International Journal of Computer Information Systems and Industrial Management Applications.
ISSN 2150-7988 Volume 15 (2023) pp. 396-407

Received: 14 January 2022; Accepted: 19 May, 2023; Published: 23 June, 2023

Hyperspectral Image Compression using Modified Convolutional Autoencoder

Satvik Agrawal[1], Sancharika Debnath[1], Santwana Sagnika[1,*], Saurabh Bilgaiyan[1], Saksham Gupta[2]

[1] School of Computer Engineering, Kalinga Institute of Industrial Technology Deemed to be University, India
satvik.agrawal2001@gmail.com, sancharikadebnath@gmail.com, santwana.sagnika@gmail.com, saurabhbilgaiyan01@gmail.com

[2] Computer Science Engineering Department, Chandigarh College of Engineering and Technology, India
dev.sakshamgupta@gmail.com

***Abstract*: Spectral imaging is a type of multi band imaging technique of the electromagnetic spectrum, used for gathering and analysis of information. Hyperspectral imaging is a technique that collects spectral information from a broad spectrum of wavelengths for the same spatial area of each pixel. Due to its multiple bands, and spectral and spatial redundancy, the image size is immense. Processing these images requires an enormous amount of memory. Our paper proposes a lossy technique to encode and compress the hyperspectral images by the help of deep convolutional networks, autoencoders and attention layers. The encoder uses convolutional and max pooling layers, connected to a singular attention layer to encode the data, and the decoder has a single dense layer preceding transpose convolutional and upsampling layers to decode the coded data back into the hyperspectral images. The method is tested on different images taken by the Airborne Visible/ Infrared Imaging Spectrometer (AVIRIS), Reflective Optics System Imaging Spectrometer (ROSIS), and NASA EO1 satellite. The method achieves superior results than existing work by up to a 5% increase in the PSNR and up to 200 times increase in the compression ratio.**

***Keywords*:** Hyperspectral image processing, autoencoder, image compression, deep learning, neural network**.**

I. Introduction

Spectral imaging utilizes multiple bands over the electromagnetic spectrum. It denotes a group of analytic techniques using multiple bands from the spectrum, and collects both spectroscopic information and imaging information at the same time. The image data from few broad wavelengths (approx. 3 to 15 bands) are captured via Multispectral remote sensors. Comparatively, the image data from the numerous spectral band covering an extended range of wavelength from 400 to 2500nm is simultaneously collected via hyperspectral (HS) remote sensors [1]. Hyperspectral imaging is a technique based on spectroscopy.

Multiple bands of image data are gathered at multiple wavelengths for each spatial area. This gathered data creates a hyperspectral cube, two of its dimensions signify spatial extent of the location and the third stands for spectral content. The spectral signature is the outcome of molecule absorption and particle scattering, and it allows for the differentiation of materials with various properties. Agriculture, environmental monitoring, weather prediction, military, food industry, medicinal, and forensic research are some examples of hyperspectral remote sensing applications.

Hyperspectral images store a load of data which can be processed to obtain a large spectrum of meaningful information. Storing huge spectral dimensionality of multiple spectral channels requires enormous memory. A single HIS dataset can have a size of hundreds of megabytes (MBs), since each pixel holds 16-bit or 12-bit information, the range of the pixel numbers can vary from few hundreds to millions, and as high as 22 bands can be the band numbers. [2]. For example, the dataset of calibrated images of standard Consultative Committee for Space Data Systems (CCSDS) [3] has 224 total spectral bands, 677 × 512 counts of pixels per band , and 16-bits of information is stored in each pixel. Thus, 677 × 512 × 224 × 16 bits = 148 MB is the size of this standardized image [2].

The analysis of enormous HSI while keeping the necessary valuable information is a major issue. It is a primary problem for compression algorithms because data compression is performed to decrease data redundancy [1]. High data redundancy usually equates to high compression ratios, while poor redundancy equates to low compression ratios. In multi-spectral images such as HSI, there are four types of redundancy:

I) Statistical redundancy - analysis of the probability of symbols. The methods used to analyse the probability of symbols are called entropy coding.

II) Spatial redundancy - intraband correlation assumes that the pixel information could be partially obtained by neighboring pixels. It can be removed via transformation.

III) Spectral redundancy - or interband correlation, based on the high correlation existing between neighboring bands in hyperspectral images. It can be removed by spatial decorrelation.

IV) Visual redundancy - driven by the detail that human eyes are not overly sensitive to high frequencies, data quantization is used to achieve compression based on visual redundancy. The utilization of interband or intraband correlations allows compression techniques to be divided into

2D and 3D methods. 2D image compression methods typically employ intraband and interband correlations individually, while 3D techniques use both inter-and intraband correlations together [4].

Hyperspectral images are merged with many recent technologies for advancement in this domain. In the past few years, research in this discipline has increased with the aid of diverse kinds of neural networks and autoencoders. A neural network is a basically sequence of calculating units that endeavor to acknowledge inherent relationships among a group of data through a process that imitates the operation of the human brain, Therefore, neural networks are interlinked systems of neurons, either organic or artificial [5]. An autoencoder is a model implementing unsupervised deep learning that trains the network to disregard signal noise and generate efficient data representations, known as encoding. The primary goal of an autoencoder is to convert high-dimensional space data to lower dimensions.

The paper proposes a modified lossy deep convolutional neural network autoencoder framework that includes an attention layer to encode and later decode hyperspectral image data. The proposed method uses convolutional layers and max pooling layers for the encoder along with an attention layer, and convolution transpose layers in the decoder. The task of the encoding convolutional and max pooling layers is to help decrease the size of the image data and also the overall space taken by the data. The major contributions of this work can be highlighted as:

- Compression of hyperspectral images using customized autoencoder.
- Greater accuracy achievement from other state of art architectures.
- Fusion of convolutional network and autoencoder along with attention layer for better performance.

The novelty of this paper can be encapsulated as:

- The proposed autoencoder is an efficient model.
- The results surpass state of the art architectures.
- Overall, training time is highly effective.

The remaining paper is organized as follows. Section 2 elaborates the recent related works in the domain of HSI compression. The technologies used by the proposed model are discussed in Section 3. In Section 4, the proposed convolutional autoencoder architecture is illustrated. The experimental setup and the obtained results of the experiments are shown in Section 5. The results achieved by the proposed model are explained and detailed in Section 6. Lastly, the conclusions along with future work are briefed in Section 7.

II. Literature Review

HSI compression is a prominent emerging topic of research for the space industry due to its immense memory size. There are several articles published in recent years for this specialized work. Many scientific communities are inclined towards lossless algorithms to preserve the required information as much as possible in compression. However, in need of higher compression and to reduce the memory size, lossy compression algorithms are adopted.

To improve the lossy compression performance of HSI, Ertem et al. (2020) [6] used an enhanced Spectral-Spatial Adaptive Sparse Representation (SSASR) known as modified SSASR. This uses a single PCA transformation in place of multiple transformations to enable compaction property. The model can eliminate the need for encoding super pixel maps with the aid of an ordering scheme of sparse coefficients. In addition, MSSASR preserved anomaly regions better than other algorithms. Barrios et al. (2020) [7] proposed a hardware implementation for lossy Multispectral and HSI compressors for onboard space missions. It extends the CCSDS 123.0-B-1 lossless standard. The High-Level Synthesis (HLS) techniques are used for algorithm implementation to increase productivity of design by raising the abstract level. The model is deployed onto ARTICo, and Xilinx Zynq Ultra Scale +Field-Programmable Gate Array (FPGA)-based MPSoC is used to test the compression solution. It gives improved results compared to the state of the art algorithms in terms of both computational cost and compression quality.

