VQPCA[57] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| Xu et al.[66] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| Gunasheela and Prasantha[62] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| CSDL_JP2[67] | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| SHSIR[62] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| HSI-CSR[68] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✗ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ |
| PSO-NTD[71] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ |
| DWT-TD(ALS)-RLE[72] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| MBSRDL[73] | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ |
| CNN-NTD [31] | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ |

**Table 9** (*Continued*).

| | Compression type | | | Platform type | | ROI | Application | | Strategy | | Environment | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Algorithms | Lossy | Loss-less | Near lossless | On-board | DC | | Tran-smit | Store | 3-D | 2-D | Seq | SM/DM | GPU | FPGA |
| Aidini et al.[74] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ |
| SSASR[77] | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✓ | ✗ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ |
| Fu et al.[78] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| Online learning[58] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| SpaRSA[76] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| Zhu et al.[79] | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ |
| CLMS[36] | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| Fast-lossless-4D predictor[81] | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| ANN[83] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| DBN[86] | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| BIP-MLPNN[84] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| Prequantization[87] | ✗ | ✓ | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| CNN-LZMA[88] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ |

## 3 Discussion

This study is focused on the recent techniques in the first place, thereby limiting the number of papers to 63 and omitting others to give a fresh outlook of the problem. The perspective of this work is limited to remote sensing applicability of HSIs, excluding the algorithms used for compression of images of other domains, say, medical, food processing, security, etc. The summary of traditional techniques used before 2006 can be obtained in the book edited by Motta et al.[97] It contains the detailed analysis of various lossless and near-lossless compression techniques including prediction-based, transform-based, and VQ-based. Sanjith and Ganesan[98] presented a review of HSI compression algorithms focusing on the methodology. It considers the techniques that can be used for onboard compression of HSI without categorizing the algorithms in detail. An analysis of the algorithms based on statistical or wavelet-based techniques is presented by Babu et al.[99] The perspective of the review is strictly based on the results obtained from the algorithms, and it also considers the standard techniques used for video compression. Majority of the algorithms have their focus on onsite compression to reduce the transmission overhead. Dusselaar and Paul[100] summarized the available literature with experimental data on the datasets. The focus of the survey was limited to specific processes based on inter- and intra-band compression and different coding techniques. A study of satellite image compression techniques is presented by Gunasheela and Prasantha.[101] It provides a quantitative analysis of the algorithms with respect to evaluation metrics such as complexity, peak signal-to-noise ratio (PSNR), error, bitrate, and CR. A comprehensive study of compression techniques focusing on the medical images is given by Hussain et al.[92] It presents summary of various algorithms along with limitations and compression rates. A review of the lossless compression techniques based on FPGA implementation is made available by Rusyn et al.[40] It mentioned the recommendations for the development of onboard compression hardware along with advantages and disadvantages of each technique.

The modern compression standard used by satellites in space mission is developed by CCSDS-123.0-B-2[102] for compression of HSI. It used a closed-loop quantization scheme providing low-complexity near-lossless compression performance. It gives user the capacity to control the compression rate by predeciding the values of relative and absolute error. Performance is slightly decreased due to unavailability of the original data samples at the decompression end, and prediction is made possible only with the help of representative pixel values.

In this review, we divided major algorithms into eight different categories depending on the similarities and dissimilarities, discussed as a part of their definitions. Figure 9 shows a chart showing the frequency of algorithms in various categorization types. It can be observed that most of the techniques fall into transform-based compression. The reason for the same being the extension of traditional 2-D compression. This is followed by prediction-based compression, which is the most favorable technique for HSIs, and it provides optimum performance in terms of CR and BR. Few algorithms in VQ technique are due to the fact that it is only applied in combination with other methods providing exclusive benefits to existing ones. Multitemporal compression includes only three algorithms as its development is still in the nascent phase.

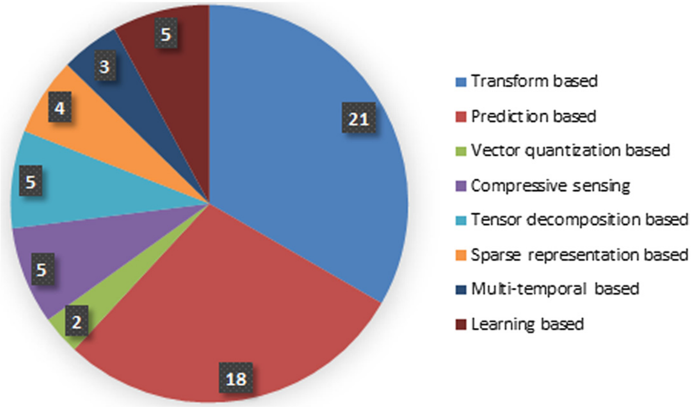

**Fig. 9** Chart representing number of algorithms in different category.

Learning-based compression is used very often due to the features of machine learning and deep-learning techniques. It also provides optimum performance in terms of application-specific parameters such as classification accuracy (CA), cluster metrics, and anomaly detection.

The analysis of Table 9 concludes that parallel and hardware implementation of compression algorithm can be explored as it has many benefits such as reduced computation time, reduced computation power, improved performance, etc. Algorithms under ROI-based compression are very few though they provide better BR and compression performance. Similarly, researchers have not much focused on near-lossless compression algorithms as they are application dependent and require clear objectives at the initial stage. This classification can help to understand any HSI compression algorithm better and work on application-specific compression.

