BTC-Net: Efficient Bit-Level Tensor Data Compression Network for Hyperspectral Image

Xichuan Zhou, *Senior Member, IEEE*, Xuan Zou, Xiangfei Shen, Wenjia Wei, Xia Zhu, and Haijun Liu, *Member, IEEE*

***Abstract*—Now it is still a challenge to compress high-throughput hyperspectral tensor image data on lightweight air-carried/spaceborne remote sensing systems, primarily due to insufficient computational resources and limited transmission bandwidth. To address this challenge, we propose a bit-level tensor data compression network (BTC-Net) that provides higher compression performance by leveraging a data-driven lightweight quantized neural encoder with two-stage bit compression. The BTC-Net achieves semantic near-lossless high reconstruction quality at low compression bit rates thanks to its optimized decoder, which uses a channelwise attention-based enhancement module to recover hyperspectral tensor data. Experimental results on different hyperspectral datasets show that the BTC-Net could achieve an extremely low compression bit rate of fewer than 0.04 bits per pixel per band (bpppb) with the state-of-the-art (SOTA) reconstruction performances. The demo of BTC-Net will be publicly available online at: https://github.com/zx20173646/BTCNet.**

***Index Terms*—Bit-level compression, deep neural network (DNN), feature enhancement, tensor data.**

I. Introduction

As THE typical high-dimensional tensor data, hyperspectral data are a hotspot line of science research and engineering applications, such as terrain classification [1], [2], object recognition [3], environmental monitoring [4], *etc*. Hyperspectral images (HSIs) record hundreds of continuously spaced narrow spectral bands in each pixel leading to high dimensionality issues. In natural scenes, directly transmitting observed hyperspectral data is sometimes infeasible due to the limited bandwidth of data transmission. But its high spatial and spectral correlations normally result in redundancy [5], [6], which triggers data compression necessities. After compression and transmission, effective reconstruction of compressed data is also necessary for subsequent research and applications of hyperspectral data [7].

Manuscript received 28 July 2023; revised 25 September 2023 and 16 November 2023; accepted 2 December 2023. Date of publication 5 December 2023; date of current version 15 December 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 61971072, Grant 62001063, and Grant U2133211; in part by the Graduate Research and Innovation Foundation of Chongqing, China, under Grant CYB22068; and in part by the Fundamental Research Funds for the Central Universities under Grant 2023CDJXY-037. *(Corresponding author: Haijun Liu.)*

Xichuan Zhou, Xuan Zou, Xiangfei Shen, and Haijun Liu are with the School of Microelectronics and Communication Engineering, Chongqing University, Chongqing 400044, China (e-mail: haijun_liu@126.com).

Wenjia Wei and Xia Zhu are with Huawei Technologies Company Ltd., Shenzhen 518116, China.

Digital Object Identifier 10.1109/TGRS.2023.3339843

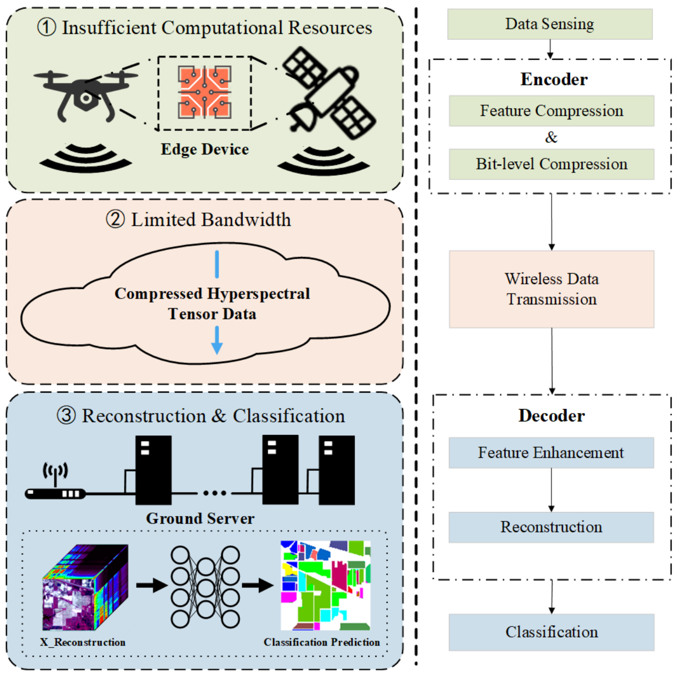


Fig. 1. In the UAV or miniterized satellite based hyperspectral data acquisition scenes, limited wireless bandwidth means that data compression (encoder) is necessary before transmission; Insufficient computational resources on the edge device require compression operation to be low-complexity; Further studies and applications after transmission require effective reconstruction (decoder) from the compressed signal.

Compression and reconstruction of hyperspectral data have attracted the interests of many researchers [5], [8]. However, several challenges can not be ignored in this field, as shown in Fig. 1.

1) *Insufficient Computational Resources:* Since the throughput of a typical hyperspectral sensor is about 50 MB/s, it is normally hard for most air-carried edge devices with insufficient computational and storage resources to process hyperspectral data [9]. Therefore, efficient hyperspectral data process methods need to be taken into consideration.
2) *Limited Bandwidth:* Data transmission always faces the problem of limited bandwidth [5], and this is more severe in the case of hyperspectral data wireless transmission due to its high throughput. For example, the hyperspectral data transmission of the unmanned aerial vehicle (UAV) mainly utilizes the frequency bands of

© 2023 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

2.4 and 5.8 GHz. However, the single-sector transmission rates of 5.8-GHz wireless technology are only up to 54 Mb/s (6.75 MB/s). Moreover, some satellites' downlink communication bandwidth can be as low as 2 Mb/s (0.25 MB/s). Therefore, high compression performance is important for hyperspectral data wireless transmission.

3) *Effective Reconstruction:* The performance of recovering compressed hyperspectral data also needs to be considered. Effective reconstruction results imply better data quality, which benefits the subsequent high-level research and applications, such as the tasks of classification, recognition, and detection. However, there is typically a trade-off between the compression and reconstruction performance [10].

To address the above issues, a series of hyperspectral data compression approaches are proposed including frequency domain-based methods [11], [12], compressive sensing (CS)-based methods [6], [13], and deep neural network (DNN)-based methods [5]. The frequency domain-based approaches have been applied to the compression of high-dimensional tensor data such as videos and HSIs. They split the input data into several blocks, and then adopt the transformation operations combined with extensive technologies to implement the compression [14], [15]. However, they focus on bit-level lossy compression without effective operations to compensate for the degradation of reconstruction performance, resulting in poor reconstruction quality when the compression bit rate is low.

CS [16], [17] has drawn wide attention in recent years, which holds the assumption that signals are sparse or near-sparse in some specific domains and can be recovered from the compressed measurements. The reconstruction of compressed measurements, however, is an ill-posed problem for CS approaches. This issue leads to sparsity-based or total variation (TV)-based regularization techniques [18], [19], [20]. Nevertheless, most of the above CS-based methods have three drawbacks. First, the compression operation of the above methods ignores the potential of bit-level compression, which limits compression performances. Second, the sparsity, TV, or low-rankness assumptions of high-dimensional tensor data in transformation domains are highly demanded, but barely fully capture the insightful prior knowledge underlying the data latent [21]. It may lead to suboptimal reconstruction results, especially in the case of low compression ratios (CRs). Third, in the stage of recovery from the compressed measurements of TV-based methods, the optimization process of parameter tuning requires a heavy computation burden, resulting in the inefficiency of reconstruction.

In recent years, the successful applications of DNN in the field of computer vision [22], [23] trigger its potential in high-dimensional tensor data compression [24], [25], [26]. Compared with the frequency domain-based, DNN-based methods have the ability to conduct bit-level compression. The degradation of reconstruction performance also can be compensated by introducing learnable parameters during the training process [27]. Compared with the CS-based methods, DNN-based methods show two clear advantages. One is that DNN-based methods require fewer assumptions on the prior knowledge underlying tensor data, implying that they can directly learn latent relevant information from training samples and be trained in an end-to-end manner [28]. The other is that the trained DNN models have fixed all parameters, which can efficiently complete the reconstruction from compressed measurements with just one feedforward operation.

These observations motivate us to introduce a DNN-based framework to achieve bit-level compression without sacrificing the capabilities of effective reconstruction from compressed signals. In this article, a new DNN-based framework named bit-level tensor data compression network (BTC-Net) is proposed, which is trained in an end-to-end way. BTC-Net includes a lightweight quantized neural encoder with two-stage data compression and an attention-based super-resolution (SR) decoder. Concretely, to address the challenge of insufficient computational resources on edge devices, a lightweight quantized encoder with fixed-point convolutional weights is first introduced to compress hyperspectral data into multibit fixed-point feature data. Then, to mitigate the impact of limited transmission bandwidth, the compressed feature data would be further compressed by the Huffman coder from the bit-level perspective. Finally, based on the bit-level compressed data, to improve the reconstruction performances, a channelwise attention feature enhancement (CAFE) module is carried out to form our decoder to enhance the representation ability of compressed features, which combines a residual and dense feature enhancement (RDFE) block and a feature channelwise attention (FCA) block.

The advantages of our BTC-Net are summarized as follows.

1) *High Efficiency:* To meet efficiency requirements on edge devices with insufficient resources, we propose a lightweight quantized neural encoder that combines a low-complexity feature compressor and a data-driven quantizer. The compressor is made up of three down-sampled convolutional-LeakyReLU layers that process the quantized input data, using learned quantized convolutional weights provided by the quantizer. Our encoder has a minimal number of fixed-point neural network parameters, totaling only 0.228M, and the computational complexity of encoding is low. This makes our lightweight encoder suitable for deployment on many edge AI chips.
2) *Low Bit Rate:* To mitigate the burden of limited transmission bandwidth, our encoder employs a two-stage compression strategy. The first stage involves feature compression with a low CR, which is accomplished through the use of convolutional operations in the feature compressor. In the second stage, the data-driven quantization is utilized along with Huffman coding to achieve bit-level compression at low bit rates. By adopting this two-stage compression strategy, we are able to achieve an extremely low compression bit rate [less than 0.04 bits per pixel per band (bpppb)], thereby alleviating the constraints posed by data transmission bandwidth.
3) *Semantic Near-Lossless Reconstruction:* We create an attention-based SR decoder that combines the feature enhancement backbone and the spatial–spectral SR

module. The feature enhancement backbone includes a novel mechanism called FCA, which improves feature representation using parallel pooling with combined group convolution and multilayer perceptron. Experimental results demonstrate that the state-of-the-art (SOTA) reconstruction performance can be achieved even at very low compression bit rates. Specifically, classification experiments on the reconstructed data show that the loss of classification accuracy due to compression and reconstruction using BTC-Net is less than 1%, indicating near-lossless semantic reconstruction outcomes.

The rest of this article is organized as follows. Section II reviews related literature. Section III presents the details of our BTC-Net architecture. Then, we demonstrate the experimental settings and results in Section IV. Finally, we conclude the observations in Section V with some remarks.

II. Related Work

In this section, we will briefly review aforementioned three categories of existing compression approaches and also review the hyperspectral classification techniques related to our experiments.

A. Frequency Domain-Based Approaches

Frequency domain-based approaches are originally used for lossy compression of 2-D image data and show reliable performances in which JPEG [29] and JPEG2000 [30] are two classic methods. JPEG transforms images by discrete cosine transform (DCT), and JPEG2000 mainly uses discrete wavelet transform (DWT) to achieve higher compression performance than JPEG. Inspired by the utilization of DWT in image compression, Pearlman et al. [31] proposed a set-partitioned embedded block (SPECK) compression method in which the input image was transformed into blocks by the DWT, and then these blocks were split into subblocks to obtain compressed coefficients. 3-D SPECK [32] was an extensive version of SPECK, which implemented 3-D DWT to the hyperspectral bands and extended the operations of block splitting to the 3-D case for finding the compressed coefficient. Similarly, the extension of JPEG2000 for HSI compression also has been proposed, which compresses each spectral band separately. However, this method only exploited the spatial redundancy, ignoring the spectral redundancy of HSI. To improve the compression performance, Rucker et al. [33] proposed the DWT + JPEG2000, which first implemented wavelet-based transform to decorrelate the spectral bands and then compressed the transformed data in the spatial dimension. Du and Fowler [11] and Du et al. [12] used principal component analysis (PCA) for spectral decorrelation and proposed PCA + JPEG2000 [15].