In the field of lossless compression, Zhu et al. (2020) [8] presented an improved Conventional least square (CRLS) with adaptive predictor selection and adaptive band selection (CRLS-APS-ABS). It uses several strategies for improvement. In order to enhance the correlation among the reference bands and prediction bands, an adaptive band selection strategy is utilized. To attain better similarity of prediction context, an adaptive predictor selection strategy is employed. Then the k-means clustering technique uses the spectral vector correlation coefficient to measure similarity. The outcome of the prediction process is improved by the double snake scan mode and the recursive local average estimation method is utilized for accelerated calculation of the local average. Báscones et al. (2018) presented a compression algorithm that decorrelates the image spectrally by using Vector Quantization and Principal Component Analysis (VQPCA). The spatial correlations are then exploited for compression by applying JPEG2000 to the Principal Components [9]. The optimized choice of parameters maximized the distortion-ratio performance. They also proposed a formula to determine the algorithm's configuration for obtaining a result ranging from heavily compressed-low SNR images to low compressed-near lossless images.

Machine Learning strategies, especially in instances of Deep Learning, enhance the performance of compression techniques, as observed from literature. It is proven that the concept of autoencoder works much better for compression in every domain. Cheng et al. (2018) [10] presents an architecture for lossy image compression using Conventional autoencoder (CAE) to attain higher efficiency while coding. Initially they designed a new CAE architecture and trained the autoencoder using a loss function based on rate-distortion. Next, feature maps were rotated using principal components analysis (PCA) and quantization and entropy coder were applied to generate the codes, achieving a 13.7% BD-rate decrease on images from Kodak database as compared to images from JPEG2000. But the moderate complexity was similar to JPEG2000. In terms of compression in the medical domain, Mishra et al. (2020) [11] used a two-staged autoencoder for compressor-decompressor for the purpose of compressing malaria RBC cell image patches, which showed significant improvement over state-of-art algorithms. The proposed dual autoencoder model is a good ROI-based loss compression method with minimum information loss.

In the field of Hyperspectral image compression, Denge et al. (2020) [12] supplied a generative neural network to learn the probability distribution of data from random latent code. The proposed model stores the HSI, the random normal distribution that supplies the maximum entropy. As a result, both the compression quality and ratio are controlled directly through the model. It also uses the pruning technique to eliminate weights that do not affect the compression ratio. To minimize the time and memory complexities constraints, a complexity-reduced variational autoencoder was designed by Oliveira et al. (2021) [13]. They also put forward a simple entropy model that contained a single parameter to support the input image's adaptability. It outperformed the CCSDS 122.0-B and had a reasonable rate distortion performance. This model is able to identify the least global amount of filters for every rate due to its bottleneck size. A lossy compression scheme is proposed by Ouahioune et al. (2021) [14], a combination of 3D wavelet transforms and a super-resolution technique based on wavelet learning called wavelet learning-based super-resolution compression (WSRC). They down-sampled the image during encoding which reduced the loss of information and up-sampled it by a super-resolution strategy at the decoder, thus increasing the information compensation. The outcome of the algorithm supplies a prominent performance, preserves the spectral signature, and generates high-quality images. But with a better resolution (high-quality), compression is not much supported in terms of memory size.

An extended work of Wang et al. (2019) [15] was proposed by Ayma et al. (2020) [16], a dimensional reduction implementation for hyperspectral imaging using orthogonal autoencoder (HOAE) to decrease the redundancy. This technique helped to learn orthogonal characteristics in a lower-dimensional vector space by including orthogonal constraints inside the loss function. The orthogonal features aided in achieving better classification rates as compared to ultramodern and typical autoencoders. A lossy Hyperspectral compression using the concept of the autoencoder was presented by Dua et al. (2021) [2]. It was represented by a combination of max-pooling layer and convolutional layer for dimensional reduction. The lossy original image is constructed again by utilizing the transpose of the convolution layer. An adaptive arithmetic coder was used to entropy code the compressed picture for transport or storage. A total improvement of 28% was achieved in PSNR and a 21 times increase in compression ratio was calculated using this model. The compression result is evaluated through classification using ultramodern classification algorithms. The proposed algorithm proved to be amazingly effective due to its negligible difference in classification accuracy. A technique of compressing hyperspectral data using a 1D-Convolutional Autoencoder was introduced by Kuester et al. (2021) [17]. The spatial domain is not utilized during compression in order to avoid the falsification of the relationship between the spectral dimensions and therefore affect the accuracy of reconstruction by the model. The given approach locates and extracts compression-relevant characteristics in an efficient manner. The 1D-Convolutional Autoencoder surpasses the Nonlinear Principal Component Analysis (NPCA) and the Deep Autoencoder in terms of reconstruction accuracy.

For compression of images, there exist several techniques, but those techniques cannot be used for HSI compression due to their spatial and spectral features. Commonly seen drawback is the splitting of the dataset [2], [6], [8], [13]. On splitting the data, the number of neighbors decreases. As a result, the prediction is limited and affects compression (for learning, input data is less), and apart from poor predictions, the gradient loss also gets affected. Pre-processing of images before feeding them to models provides better results [9], [14], [16], [17]. In recent years, many architectures have presented for HSI compression. Few of them are discussed earlier, and the main intuition used behind HSI compression is the property of dimension reduction [6]-[8]. In the growing era of artificial intelligence, one can compress images effectively with significantly lower computational power [10]-[12]. In this paper, we tried to overcome a few of these drawbacks by avoiding data splitting followed by modified machine learning architecture utilization for better comparison results and experimenting with reshaping and normalization techniques for pre-processing.

III. Basic Concepts

The different techniques used for the proposed Hyperspectral Image compression are discussed in this section [18]. This section is organized as: *A.* Convolutional Neural Network (CNN) *B.* Convolutional Autoencoder, *C.* Entropy Coding.

A. Convolutional Neural Network (CNN)

Convolutional Neural Networks (CNNs) are a version of traditional Artificial Neural Networks (ANNs), consisting of self-optimized neurons [19]. A CNN, also known as ConvNet, is a specialized Deep Learning model, a sub type of ANN that uses an image based input and applies importance-based learnable weights and biases on various aspects of the grid pattern data, like images and can distinguish between each of them. The CNN algorithms with some slight modifications are used for uncountable purposes. Few of the most common uses of CNN are food detection [20], hate-speech identification system [22], in numbers of medicine purposes [18], [22], image classification [23], and recognition [24], texture synthesis [25], facial analysis [26], [27] and many more. CNNs have many forthcoming spectra as improvement in the application within radiology, the study of vulnerability in deep neural networks like adversarial examples [28], pre-trained networks for large required datasets like medical datasets can be proposed. The principal function of CNN is extracting features from the input with the help of back-propagation using multiple rudimentary steps like convolution layers, pooling layers, and fully connected or dense layers [29].

The concept of CNN came from the convolution theorem based on the mathematical operator which is anointed convolution, a specialized category of linear operation. So, mathematically using the convolution theorem, the convolutional layer can be explained and defined as:

$$(u * v)(\alpha) \triangleq \frac{\int_{-\infty}^{\infty} u(\psi)v(\alpha-\psi)d\psi}{u(\alpha)*v(\alpha)} \quad (1)$$

Mathematically, in equation 1, convolution is the integral of all the space present in one function, u at α time of another function, v at α of the continuous variable. The integration is from minus infinity to plus infinity over all the dimensions of

the variable α (can be 1D or 3D variable). The crossed circle operation writes down the convolution operation. Figure 1 represents a typical convolutional neural network with its hidden layers.