It can be observed from Fig. 10 that majority of the algorithms have used PSNR as the first metric for evaluation. Quality of image is an essential factor for compression algorithms, which is calculated by PSNR. Size of the compressed image is calculated in the form of number of bits used to store a pixel multiplied by total number of pixels. Second important metric used to compare the performance of algorithms is BR that represents the number of bits processed per unit time. It can also be observed that 13% of algorithms use CR as a metric to measure the performance of an algorithm by obtaining a ratio of the size of decompressed image to size of compressed image in bits. SNR is also used by most of the algorithms followed by CR that represents the amount of information present in the reconstructed image compared to the presence of noise/error. As mentioned in the previous section, time taken in the process of compression and decompression is a crucial point to be taken care in majority of the applications. Execution time is used by 10% of the algorithms as an evaluation metric, which is followed by mean square error (MSE). HSIs lack psychovisual properties and are used by particular applications such as classification, anomaly/target detection. So, another metric used to analyze the quality of decompressed image relies upon under classification parameters, such as CA, ROC curve, etc. Absolute and relative errors and structural and spectral parameters are also used by few algorithms. Parallel compression algorithms use speedup, throughput, power, and compression time to evaluate the performance. It can also be concluded that few important metrics have not been considered by these algorithms, such as edge preservation index (EPI). EPI is a measure of the number of edges preserved in the image reconstructed after compression. Although it is widely used in medical images, it can improve the quality of HSIs also by maintaining the minute details.

An analysis of the datasets used by the techniques considered in this review is shown in Fig. 11. The detailed table with each algorithm and dataset can be found in the supplementary section, Annexure 1. Public dataset in the diagram is a combination of HSIs obtained from different sensors and available in public domain. Various sample images are calibrated and uncalibrated Yellowstone scenes from AVIRIS sensor, Cuprite, Jasper Ridge, Moffett, Washington DC, Lunar Lake, low altitude, and CCSDS standard HSIs. It also includes Indian Pines, Salinas,

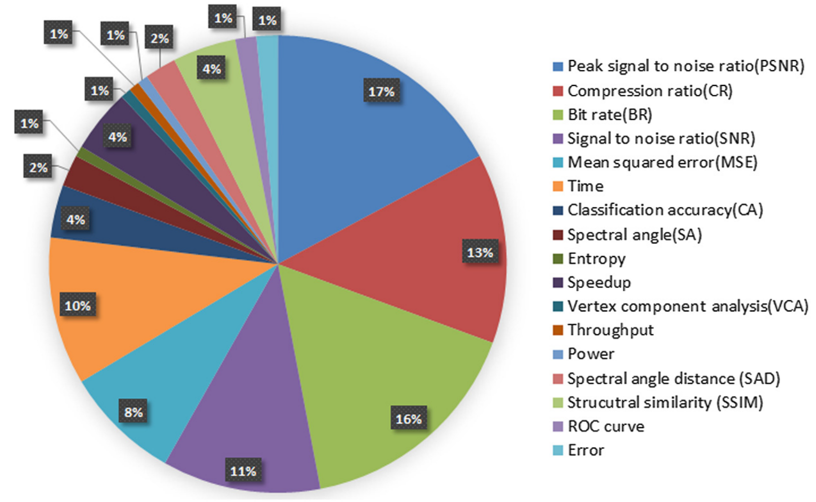

**Fig. 10** Statistics of metrics used by algorithms.

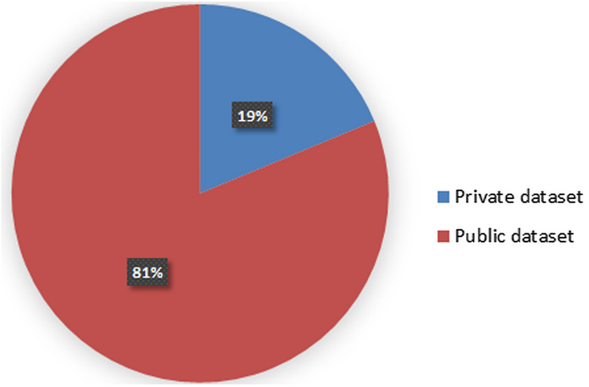

**Fig. 11** Statistics of datasets used by algorithms.

Kennedy Space Center, Pavia Centre and University, Urban data scenes, Levada sequence, etc. Datasets from HyperION, HICO, Modis, HYDICE sensors, and multispectral images of various buildings, cities, and mountains. 81% of the algorithms have used these publicly available dataset, which makes the validation easy. Results can be verified by implementing the algorithms in the same environment on the same datasets as supplied by individual authors. Similarly, it can be analyzed that 19% of the techniques have used self prepared HSIs for evaluation of their algorithms. These datasets may or may not be available in public domain. Some other datasets used by HSI compression techniques are aerial view of the Suwanee natural reserve (SUW) with a size of 360 × 320 × 1200, Beltsville crop fields, Maryland, (BEL), whose size is 360 × 320 × 600, satellite view of Reno, Nevada (REN), size 356 × 320 × 600, an image of the Cuprite Hills (CUP) of size 188 × 350 × 350, scene of Hawaii, (HAW) with 614 bands and image from Maine, (MAI) with 680 bands, and seven images from Yellowstone National Park (YEL) each of size 224 × 512 × 677, where values indicate number of rows, number of columns, and number of channels, respectively.