B. CS-Based Approaches

CS-based approaches utilize the prior knowledge that the high-dimensional tensor image can be sparsely or near-sparsely represented in some specific domains [13], [34], [35]. Based on this fact, Duarte and Baraniuk [18] developed Kronecker products between the sparse basis and sensor matrix for high-dimensional tensor data CS. Waters et al. [36] proposed a low rank and sparse matrix recovery method by considering the global interchannel correlation and intrachannel smoothness, and applied this method to the compression recovery tasks of both videos and HSIs. Li et al. [37] discussed the spectral correlation in HSIs. Based on shown certain joint sparseness in the spatial dimension under the wavelet representation, they proposed a method that utilized low rank and joint sparse matrix. After that, Golbabaee and Vandergheynst [38] combined joint norm and TV regularizations to utilize the correlations among spectral bands and local smoothness prior in hyperspectral data. Different from the foregoing works, Yang et al. [10] directly used tensor-based sparse representation with nonlinear compressed operations, and obtained comparable results. To improve the reconstruction performances under low CRs, Wang et al. [6] proposed a joint tensor tucker decomposition and weighted TV regularization (JTenRe3DTV) to decorrelate and smooth reconstructed HSIs. To well balance the trade-off between 3DTV assumptions and intrinsic properties of hyperspectral data, Peng et al. [13] extended an enhanced 3DTV regularization (E3DTV) term that encoded the correlation and difference among all channels of HSI data.

C. DNN-Based Approaches

Based on DNN architectures, plenty of methods have been applied for various kinds of high-dimensional tensor data compression and reconstruction. The initial investigation of DNN architecture for temporal video CS was proposed by Iliadis et al. [24] that used a deep fully connected network to efficiently and effectively learn the nonlinear mapping between video sequences and the compressed data. Shi et al. [25] proposed a convolutional neural network (CNN) based video CS framework that explores intraframe and interframe correlations of video data and achieves competitive reconstruction performance. Ran et al. [26] proposed a dual-domain-based DNN with parallel and interactive branch structures for magnetic resonance imaging, which utilized the inherent relationship between k-space and the spatial domain. Nevertheless, this method only obtained beneficial reconstruction results along with poor compression performances (high CR values). Jiang et al. [39] made a step forward on the RGB image SR by introducing a spatial–spectral prior network, which utilizes the spatial information and spectral correlation underlying HSIs to upsample compressed data using group convolution in a progressive manner. Considering insufficient computational resources on the miniaturized satellite and limited data transmission bandwidths between space and the ground, Hsu et al. [5] proposed a DNN-based joint encoding and decoding framework of HSIs, named deep compressed sensing network (DCSN), maintaining efficient compression and desirable reconstruction performances. However, the compression operations of DCSN focused merely on the feature level without giving importance to bit-level compression potentials. Further, during the decoding stage, DCSN equally treated the channelwise features and ignored distinguishing important features, which constrained the reconstruction performance.

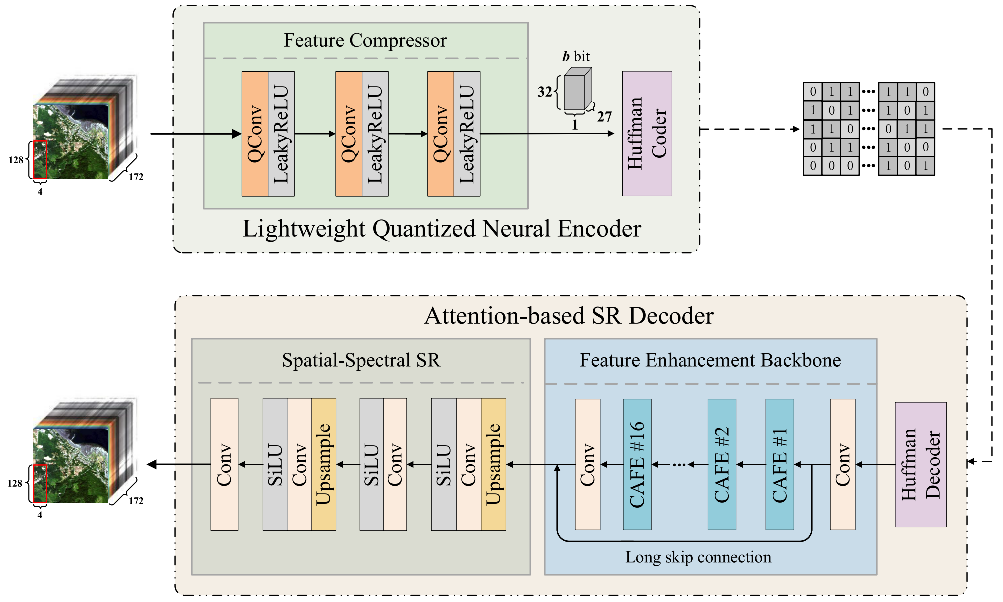


Fig. 2. Network architecture of the proposed BTC-Net for HSIs compression and reconstruction. (Top) Sensed hyperspectral data are compressed by the lightweight quantized neural encoder to obtain the bit-level compressed signal, which would be transmitted to the attention-based SR decoder. (Bottom) Decoder first enhances the representation of compressed features, and then reconstruct both spatial and spectral information. Qconv denotes the convolutional layer with the quantized activations and weights. CAFE denotes channelwise attention feature enhancement block.

D. Deep Learning-Based Hyperspectral Classification

To assess the preservation of semantic information in reconstructed HSIs compared to the originals, the hyperspectral classification task serves as a suitable measure of the semantic preservation capacity of compression approaches. This section provides a brief overview of key aspects of hyperspectral classification. In HSIs, each pixel can be treated as a high-dimensional vector, with its entries representing the spectral reflectance at specific wavelengths. This property enables classifiers to assign a unique label based on the pixel's spectral characteristics [40]. Early studies in HSI classification primarily focused on machine learning techniques that used classification hyperplanes based on spectral signatures from training samples. These methods included support vector machines [41], ensemble learning [42], [43], and sparse/collaborative representation [44], [45]. More recently, deep learning-based techniques have emerged as a powerful approach for extracting informative features from HSIs in a hierarchical manner. These techniques leverage multiple layers, with earlier layers capturing shallow features and deeper layers representing more complex and abstract features. Examples include autoencoders [46], CNNs [47], [48], and generative adversarial networks [49]. Transformers, which have achieved significant success in natural language processing and other domains, have also shown promise in hyperspectral classification due to their ability to model global context effectively. For instance, Hong et al. [50] utilized the sequence attributes of spectral signatures from neighboring bands using a transformer-based backbone network. Zhong et al. [51] proposed a spectral–spatial transformer architecture through a factorized architecture search framework to determine layer-level operations and block-level orders. Tu et al. [52] developed a transformer architecture based on local semantic feature aggregation, allowing efficient representation of long-range dependencies in multiscale features. Additionally, to handle the adversarial sample attacks, Tu et al. [53] proposed a robust class context-aware network by constructing a learnable affinity matrix. This matrix captures both intraclass and interclass relationships and serves as a class-contextual prior.

III. Proposed Method

To achieve efficient hyperspectral data transmission between the sender and receiver, it is necessary to well compress and then reconstruct the hyperspectral data. To address the challenges of insufficient computational resources, limited transmission bandwidth, and effective reconstruction requirement for the HSI data, we propose an end-to-end DNN-based method, BTC-Net. It could achieve high performance of both compression and reconstruction of the hyperspectral data.

A. Overview of the Proposed BTC-Net

As illustrated in Fig. 2, our proposed BTC-Net mainly consists of a lightweight quantized neural encoder and an attention-based SR decoder. Original HSI is first preprocessed and then put into the encoder for efficient bit-level compression. Subsequently, the compressed bit stream would be transmitted to the receiver and reconstructed in our decoder by performing SR along both spatial and spectral dimensions.

The lightweight quantized neural encoder achieves feature-wise and bit-level compression with a lightweight structure. A three-layer quantized CNN aims to compress the feature dimension of the input hyperspectral data, and represent the compressed data with fewer bits, which can be deployed on the edge devices of insufficient computational resources.

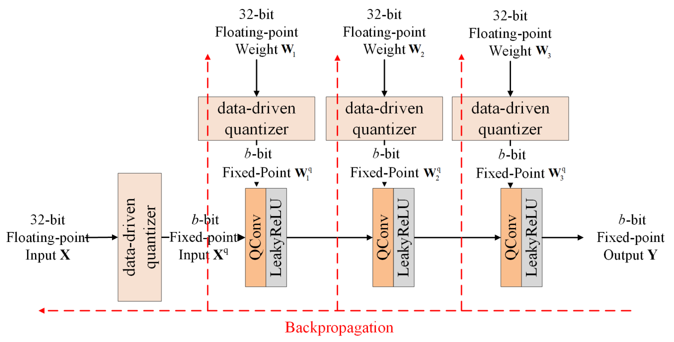

32-bit Floating-point Weight W_1
32-bit Floating-point Weight W_2
32-bit Floating-point Weight W_3
data-driven quantizer
b-bit Fixed-Point W_1^q
b-bit Fixed-Point W_2^q
b-bit Fixed-Point W_3^q
32-bit Floating-point Input X
data-driven quantizer
b-bit Fixed-point Input X^q
QConv
LeakyReLU
b-bit Fixed-point Output Y
Backpropagation

Fig. 3. Quantization diagram of the feature compressor in the training stage.

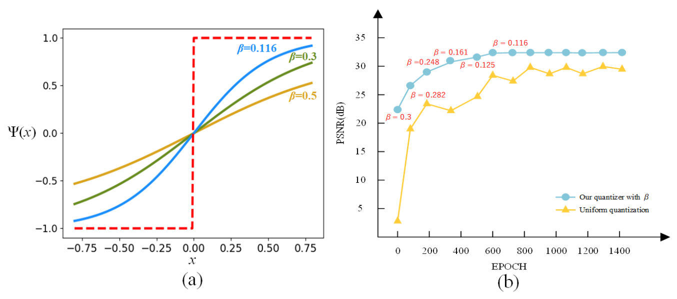


Fig. 4. In the data-driven quantization algorithm, as the characteristic variable β gets smaller during the training, the asymptotic function progressively approximates the staircase function (the red dashed line). (a) Characteristic of β. (b) Performance improvement with β during training.

To further decrease the bit rate of quantized results, we add the Huffman coder after this CNN.

Our attention-based SR decoder will first conduct a Huffman decoding after receiving the compressed bit stream. Then the decoded results subsequently are enhanced by the feature enhancement backbone. Finally, the hyperspectral data would be reconstructed by the SR operations of the spatial–spectral SR module. Specifically, the backbone mainly contains 16 CAFE blocks. A long skip connection is employed to convey the low-frequency information of features between the head and tail of the backbone. After sufficient feature enhancement by the backbone, the SR operations in the dimensions of space and spectrum would be performed alternately. The SR in the spatial dimension is implemented with the interpolation operations, which reconstructs the spatial information of the compressed features. Meanwhile, the SR in the spectral dimension will decode the spectral information and recover the spectral resolution.

B. Lightweight Quantized Neural Encoder

Our proposed encoder is characterized by lightweight and high-compression performance. It mainly comprises of a three-layer CNN whose weights and activations are quantized by our data-driven quantizer, leading to the compact structure and the implementation of both effective featurewise and bit-level compression.

1) Neural Network Quantization During Training: Fig. 3 illustrates the process of quantization that we employ during the training phase. Our data-driven quantizer maps the original 32-bit floating-point activations and weights into fixed-point data, which is represented by b bits. After quantization, the corresponding value of the loss function can be calculated, and appropriate fixed-point weights would be learned through back-propagation. These weights are then retained and utilized to process the input HSI data during the inference stage, which achieves the lightweight of the encoder while compressing the data into a low bit rate.

2) Theory of Quantization: For a 32-bit floating-point value to be quantized with b bits using uniform quantization, it is first clipped in (l, u) before quantization. Then the clipping range (l, u) is uniformly divided into $2^b - 1$ intervals R_i, $i \in \{1, 2, \dots, 2^b - 1\}$, and the length of the interval is $\delta = (u - l/2^b - 1)$.