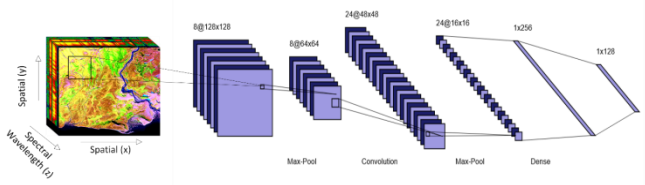

Figure 1. Convolutional Neural Networks (CNN)

B. *Convolutional Autoencoder*

Autoencoder (AE) is a self-taught learning unsupervised neural network framework, composed of encoding stages and decoding stages to reconstruct the given input patterns by dropping the noises and data dimension reduction. AE takes a fragment of input patterns and produces a discrete latent code [30]. They execute by representing the input in a latent-space after its compression (encoding the data) of a 1-D vector, known as the bottleneck, and reconstructing the output from this generated representation (decoding the data) [31]. The higher level of network hardwiring by adding convolutional operation in the existing autoencoder network is understood as Convolutional Autoencoder (CAE) [46]. Figure. 2 illustrates the structural framework of a convolutional autoencoder.

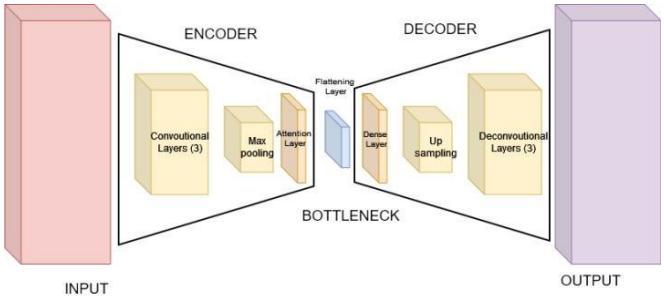

Figure 2. Convolutional Auto Encoder

AE is used for de-noising and dimension reduction purposes. AE is applicable for a wide range of purposes in the modern world in remote sensing [32], classification [33], deception detection [34], medicines [35], removal of noise [36] and watermark [37], image reconstruction [38] and generation [39]. The greater the accuracy of the autoencoder, the more similar the generated and the original data are [40]. Therefore, AE is divided into two parts: *1.* Encoder and *2.* Decoder.

1) Encoder

An encoder is a fully connected network that uses a feed-forward system to compress an input by encoding the input image in a reduced dimension and representing it in a latent space. The mapping functions are non-linear. The standard form is:

$$m_i = f(y_i) = \frac{1}{1+\exp(-w_1 y_i + v_1)} \quad (2)$$

In equation 2, is a function w_1 and is the encoding weight. Its biased vector is v_1.

An encoder is a self-learning forward propagating convolutional neural network consisting of many building blocks of contrasting functions as its layer. The blocks used for the layers in the encoder of the proposed framework are: *a)* Convolutional layers, *b)* Activation function, *c)* Pooling, *d.* Attention layers, and *e)* Flattening layers.

a) Convolutional layers

The convolution layer is the initial layer of CNN that extracts the features from the input. It has sets of filters known as kernels. Convolutional layers using learnable weights and bias forward propagate on the training dataset. As shown in equation 3 it follows the convolutional theorem that is the product of Fourier transform, F(k), G(k) in Fourier space of two function, f(r), g(r) is same as the convolution of both functions, i.e.,

$$f(r) \otimes\otimes\, g(r) \Leftrightarrow F(k)G(k) \quad (3)$$

b) Activation function

The activation or threshold function is a mathematical function of a node applied to a given set of inputs. In simplified terms, it is a feature to decide the activation of a given neuron in the exact way biological neurons get activated using simulation. It is a function used in the neural network to help the network learn complex features in the input data, represented in Figure 3. The task of this function is to use the weighted sum of input nodes and generate an output in the given layer of the neural network and fed to another layer or as output [44].

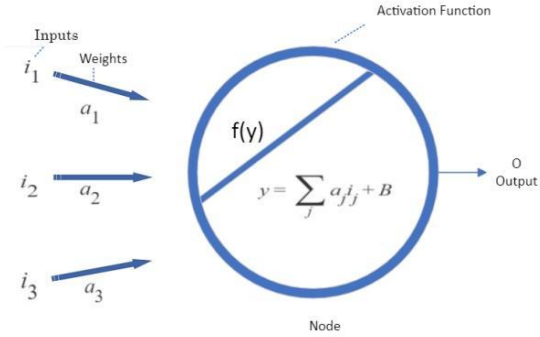

Figure 3. Activation Function

Several activation functions, diverged in terms of linear activation function and non-linear activation functions like sigmoid, Hyperbolic Tangent (tanh), exponential linear units (ELUs), etc. are used in neural networks. The proposed method made use of two activation functions. ReLU (Rectified Linear Unit) is used which supplies a linear function impression. It utilizes a derivative function and permits back-propagation with computational efficiency simultaneously. The ReLU function only works for positive input, which can be understood by equation 4 where F(y) is the function. The mathematical representation of the ReLU function is:

$$f(y) = \max(0, y) \quad (4)$$

Another activation function used is an identity function, also called a linear function. The activation is proportional to the given input and represented in equation 5.

$$f(y) = y \quad (5)$$

c) Pooling

Pooling layers are the next layer after the convolutional layers and are operated for spatial dimension reduction of feature

maps, i.e., they reduce the amount of learning parameters. A pooling layer down-samples the data size using spatial variances. Max pooling selects a maximum element from the region of the filtered feature map as represented in equation 6.

$$output = \left[1 + \frac{input+2\times padding[0]-dilation[0]\times(kernelsize[0]-1)-1}{stride[0]}\right] \quad (6)$$

d) Attention layers

In conventional neural networks, cognitive attention can be imitated using a technique called attention. The attention mechanism is the mechanism that helps the neural network to memorize a long sequence of data. The attention layer is a conduit between the encoder and the decoder that transfers information from hidden states of encoder to the decoder. It aids a neural network in the memorization of a large series of data. The model can focus selectively on significant parts of the input sequence, and hence identify the relationship between them. The attention layer is built to permit the use of the most relevant parts of the input sequence flexibly to decoder by accumulation of all encoded input vectors to a weighted combination, with the greatest weights received by the most relevant vectors [42]. Normalization of the output score of a feed-forward neural network is represented by a function that preserves the congruence between input at j and output at i generates the attention weights.

$$\beta_{ij} = \frac{\exp(a_{ij})}{\sum_{r=1}^{\tau_y}\exp(a_{ir})} \quad (7)$$

β_{ij} are the weights, τ_y is window size, and the sum of all weights within one window is equal to 1 in equation 7.

e) Flattening layer.

Flattening is the process of converting multiple-dimension input to a single-dimension array. The output generated by the convolutional layers is flattened to a single 1-D feature tensor. In simple terms, a flattening layer merges all the input layers into a single layer. The flattening layer adds an extra channel dimension to the input shape without a feature axis [43]. The use of flattening layers is to permit changes to the input shape from a vector of n-D matrixes to the correct shape for interpretation of dense layer. It collapses an input spatial dimension to a channel dimension. For example, if the output given by a convolutional layer is a 3D array of shapes, 32x16x8 then after flattening the shape will be 32*16*8=4096 units. The encoder architecture elaborated is represented in the figure. 4.