HSI compression algorithms considered in this review address the issue of the large size of the image by reducing the size. While accomplishing the main objective, many challenges are observed such as compression time, scalability, flexibility, resource usability, etc., that create differences. These challenges provide an unfilled spot that should be filled by following future research directions as discussed. The majority of algorithms do not consider the decompression phase that could address the problem of reconstruction. Some challenges are summarized as:

- Minimal research is done in developing parallel HSI compression algorithms which have reduced complexity in other domains.
- Several improvements are required in 4-D image compression, which is in its nascent phase.
- Different TD techniques, other than NTD, can be applied to reduce the size of an HSI data cube.
- With the advent of deep learning techniques, the performance of learning-based algorithms can be improved to a significant extent.
- Performance improvement in near-lossless compression.
- Less priority given to real-time image compression and its analysis.
- Minimal availability of application-oriented compression algorithms.

We propose an adaptive framework for HSI compression to overcome certain limitations described above. Design criteria for the remote sensing application may be enlisted as:

(a) Selection of the compression technique as per the requirements of application.
(b) FPGA-based methods for real-time compression.
(c) More emphasis on parallel HSI compression.
(d) Extension of 3-D algorithms to temporal domain for multitemporal HSIs.
(e) Use deep learning methods for performance improvement.

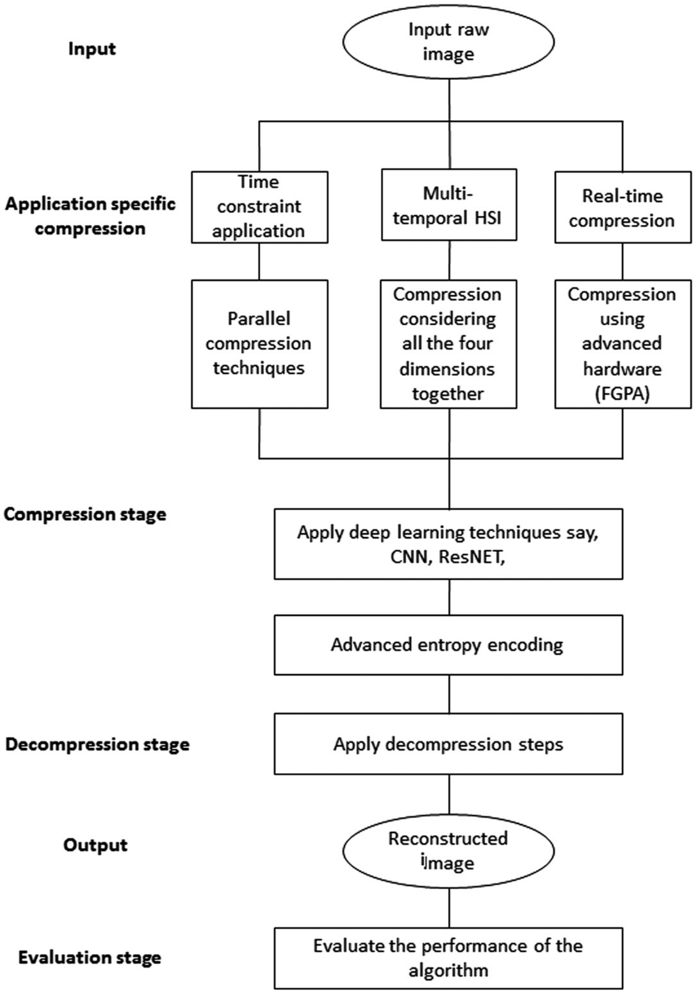
Input
Input raw image
Application specific compression
Time constraint application
Multi-temporal HSI
Real-time compression
Parallel compression techniques
Compression considering all the four dimensions together
Compression using advanced hardware (FGPA)
Compression stage
Apply deep learning techniques say, CNN, ResNET,
Advanced entropy encoding
Decompression stage
Apply decompression steps
Output
Reconstructed image
Evaluation stage
Evaluate the performance of the algorithm

**Fig. 12** Suggestive general framework.

Application-specific compression provides better performance as compared to general techniques. Figure 12 shows a suggestive general framework for such compression standard, which can be used to improve the quality of reconstructed image along with the compression algorithm. Particular methods for each design criteria can be developed as a part of the future work of the study. The evaluation stage has been included as the last step to decide the quality of observed image at the decoder.

## 4 Conclusion

Reduction in the image size is the basis of the development of the compression algorithm since it gives many benefits to the HS analysis. A large dataset is required to validate the results of such algorithms. Most of the HSIs used by the researchers are available as open-source and others at a nominal charge. The review also helps to gain theoretical knowledge about the data source. We have also categorized algorithms on the basis of parameters that could help to decide the scope, objective, implementation environment, scalability, and strategy of the compression.

A detailed study of HSI compression techniques is also covered, and future directions are discussed to overcome the observed challenges. Algorithms of different techniques are categorized with their methodology, advantages, and limitations compactly. Techniques adopted for classification can be used to evaluate and categorize any algorithm of the field. Such classifications could help in the development of advanced compression algorithms and may boost many space programs.

## 5 Appendix

The performance of any algorithm can be analyzed by the evaluation metrics used by it. The Table 10 presents a collection of metrics used by the techniques along with the dataset on which the experiments have been performed. It contains a combination of datasets available in public domain and dataset generated especially for the experiment. The abbreviations in the analysis criteria have their respective meanings.