Inspired by Choi et al. [54] and Gong et al. [55], we smoothly connect hyperbolic tangent functions (tanh) in different intervals to approximate the standard uniform quantization staircase function, avoiding the nondifferentiability of the latter and the corresponding performance degradation. For the input x that falls in the interval R_i, the asymptotic function is defined as

$$\Psi(x) = \lambda \cdot \tanh(c(x - k_i)), \quad \text{if} \quad x \in R_i \tag{1}$$

where the parameter k_i denotes the center value of the ith interval, and the parameter c plays a critical role in shaping the asymptotic function Ψ, and λ is the scaling coefficient, whose effect is to ensure that the tanh functions of Ψ in adjacent intervals can be smoothly connected, and it is defined as

$$\lambda = \frac{1}{\tanh(0.5c\delta)}. \tag{2}$$

To adaptively select appropriate values for the above parameters during training, we introduce a characteristic variable β, which aims to measure the asymptotic degree [as shown in Fig. 4(a)]. It is defined as

$$\beta = 1 - \tanh(0.5c\delta). \tag{3}$$

Then, the parameters c and λ in (1) can be represented by β

$$\lambda = \frac{1}{1-\beta} \quad \text{and} \quad c = \frac{1}{\delta} \cdot \log(\frac{2}{\beta} - 1). \tag{4}$$

So far, for the floating-point input x that falls in the interval R_i, the asymptotic function Ψ is represented as

$$\Psi(x) = \frac{1}{1-\beta} \cdot \tanh(\frac{1}{\delta} \cdot \log(\frac{2}{\beta} - 1) \cdot (x - k_i)). \tag{5}$$

As shown in Fig. 5, thanks to the asymptotic function's differentiability in the backpropagation, our data-driven quantizer can be automatically evolved to approximate the uniform quantization behavior without instability produced, thereby optimizing the performance of the quantized model [as shown in Fig. 4(b)].

With the asymptotic function $\Psi(x)$, our data-driven quantization algorithm $Q_b(x)$ maps the floating-point value x to the fixed-point value x^q with b bits, which is defined as

$$x^q = Q_b(x) = l + \delta(i + \frac{\Psi(x) + 1}{2}). \tag{6}$$

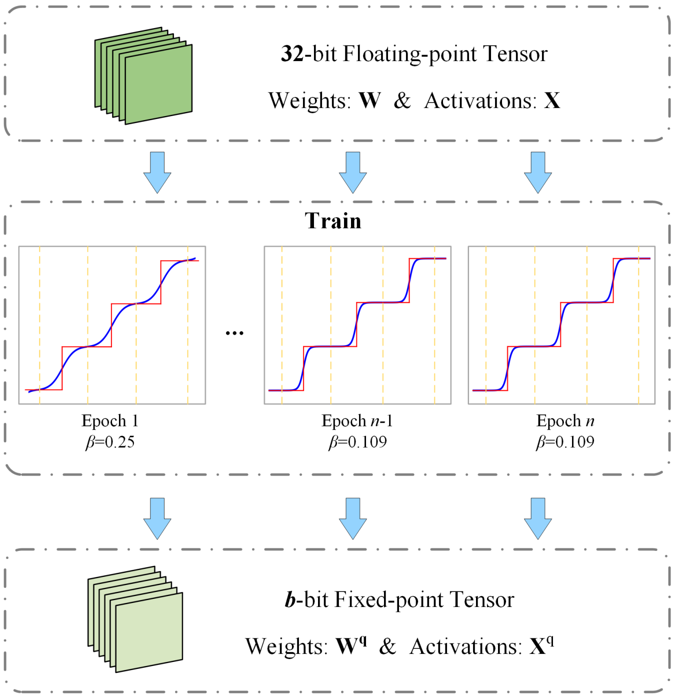

32-bit Floating-point Tensor
Weights: **W** & Activations: **X**
Train
Epoch 1
β=0.25
Epoch n-1
β=0.109
Epoch n
β=0.109
b-bit Fixed-point Tensor
Weights: $\mathbf{W}^q$ & Activations: $\mathbf{X}^q$

Fig. 5. Overview of the data-driven quantizer in the BTC-Net. The blue lines denote the asymptotic function component within our quantizer, while the red lines stand for the staircase uniform quantization, respectively. Thanks to the asymptotic function's differentiability in the backpropagation, our data-driven quantizer can be automatically evolved to approximate the uniform quantization behavior without instability being produced. That is to say, as the Epoch increases, the blue line progressively approximates the red line because the parameters β is gradually learned and optimized.

3) Quantized Neural Network: By utilizing this data-driven quantizer, the 32-bit floating-point input $\mathbf{X} \in \mathbb{R}^{C_{\text{in}} \times H_{\text{in}} \times W_{\text{in}}}$ and weights $\mathbf{W} \in \mathbb{R}^{k_h \times k_w}$ of the CNN in our encoder would be quantized into b-bit fixed-point data in the training stage

$$\mathbf{X}^q = Q_b(\mathbf{X}) \tag{7}$$
$$\mathbf{W}^q = Q_b(\mathbf{W}). \tag{8}$$

After the training loss converges, appropriate weights ($\mathbf{W}_1^q$, $\mathbf{W}_2^q$ and $\mathbf{W}_3^q$) of the compressor would be fixed and utilized in the inference stage of our encoder.

After the compression by our three-layer quantized CNN, we can obtain the output $\mathbf{Y} \in \mathbb{R}^{C_{\text{out}} \times H_{\text{out}} \times W_{\text{out}}}$

$$\mathbf{Y} = f_{\text{comp}}(f_{\text{comp}}(f_{\text{comp}}(\mathbf{X}^q, \mathbf{W}_1^q), \mathbf{W}_2^q), \mathbf{W}_3^q) \tag{9}$$

where the function $f_{\text{comp}}(\mathbf{X}^q, \mathbf{W}^q) = \rho(\mathbf{X}^q \odot \mathbf{W}^q)$ denotes the convolution-LeakyReLU operation, and the symbol $\odot$ is the convolution operation. As both the inputs and weights are quantized into fixed-point values with b bits, the convolutional results are b-bit fixed-point compressed data.

During the featurewise compression process of convolutional operations, spectral features compression is achieved due to $C_{\text{out}} < C_{\text{in}}$, which depends on the number of output channels of the convolutional layers. The spatial features compression is achieved due to $H_{\text{out}} < H_{\text{in}}$, $W_{\text{out}} < W_{\text{in}}$, which depends on the parameters of each convolutional layer. In this regard, we can easily set the CR of feature compression by adjusting the value of convolutional stride s, padding number p, kernel size $k_h \times k_w$, and the number of output channels C_{out}.

TABLE I
RELEVANT SETTINGS OF THE FEATURE COMPRESSOR TO OBTAIN THE SAMPLING RATIO OF 1%

Layer No.	Input Feature	Kernel Size	Stride	Padding	Output Feature
1	128×4×172	3×3	[2,2]	[1,0]	64×1×128
2	64×1×128	3×1	[1,1]	[1,0]	64×1×64
3	64×1×64	3×1	[2,1]	[1,0]	32×1×27

To meet the requirements of limited communication bandwidth, we set the CR of feature-level compression to 1% in this article, and the corresponding parameters for the feature compressor is listed in Table I. Thus, we obtain the compressed result $\mathbf{Y}$ from the encoder, which is an output tensor that has undergone both featurewise and bit-level compression. Compared to the input hyperspectral data $\mathbf{X}$, the overall CR is calculated based on the featurewise CR (1%) and the bit-level CR, achieving an extremely low bit rate.

4) Huffman Coder: Moreover, Huffman coding is a commonly used lossless data compression method, which uses variable-length codewords to encode source symbols. Each symbol is mapped into Table I according to its occurrence probability. The greater the occurrence probability, the fewer bits are required. For the BTC-Net, the statistical characteristics of the activation output are changed due to our data-driven quantization. This fact enables Huffman coding to compress the quantized output. Different degrees of quantization will obtain different levels of compression under the Huffman coding operation. It will be proved in Section IV that with Huffman coding and decoding, 70%–75% bits of the compressed signal can be saved without decreasing reconstruction performances.

C. Attention-Based Super-Resolution Decoder

To precisely recover hyperspectral data from low-resolution compressed features, an effective SR decoder is necessary. Many CNN-based SR technologies have been proposed, leading to the improvement of hyperspectral data reconstruction [56], [57]. However, these approaches equally treat the input low-resolution compressed features of entire channels, ignoring the fact that there is inconsistent importance of various channelwise features. Especially for the HSI data, some channels would dominate the characteristics of the hyperspectral pixels. In this regard, we introduce a channelwise attention mechanism to boost the representation capability of compressed features in the backbone. Subsequently, the spatial and spectral SR will be performed.

To successively rebuild the spatial and spectral information from compressed features, our decoder is mainly composed of two parts, which is shown in Fig. 2. First, the Huffman decoder losslessly recovers quantized data $\mathbf{Y}^q$. Then, an effective backbone incorporating a feature channelwise attention mechanism is proposed to enhance the feature expression. At last, the interpolation upsampling and convolutional operations are combined to reconstruct both the spatial and spectral information.

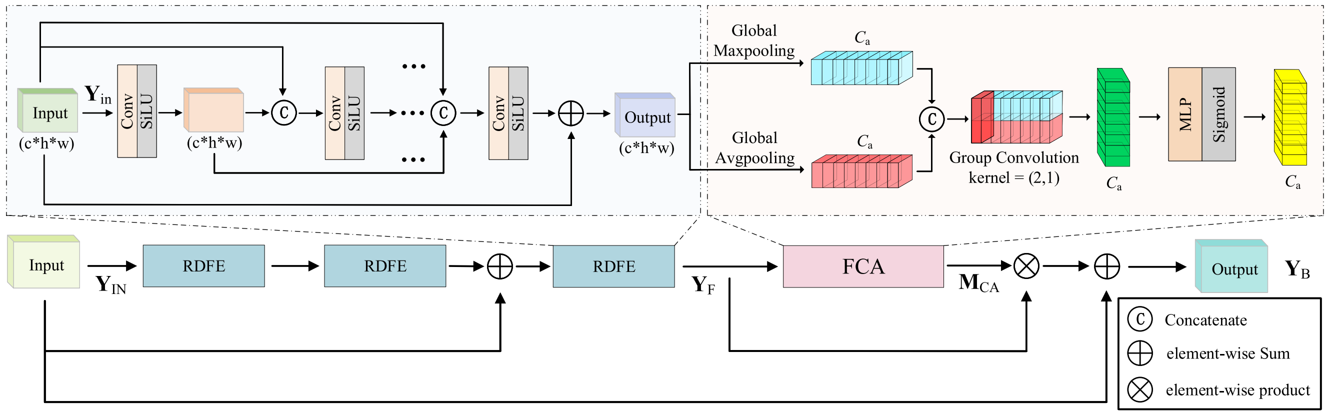


Fig. 6. Network architecture of proposed CAFE block. As the basic unit of the feature enhancement backbone in the decoder, CAFE combines the RDFE block and the FCA block to enable the decoder to enhance the representation capabilities of compressed features.

1) Feature Enhancement Backbone: The backbone network is very important for CNN-based SR tasks. Many effective backbone networks have been proposed, among which the insights of ResNet [58] and DenseNet [59] structures are referred to in this work. Their merits contribute to the construction of our backbone with the feature enhancement ability.

After Huffman decoding, compressed features will be learned by the backbone, whose basic unit is the CAFE block. As shown in Fig. 6, the CAFE block is mainly composed of two blocks, i.e., the RDFE block and the FCA block. We stack three RDFE blocks and one FCA block to form a CAFE unit with two residual connections. In the RDFE, there are five convolutional layers with kernels size of 3 × 3, followed by the activation function that uses the SiLU (a.k.a., Swish [60], [61]), which is defined as

$$f(x) = x \cdot \sigma(x) \tag{10}$$

where $\sigma(x)$ is the sigmoid function.

As shown in Fig. 6, for each layer in RDFE, the feature maps of all previous layers are densely concatenated and treated as the input itself. At last, the elementwise sum is conducted between the input feature map of the RDFE and the output of the last layer of the RDFE by a residual connection. Specifically, the output feature map $\mathbf{Y}_F$ of RDFE is defined as

$$\mathbf{Y}_F = f_{\text{RDFE}}(\mathbf{Y}_{\text{in}}, \mathbf{W}_{\text{RDFE}}) \tag{11}$$

where f_{RDFE} denotes the function of RDFE, $\mathbf{W}_{\text{RDFE}}$ and $\mathbf{Y}_{\text{in}}$ are the set of the parameters and the input feature map of the RDFE, respectively.

Here, we will introduce the FCA block. As demonstrated in Fig. 6, given the input feature with C_a channels, we first implement spatial global average-pooling f_{avg} and global max-pooling $f_{\max}$ to compress the spatial information of the feature map to obtain two $1 \times 1 \times C_a$ feature sequences. Then, we use 2×1 group convolution $f_g^{2\times1}$ to combine the two sequences, whose number of output channels also equals to C_a. Next, a multilayer perception (MLP) is added with a hidden layer of size C_a/r, where r is the compression coefficient. Thus, the output channelwise attention weight coefficient $\mathbf{M}_{\text{CA}}$ can be obtained as follows:

$$\mathbf{M}_{\text{CA}} = \sigma(f_{\text{MLP}}(f_g^{2\times1}(f_{\text{avg}}(\mathbf{Y}_F);\ f_{\max}(\mathbf{Y}_F)))) \tag{12}$$

where σ is the sigmoid activation function.