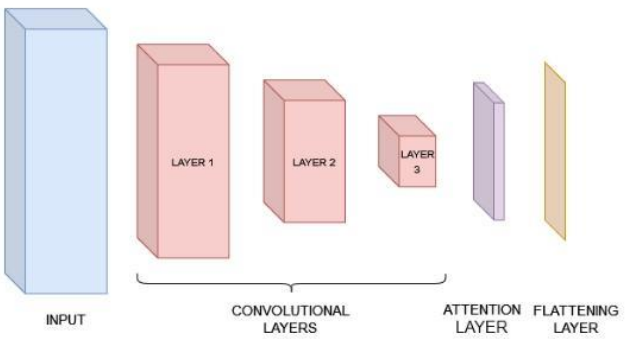

Figure 4. Encoder

2) Decoder

A decoder is the second component of an autoencoder that maps the code to reconstruct the compressed version of the in)-1put. A decoder is the inverse CNN structure, which produces output using the 1D vector. The goal of the AE is to produce an identical copy of the input. Thus, the decoder architecture has to be a mirror image of the encoder. The basis of the architecture of a decoder is that the input and output dimensionalities must be equal. The decoder takes the encoder output with the bottleneck layer, and recreates/regenerates the input. The decoder network consists of 3 layers: *a)* Dense layer, *b)* Deconvolution, *c)* Upsampling as shown in figure. 5.

a) Dense layer

A dense layer is the most basic layer in neural networks that feeds its neurons all the output of the preceding layer, supplying one output for every neuron to the next layer. It is a regular deep fully connected neural network, a frequently implemented layer. The dense layer mainly performs matrix-vector multiplication, where the used values are the self-trained parameters that can be updated through back-propagation. It applies several operations like scaling, rotation and translation on a vector, but primarily they change the vector dimensions, generating an output of an m-D vector.

b) Deconvolution

Deconvolution is a mathematical operator that transposes convolution. It is an unsupervised technique for convolutional decomposition based on a sparsity constraint [44]. Its main application is for those networks which reconstruct the input. In deconvolution, the dimension of output is more than that of input. As deconvolution is the transpose of convolution, it can be mathematically expressed in the form of linear system as shown in equation 8.

$$[o] = [K]\cdot[y] + [e] \quad (8)$$

Where, [y] is the matrix of unknown signals, [o] is a vector of observations, [K] is a known matrix, and [e] a vector of random errors.

c) Upsampling

Upsampling is a technique used to raise the sampling rate by insertion of zero-valued samples in between the original samples. Upsampling is the manipulation of signals for the artificial increment of the sampling rate. Upsampling improves resolution, anti-aliasing filter performance and reduces errors. In simple words upsampling is a simple scaling up of input image using nearest neighbours (bi-linear upsampling). Considering s is the factor for upsampling operation, then the $(i, j)^{th}$ element of the upsampling matrix $u_{a,b}$ in equation 9.

$$\{u_{a,b}\}_{i,j} = \uparrow s(\beta[i-j]) = \beta\left[\frac{i}{s} - j\right] = b[i - sj] \quad (9)$$

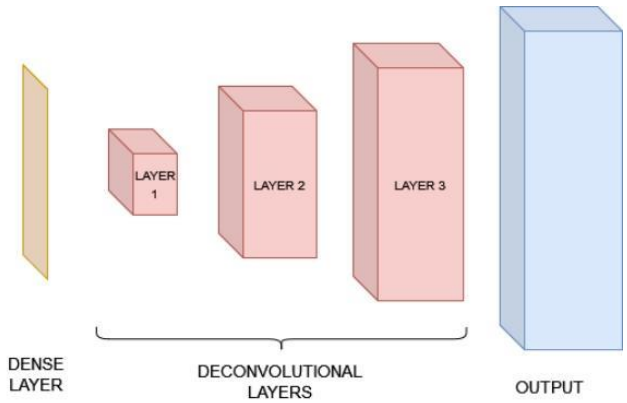

Figure 5. Decoder

IV. Proposed Method

First, normalization of HSI images is done to avoid input shape errors followed by optimization of the neural network. An Adam optimizer is applied with its parameters, the learning rate of 0.0009, default for both the exponential decay, with beta_1=0.9 and beta_2=0.999 [45], followed by model implementation.

This paper uses the dimension reduction property of the autoencoder for HSI compression. The illustration of the proposed autoencoder architecture is shown in algorithm 1. Elaborating the framework, both encoder and decoder have a set of convolutional, max-pooling, and dense layers. As the first step, input fed to three convolutional layers paired with max-pooling layers, a padding size of kernel = 2, 2 was used. The first convolutional layer has 128 filters and a ReLU activation function if the number of bands is greater than 128. The second layer with 64 filters is included, followed by the third convolutional layer of 32 kernels with the tanh activation function. Every convolutional layer is connected to a max pool layer for the reduction of pixels from earlier convolutional layers, to reduce input Dimensionality. Lastly, after the third convolutional layer, an attention layer is added followed by a flattening layer of 8192 units.

ALGORITHM 1: MODIFIED AUTOENCODER ALGORITHM FOR COMPRESSION

Result: Decoded bit streams of the compressed image
Input: Hyperspectral Image, his, of dimension (rw x cl x b), Here rw = number of rows, cl = number of columns, b = number of bands
Attention Layer: Generate custom layer using keras Default layer Class. Four function as arguments for Keras custom layer generation rule. Function build () To define weight and biases (w and B). The call () for multilayer perceptron (MLP) with tanh Followed by softmax layer. To return the shape of the built layer, a compute_output_shape () function is added, along with get_config () for gathering all the information About the custom model.
Initial Parameters: For optimized neural network Adam optimization algorithm is taken, with $\alpha = 0.0009$, $\beta1 = 0.9$ and $\beta2 = 0.999$. Here α represents the learning rate of the model, and $\beta1$ and $\beta2$ respectively are the exponential Decay rates of 1st and 2nd moment estimation.
Representation: The image is normalized using the z score And reshaped into a 4-D matrix, i.e. (1, rw, cl, b).

Algorithm:
// Bitwise Normalization
Normalize data using z- score normalization
Reshape to 4-D array # reshaping into (1, rw, cl, and b).
// Encoder Model
Conv2D layer with 220 filters and ReLU function;
Max pooling2D layer;
Conv2D layer with 128 filters and ReLU function;
Max pooling2D layer;
Conv2D layer with 64 filters and ReLU function;
Max pooling2D layer;
Conv2D layer with 32 filters and ReLU function;
Max pooling2D layer;
Attention layer;
Flattening layer;
// Decode Model
Dense layer with ReLU function;
Layer reshape;
Upsampling2D layer;
Conv2DTranspose layer with 32 filters and ReLU function;
Upsampling2D layer;
Conv2DTranspose layer with 64 filters and ReLU function;
Upsampling2D layer;
Conv2DTranspose layer with 128 filters and ReLU function;
Upsampling2D layer;
Conv2DTranspose layer with 220 filters and ReLU function;
Conv2DTranspose layer with 220 filters and Linear function;
Resize to (rw, cl)

The output obtained from the encoder is the input for the decoder. So, 8192 units are taken as the unit for the dense layer. It is followed by two deconvolutional layers, or transpose of convolutional layers, further followed by upsampling layers. Then again three deconvolutional layers are added. A regenerated image of a similar shape to that of the input image is obtained as the final output. To minimize loss as much as possible, 120 to 350 epochs are used for training. Figure 6 shows the pipeline of the proposed convolutional autoencoder.