**Table 10** Datasets and evaluation metrics used by different algorithms.

| Algorithms | Datasets used | Analysis criteria |
|---|---|---|
| Hybrid 3D-DCT-TD[12] | AVIRIS scenes | PSNR and CR |
| DWT-TD[8] | Cuprite and Moffett scenes | SNR, CA, and VCA |
| 3D-DCT-SVM[13] | Cuprite and Moffett scenes | CR and SA |
| JPEG-LS[14] | AVIRIS calibrated and uncalibrated scenes | CR |
| Kozhemiakin et al.[15] | Landsat images | MSE and CR |
| Giordano and Guccione[16] | AVIRIS scenes | CA and time |
| HyperLCA[20] | AVIRIS and Hyperion scenes | MSE, SNR, and SA |
| PCA-DCT[17] | HYDICE and AVIRIS scenes | PSNR, MSE, and CR |
| Integer HyperLCA[21] | AVIRIS and Hyperion scenes | MSE and SNR |
| Wang et al.[10] | AVIRIS low altitude images | PSNR and BR |
| Folded PCA[18] | Indian Pines | PSNR, SNR, and CA |
| Luminance transform[22] | Cuprite, Moffett Field, and Jasper ridge scenes | SNR and SA |
| WPCA[19] | Cuprite and Moffett field scenes | MSE, SNR, and ROC curve |
| FrWF[23] | Images from Minho (Portugal) | PSNR and BR |
| RWA-C[24] | Cuprite and Moffett scenes | Entropy |
| 3D-LMBTC[11] | Washington DC, Cuprite, Jasper Ridge, urban | PSNR and time |
| IKLT-IDWT[25] | Indian Pines, Salinas, Botswana, KSC, and urban scenes | PSNR, BR, Time |
| Diaz et al.[26] | Specim FX10 dataset | CR and speedup |
| SVR-DWT[27] | AVIRIS Yellowstone uncalibrated, Cuprite, Indian Pines, Washington DC, Mt. St. Helens, and Erta Ale scenes | BR, CR, SNR, PSNR, SSIM, and MSE |
| GFT[28] | AVIRIS scenes | BR and PSNR |
| Fuzzy transform[29] | Pleiades satellite's multispectral images | PSNR and time |
| Bogdan et al.[40] | Multispectral image | CR |
| SB-DSC[41] | AVIRIS scenes | PSNR, average error, and SA |
| Conoscenti et al.[38] | AVIRIS scenes | SNR, BR, and SAD |
| Zhaoa et al.[39] | Cuprite, Jasper Ridge, and Lunar Lake scenes | PSNR, BR, CA, and time |

**Table 10** (*Continued*).

| Algorithms | Datasets used | Analysis criteria |
|---|---|---|
| 3D-MBLP[42] | AVIRIS and CCSDS datasets | CR |
| Binary tree-based decomposition[43] | Indian Pines, Salinas, Pavia Centre and University, Cuprite, KSC, and Botswana scenes | CR |
| Shen et al.[44] | Indian Pines, KSC, Salinas, and Pavia University scenes | BR |
| RLS-OPB-P[35] | AVIRIS scenes | CR, time, and speedup |
| Fjeldtvedt et al.[45] | Modis, HyperION, AVIRIS, and HICO scenes | Power and speedup |
| Barrios et al.[46] | AVIRIS Scenes | CR and time |
| LSTM-RNN[47] | Indian pines, Pavia University, Salinas and KSC scenes | MSE |
| SuperRLS[34] | AVIRIS scenes | BR, time, and speedup |
| C-DPCM-RNN[33] | AVIRIS scenes | BR |
| Li et al.[48] | AVIRIS scenes | BR and speedup |
| Afjal et al.[49] | Indian Pines and three Landsat multispectral images | BR and entropy |
| Rodriguez et al.[50] | Indian Pines and Yellowstone scenes | Throughput, power, and rate |
| Cang and Wang[51] | HJ-1A satellite dataset | MSE and PSNR |
| Bascones et al.[52] | SUW, BEL, REN, CUP, HAW, MAI, and YEL scenes | PSNR, BR, and time |
| Li et al.[60] | AVIRIS scenes | SNR and BR |
| VQPCA[57] | SUW, DHO, BEL, REN, CUP, and Cuprite scenes | SNR and CR |
| Xu et al.[66] | Cuprite, Jasper Ridge, Lunar Lake, and low altitude scenes | SNR and time |
| Gunasheela and Prasantha[62] | Urban dataset | PSNR and SA |
| CSDL_JP2[67] | Pavia University | PSNR, SSIM, and BR |
| SHSIR[62] | Urban dataset | SSIM |
| HSI-CSR[68] | Pavia University | PSNR, SSIM, mean feature similarity, SAM, and weighted sum of MSE |
| PSO-NTD[71] | AVIRIS scenes | MSE, SNR, and CR |
| DWT-TD(ALS)-RLE[72] | Colorado River scenes | PSNR, CR, and time |
| MBSRDL[73] | Pavia Centre and University | PSNR, SSIM, and time |
| CNN-NTD[31] | Multispectral images containing buildings, cities, and mountains | PSNR, BR, and time |
| Aidini et al.[74] | EUROSAT multispectral images | PSNR and accuracy |
| SSASR[77] | AVIRIS scenes | BR, CA, SAD, and time |
| Fu et al.[78] | Indian Pines, Washington DC, Moffett, and Jasper Ridge scenes | SNR and SAD |

**Table 10** (*Continued*).

| Algorithms | Datasets used | Analysis criteria |
|---|---|---|
| Online learning[58] | Jasper Ridge, Cuprite, Lunar Lake, and low altitude scenes | SNR |
| SpaRSA[76] | AVIRIS and Hyperion scenes | PSNR |
| Zhu et al.[79] | Mississippi State University, CASI, Hyperspec sensor images | SNR, CR, BR, and ROC curve |
| CLMS[36] | Levada sequence scenes | BR |
| Fast-Lossless-4D Predictor[81] | Time lapse and AAMU scenes | BR |
| ANN[83] | AVIRIS scenes | CR and speedup |
| DBN[86] | AVIRIS scenes | BR and time |
| BIP-MLPNN[84] | AVIRIS scenes | MSE, PSNR, and SSIM |
| Prequantization[87] | Cuprite, Jasper Ridge, and Moffett scenes | Absolute error and relative error |
| CNN-LZMA[88] | SIPI dataset and six aerial images | CR and BR |

## Acknowledgments

The authors declare that they have no conflict of interest.