On the one hand, for the channelwise attention weight sequence $\mathbf{M}_{\text{CA}}$, the output of the last RDFE block in the CAFE can be weighted. On the other hand, after stacking three RDFE blocks and the FCA block, the skip connection is added between the head and the tail of the CAFE block, by which the low-frequency information can be transmitted directly. Here, one CAFE block can be formulated as

$$\mathbf{Y}_B = \mathbf{Y}_{\text{IN}} + \mathbf{Y}_F \otimes \mathbf{M}_{\text{CA}} \tag{13}$$

where $\mathbf{Y}_{\text{IN}}$ is the input of the CAFE, $\mathbf{Y}_F$ is the output of the third stacked RDFE block in the CAFE, $\mathbf{M}_{\text{CA}}$ is the channelwise attention weights, and $\otimes$ is the elementwise product. As displayed in Fig. 2, we stack 16 CAFE blocks to form the backbone network of the SR decoder.

2) Spatial–Spectral Super-Resolution: After obtaining the intermediate feature map $\mathbf{Y}_B$ from the feature enhancement backbone, the spatial and spectral resolution of $\mathbf{Y}_B$ will be reconstructed. Within the spatial–spectral SR module, four convolutional layers are arranged, and the number of their kernels increases gradually to ensure the spectral resolution of the output $\widehat{\mathbf{X}}$ is enlarged to the same level of the input $\mathbf{X}$. Meanwhile, two 2× interpolation upsample blocks are inserted to recover the spatial resolution of compressed features. The function of our spatial–spectral SR module can be defined as follows:

$$\widehat{\mathbf{X}} = f_{\text{SSR}}(\mathbf{Y}_B). \tag{14}$$

This process will reconstruct a high-quality strip-like hyperspectral data $\widehat{\mathbf{X}}$. Since the reconstruction of stripes is irrelevant to each other, parallel computing can be implemented to accelerate the reconstruction process.

D. Loss Function

Since $\mathcal{L}_1$ loss can sensitively capture small errors between reconstructed $\widehat{\mathbf{X}}$ and original input $\mathbf{X}$, we adopt this loss to measure the reconstruction accuracy of our network

$$\mathcal{L}_1 = \sum_{i=0}^{B-1} \sum_{j=0}^{L-1} \left\| \widehat{\mathbf{X}}_j^i - \mathbf{X}_j^i \right\|_1 \tag{15}$$

where B is the batch size, and $L = W_{\text{in}} \times H_{\text{in}}$ indicates the number of pixels in every stripe-like hyperspectral data.

IV. Experimental Results

A. Experimental Setup

1) Overview of Datasets: In this section, we trained and tested the proposed BTC-Net method with the data[1] collected by the HSI sensor Airborne Visible/Infrared Imaging Spectrometer (AVIRIS). These HSI data contain 224 spectral bands (wavelengths range between 400 and 2500 nm). Due to the Water vapor in the atmosphere absorbing specific wavelengths of electromagnetic radiation, it results in distinct absorption spectral curves and distortions in the intensity and shape of the target's reflectance spectrum. Those water vapor absorption and low SNR bands (1–10, 104–116, 152–170, and 215–224) are required to be removed prior to the subsequent tasks in the hyperspectral preprocessing workflow [5], [62]. After removing water vapor absorption and low SNR bands, 172 bands are used in the experiment. Besides, all the negative values in experimental data are artificially set to zero, in accordance with the nonnegative properties of the HSI. These data are collected over some sites in places of America and Canada, including California, Oregon, Florida, Arizona, Michigan, Yellowstone in America, and Ontario, Alberta, and British Columbia in Canada.

2) Evaluation Indices: Spectral angle mapper (SAM) [63], root-mean-square error (RMSE) [64], and peak signal-to-noise ratio (PSNR) [65] were used to measure the reconstruction performance of different methods. For the convenience of mathematical expression, we represent the HSI as a matrix $\mathbf{X} \in \mathbb{R}^{C_{\text{in}} \times L}$, whose rows and columns represent the spectral bands and pixel vectors, respectively, where $L = H_{\text{in}} \times W_{\text{in}}$. Specifically, $\mathbf{X}_{c,:}$ denotes the cth band of the ground-truth $\mathbf{X}$, while $\mathbf{X}_{:,j}$ denotes the jth pixel of $\mathbf{X}$.

1) SAM, measures the spectral distortion, whose value is smaller, the spectral reconstruction is better

$$\text{SAM} = \frac{1}{L} \sum_{j=1}^{L} \arccos\left(\frac{\hat{\mathbf{X}}_{:,j}^T \mathbf{X}_{:,j}}{\left\|\hat{\mathbf{X}}_{:,j}^T\right\|_2 \cdot \left\|\mathbf{X}_{:,j}^T\right\|_2} \right). \tag{16}$$

2) The PSNR measures the spatial quality, given in dB

$$\text{PSNR} = \frac{1}{C_{\text{in}}} \sum_{c=1}^{C_{\text{in}}} 10 \log_{10} \left(\frac{\max\{\mathbf{X}_{c,:}\}}{\frac{1}{L}\left\|\widehat{\mathbf{X}}_{c,:} - \mathbf{X}_{c,:}\right\|_2^2} \right). \tag{17}$$

3) The RMSE measures the global quality, whose value is smaller, the global quality of reconstruction is higher

$$\text{RMSE} = \sqrt{\frac{1}{C_{\text{in}}} \sum_{c=1}^{C_{\text{in}}} \frac{1}{\sqrt{L}} \left\|\widehat{\mathbf{X}}_{c,:} - \mathbf{X}_{c,:}\right\|_2}. \tag{18}$$

[1]aviris.jpl.nasa.gov/dataportal/

3) Hyperparameters Setting: We trained and tested the proposed BTC-Net using Pytorch libraries. The RAdam optimizer without decay was utilized, and the learning rate was set to $4e^{-4}$. The models were trained with a batch size of 32. Predicated on the belief that in hyperspectral data acquisition, the pushbroom system scans the image using an acquisition behavior of strip-by-strip [66], [67], thus the input HSI data was cropped into stripe-like patches of $128 \times 4 \times 172$. During the training, standard data augmentation strategies (e.g., random flipping) were adopted to improve the generalization ability of the proposed models. To maintain consistency, the feature CR of all the methods in the experiments was fixed at 1%.

B. Verification of Hyperspectral Compression Performance

1) Dataset Setting: In this experiment, HSI data collected from these sites were cropped into the size of $256 \times 256 \times 172$, leading to a total number of 1183 subimages. We divided the existing hyperspectral subimages into four datasets according to their ground features, which is denoted as follows.

1) The dataset of urban areas is named *Urban*, including 101 subimages.
2) The dataset of vegetation regions (farm or grass) is named *Farm*, including 265 subimages.
3) The dataset of lake/coastline areas is named *Lake*, including 110 subimages.
4) The dataset of mountain areas is named *Mountain*, including 707 subimages.

We randomly selected 90% and 10% HSIs of each dataset as the training set and testing set, respectively. Specifically, the number of subimages in the training set for Urban, Farm, Lake, and Mountain are 91, 239, 99, and 636, respectively. Finally, we gathered these four datasets to form the dataset named *Total*.

The proposed BTC-Net was trained on the training set of *Total* and then evaluated on the corresponding test set to verify the overall performance. It was also trained on the training set of each dataset and then evaluated on the testing sets of the above four datasets, to verify the generalization ability of the proposed method for different ground features.

2) Compression Performance Comparisons Among Methods: In this section, our BTC-Net (b bit) method was compared with seven SOTA methods under the feature CR of 1%. The SOTA methods include joint nuclear/TV norm minimization (JTTV) [38], self-learning tensor nonlinear CS (SL-TNCS) [10], joint tensor/reweight 3DTV norm minimization (JTenRe3-DTV) [6], E3DTV [13], PCA + JPEG2000 [15], DCSN [5], spatial–spectral prior SR (SSPSR) [39], and the nonlocal meets global paradigm (NGmeet) [68]. The first four methods adopted the randomly permuted Hadamard transform as the compressive strategy. NGmeet [68] is a regularization technique, which implements compression by spectral downsampling. PCA + JPEG2000 [15] is a frequency domain-based method, in which the PCA algorithm is used to decorrelate spectral dimension of HSIs, and the JPEG2000 is adopted for the compression in spatial dimension. DCSN [5] and SSPSR [39] are both DNN-based methods, which

TABLE II
QUANTITATIVE COMPARISON OF ALL COMPETING METHODS ON FIVE HSI CUBES UNDER THE FEATURE COMPRESSION OF 1%

Test set / Model	Moffett	California(U)	HyspIRI17(F)	FtMyers(L)	HyspIRI40(M)	Huffman CR$\downarrow$	Bit rate (bpppb)$\downarrow$
	SAM$\downarrow$ /RMSE$\downarrow$ /PSNR$\uparrow$						
JTTV [38]	11.83/418.60/25.776	8.013/284.61/25.019	16.80/368.89/22.769	7.742/245.52/23.982	2.844/158.59/22.524	-	0.32
SLTNCS [10]	9.457/364.15/27.151	7.532/267.55/25.055	6.751/225.97/24.539	9.373/217.51/26.359	4.342/166.35/22.169	-	0.32
PCA + JPEG2000 [15]	8.793/307.37/31.197	7.2373/238.834/30.495	2.756/134.12/32.073	4.401/113.57/33.895	1.409/51.021/32.498	-	0.039
E3DTV [13]	6.133/222.90/32.383	4.508/151.86/30.816	2.874/144.18/31.456	4.839/117.11/33.380	2.029/64.302/30.286	-	0.32
JTenRe3DTV [6]	5.533/188.79/33.702	4.588/142.56/30.832	2.528/91.848/32.789	2.116/60.931/36.284	2.113/77.493/29.060	-	0.32
SSPSR [39]	4.579/171.73/34.502	2.431/91.261/33.853	1.427/71.351/34.748	2.478/81.236/35.875	1.321/44.601/33.652	-	0.32
NGmeet [68]	4.493/165.38/34.722	2.635/96.376/33.488	1.382/65.033/35.171	4.071/96.781/34.407	0.991/39.632/34.113	-	0.32
DCSN [5]	**3.161**/139.06/35.243	2.144/88.781/34.045	1.369/65.989/34.933	1.354/56.239/36.806	1.017/41.179/33.902	-	0.32
BTC-Net (32 bit)	3.197/**138.36/35.273**	**2.033/86.101/34.462**	**1.322/63.317/35.434**	**1.202/52.441/37.509**	**0.944/37.002/34.827**	-	0.32
BTC-Net (11 bit)	3.211/140.91/35.021	2.040/86.153/34.451	1.332/63.471/35.366	1.211/52.699/37.453	0.952/37.081/34.768	34.41%	0.039
BTC-Net (10 bit)	3.300/141.03/35.009	2.061/86.301/34.442	1.334/63.482/35.364	1.214/52.706/37.442	0.955/37.111/34.756	33.01%	0.033
BTC-Net (9 bit)	3.304/141.27/35.001	2.076/86.313/34.439	1.339/63.671/35.355	1.216/52.714/37.436	0.963/37.318/34.749	31.63%	0.028
BTC-Net (8 bit)	3.358/141.61/34.979	2.121/87.614/34.207	1.359/64.621/35.015	1.283/52.919/37.231	0.991/37.742/34.625	30.43%	0.027
BTC-Net (7 bit)	3.403/142.97/34.853	2.171/88.826/34.004	1.402/65.995/34.906	1.294/53.889/37.018	1.007/38.718/34.029	**29.05%**	**0.020**

was trained on the same training set of BTC-Net (*b* bit) before comparison. We used five HSI cubes for performance evaluation, which are of 256 × 256 × 172. One of the chosen HSI data was Moffett Field, and others were from the test set of *Urban*, *Farm*, *Mountain*, and *Lake*, respectively. We denoted them as California(U), HyspIRI17(F), FtMyers(L), and HyspIRI40(M) according to the data collection site.

Note that the compression indices (Huffman CR and bit rate) were calculated by averaging on these five HSI cubes. The compressed signal of six SOTA methods except for PCA + JPEG2000 [15] was represented by 32-bit float-point data, whose bit rate should be converted into 0.32 bpppb. For the convenience of comparison, we set the compression bit rate of PCA + JPEG2000 [15] as 0.039 bpppb, which was the same as the bit rate of BTC-Net (11 bit).

Table II lists the reconstruction results. We can see that: 1) our proposed BTC-Net method achieved better performance than other SOTA methods on five HSIs, with obvious advantages in both compression and reconstruction performance; 2) considering the compression performance of BTC-Net (*b* bit), which combines quantization and Huffman coding, a high bit rate (0.020–0.039 bpppb) was achieved, which means 8 to 16 times higher bit rate than other SOTA methods; and 3) in the case of a low bit rate of the compressed signal, BTC-Net (*b* bit) still maintained good reconstruction performance. For example, BTC-Net (7 bit) with the extremely low bit rate of 0.020 bpppb, still achieved superior reconstruction results than four TV-based SOTA methods and the frequency domain-based PCA + JPEG2000 [15].