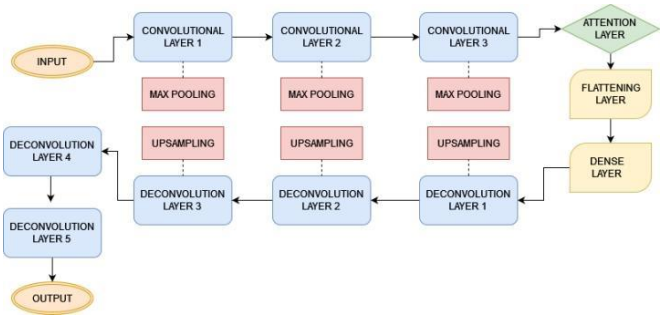

Figure 6. Proposed Convolutional Autoencoder Architecture

V. Implementation and Results

To validate the proposed model simulation results are presented in this section. Here the details of the datasets used, the experimental configurations, and the obtained results are mentioned. The RGB images of the datasets are shown in Figure 7. The section is subdivided as:
A. Dataset, *B.* Experimental setup, *C.* Comparison and Analysis of Algorithms

A. Dataset

To assess the proposed compression technique, widely used Hyperspectral datasets are adopted. The framework is performed in four real-world datasets: Indian Pines, Kennedy Space Center (KSC), the Salinas Scene, and the University of Pavia datasets. For research purposes, these datasets are publicly available in [41]. Each dataset has multiple sets of a single image. Table 1 specifies the number of sets present in each dataset. We will briefly discuss each HSI dataset in its sub division: *1* Indian Pines, *2.* Kennedy Space Center (KSC), *3.* Salinas Scene, and *4.* University of Pavia

Table 1. Hyperspectral Imaging Dataset Used

DATASET	PIXEL	BAND	CLASS	IMAGE No.
Indian Pines	145*145	200	16	200
Kennedy Space Center (KSC)	512*614	176	13	126
Salinas Scene	512*217	204	16	204

1) Indian Pines

Indian Pines is the first dataset used for hyperspectral image compression in this paper. In June 1992, NASA used an AVIRIS sensor to gather the hyperspectral scene over Indian Pines, North-western Indiana test site. The images are of 145X145 pixels and have 220 spectral reflectance bands in the wavelength range 0.4–2.5 10^ (-6) meters (µm). The 220 wavebands of the image were continuously captured via AVIRIS on the specified wavelength and at approximately 20m of spatial resolution. Recently, by removing the water absorption region covering bands ({104-108}, {150-163}, 220), the dataset is corrected by reducing 220 bands to 200 bands. The ground truth is categorized into sixteen classes, shown in table 1.

2) Kennedy Space Center (KSC)

The second real-world dataset used is the site of mixed vegetation over the Kennedy Space Center (KSC), Florida, and USA which was acquired in 1996 by the National Aeronautics and Space Administration (NASA) Airborne Visible/Infrared Imaging Spectrometer instrument. It had 224 spectral bands when it was collected ranging from 0.4 to 2.5µm wavelengths, with a spatial size of 512 × 614 pixels with 5211 labelled pixels, and 18m spatial resolution. Due to the low signal-to-noise ratio (SNR) and water absorption, 48 bands are discarded during pre-processing. As a result, for the classification, 176 spectral bands are used. In the original dataset and the ground truth, 13 land cover types are represented in table 1.

3) Salinas Scene

The third dataset used in this paper is the Salina scene, acquired over Salinas Valley, California, USA, by 224 band AVIRIS Imagining spectrometer sensor, and characterized via high spatial resolution of 3.7-meter per pixel. The dimension of the Salinas scene was 512X217 pixels with 224 spectral bands, but 20 bands of water absorption region were discarded: {108-112}, {154,167}, 224. As shown in table 1, 16 classes are categorized on the ground truth.

4) University of Pavia

In this paper it is the fourth HSI dataset used for compression. The University of Pavia dataset is an HSI dataset acquired over Pavia, northern Italy in 2003 by an airborne reflectance optical spectrometer sensor, the Reflective Optics Spectrographic Imaging System, ROSIS-03. Initially, the Pavia University dataset was of 610*610 pixels with 115 spectral bands, but some images on the sample had no information in it, so it was discarded to 610X340 pixels and due to the presence of noise, 15 wavebands were removed, so the corrected dataset consisted of 610X340 pixels and 103 spectral bands. The ground truth is differentiated into 9 classes shown in table 1.

B. Experimental setup

The proposed compression framework and all other architectures are implemented in the personal hardware workstation powered by windows 11 with Intel(R) Core (TM) i5-8250U 8th generation and CPU @ 1.80 GHz and 16GB Random Access Memory (RAM), and along with this, all the neural network libraries like Keras, TensorFlow used are coded in python 3.8.8. For the effectiveness evaluation of the proposed framework, two measurement metrics used are Peak Signal to Noise Ratio (PSNR), and Compression Ratio (CR).

The ratio of maximum signal power possible and the corrupting noise power affecting the representative's fidelity is known as PSNR. PSNR is utilized for measuring quality of the reconstructed lossy compression codecs, also known as image

Indian Pines | Kennedy Space Center (KSC) | Salinas Scene | University of Pavia

Figure 7. RGB image of the HSI Dataset used

quality metric. Equation 10 expressed PSNR mathematically, where is the maximum pixel value possible and mse represents mean square error, which is a quantification of the error difference between the original and the reconstructed image.

$$PSNR = \frac{M_{P_p}}{mse} \quad (10)$$

Compression Ratio (CR) also called Data Compression Ratio, and compression power is the measurement metric of relative reduction of data representation size achieved by compression techniques. In simple words, CR is the ratio of uncompressed to compressed size as shown in equation 11.

$$CR = \frac{Uncompressed\ (Original) size\ of\ image}{Compressed\ size\ of\ image} \quad (11)$$

The intention of compression is to minimize the amount of input data, simultaneously preserving as much information as feasible. To put it another way, it means optimizing the lossy data with maximization of CR and PSNR.

A. Comparison and Analysis of Algorithms

This section discusses the comparative comparison carried out to inspect the quantitative superiority of the proposed HIS compression framework including the state-of-art: NFTD + PCA, and 3D DWT+SVR. We explored and analysed the behaviour of AE for better performance on the basis of PSNR and CR.

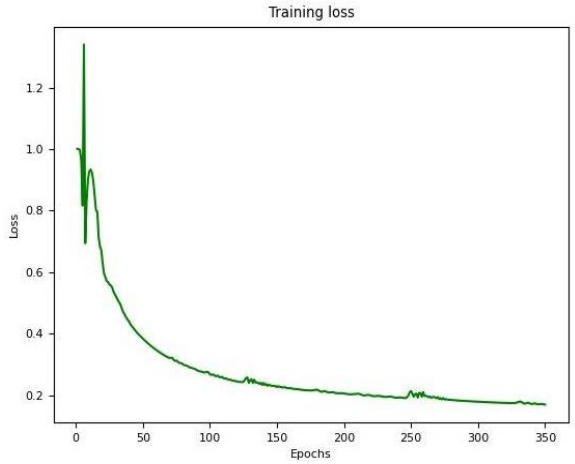

Figure 8. Model Loss

1) Construction

Initially, the proposed architecture's performance is assessed via the calculation of the obtained loss against the number of epoch functions. Training and testing splitting was randomly distributed for better performance. The obtained graph of gained loss (y-axis) based on epochs (x-axis) is shown in Figure 8. From the graph we can suggest, in between 0 to 25 epochs the training loss was immense with lots of fluctuations and after 25 epochs the loss gradually declined and in between 300 to 350 it started attending consistency, due to its static level, the optimal number of epochs selected for the model was 350 epochs. Due to variation in dataset dimension, each filter size (kernel size) is tuned as per the input size. To avoid data loss and image splitting, we have not considered partitioning the datasets into the training and testing phase as it will result in the splitting of spectral dimensions that will cause image distortion. Initially, the proposed architecture comprised of three convolutional layers, with each being connected to a max-pooling layer, then bottleneck layers as encoders, and in the decoder, five de-convolutional layers (transposed convolution layers) along with upsampling.