## References

1. M. J. Khan et al., "Modern trends in hyperspectral image analysis: a review," *IEEE Access* **6**, 14118–14129 (2018).
2. B. Park and R. Lu, *Hyperspectral Imaging Technology in Food and Agriculture*, Springer (2015).
3. C. T. Willoughby, M. A. Folkman, and M. A. Figueroa, "Application of hyperspectral imaging spectrometer systems to industrial inspection," *Proc. SPIE* **2599**, 264–272 (1996).
4. H. Saari et al., "Novel miniaturized hyperspectral sensor for UAV and space applications," *Proc. SPIE* **7474**, 74741M (2009).
5. H. Grahn and P. Geladi, *Techniques and Applications of Hyperspectral Image Analysis*, John Wiley & Sons (2007).
6. E. Magli, "Multiband lossless compression of hyperspectral images," *IEEE Trans. Geosci. Remote Sens.* **47**(4), 1168–1178 (2009).
7. S. Liew, "Principles of remote sensing," 2019, https://crisp.nus.edu.sg/~research/tutorial/image.htm (accessed 10 March 2019).
8. A. Karami, M. Yazdi, and G. Mercier, "Compression of hyperspectral images using discrete wavelet transform and tucker decomposition," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **5**(2), 444–450 (2012).
9. J. A. Saghri, S. Schroeder, and A. G. Tescher, "Adaptive two-stage Karhunen–Loeve–transform scheme for spectral decorrelation in hyperspectral bandwidth compression," *Opt. Eng.* **49**(5), 057001 (2010).
10. X. Wang et al., "Distributed source coding of hyperspectral images based on three-dimensional wavelet," *J. Indian Soc. Remote Sens.* **46**(4), 667–673 (2018).
11. S. Bajpai et al., "Low memory block tree coding for hyperspectral images," *Multimedia Tools Appl.* **78**, 27193–27209 (2019).
12. A. Karami, M. Yazdi, and A. Z. Asli, "Hyperspectral image compression based on tucker decomposition and discrete cosine transform," in *2nd Int. Conf. Image Process. Theory, Tools and Appl.*, IEEE, pp. 122–125 (2010).

13. A. Karami, S. Beheshti, and M. Yazdi, "Hyperspectral image compression using 3D discrete cosine transform and support vector machine learning," in *11th Int. Conf. Inf. Sci., Signal Process. and Their Appl.*, IEEE, pp. 809–812 (2012).
14. B. U. Töreyn et al., "Lossless hyperspectral image compression using wavelet transform based spectral decorrelation," in *7th Int. Conf. Recent Adv. Space Technol.*, IEEE, pp. 251–254 (2015).
15. R. Kozhemiakin et al., "Lossy compression of landsat multispectral images," in *5th Mediterr. Conf. Embedded Comput.*, IEEE, pp. 104–107 (2016).
16. R. Giordano and P. Guccione, "ROI-based on-board compression for hyperspectral remote sensing images on GPU," *Sensors* **17**(5), 1160 (2017).
17. R. J. Yadav and M. Nagmode, "Compression of hyperspectral image using PCA–DCT technology," in *Innovations in Electronics and Communication Engineering*, H. Saini, R. Singh, and K. Reddy, Eds., pp. 269–277, Springer, Singapore (2018).
18. S. Mei et al., "Low-complexity hyperspectral image compression using folded PCA and JPEG2000," in *IEEE Int. Geosci. and Remote Sens. Symp.*, IEEE, pp. 4756–4759 (2018).
19. A. C. Karaca and M. K. Güllü, "Target preserving hyperspectral image compression using weighted PCA and JPEG2000," *Lect. Notes Comput. Sci.* **10884**, 508–516 (2018).
20. R. Guerra et al., "A new algorithm for the on-board compression of hyperspectral images," *Remote Sens.* **10**(3), 428 (2018).
21. R. Guerra et al., "A hardware-friendly algorithm for the on-board compression of hyperspectral images," in *9th Workshop Hyperspectral Image and Signal Process.: Evol. Remote Sens.*, IEEE, pp. 1–5 (2018).
22. E. Can et al., "Compression of hyperspectral images using luminance transform and 3D-DCT," in *IEEE Int. Geosci. and Remote Sens. Symp.*, IEEE, pp. 5073–5076 (2018).
23. S. Khan et al., "Fractional wavelet filter based discrete wavelet transform and SPIHT for hyperspectral image compression," *Int. J. Inf. Syst. Manage. Sci.* **2**(1) (2019).
24. E. Ahanonu, M. Marcellin, and A. Bilgin, "Clustering regression wavelet analysis for lossless compression of hyperspectral imagery," in *Data Compression Conf.* (2019).
25. R. Nagendran and A. Vasuki, "Hyperspectral image compression using hybrid transform with different wavelet-based transform coding," *Int. J. Wavelets Multiresolution Inf. Process.* **18**(1), 1941008 (2020).
26. M. Díaz et al., "Real-time hyperspectral image compression onto embedded GPUs," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **12**(8), 2792–2809 (2019).
27. N. Zikiou, M. Lahdir, and D. Helbert, "Support vector regression-based 3D-wavelet texture learning for hyperspectral image compression," *Visual Comput.* **36**, 1473–1490 (2020).
28. S. Jafari, "Graph transforms for hyperspectral image compression," PhD Thesis, Politecnico di Torino (2019).
29. D. Monica and A. Widipaminto, "Fuzzy transform for high-resolution satellite images compression," *Telkomnika* **18**(2), 1130–1136 (2020).
30. M. Ricci and E. Magli, "Predictor analysis for onboard lossy predictive compression of multispectral and hyperspectral images," *J. Appl. Remote Sens.* **7**(1), 074591 (2013).
31. J. Li and Z. Liu, "Multispectral transforms using convolution neural networks for remote sensing multispectral image compression," *Remote Sens.* **11**(7), 759 (2019).
32. J. S. Mielikainen, P. J. Toivanen, and A. Kaarna, "Linear prediction in lossless compression of hyperspectral images," *Opt. Eng.* **42**(4), 1013–1018 (2003).
33. J. Luo et al., "Lossless compression for hyperspectral image using deep recurrent neural networks," *Int. J. Mach. Learn. Cybern.* **10**, 2619–2629 (2019).
34. A. C. Karaca and M. K. Güllü, "Superpixel based recursive least-squares method for lossless compression of hyperspectral images," *Multidimension. Syst. Signal Process.* **30**(2), 903–919 (2019).
35. C. Li, "Parallel implementation of the recursive least square for hyperspectral image compression on GPUs," *KSII Trans. Internet Inf. Syst.* **11**(7), 3543–3557 (2017).
36. H. Shen, W. D. Pan, and Y. Dong, "Efficient lossless compression of 4D hyperspectral image data," in *Proc. 3rd Int. Conf. Adv. Big Data Anal.*, Las Vegas, Nevada, pp. 25–28 (2016).