Visualization could help intuitively evaluate the reconstruction performance of various methods. Thus we conducted visualization experiments comparing JTenRe3-DTV [6], E3DTV [13], PCA + JPEG2000 [15], DCSN [5], SSPSR [39], and NGmeet [68]. visualization experiment on. For each HSI of visualization, we selected the 25th, the 15th, and the 6th band as the three channels of the RGB image. In Fig. 7, it was apparent that under the feature CR of 1%, the reconstructed HSIs by BTC-Net (11 bit) and DCSN offered a better visual experience, very close to the ground truth, but our BTC-Net (11 bit) achieved a lower compression bit rate of encoding. From the reconstructed HSIs of four TV-based methods, some problems can be observed, such as the noise points in the reconstructed HSIs of E3DTV [13] and JTenRe3DTV [6], leading to the loss of edge details and texture information. The reconstruction performance of PCA + JPEG2000 [15] was relatively poor, which is reflected in the loss of main information and the color-jitter problem in the images. Meanwhile, the same problems of color-jitter could also be easily found in the RGB results of reconstruction by SSPSR [39] and NGmeet [68]. The visualization results showed that the BTC-Net (11 bit) method preserved a large number of details of the original HSIs, which denoted that our model could focus on the important spatial and spectral information during reconstruction.

Fig. 8 presents the PSNR and RMSE values for every band of the reconstructed HSIs by our BTC-Net method and the comparison methods. In the PSNR figures (top), the performance of proposed BTC-Net (32 bit) and BTC-Net (11 bit) outperformed almost the other six SOTA methods for the vast majority of bands. Meanwhile, in the RMSE figures (bottom), our BTC-Net (32 bit) and BTC-Net (11 bit) always maintained a relatively low level of RMSE values. Furthermore, no matter in figures of PSNR or RMSE, the lines of BTC-Net (32 bit) and BTC-Net (11 bit) are very close to each other, which indicates that 11 bits of quantization do not cause obvious reconstruction performance degradation.

3) Computational Complexity Analysis: Due to the resources of computational and storage on the air-carried devices being insufficient, the efficiency and occupation of our encoder need to be concerned. The number of neural network parameters in our lightweight quantized neural encoder is 0.228 M, whose memory occupation is only 0.3135 MB (quantization of 11 bits) and better than 0.9120 MB of DCSN [5]. Furthermore, the Huffman coding can be designed to be a part of the hardware, which occupies a few resources [69], [70].

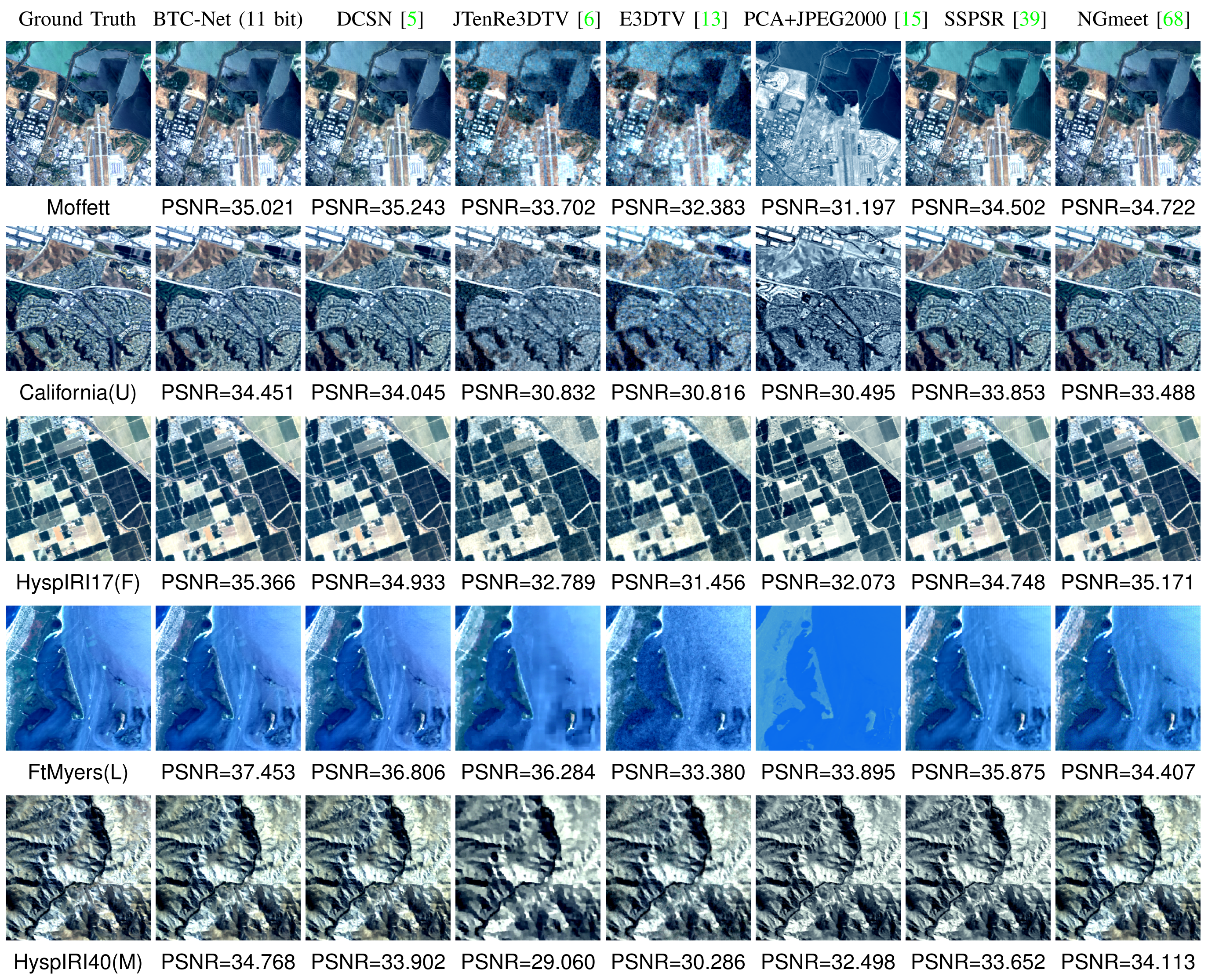


Fig. 7. Visualization results of the ground truth (first column) and the reconstructed HSIs by BTC-Net with 11-bit quantization (second column, trained on the training set of Total), DCSN (third column, trained on the training set of Total), JTenRe3DTV (fourth column), E3DTV (fifth column), PCA + JPEG2000 (sixth column), JTTV (seventh column), and SLTNCS (eighth column). (From top to bottom) Five rows of HSIs are Moffett, California(U), HyspIRI17(F), FyMyers(L), and HyspsIRI40(M), respectively.

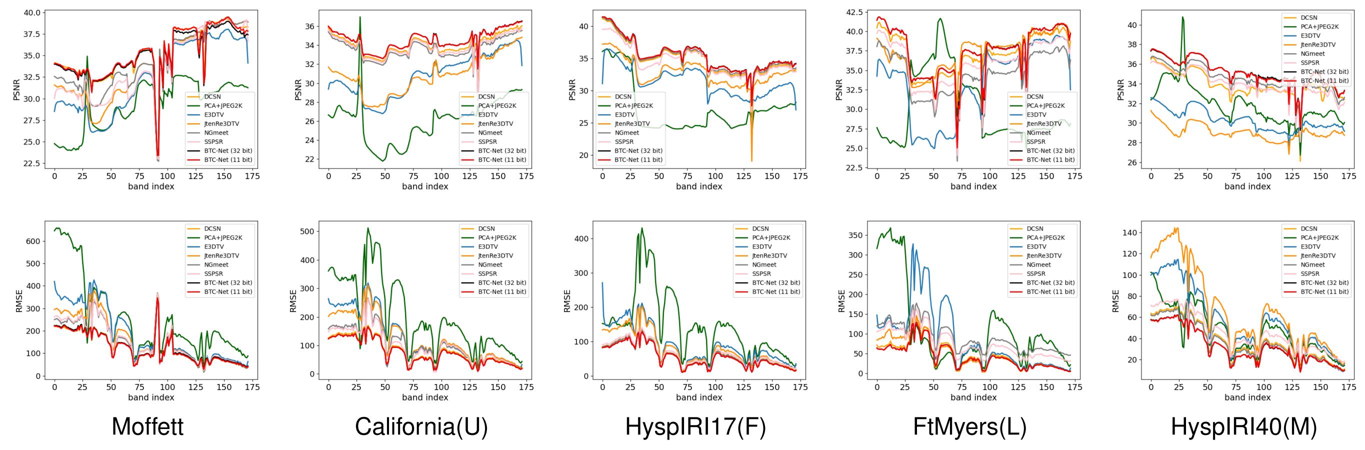


Fig. 8. PSNR and RMSE comparisons of different competing methods along all bands of five HSIs.

The running time of the whole process (both encoding and decoding) of the comparison methods were tested, as shown in Table III. For the DNN-based methods, we reported the inference time after the pretrained models were deployed. For

TABLE III
RUNNING TIME (IN SECONDS) OF DIFFERENT METHODS UNDER FEATURE CR OF 1%

Categories	Methods	Running time (s)	Bit rate (bpppb)↓
DNN-based	BTC-Net (11 bit)	2.034	**0.039**
	BTC-Net (11 bit)-w/o Huffman	1.449	0.11
	DCSN [5]	**1.438**	0.32
	SSPSR [39]	1.635	0.32
Frequency Domain-based	PCA + JPEG2000 [15]	2.911	0.039
Regularization-based	NGmeet [68]	5.2103	0.32
	JTTV [38]	148.2306	0.32
	SLTNCS [10]	152.7555	0.32
	E3DTV [13]	1462.0309	0.32
	JTenRe3DTV [6]	2836.491	0.32

TABLE IV
ABLATION STUDY EVALUATED ON THE *Total* DATASET. DIFFERENT COMPONENTS ARE GRADUALLY ADDED TO THE BASELINE TO SHOW THEIR EFFECTIVENESS

	Compressor	Quantizer (11 bit)	Huffman	Decoder	CA-based decoder	SAM↓ / RMSE↓ / PSNR↑	Huffman CR↓	Bit rate↓
Baseline	✓			✓		1.671/85.766/34.088	-	0.32
Baseline-I	✓				✓	**1.634/83.656/34.447**	-	0.32
Baseline-II	✓		✓		✓	**1.634/83.656/34.447**	-	0.32
Baseline-III	✓	✓			✓	1.640/83.707/34.422	-	0.11
Baseline-IV	✓	✓		✓		1.675/86.183/34.053	-	0.11
BTC-Net (our)	✓	✓	✓		✓	1.640/83.707/34.422	**35.1%**	**0.039**

regularization-based methods, we reported the whole optimization time since their training and testing procedures are tied together for processing a single HSI. These experiments were performed on a desktop computer equipped with an Intel i5-10400 CPU with 8-GB running memory. The running time was calculated by averaging that on the five HSI cubes mentioned in Table II.

We can see that the following holds.

1) Since the training and testing procedures of regularization-based methods are tied together for processing a single HSI, our BTC-Net (11 bit) achieved much faster processing speed in inference stage while displayed superior compression performance, compared with JTTV [38], SLTNCS [10], E3DTV [13], and JTenRe3DTV [6], which demonstrates the effectiveness of DNN-based methods for hyperspectral data compression in a data-driven manner. Even compared with the fast execution speed of PCA + JPEG2000 [15] and NGmeet [68], our BTC-Net performs better at the extremely low compression bit rate.
2) Compared to the DNN-based method, DCSN [5] and SSPSR [39], our BTC-Net (11 bit) could achieve much better compression performance and less running time cost than SSPSR [39], but consumed 0.6 s more running time than DCSN [5], because the introduction of the Huffman coder. When we removed the Huffman codec, our BTC-Net (11 bit)-w/o Huffman method still outperformed DCSN in the compression performance with the similar running time, which demonstrates the effectiveness of our proposed bit-level quantized neural encoder for hyperspectral data compression.

4) Ablation Study: We evaluated the effectiveness of our proposed BTC-Net method, mainly focusing on three components, the data-driven quantizer, the Huffman coder, and the channelwise attention-based decoder. To accurately describe the compression capability of the proposed BTC-Net, we adopted two kinds of compression indices. One is the Huffman CR, which is calculated by dividing the number of Huffman encoded bits by the number of input bits. The other is the bit rate, whose effect is to measure the compression performance of the encoder and its value is quantified with the bpppb. It must be noted that Huffman CR and the bit rate are calculated based on the models evaluated on the testing set of *Total*.