2) Filters

The proposed CAE framework uses N input neurons with H hidden neurons, where N stands for the number of spectral bands in each dataset. As the framework is completely connected, every hidden layer is attached to every input neuron. Thus, on the whole, the N connection acts as a filter, since it filters out some information from the input representing wavelengths and simultaneously overemphasize others. For improvement in the model, we experimented with various hidden layers, such as the attention layer, flattening layer, and dense layer in both encoder and decoder. Six experimental models are compared in table 2. The size of every layer varies as per the dataset dimension. From the table, it can be understood that the autoencoder with attention layer followed by flattening and dense layer performs best compared to other architectures on the basis of CR and PSNR, and size of dense layer depends proportionally to the size of data.

3) Comparison

For efficiency evaluation of our proposed framework, we evaluated based on compression ratio and PSNR obtained from three state-of-art algorithms. We implemented the NFTD + PCA by taking principal components as 16, as the dataset consists of large spectral bands. Along with this, we implemented 3D DWT+SVR and CAE proposed in [2]. In table 3, a comparison of the average results is represented because of its stability. The experiment was executed 20 times so that it authenticates the coherence of the proposed architecture. Table 4 shows the top 10 achieved PSNR for the University of Pavia dataset. It is observed that the modified CAE outperforms all other algorithms on the basis of CR and PSNR. The proposed architecture performs incredibly well for the Salinas dataset. The reason for the better performance is its 3.7 m per pixel spatial resolution. In table 5, we compare the original size of each file to the obtained compressed file size in bytes. The compressed file size of the dataset obtained by the University of Pavia decreased the most, and the original file size decreased by 9655 times.

The size of the original Salinas dataset undergoes around 7950 times decrement and for the KSC dataset, the original file size is compressed by 1177 times, and 2713 times reduction is seen in Indian pines compressed file size. The results obtained from the modified CAE and other state-of-art algorithms are graphically represented in Figure 9. Figure 9a shows that in the

Table 2. Filter comparison

MODELS (M)	ATTENTION LAYER	DENSE LAYER		CR	PSNR
		ENCODER	DECODER		
DIMENSION		**8192**			
M1	✓	×	×	441.238	53.008
M2	×	✓	×	389.043	51.917
M3	×	×	✓	502.576	53.893
M4	✓	✓	×	1105.673	48.128

Table 3. Comparison

DATASET	NFTD+PCA		3D DWT+SVR		CAE		MODIFIED CAE	
	PSNR	CR	PSNR	CR	PSNR	CR	PSNR	CR
Indian Pines	49.68	130.25	44.19	27.02	55.19	128.33	55.89	2713.95
Kennedy Space Center (KSC)	47.06	235.12	41.56	26.35	53.32	371.67	53.55	5756.14
Salinas Scene	46.47	272.23	42.28	21.53	54.46	447.42	58.13	7950.95
University of Pavia	47.11	137.51	43.59	25.95	55.26	157.71	56.74	3194.47

Table 4. Top 10 achieved PSNR for the University of Pavia dataset

University of Pavia		
EXPERIMENT NO.	PSNR	CR
EXPERIMENT 1	52.06	6521.80
EXPERIMENT 3	59.30	7049.02
EXPERIMENT 4	51.00	8561.31
EXPERIMENT 7	54.21	6958.42
EXPERIMENT 8	52.69	4237.65
EXPERIMENT 9	57.28	7562.60
EXPERIMENT 11	55.13	6851.32
EXPERIMENT 12	51.33	7963.14
EXPERIMENT 15	50.3	6753.78
EXPERIMENT 19	52.6	6025.36

Table 5. Size Comparison

DATASET	ORIGINAL SIZE (in Bytes)	COMPRESSED SIZE (in Bytes)
Indian Pines	6296374	2320
Kennedy Space Center (KSC)	56824624	48272
Salinas Scene	27605707	3472
University of Pavia	95320136	9872

proposed modified CAE, represented by yellow bar, CR is increased by approx. 18 times than the state-of-art CAE method, 200 times increment from 3D DWT+SVR, and 25 times of NFTD+PCA. The CR for the Salinas dataset achieved is 355 times better in modified CAE than 3D DWT+SVR.

Figure 9b compares the PSNR gained in the proposed architecture, signified by yellow bar, to other state-of-art compression architecture and confirms that the modified CAE performed better by around 5% in terms of PSNR. The estimation for only 0.23 unit increment in PSNR for the KSC dataset is that due to the larger dimension (512 x 614 x 176), the quality improvement is slighter during compression.

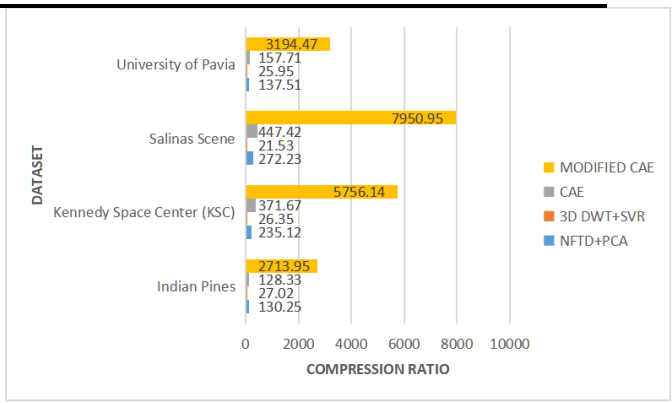

Figure 9a. Comparison of CR of proposed model with state-of-art algorithms

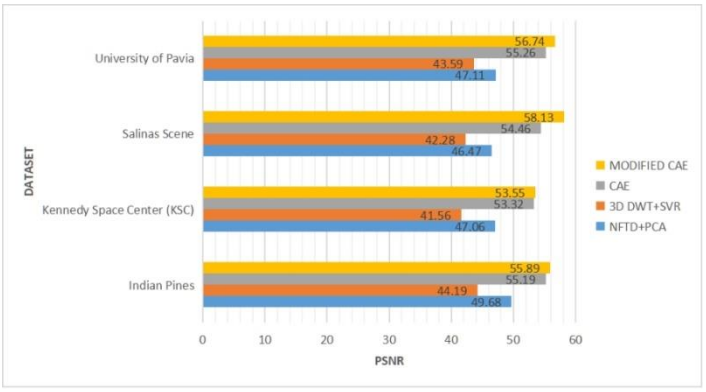

Figure 9b. Comparison of PSNR of proposed model with state-of-art algorithms

Figure 9. Comparison graph for CR, PSNR

The absolute difference between modified CAE and CAE in PSNR achieved for the Salinas dataset is 3.67 units. The reason behind the improvement in CR and PSNR is the addition of the attention layer in the encoder that memorizes large sequences of spatial features, which consequently improves the compression rate after decoding. The illustration of the original image along with the image reconstructed by the modified CAE is shown in Figure 10.

XI. Discussion

The main problem with hyperspectral image, apart from its high cost, is the large storage requirement due to its multiple bands and spectral and spatial redundancy. As compared to RGB images (3 band imagery), usually HSI consists of more than 100 bands, for storage of spectral information from each band. As a result, the data size is immense.

The main innovation and the primary goal of this work is to compress the data with as less loss of information as possible

utilizing the strength of an unsupervised artificial neural network, autoencoder, i.e., reconstruction of compression data from the reduced encoded representation as similar to the original input as possible by extracting key features of the hyperspectral Images and storing it into a 3D tensor. The depressed version can be reconstructed in its original using the decoder, as using transpose convolutional layers reverses the steps of CNN.