37. R. Sugiura et al., “Optimal Golomb-rice code extension for lossless coding of low-entropy exponentially distributed sources,” *IEEE Trans. Inf. Theory* **64**(4), 3153–3161 (2018).
38. M. Conoscenti, R. Coppola, and E. Magli, “Constant SNR, rate control, and entropy coding for predictive lossy hyperspectral image compression,” *IEEE Trans. Geosci. Remote Sens.* **54**(12), 7431–7441 (2016).
39. D. Zhao, S. Zhu, and F. Wang, “Lossy hyperspectral image compression based on intra-band prediction and inter-band fractal encoding,” *Comput. Electr. Eng.* **54**, 494–505 (2016).
40. B. Rusyn et al., “Lossless image compression in the remote sensing applications,” in *IEEE First Int. Conf. Data Stream Mining and Process.*, IEEE, pp. 195–198 (2016).
41. M. B. Nm, S. Sujatha, and A.-S. K. Pathan, “Skip block based distributed source coding for hyperspectral image compression,” *Multimedia Tools Appl.* **75**(18), 11267–11289 (2016).
42. R. Pizzolante and B. Carpentieri, “Multiband and lossless compression of hyperspectral images,” *Algorithms* **9**(1), 16 (2016).
43. S. Shahriyar et al., “Lossless hyperspectral image compression using binary tree based decomposition,” in *Int. Conf. Digital Image Comput.: Tech. and Appl.*, IEEE, pp. 1–8 (2016).
44. H. Shen, W. D. Pan, and D. Wu, “Predictive lossless compression of regions of interest in hyperspectral images with no-data regions,” *IEEE Trans. Geosci. Remote Sens.* **55**(1), 173–182 (2017).
45. J. Fjeldtvedt, M. Orlandić, and T. A. Johansen, “An efficient real-time FPGA implementation of the CCSDS-123 compression standard for hyperspectral images,” *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **11**(10), 3841–3852 (2018).
46. Y. Barrios et al., “Hardware implementation of the CCSDS 123.0-B-1 lossless multispectral and hyperspectral image compression standard by means of high level synthesis tools,” in *9th Workshop Hyperspectral Image and Signal Process.: Evol. Remote Sens.*, IEEE, pp. 1–5 (2018).
47. Z. Jiang, W. D. Pan, and H. Shen, “LSTM based adaptive filtering for reduced prediction errors of hyperspectral images,” in *6th IEEE Int. Conf. Wireless Space and Extreme Environ.*, IEEE, pp. 158–162 (2018).
48. J. Li, J. Wu, and G. Jeon, “GPU acceleration of clustered DPCM for lossless compression of hyperspectral images,” *IEEE Trans. Ind. Inf.* **16**, 2906–2916 (2020).
49. M. I. Afjal, M. Al Mamun, and M. P. Uddin, “Band reordering heuristics for lossless satellite image compression with 3D-calic and CCSDs,” *J. Visual Commun. Image Represent.* **59**, 514–526 (2019).
50. A. Rodriguez et al., “Scalable hardware-based on-board processing for run-time adaptive lossless hyperspectral compression,” *IEEE Access* **7**, 10644–10652 (2019).
51. S. Cang and A. Wang, “Research on hyperspectral image reconstruction based on GISMT compressed sensing and interspectral prediction,” *Int. J. Opt.* **2020**, 7160390 (2020).
52. D. Báscones, C. González, and D. Mozos, “An extremely pipelined FPGA implementation of a lossy hyperspectral image compression algorithm,” *IEEE Trans. Geosci. Remote Sens.* (2020).
53. L. Ke and M. W. Marcellin, “Near-lossless image compression: minimum-entropy, constrained-error DPCM,” *IEEE Trans. Image Process.* **7**(2), 225–228 (1998).
54. D. Manak et al., “Efficient hyperspectral data compression using vector quantization and scene segmentation,” *Can. J. Remote Sens.* **24**(2), 133–143 (1998).
55. R. Gray, “Vector quantization,” *IEEE ASSP Mag.* **1**(2), 4–29 (1984).
56. S.-E. Qian et al., “3D data compression of hyperspectral imagery using vector quantization with NDVI-based multiple codebooks,” in *Sens. and Managing Environ. IEEE Int. Geosci. and Remote Sens. Symp. Proc.*, IEEE, Vol. 5, pp. 2680–2684 (1998).
57. D. Báscones, C. González, and D. Mozos, “Hyperspectral image compression using vector quantization, PCA and JPEG2000,” *Remote Sens.* **10**(6), 907 (2018).
58. W. Jifara et al., “Hyperspectral image compression based on online learning spectral features dictionary,” *Multimedia Tools Appl.* **76**(23), 25003–25014 (2017).
59. R. Bal, A. Bakshi, and S. Gupta, “Performance evaluation of optimization techniques with vector quantization used for image compression,” in *Harmony Search and Nature Inspired*