1) *Effectiveness of Different Components:* We adopted the BTC-Net without the quantization, Huffman coder, and attention mechanism as the baseline. Then we gradually applied our data-driven quantizer (11 bit), Huffman coder, and the channelwise attention-based decoder (CA-based decoder in Table IV) modules to the baseline model to demonstrate their effectiveness. The results on the *Total* dataset are shown in Table IV, from which we can see that the following holds.
   a) Based on the baseline, baseline-I replaced the decoder of DCSN [5] with our designed CA-based decoder. Compared to the baseline, the reconstruction of baseline-I was improved in terms of

TABLE V

RECONSTRUCTION PERFORMANCE COMPARISON OF OUR PROPOSED BTC-NET WITH THE DIFFERENT NUMBER OF BITS (7–11) FOR QUANTIZING THE RESULTS OF FEATURE COMPRESSION

Test set / Model	*Urban*	*Farm*	*Lake*	*Mountain*	*Total*	Huffman CR↓	Bit rate (bpppb)↓
	SAM↓ /RMSE↓ /PSNR↑						
BTC-Net (7 bit)	2.320/123.222/33.816	1.814/74.054/32.567	1.647/74.673/35.656	1.657/85.503/34.356	1.752/85.242/34.034	**30.3%**	**0.021**
BTC-Net (8 bit)	2.311/123.105/33.854	1.789/73.628/32.601	1.567/74.051/35.744	1.575/85.132/34.562	1.685/84.767/34.181	31.5%	0.025
BTC-Net (9 bit)	2.268/122.404/33.941	1.783/73.167/32.712	1.544/73.660/35.854	1.559/84.427/34.682	1.669/84.187/34.292	32.8%	0.03
BTC-Net (10 bit)	2.264/122.347/33.954	1.778/72.942/32.737	1.524/73.546/35.861	1.545/84.152/34.795	1.657/83.953/34.369	33.9%	0.034
BTC-Net (11 bit)	**2.261/122.134/33.959**	**1.776**/72.710/**32.761**	1.500/73.305/**35.887**	1.524/83.925/34.875	1.640/83.707/34.422	35.1%	0.039
BTC-Net (32 bit)	2.266/122.176/33.952	1.783/**72.677**/32.748	**1.493/73.248**/35.880	**1.512/83.863/34.917**	**1.634/83.656/34.447**	-	0.32
DCSN [5]	2.336/126.939/33.404	1.870/76.207/32.206	1.598/76.870/35.353	1.515/85.118/34.650	1.671/85.930/34.071	-	0.32

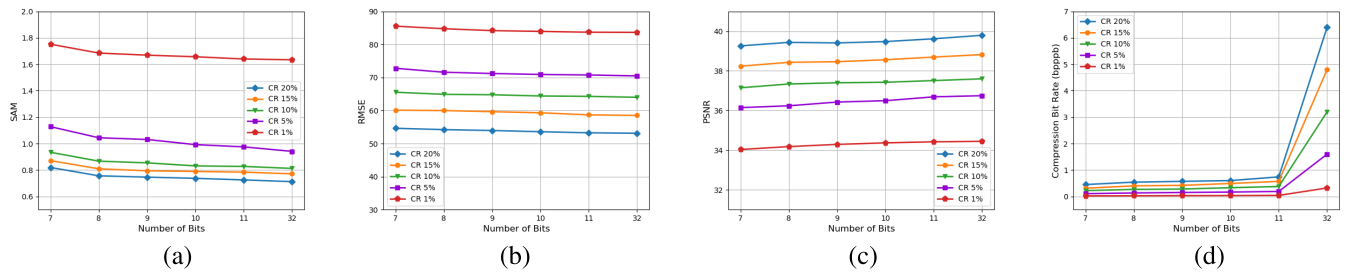


Fig. 9. Performance evaluation of the proposed BTC-Net method of different feature CRs (1%, 5%, 10%, 15%, and 20%), with different number of bits (7–11 and 32) for representing the weights and activations of feature compressor. On the metrics (a) SAM, (b) RMSE, (c) PSNR, and (d) compression bit rate.

SAM/RMSE/PSNR, which proves the effectiveness of our CA-based decoder with channelwise attention to enhance the hyperspectral features from the channel perspective. Similar conclusions also can be drawn from the comparison between baseline-III and baseline-IV.

b) Based on baseline-I, baseline-II added the Huffman coder, but it did not generate the gain in compression performance (Huffman CR is “-”).

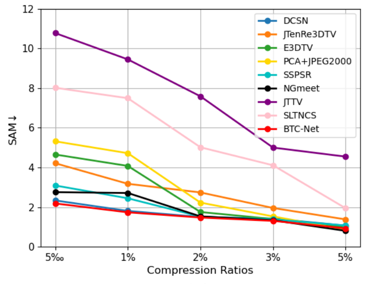
c) Based on baseline-I, baseline-III quantized the three-layer CNN with 11 bits, reducing the bit rate of encoded results remarkably (from 0.32 to 0.11 bpppb), with a slight drop in reconstruction performances in terms of SAM/RMSE/PSNR. It demonstrates the effectiveness of quantizer for feature compression to address the challenge of constrained transmission bandwidth. Similar conclusions also can be drawn from the comparison between baseline and baseline-IV.

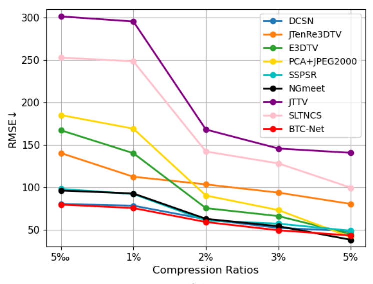
d) Based on baseline-III, our proposed BTC-Net method added the lossless Huffman coder. It can further save 64.9% ($= 1 - 35.1\%$) bits to represent the encoding results, simultaneously maintaining the reconstruction performance, which means that Huffman coding becomes effective when the quantizer is utilized.

e) Our BTC-Net remarkably outperformed the baseline, with a better reconstruction performance in terms of SAM/RMSE/PSNR, and especially with a much lower bit rate (0.32 versus 0.039 bpppb). It demonstrates the effectiveness of our BTC-Net by incorporating the data-driven quantization algorithm, Huffman coder, and channelwise attention-based decoder.

2) *Bit Number b of Quantization:* The bit number b of our quantizer has a direct impact on the compression bit rate. However, there is a trade-off between compression and reconstruction performance, where a lower bit rate typically results in worse reconstruction performance. To determine the optimal bit number, we conducted experiments by training on the Total dataset and evaluating on five different testing sets. The proposed BTC-Net uses quantization bits ranging from 7 to 11 under the feature CR of 1%, as listed in Table V.

We can see that: 1) the BTC-Net (32 bit) achieved the best reconstruction performance on the testing sets of *Total*, and its reconstruction performance was very close to that of BTC-Net (11 bit) on the other four testing sets; 2) the BTC-Net (b bit) method was able to achieve effective reconstruction of HSI at an extremely low compression bit rate (0.021–0.039 bpppb), resulting in a CR of nearly 800 to 1600 times compared to 32-bit full-precision data; 3) on the other hand, as the number of quantization bits decreases, the reconstruction performance of our BTC-Net method truly dropped as well; 4) the Huffman coding after quantization in our encoder led to the CR of about 30%–35%, which means about 65%–70% representation bits of the compressed signals can be saved before transmission; and 5) it is worth noting that, even when the number of quantization bits equals to 8, both the compression efficiency and reconstruction performance (RMSE and


(a)


(b)

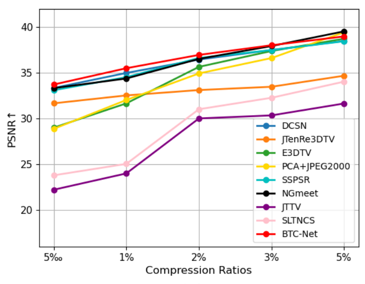

(c)

Fig. 10. Performance comparisons of the proposed BTC-Net and competitive methods at different sampling rates (5‰, 1%, 2%, 3%, and 5%) on the metrics (a) SAM, (b) RMSE, and (c) PSNR.

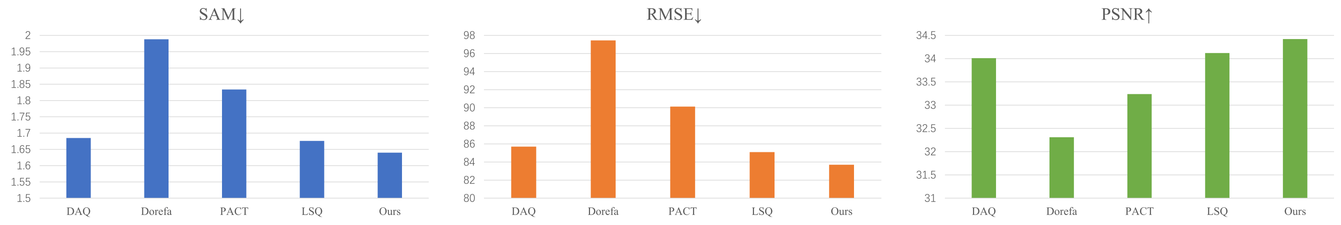


Fig. 11. Comparison of different quantization methods with 11 bit in our proposed BTC-Net framework. The reconstruction performance in terms of SAM, RMSE, and PSNR.

PSNR) of our BTC-Net method were superior to the SOTA DNN-based method DCSN [5].

3) *Effect of Feature CRs:* To discuss the robustness of the data-driven quantizer, the experiments of BTC-Net with 7–11 quantization bits under different feature CRs (1%, 5%, 10%, 15%, 20%) has been conducted. As shown in Fig. 9, both the reconstruction and compression performance are demonstrated. Comparing floating-point (32 bits) and quantized feature compressor (7–11 quantization bits), we can observe that the following holds.
   a) The higher the feature CR value, on the one hand, the more accurate the reconstruction will be in terms of performance. On the other hand, our data-driven quantization still causes varying levels of reconstruction accuracy loss for feature CRs of 5%, 10%, 15%, and 20%, but these losses are all minor and comparable to the situation at a CR of 1%.
   b) The impact of data-driven quantization and Huffman coding on compression performance are considerable, as shown in Fig. 9(d). For instance, the model with our data-driven quantization of 7 bits reduces the compression bit rates from 6.4 to 0.45 bpppb when compared to the floating-point BTC-Net at the feature CR of 20%.

4) *Comparison of Methods at Different CRs:* To observe the comparison of methods at different CRs, we conducted comparative experiments to evaluate the performance of their proposed BTC-Net alongside other competitive methods at different CRs (5‰, 1%, 2%, 3%, 5%). As shown in Fig. 10, it can be observed that the proposed BTC-Net achieved SOTA performance among all competing methods at most sampling rates, especially at the extreme case of 5‰. For a CR of 5%, NGmeet provided relatively better results. This observation can be attributed to the fact that NGmeet handles the observed data from the subspace with a nonlocal low-rankness prior, allowing for better reconstruction of the compressed data from a relatively limited information loss compared to a CR of 5‰.

5) *Effectiveness of Quantizer:* We compared the effectiveness of several quantization algorithms by utilize them to quantize the encoder of our baseline and its compressed feature output, including distance aware quantization (DAQ) [27], dorefa-net [71], parameterized clipping activation (PACT) [54], learned step size quantization (LSQ) [72], and our quantizer. As shown in Fig. 11, our quantizer outperformed all the other compared quantization methods in terms of SAM, RMSE, and PSNR reconstruction performance indices. Among the aforementioned quantizers, Dorefa-Net [71] performed poorly due to being nearly equivalent to uniform quantization without any learnable parameters. In contrast, our quantizer optimizes in a data-driven way without adding any computational burden, making the training process stable and efficient compared to the other three more complex algorithms.

C. Semantic Loss Test

1) Dataset Setting: To measure the semantic loss of the reconstructed HSIs compared with the original ones, we implemented this classification experiment, and selected two widely used classification datasets, including *Indian Pines* (IP) and *Salinas*, which were also captured by the same sensor AVIRIS like previous datasets. For the convenience of compression and reconstruction, the size of IP and *Salinas* were cropped into 128 × 128× 172 and 512 × 128 ×172, respectively, maximizing the retention of labeled sample data.