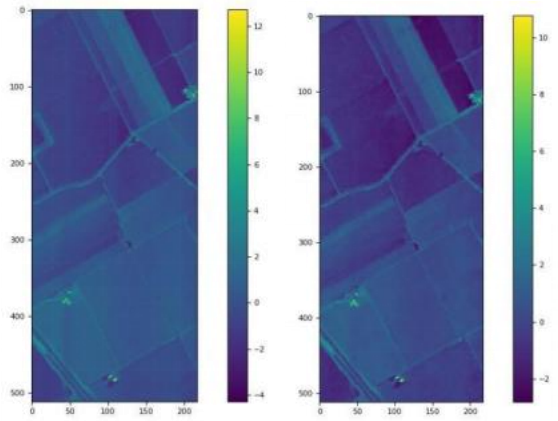

Figure 10. Original image (left) and reconstructed image (right)

However, [2] used entropy coding as the bottleneck to minimize the number of its bits for a unique representation of input data. It is a lossless coding technique that uses code words for decoding, and even a corrupted single bit in the code word can make the entire message corrupted. Along with this drawback, another one, is its limitation on the precision of the encoded number, thus limiting the number of symbols within the code word to be encoded. The experiments show that the architecture proposed in the paper outperforms state-of-art compression techniques, using the following configuration: input of size according to its data dimension is fed to three consecutive convolutional layers accompanied by a max-pooling layer of each, further connected to an attention layer as encoder and flattening layer as the bottleneck, and the decoder consists of a dense layer followed by five deconvolutional layers along with upsampling layers for reconstructing the compressed image. The baseline of image compression is compressing the data and then reconstructing the image by decompression for compression of data encoder is required, and for decompression of data decoder is needed.

An autoencoder is the sole composition of encoder and decoder. In general, the efficient neural network for dealing with an image is a convolutional neural network without CNN computational time, and computational power required will be immense and along with this, the neighborhood information of each pixel will be lost, so considering both the intuition, the combination of autoencoder with CNN is the most coherent architecture for image compression. The advantages of the proposed architecture, modified CAE are as follows:

- Convolutional Autoencoder: Autoencoders are data compression models, used to encode input data to smaller dimension representatives. The dimension reduction attribute of autoencoder in addition to the feature extraction attribute of CNN results in an efficient image compression technique.
- Attention mechanism: The ability to identify the information most relevant to accomplishing the given task from input makes the attention mechanism incredibly suitable for identifying features and providing it to CNN that performs feature extraction. The method confirms that the addition of the attention layer and flattening layer to the autoencoder assist better image compression than state-of-art algorithms.

XII. Conclusion & Future Work

In this work, we employ a modified convolutional autoencoder for compression of hyperspectral images. Along with a fundamental convolutional autoencoder framework, a few modifications are made like an attention layer is added to the encoder, a flattening layer as the bottleneck, and a dense layer is connected in the decoder. The algorithm is tested for its compression performance on four benchmark hyperspectral datasets that are Kennedy Space Center, Indian Pines, Salinas Scene, and the University of Pavia dataset. The achieved results depict that the proposed algorithm is able to perform significantly better than the compared work with respect to both compression ratio and peak-signal-to-noise ratio. As the properties of each used image differ from others, the obtained result varies for each dataset. From the state-of-art, CAE CR increased by around 18 times and a 5% gain is obtained in PSNR for the proposed architecture, and apart from this, a maximum of 7950 times drop is seen in the compressed file size to that of the original file size.

In terms of future work on compression of hyperspectral images, a variety of autoencoders can be trained for compression other than convolutional autoencoders, and neural network architectures like recurrent neural networks, and many other approaches can be implemented. The architectures can be further modified by fine-tuning and pre-processing the hyperspectral images, and the model can be better optimized with Stochastic Gradient Descent (SGD) as a parameter followed by cross-entropy for loss function. Keeping in mind the dimension of each image, better performance can be achieved with complex neural networks with higher computational power and powerful GPU. By the usage of higher computing power and with improved and modified hardware, the computational time can be decreased.

References

[1] R. Dusselaar & M. Paul, "Hyperspectral image compression approaches: opportunities, challenges, and future directions: discussion," *JOSA A*, vol. 34, no. 12, pp. 2170-2180, 2017.

[2] Y. Dua, R. S. Singh, K. Parwani & S. Lunagariya, V. Kumar, "Convolution Neural Network based lossy compression of hyperspectral images," *Signal Processing: Image Communication*, vol. 95, p. 116255, 2021.

[3] *NASA*, 123.0-b-info testdata, 2015, URL: [link]

[4] "Hyperspectral Image Processing," in *Science Direct Topics, Elsevier*, [link].

[5] R. Yamashita, M. Nishio, R. K. G. Do & K. Togashi, "Convolutional neural networks: an overview and application in radiology," *Insights into imaging*, vol. 9, no. 4, pp. 611-629, 2018.

[6] A. Ertem, A. C. Karaca, O. Urhan & M. K. Güllü, "Superpixel based compression of hyperspectral image with modified dictionary and sparse representation,"

International Journal of Remote Sensing, vol. 41, no. 16, pp. 6307-6324, 2020.

[7] Y. Barrios, A. Rodríguez, A. Sánchez, A. Pérez, S. López & A. Otero, R. Sarmiento, "Lossy hyperspectral image compression on a re-configurable and fault-tolerant fpga-based adaptive computing platform," *Electronics*, vol. 9, no. 10, p. 1576, 2020.

[8] F. Zhu, H. Wang, L. Yang, C. Li & S. Wang, "Lossless Compression for Hyperspectral Images based on Adaptive Band Selection and Adaptive Predictor Selection," *KSII Transactions on Internet and Information Systems (TIIS)*, vol. 14, no. 8, pp. 3295-3311, 2020.

[9] D. Báscones, C. González & D. Mozos, "Hyperspectral image compression using vector quantization, PCA and JPEG2000," *Remote sensing*, vol. 10, no. 6, p. 907, 2018.

[10] Z. Cheng, H. Sun, M. Takeuchi & J. Katto, "Deep convolutional autoencoder-based lossy image compression," in *Picture Coding Symposium (PCS)*, 2018, pp. 253-257.

[11] D. Mishra, S. K. Singh & R. K. Singh, "Lossy Medical Image Compression using Residual Learning-based Dual Autoencoder Model," in *IEEE 7th Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON)*, pp. 1-5, 2020.

[12] C. Deng, Y. Cen, and L. Zhang, "Learning-based hyperspectral imagery compression through generative neural networks," *Remote Sensing*, vol. 12, no. 21, p. 3657, 2020.

[13] V. Alves de Oliveira, M. Chabert, T. Oberlin, C. Poulliat, M. Bruno, C. Latry & R. Camarero, "Reduced-Complexity End-to-End Variational Autoencoder for on Board Satellite Image Compression," *Remote Sensing*, vol. 13, no. 3, p. 447, 2021.

[14] M. Ouahioune, S. Ameur & M. Lahdir, "Enhancing hyperspectral image compression using learning-based super-resolution technique," *Earth Science Informatics*, vol. 14, no. 3, pp. 1173-1183, 2021.

[15] W. Wang, D. Yang, F. Chen, Y. Pang, S. Huang & Y. Ge, "Clustering With Orthogonal AutoEncoder," *IEEE Access*, vol. 7, pp. 62421-62432, 2019.

[16] V. H. Ayma, V. A. Ayma & J. Gutierrez, "Dimensionality Reduction via an Orthogonal Autoencoder Approach for Hyperspectral Image Classification," in *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, vol. 43, 2020.

[17] J. Kuester, W. Gross & W. Middelmann, "1D-Convolutional Autoencoder Based Hyperspectral Data Compression," in *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 43, pp. 15-21, 2021.

[18] M. Kowsher, M. A. Alam, M. J. Uddin, F. Ahmed, M. W. Ullah & M. R. Islam, "Detecting Third Umpire Decisions & Automated Scoring System of Cricket," in *International Conference on Computer, Communication, Chemical, Materials and Electronic Engineering (IC4ME2)* , pp. 1-8, 2019.