*Optimization Algorithms*, N. Yadav et al., Eds., Vol. 741, pp. 879–888, Springer, Singapore (2019).

60. R. Li, Z. Pan, and Y. Wang, "The linear prediction vector quantization for hyperspectral image compression," *Multimedia Tools Appl.* **78**(9), 11701–11718 (2019).
61. S.-E. Qian et al., "Vector quantization using spectral index-based multiple subcodebooks for hyperspectral data compression," *IEEE Trans. Geosci. Remote Sens.* **38**(3), 1183–1190 (2000).
62. K. Gunasheela and H. Prasantha, "Compressive sensing approach to satellite hyperspectral image compression," in *Information and Communication Technology for Intelligent Systems*, S. Satapathy and A. Joshi, Eds., Vol. 106, pp. 495–503, Springer, Singapore (2019).
63. Z. Zha et al., "Compressed sensing image reconstruction via adaptive sparse nonlocal regularization," *Visual Comput.* **34**(1), 117–137 (2018).
64. F. Magalhães et al., "High-resolution hyperspectral single-pixel imaging system based on compressive sensing," *Opt. Eng.* **51**(7), 071406 (2012).
65. L. Liu et al., "Karhunen–Loève transform for compressive sampling hyperspectral images," *Opt. Eng.* **54**(1), 014106 (2015).
66. K. Xu et al., "Distributed lossy compression for hyperspectral images based on multilevel coset codes," *Int. J. Wavelets Multiresolution Inf. Process.* **15**(2), 1750012 (2017).
67. S. Kumar et al., "Onboard hyperspectral image compression using compressed sensing and deep learning," *Lect. Notes Comput. Sci.* **11130**, 30–42 (2018).
68. J. Xue et al., "Nonlocal tensor sparse representation and low-rank regularization for hyperspectral image compressive sensing reconstruction," *Remote Sens.* **11**(2), 193 (2019).
69. E. Candes and T. Tao, "Decoding by linear programming," *IEEE Trans. Inf. Theory* **51**(12), 4203–4215 (2005).
70. Y.-D. Kim and S. Choi, "Nonnegative tucker decomposition," in *IEEE Conf. Comput. Vision and Pattern Recognit.*, IEEE, pp. 1–8 (2007).
71. A. Karami, R. Heylen, and P. Scheunders, "Hyperspectral image compression optimized for spectral unmixing," *IEEE Trans. Geosci. Remote Sens.* **54**(10), 5884–5894 (2016).
72. K. Rajan and V. Murugesan, "Hyperspectral image compression based on DWT and TD with ALS method," *Int. Arab J. Inf. Technol.* **13**(4), 435–442 (2016).
73. Y. Chong et al., "Block-sparse tensor based spatial-spectral joint compression of hyperspectral images," *Lect. Notes Comput. Sci.* **10956**, 260–265 (2018).
74. A. Aidini et al., "Hyperspectral image compression and super-resolution using tensor decomposition learning," in *53rd Asilomar Conf. Signals, Syst., and Comput.*, IEEE, pp. 1369–1373 (2019).
75. J. E. Fowler, "Compressive-projection principal component analysis," *IEEE Trans. Image Process.* **18**(10), 2230–2242 (2009).
76. İ. Ülkü and E. Kizgut, "Large-scale hyperspectral image compression via sparse representations based on online learning," *Int. J. Appl. Math. Comput. Sci.* **28**(1), 197–207 (2018).
77. W. Fu et al., "Adaptive spectral–spatial compression of hyperspectral image with sparse representation," *IEEE Trans. Geosci. Remote Sens.* **55**(2), 671–682 (2017).
78. C. Fu, Y. Yi, and F. Luo, "Hyperspectral image compression based on simultaneous sparse representation and general-pixels," *Pattern Recognit. Lett.* **116**, 65–71 (2018).
79. W. Zhu, Q. Du, and J. E. Fowler, "Multitemporal hyperspectral image compression," *IEEE Geosci. Remote Sens. Lett.* **8**(3), 416–420 (2011).
80. Z. Wang, N. M. Nasrabadi, and T. S. Huang, "Spatial–spectral classification of hyperspectral images using discriminative dictionary designed by learning vector quantization," *IEEE Trans. Geosci. Remote Sens.* **52**(8), 4808–4822 (2014).
81. H. Shen, Z. Jiang, and W. Pan, "Efficient lossless compression of multitemporal hyperspectral image data," *J. Imaging* **4**(12), 142 (2018).
82. G. S. Rao, G. V. Kumari, and B. P. Rao, "Image compression using neural network for biomedical applications," in *Soft Computing for Problem Solving*, J. Bansal et al., Eds., Vol. 817, pp. 107–119, Springer, Singapore (2019).
83. Y. M. Masalmah et al., "A framework of hyperspectral image compression using neural networks," in *Latin Am. and Caribbean Conf. Eng. and Technol. Proc.*, Vol. 13, University del Turabo, Puerto Rico (2015).