TABLE VI
SEMANTIC LOSS TEST ON AVERAGED CLASSIFICATION ACCURACY (%) WITH THE STANDARD DEVIATION TABULATED IN PARENTHESIS. THE CLASSIFICATION RESULTS ARE OBTAINED FROM THE BASIC CNN CLASSIFIER TRAINED ON ORIGINAL AND RECONSTRUCTED DATASETS

Metric	*IP* Dataset					*Salinas* Dataset				
	Orig. *IP*	BTC-Net	DCSN	PCA+JPEG2K	E3DTV	Orig. *Salinas*	BTC-Net	DCSN	PCA+JPEG2K	E3DTV
Class 1	87.32 ± 8.51	84.88 ± 8.64	75.12 ± 14.59	38.05 ± 20.44	32.20 ± 17.31	99.96 ± 0.13	98.00 ± 2.23	72.52 ± 24.63	77.51 ± 14.47	77.11 ± 14.57
Class 2	91.63 ± 1.88	91.80 ± 1.61	91.45 ± 2.40	77.24 ± 3.89	75.05 ± 2.92	99.99 ± 0.01	99.89 ± 0.13	99.93 ± 0.10	96.95 ± 4.85	95.92 ± 5.43
Class 3	92.92 ± 2.61	88.33 ± 5.25	85.90 ± 8.05	81.45 ± 11.50	77.69 ± 11.23	90.33 ± 19.84	90.28 ± 16.74	90.63 ± 18.80	77.90 ± 23.65	77.70 ± 24.46
Class 4	87.79 ± 6.42	87.75 ± 6.47	83.33 ± 10.65	76.15 ± 13.03	65.12 ± 19.47	99.78 ± 0.29	99.62 ± 0.27	99.76 ± 0.22	96.00 ± 2.65	95.68 ± 3.12
Class 5	95.60 ± 3.26	95.88 ± 2.18	94.95 ± 2.46	90.88 ± 5.60	89.70 ± 7.33	99.74 ± 0.29	99.65 ± 0.31	99.49 ± 0.51	94.62 ± 2.49	93.88 ± 2.98
Class 6	98.80 ± 1.04	98.72 ± 0.89	98.46 ± 1.24	96.24 ± 2.02	95.40 ± 3.33	100	100	100	99.94 ± 0.11	99.92 ± 0.10
Class 7	80.40 ± 18.02	73.60 ± 18.43	54.80 ± 23.19	66.00 ± 18.70	54.40 ± 29.40	99.99 ± 0.02	99.90 ± 0.09	99.97 ± 0.04	99.50 ± 0.30	99.27 ± 0.35
Class 8	98.08 ± 2.81	96.99 ± 4.07	96.15 ± 3.73	97.69 ± 2.91	96.67 ± 3.52	97.37 ± 1.74	95.79 ± 1.43	95.78 ± 1.54	88.91 ± 2.42	89.07 ± 2.76
Class 9	45.00 ± 11.51	42.78 ± 18.43	38.89 ± 13.83	28.33 ± 13.02	23.89 ± 13.39	99.98 ± 0.03	99.91 ± 0.09	99.83 ± 0.09	98.22 ± 2.40	98.30 ± 2.06
Class 10	89.91 ± 6.27	90.19 ± 2.35	88.53 ± 0.96	79.72 ± 9.48	79.77 ± 9.73	98.36 ± 0.77	98.22 ± 0.86	93.47 ± 3.77	60.03 ± 6.30	53.53 ± 4.75
Class 11	95.90 ± 1.73	95.38 ± 1.68	93.39 ± 1.93	87.52 ± 3.10	86.89 ± 3.87	92.55 ± 6.83	88.62 ± 6.64	38.58 ± 17.75	52.02 ± 17.02	51.34 ± 16.29
Class 12	88.80 ± 6.90	87.95 ± 3.57	84.37 ± 6.06	70.32 ± 9.94	69.31 ± 9.77	99.98 ± 0.06	99.98 ± 0.06	99.99 ± 0.03	96.23 ± 2.17	96.04 ± 2.15
Class 13	98.10 ± 1.82	98.26 ± 1.90	96.63 ± 2.24	93.59 ± 7.06	92.34 ± 5.28	98.23 ± 2.47	98.19 ± 2.43	68.30 ± 35.98	79.81 ± 9.58	81.19 ± 13.09
Class 14	96.42 ± 3.94	97.78 ± 1.31	97.56 ± 1.20	93.58 ± 4.93	94.64 ± 3.24	97.31 ± 1.66	96.84 ± 1.03	90.79 ± 6.71	97.31 ± 1.86	97.24 ± 1.75
Class 15	94.73 ± 4.10	92.94 ± 4.94	89.83 ± 3.84	90.35 ± 7.15	89.02 ± 6.18	86.97 ± 14.58	82.93 ± 14.74	85.21 ± 9.19	71.88 ± 12.34	69.28 ± 13.10
Class 16	86.87 ± 6.18	92.65 ± 5.40	93.49 ± 3.54	84.58 ± 12.63	85.78 ± 11.88	—	—	—	—	—
OA	93.80 ± 1.04	93.40 ± 1.05	91.76 ± 1.53	84.72 ± 1.88	83.41 ± 2.22	97.89 ± 1.03	97.05 ± 0.94	94.91 ± 1.41	89.61 ± 1.61	88.88 ± 1.79
AA	89.27 ± 1.77	88.49 ± 1.75	85.18 ± 2.78	78.23 ± 3.59	75.49 ± 4.29	97.37 ± 1.49	96.52 ± 1.59	88.95 ± 4.41	85.79 ± 3.43	85.03 ± 3.64
Kappa	92.84 ± 1.21	92.38 ± 1.21	90.49 ± 1.77	82.37 ± 2.16	80.85 ± 2.57	97.62 ± 1.17	96.68 ± 1.07	94.26 ± 1.60	88.29 ± 1.82	87.45 ± 2.03
Semantic Loss on OA	—	**-0.4**	-2.04	-9.08	-10.39	—	**-0.84**	-2.98	-8.28	-9.01
Semantic Loss on AA	—	**-0.78**	-4.09	-11.04	-13.78	—	**-0.85**	-8.42	-11.58	-12.34
Semantic Loss on Kappa	—	**-0.46**	-2.35	-10.47	-11.99	—	**-0.94**	-3.36	-9.33	-10.17
Bit Rate	—	**0.039**	0.32	**0.039**	0.32	—	**0.039**	0.32	**0.039**	0.32

2) Semantic Performance Comparisons Among Methods: Before the classification experiment, the datasets (including the original and the reconstructed hyperspectral data of both IP and *Salinas*) were divided into training sets and testing sets. For each dataset, 10% of the total samples per class were used as training samples, while the remaining 90% of the total samples per class were treated as testing samples. To ensure a fair comparison, we followed two important principles. First, to avoid sample bias, we randomly generated the indices of the training samples on IP and *Salinas*. The experiment was performed across ten independent runs, and the average values were taken as the final classification results, with the standard deviation tabulated in parenthesis. Second, we designed a nine-layer CNN as our basic classifier. This classifier was trained on the training sets of the original hyperspectral data and tested on the testing sets of the original data and all reconstructed datasets.

The optimizer Adam was used during training. The learning rate and the weight decay were set to 0.0005 and 0.0002, respectively. The batch size was 1 and the number of learning epochs was 500. The HSIs reconstructed by four methods BTC-Net, DCSN [5], PCA + JPEG2000 [15], and E3DTV [13] would participate in the comparison of classification accuracy. Three quantitative metrics, overall accuracy (OA), average accuracy (AA), and κ coefficient (Kappa), were used to evaluate the performance of our classifier on different testing sets.

Table VI (left) tabulates the classification results on the testing sets of the IP dataset. The highest OA of 93.80% was achieved on the testing set of the original IP HSI. However, the OA on the other four testing sets decreased to varying degrees. BTC-Net demonstrated a loss of OA of only 0.4% (less than 1%), with losses of 0.78% in AA and 0.46% in the Kappa coefficient. This indicates that the reconstruction quality of BTC-Net on the IP dataset achieved near-lossless semantics at low compression bit rates (0.039 bpppb), surpassing the performance of the other three methods. DCSN exhibited the second-best results compared to BTC-Net, while the traditional methods, PCA-JPEG2000 and E3DTV, showcased relatively higher semantic loss values exceeding 4%.

Table VI (right) presents the classification results on the testing sets of the *Salinas* dataset. Obviously, the best performance was achieved on the testing set of original *Salinas*. The lowest semantic loss performance can be observed in BTC-Net, with reductions of 0.84%, 0.85%, and 0.94% in terms of OA, AA, and Kappa coefficient, respectively, when compared to the best results. On the other hand, there is a noticeable decline in classification performance for the reconstructed HSIs by DCSN [5], PCA + JPEG2000 [15], and E3DTV [13]. These methods exhibited reductions of 2.98%, 8.28%, and 9.01% in OA, respectively.

The trained classifier achieved the highest classification accuracy on the testing sets of the original HSIs. Interestingly, when comparing it with the best results, the classification

TABLE VII
RECONSTRUCTION PERFORMANCE COMPARISON OF BTC-NET (11 bits) AND DCSN [54] ON THE DATASETS *Hawaii*, *the Bay Area*, AND *Saskatchewan*, IN TERMS OF SAM/RMSE/PSNR

Test set / Train set	Model	*Hawaii*	*The Bay Area*	*Saskatchewan*	*Nevada*	Averaged
		SAM↓ /RMSE↓ /PSNR↑				
California	BTC-Net	2.707/86.192/32.030	2.692/84.901/32.324	2.778/88.932/31.158	3.131/95.411/30.564	2.827/88.859/31.519
	DCSN	2.833/87.219/31.654	2.761/85.832/32.012	2.851/89.647/30.845	3.209/96.135/30.226	2.913/89.708/31.184
Arizona	BTC-Net	2.939/103.85/30.775	2.903/103.16/31.048	2.984/106.81/30.464	3.326/118.28/29.164	3.038/108.03/30.363
	DCSN	3.015/104.18/30.622	2.997/103.65/31.005	3.106/107.23/30.356	3.404/118.87/29.031	3.131/108.48/30.248
Ontario	BTC-Net	3.264/112.42/29.044	3.032/107.43/29.807	3.334/113.06/28.779	3.157/111.97/29.267	3.197/111.22/29.224
	DCSN	3.341/113.56/28.996	3.117/108.06/29.611	3.409/113.78/28.551	3.264/112.83/29.009	3.282/112.06/29.041
Yellowstone	BTC-Net	2.906/108.13/30.378	2.761/104.54/31.129	2.924/108.40/30.291	3.508/121.22/29.061	3.025/110.57/30.315
	DCSN	2.973/109.68/30.117	2.893/105.60/30.743	3.057/110.08/30.144	3.592/122.71/28.905	3.129/112.02/29.977
Averaged	BTC-Net	2.954/102.65/30.557	2.847/100.01/31.077	3.005/104.30/30.173	3.281/111.72/29.514	3.019/104.67/30.355
	DCSN	3.040/103.66/30.347	2.942/100.79/30.843	3.106/105.18/29.974	3.367/112.64/29.293	3.114/105.57/30.117

accuracy loss on the HSIs compressed and reconstructed by BTC-Net was less than 1%. This indicates that the encoding and decoding procedures of BTC-Net preserve near-lossless semantic information at a low compression bit rate (achieving 0.039 bpppb), making it suitable for classification purposes. On the other hand, PCA + JPEG2000 [15] primarily employs PCA for compression, resulting in spectral degradation and misjudgment during classification. E3DTV [13] introduces an enhanced 3-D TV regularization, but under low sampling rates, it may lead to poor performance due to spatial-spectral information loss. As a consequence, the classification accuracy declines on the reconstructed data.

D. Verification of Hyperspectral Generalization Performance

1) Dataset Setting: Clearly, the new generalization experiment follows the paradigm that we train the model on different types of areas and evaluate the interference capacity of the model on new locations. Four kinds of datasets captured from different areas, i.e., *California*, *Arizona*, *Ontario*, and *Yellowstone*, treated as training set, where the numbers of subimages in training sets are 162, 132, 121, 129, respectively. Then we train the models and evaluate them on the testing sets of *Hawaii*, *the Bay Area*, *Saskatchewan*, and *Nevada*, respectively. Four kinds of unseen datasets captured from new locations, i.e., Hawaii, the Bay Area in America, Nevada, and Saskatchewan in Canada, are treated as test set. Four test sets are referred to as *Hawaii*, *the Bay Area*, *Saskatchewan*, and *Nevada*, respectively.

2) Generalization Performance Comparisons Among Methods: The results for evaluating the generalization capacity are summarized in Table VII. The findings from Table VII can be summarized as follows. First, the average PSNR values of the BTC-Net model on each testing set range from 29 to 32 dB, slightly outperforming the baseline DCSN model. Additionally, when considering SAM and RMSE metrics, the proposed BTC-Net consistently delivers better results compared to the DCSN model. These outcomes highlight the superior generalization performance of the BTC-Net on previously unseen datasets.