[19] K. O'Shea & R. Nash, "An introduction to convolutional neural networks," *ArXiv pre-print arXiv: 1511.08458*, 2015.

[20] H. Kagaya, K. Aizawa & M. Ogawa, "Food detection and recognition using convolutional neural network," in *Proceedings of the 22nd ACM international conference on Multimedia*, pp. 1085-1088, 2014.

[21] B. Gambäck & U. K. Sikdar, "Using convolutional neural networks to classify hate-speech," in *Proceedings of the first workshop on abusive language online*, pp. 85-90, 2017.

[22] S. M. Anwar, M. Majid, A. Qayyum, M. Awais, M. Alnowami & M. K. Khan, "Medical image analysis using convolutional neural networks: a review," *Journal of medical systems*, vol. 42, no. 11, pp. 1-13, 2018.

[23] F. Sultana, A. Sufian, and P. Dutta, "Advancements in image classification using convolutional neural network," in *Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*, pp. 122-129, 2018.

[24] S. Hijazi, R. Kumar & C. Rowen, "Using convolutional neural networks for image recognition," *Cadence Design Systems Inc.: San Jose*, CA, USA, 2015.

[25] L. Gatys, A. S. Ecker & M. Bethge, "Texture synthesis using convolutional neural networks," in *Advances in neural information processing systems*, pp. 28, 2015.

[26] B. Fasel, "Robust face analysis using convolutional neural networks," in *Object recognition supported by user interaction for service robots*, vol. 2, pp. 40-43, 2002.

[27] C. Pramerdorfer & M. Kampel, "Facial expression recognition using convolutional neural networks: state of the art," *ArXiv preprint* arXiv: 1612.02903, 2016.

[28] I. J. Goodfellow, J. Shlens & C. Szegedy, "Explaining and harnessing adversarial examples," *ArXiv preprint* arXiv: 1412.6572, 2014.

[29] R. Yamashita, M. Nishio, R. K. G. Do, et al., "Convolutional neural networks: an overview and application in radiology," *Insights Imaging*, vol. 9, pp. 611-629, 2018.

[30] A. Habibian, T. V. Rozendaal, J. M. Tomczak & T. S. Cohen, "Video compression with rate-distortion autoencoders," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7033-7042, 2019.

[31] "Deep inside Autoencoders," Towards Data Science, available at: [Link].

[32] W. Li, H. Fu, L. Yu, P. Gong, D. Feng, C. Li & N. Clinton, "Stacked Autoencoder-based deep learning for remote-sensing image classification: a case study of African land-cover mapping," *International journal of remote sensing*, vol. 37, no. 23, pp. 5632-5646, 2016.

[33] G. Abdi, F. Samadzadegan & P. Reinartz, "Spectral-spatial feature learning for hyperspectral imagery classification using deep stacked sparse autoencoder," *Journal of Applied Remote Sensing*, vol. 11, no. 4, p. 042604, 2017.

[34] H. Fu, P. Lei, H. Tao, L. Zhao & J. Yang, "Improved semi-supervised autoencoder for deception detection," *PloS one*, vol. 14, no. 10, p. e0223361, 2019.

[35] S. Saravanan & J. Sujitha, "Deep medical image reconstruction with autoencoders using deep Boltzmann machine training," *EAI Endorsed Transactions on Pervasive Health and Technology*, vol. 6, no. 24, p. e2, 2020.

[36] L. Yasenko, Y. Klyatchenko & O. Tarasenko-Klyatchenko, "Image noise reduction by denoising autoencoder," in *IEEE 11th International*

Conference on Dependable Systems, Services and Technologies (DESSERT), pp. 351-355, 2020.
[37] K. Haribabu, G. R. K. S. Subrahmanyam & D. Mishra, "A robust digital image watermarking technique using autoencoder based convolutional neural networks," in *IEEE Workshop on Computational Intelligence: Theories, Applications and Future Directions (WCI)*, pp. 1-6, 2015.
[38] Y. Yang, Q. J. Wu & Y. Wang, "Autoencoder with invertible functions for dimension reduction and image reconstruction," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 48, no. 7, pp. 1065-1079, 2018.
[39] W. Xu, S. Keshmiri, and G. Wang, "Adversarially approximated autoencoder for image generation and manipulation," *IEEE Transactions on Multimedia*, vol. 21, no. 9, pp. 2387-2396, 2019.
[40] "Autoencoder Image Compression with Keras," *Paper space*, available at: [Link]
[41] "Hyperspectral Remote Sensing Scenes," *Center for Coastal and Watershed Studies*, available at: [Link]
[42] "The Attention Mechanism from Scratch," *Machine Learning Mastery*, available at: [Link]
[43] "Flatten Layer - Keras Documentation," *Keras*, available at: [Link]
[44] S. Issa and A. R. Khaled, "Lower Limb Activity Prediction Using EMG Signals and Broad Learning," *International Journal of Computer Information Systems and Industrial Management Applications*, vol. 14, pp. 162-172, 2022.
[45] "Adam - Keras Documentation," Keras, available at: [Link]
[46] S. Dolgikh, "Generative Conceptual Representations and Semantic Communications," *International Journal of Computer Information Systems and Industrial Management Applications*, vol. 14, pp. 239-248, 2022.

Author Biographies

Satvik Agrawal was born in Ajmer, Rajasthan, India in the year 2001. He completed a majority of his schooling from Pune after which he enrolled at Kalinga Institute of Industrial Technology, Bhubaneshwar to study computer science engineering. He has worked on numerous papers related to machine learning and is on track to graduate with exemplary grades.

Sancharika Debnath, a native of West Bengal, India, was born in 2001 and is currently in her final year of undergraduate studies at Kalinga Institute of Industrial Technology, pursuing a B.Tech. Degree in Information Technology. With a passion for cutting-edge technologies, her interests lie in various domains of the field, including Unsupervised Learning, Transfer Learning, Machine Learning, Deep Learning, and Web Development. She actively applies her knowledge and skills in these areas to tackle real-world challenges, such as text mining, recommendation systems, fraud detection, and more.

Santwana Sagnika is an Assistant Professor at School of Computer Engineering, KIIT Deemed to be University. She has completed her Ph.D. in Computer Science and Engineering from KIIT Deemed to be University, Bhubaneswar, India. Her areas of interest include Natural Language Processing, Machine Learning, Optimization techniques and Cloud Computing. She has published more than 25 papers in renowned journals and conferences and is a regular reviewer for Springer, Elsevier, Taylor and Francis, and SAGE journals.

Saurabh Bilgaiyan is currently working as an Assistant Professor in KIIT deemed to be University, Bhubaneswar, India since 2016. He has completed his Master's and Ph.D. in Computer Science Engineering from KIIT, Deemed to be University, Bhubaneswar, India in 2014 and 2018 respectively. Bachelor's degree of B.E. in Information Technology from B.I.R.T., Bhopal, India in 2012. Dr. Saurabh Bilgaiyan has published more than 50 research papers in various reputed International Journals, Conferences, and edited books. His area of interest includes soft computing, software engineering, cloud computing, image processing, and machine learning. He has reviewed various manuscripts in more than 25 International conferences and Journals including Soft Computing Springer, IEEE Access, IETE Journal of Research Taylor and Francis, Future Generation Computer System Elsevier, Concurrency and Computation: Practice and Experience Wiley, etc.

Saksham Gupta, born in 1998, did most of his schooling and Bachelor's of Engineering in computer science from Chandigarh. He then shifted to Hyderabad in 2023 as a student in the Indian School of Business where he is studying management. He has worked in the fields of Computer Vision, Neural Networks and Machine Learning.