84. R. Dusselaar and M. Paul, "A block-based inter-band predictor using multilayer propagation neural network for hyperspectral image compression," arXiv:1902.04191 (2019).
85. J. W. Chai, J. Wang, and C.-I. Chang, "Mixed principal-component-analysis/independent-component-analysis transform for hyperspectral image analysis," *Opt. Eng.* **46**(7), 077006 (2007).
86. Z. Jiang, W. D. Pan, and H. Shen, "Universal Golomb–rice coding parameter estimation using deep belief networks for hyperspectral image compression," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **11**(10), 3830–3840 (2018).
87. D. Valsesia and E. Magli, "High-throughput onboard hyperspectral image compression with ground-based CNN reconstruction," *IEEE Trans. Geosci. Remote Sens.* **57**, 9544–9553 (2019).
88. B. Sujitha et al., "Optimal deep learning based image compression technique for data transmission on industrial internet of things applications," *Trans. Emerging Telecommun. Technol.* e3976 (2020).
89. M. Nelson and J.-L. Gailly, *The Data Compression Book*, M & T Books, New York (1996).
90. Y. Li et al., "Distributed lossless compression algorithm for hyperspectral images based on the prediction error block and multiband prediction," *Opt. Eng.* **55**(12), 123114 (2016).
91. R. Ansari, E. Ceran, and N. D. Memon, "Near-lossless image compression techniques," *Proc. SPIE* **3309**, 731–742 (1998).
92. A. J. Hussain, A. Al-Fayadh, and N. Radi, "Image compression techniques: a survey in lossless and lossy algorithms," *Neurocomputing* **300**, 44–69 (2018).
93. R. Chandra et al., *Parallel Programming in OpenMP*, Morgan Kaufmann (2001).
94. W. Gropp, R. Thakur, and E. Lusk, *Using MPI-2: Advanced Features of the Message Passing Interface*, MIT Press (1999).
95. D. Pellerin and S. Thibault, *Practical FPGA Programming in C*, Prentice Hall Press (2005).
96. J. Sanders and E. Kandrot, *CUDA by Example: An Introduction to General-Purpose GPU Programming*, Addison-Wesley Professional (2010).
97. G. Motta, F. Rizzo, and J. A. Storer, *Hyperspectral Data Compression*, Springer Science & Business Media (2006).
98. S. Sanjith and R. Ganesan, "A review on hyperspectral image compression," in *Int. Conf. Control, Instrum., Commun. and Comput. Technol.*, IEEE, pp. 1159–1163 (2014).
99. K. S. Babu et al., "Hyperspectral image compression algorithms–a review," in *Artificial Intelligence and Evolutionary Algorithms in Engineering Systems*, L. Suresh, S. Dash, and B. Panigrahi, Eds., Vol. **325**, pp. 127–138, Springer, New Delhi (2015).
100. R. Dusselaar and M. Paul, "Hyperspectral image compression approaches: opportunities, challenges, and future directions: discussion," *J. Opt. Soc. Am. A* **34**(12), 2170–2180 (2017).
101. K. Gunasheela and H. Prasantha, "Satellite image compression-detailed survey of the algorithms," *Lect. Notes Networks Syst.* **14**, 187–198 (2018).
102. A. B. Kiely et al., "The new CCSDS standard for low-complexity lossless and near-lossless multispectral and hyperspectral image compression," in *Proc. Onboard Payload Data Compression Workshop*, pp. 1–7 (2018).

**Yaman Dua** is a research scholar in the Department of Computer Science and Engineering, Indian Institute of Technology (BHU), Varanasi, India. He received his bachelor's degree in information technology from Dr. A.P.J. Abdul Kalam Technical University, India. His research interests include image compression, machine learning, and parallel computing.

**Vinod Kumar** is a research scholar in the Department of Computer Science and Engineering, Indian Institute of Technology (BHU), Varanasi, India. He received his master's degree in electronics and electrical communication engineering from Indian Institute of Technology, Kharagpur. He received his bachelor's degree in electronics engineering from Indian Institute of Technology

(BHU), Varanasi, India. His research interests include image classification, high-performance computing, and deep learning.

**Ravi Shankar Singh** is an associate professor in the Department of Computer Science and Engineering, Indian Institute of Technology (BHU), Varanasi, India. He has published over 70 papers in national and international journals and conferences in research areas including data structures, algorithms, and high-performance computing.