Second, among the training sets, the BTC-Net model trained on the *California* dataset demonstrates the best overall performance (as indicated by the first Averaged row) due to the larger number of images available in the *California* training set. In contrast, owing to insufficient training samples, e.g., 121 of *Ontario*, the averaged PSNR is only 29.224 dB.

Third, upon comparing Table VII with Table II, we observe a moderate drop in quantitative values when evaluating the generalization capacity on unseen datasets (as shown in Table VII). However, the proposed BTC-Net still maintains an acceptable level of generalization performance, as evidenced by the results.

V. CONCLUSION

In this article, we proposed a novel DNN-based BTC-Net. BTC-Net consists of a lightweight quantized neural encoder with two-stage data compression and an attention-based SR decoder. The lightweight encoder combines data-driven quantization, convolution-activation operations, and Huffman coding to implement two-stage compression, which efficiently achieves bit-level compression on hyperspectral tensor data. The attention-based SR decoder utilizes a channelwise attention mechanism, which enables the feature enhancement backbone to focus on dominant features, thereby improving the feature representation abilities of the compressed features. Afterward, SR is employed to recover information in both the spatial and spectral dimensions. Experiments on hyperspectral datasets demonstrate that our BTC-Net achieves desirable bit-level compression performance at very low compression bit rates (less than 0.04 bpppb). Our method also achieves semantic near-lossless reconstruction quality on classification datasets, surpassing other SOTA methods.

REFERENCES

[1] C. Yu, J. Huang, M. Song, Y. Wang, and C.-I. Chang, "Edge-inferring graph neural network with dynamic task-guided self-diagnosis for few-shot hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5535613.
[2] S. Hou, H. Shi, X. Cao, X. Zhang, and L. Jiao, "Hyperspectral imagery classification based on contrastive learning," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5521213.
[3] S. Zhang, H. Huang, and Y. Fu, "Fast parallel implementation of dual-camera compressive hyperspectral imaging system," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 11, pp. 3404–3414, Nov. 2019.

[4] A. F. H. Goetz, "Three decades of hyperspectral remote sensing of the Earth: A personal view," *Remote Sens. Environ.*, vol. 113, pp. S5–S16, Sep. 2009.

[5] C.-C. Hsu, C.-H. Lin, C.-H. Kao, and Y.-C. Lin, "DCSN: Deep compressed sensing network for efficient hyperspectral data transmission of miniaturized satellite," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 9, pp. 7773–7789, Sep. 2021.

[6] Y. Wang, L. Lin, Q. Zhao, T. Yue, D. Meng, and Y. Leung, "Compressive sensing of hyperspectral images via joint tensor tucker decomposition and weighted total variation regularization," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 12, pp. 2457–2461, Dec. 2017.

[7] J. Jiang, H. Sun, X. Liu, and J. Ma, "Learning spatial-spectral prior for super-resolution of hyperspectral imagery," *IEEE Trans. Comput. Imag.*, vol. 6, pp. 1082–1096, 2020.

[8] J. Xue, Y. Zhao, W. Liao, and J. Chan, "Nonlocal tensor sparse representation and low-rank regularization for hyperspectral image compressive sensing reconstruction," *Remote Sens.*, vol. 11, no. 2, p. 193, Jan. 2019.

[9] C. Li, T. Sun, K. F. Kelly, and Y. Zhang, "A compressive sensing and unmixing scheme for hyperspectral data processing," *IEEE Trans. Image Process.*, vol. 21, no. 3, pp. 1200–1210, Mar. 2012.

[10] S. Yang, M. Wang, P. Li, L. Jin, B. Wu, and L. Jiao, "Compressive hyperspectral imaging via sparse tensor and nonlinear compressed sensing," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 11, pp. 5943–5957, Nov. 2015.

[11] Q. Du and J. E. Fowler, "Hyperspectral image compression using JPEG2000 and principal component analysis," *IEEE Geosci. Remote Sens. Lett.*, vol. 4, no. 2, pp. 201–205, Apr. 2007.

[12] Q. Du, N. Ly, and J. E. Fowler, "An operational approach to PCA+JPEG2000 compression of hyperspectral imagery," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 6, pp. 2237–2245, Jun. 2014.

[13] J. Peng, Q. Xie, Q. Zhao, Y. Wang, L. Yee, and D. Meng, "Enhanced 3DTV regularization and its applications on HSI denoising and compressed sensing," *IEEE Trans. Image Process.*, vol. 29, pp. 7889–7903, 2020.

[14] W. Fu, S. Li, L. Fang, and J. A. Benediktsson, "Adaptive spectral–spatial compression of hyperspectral image with sparse representation," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 2, pp. 671–682, Feb. 2017.

[15] Q. Du, N. Ly, and J. E. Fowler, "An operational approach to PCA+JPEG2000 compression of hyperspectral imagery," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 6, pp. 2237–2245, Jun. 2014.

[16] E. J. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. Inf. Theory*, vol. 52, no. 2, pp. 489–509, Feb. 2006.

[17] D. L. Donoho, A. Maleki, and A. Montanari, "Message-passing algorithms for compressed sensing," *Proc. Nat. Acad. Sci. USA*, vol. 106, no. 45, pp. 18914–18919, Nov. 2009.

[18] M. F. Duarte and R. G. Baraniuk, "Kronecker product matrices for compressive sensing," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Mar. 2010, pp. 3650–3653.

[19] C. Li, T. Sun, K. F. Kelly, and Y. Zhang, "A compressive sensing and unmixing scheme for hyperspectral data processing," *IEEE Trans. Image Process.*, vol. 21, no. 3, pp. 1200–1210, Mar. 2012.

[20] G. Martín, J. M. Bioucas-Dias, and A. Plaza, "HYCA: A new technique for hyperspectral compressive sensing," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 5, pp. 2819–2831, May 2015.

[21] Y. Chen, T.-Z. Huang, W. He, N. Yokoya, and X.-L. Zhao, "Hyperspectral image compressive sensing reconstruction using subspace-based nonlocal tensor ring decomposition," *IEEE Trans. Image Process.*, vol. 29, pp. 6813–6828, 2020.

[22] X. Deng and P. L. Dragotti, "Deep convolutional neural network for multi-modal image restoration and fusion," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 10, pp. 3333–3348, Oct. 2021.

[23] K. Song, H. Yang, and Z. Yin, "Multi-scale attention deep neural network for fast accurate object detection," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 10, pp. 2972–2985, Oct. 2019.

[24] M. Iliadis, L. Spinoulas, and A. K. Katsaggelos, "Deep fully-connected networks for video compressive sensing," *Digit. Signal Process.*, vol. 72, pp. 9–18, Jan. 2018.

[25] W. Shi, S. Liu, F. Jiang, and D. Zhao, "Video compressed sensing using a convolutional neural network," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 2, pp. 425–438, Feb. 2021.

[26] M. Ran et al., "MD-Recon-net: A parallel dual-domain convolutional neural network for compressed sensing MRI," *IEEE Trans. Radiat. Plasma Med. Sci.*, vol. 5, no. 1, pp. 120–135, Jan. 2021.

[27] D. Kim, J. Lee, and B. Ham, "Distance-aware quantization," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 5251–5260.

[28] X. Wang, J. Chen, Q. Wei, and C. Richard, "Hyperspectral image super-resolution via deep prior regularization with parameter estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 4, pp. 1708–1723, Apr. 2022.

[29] G. K. Wallace, "The JPEG still picture compression standard," *Commun. ACM*, vol. 34, no. 4, pp. 30–44, Apr. 1991.

[30] M. Charrier, D. S. Cruz, and M. Larsson, "JPEG2000, the next millennium compression standard for still images," in *Proc. IEEE Int. Conf. Multimedia Comput. Syst.*, Jun. 1999, pp. 131–132.

[31] W. A. Pearlman, A. Islam, N. Nagaraj, and A. Said, "Efficient, low-complexity image coding with a set-partitioning embedded block coder," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 11, pp. 1219–1235, Nov. 2004.

[32] X. Tang and W. A. Pearlman, "Three-dimensional wavelet-based compression of hyperspectral images," in *Hyperspectral Data Compression*. Hoboken, NJ, USA: Wiley, 2006, pp. 273–278.

[33] J. T. Rucker, J. E. Fowler, and N. H. Younan, "JPEG2000 coding strategies for hyperspectral data," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Nov. 2005, pp. 128–131.

[34] Y. Liu, Q. Liu, M. Zhang, Q. Yang, S. Wang, and D. Liang, "IFR-Net: Iterative feature refinement network for compressed sensing MRI," *IEEE Trans. Comput. Imag.*, vol. 6, pp. 434–446, 2020.

[35] I. Noor and E. L. Jacobs, "Adaptive compressive sensing algorithm for video acquisition using a single-pixel camera," *J. Electron. Imag.*, vol. 22, no. 2, May 2013, Art. no. 021013.

[36] A. E. Waters, A. C. Sankaranarayanan, and R. Baraniuk, "Sparcs: Recovering low-rank and sparse matrices from compressive measurements," in *Proc. Conf. Workshop Neural Inf. Process. Syst.*, 2011, pp. 1089–1097.

[37] C. Li, T. Sun, K. F. Kelly, and Y. Zhang, "A compressive sensing and unmixing scheme for hyperspectral data processing," *IEEE Trans. Image Process.*, vol. 21, no. 3, pp. 1200–1210, Mar. 2012.

[38] M. Golbabaee and P. Vandergheynst, "Joint trace/TV norm minimization: A new efficient approach for spectral compressive imaging," in *Proc. 19th IEEE Int. Conf. Image Process.*, Sep. 2012, pp. 933–936.

[39] J. Jiang, H. Sun, X. Liu, and J. Ma, "Learning spatial-spectral prior for super-resolution of hyperspectral imagery," *IEEE Trans. Comput. Imag.*, vol. 6, pp. 1082–1096, 2020.

[40] S. Li, W. Song, L. Fang, Y. Chen, P. Ghamisi, and J. A. Benediktsson, "Deep learning for hyperspectral image classification: An overview," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 9, pp. 6690–6709, Sep. 2019.

[41] F. Melgani and L. Bruzzone, "Classification of hyperspectral remote sensing images with support vector machines," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 8, pp. 1778–1790, Aug. 2004.

[42] A. Merentitis, C. Debes, and R. Heremans, "Ensemble learning in hyperspectral image classification: Toward selecting a favorable bias-variance tradeoff," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 4, pp. 1089–1102, Apr. 2014.

[43] W. Feng and W. Bao, "Weight-based rotation forest for hyperspectral image classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 11, pp. 2167–2171, Nov. 2017.

[44] X. Shen, W. Bao, H. Liang, X. Zhang, and X. Ma, "Grouped collaborative representation for hyperspectral image classification using a two-phase strategy," *IEEE Geosci. Remote Sens. Lett.*, vol. 19, pp. 1–5, 2022.

[45] J. Peng et al., "Low-rank and sparse representation for hyperspectral image processing: A review," *IEEE Geosci. Remote Sens. Mag.*, vol. 10, no. 1, pp. 10–43, Mar. 2022.

[46] Y. Chen, Z. Lin, X. Zhao, G. Wang, and Y. Gu, "Deep learning-based classification of hyperspectral data," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 6, pp. 2094–2107, Jun. 2014, doi: 10.1109/JSTARS.2014.2329330.

[47] H. Liang, W. Bao, X. Shen, and X. Zhang, "HSI-mixer: Hyperspectral image classification using the spectral–spatial mixer representation from convolutions," *IEEE Geosci. Remote Sens. Lett.*, vol. 19, pp. 1–5, 2022, doi: 10.1109/LGRS.2022.3200145.

[48] X. Li, M. Ding, and A. Pižurica, "Deep feature fusion via two-stream convolutional neural network for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 4, pp. 2615–2629, Apr. 2020, doi: 10.1109/TGRS.2019.2952758.

[49] H. Liang, W. Bao, X. Shen, and X. Zhang, "Spectral–spatial attention feature extraction for hyperspectral image classification based on generative adversarial network," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 10017–10032, 2021, doi: 10.1109/JSTARS.2021.3115971.

[50] D. Hong et al., "SpectralFormer: Rethinking hyperspectral image classification with transformers," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5518615, doi: 10.1109/TGRS.2021.3130716.

[51] Z. Zhong, Y. Li, L. Ma, J. Li, and W.-S. Zheng, "Spectral–spatial transformer network for hyperspectral image classification: A factorized architecture search framework," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5514715, doi: 10.1109/TGRS.2021.3115699.

[52] B. Tu, X. Liao, Q. Li, Y. Peng, and A. Plaza, "Local semantic feature aggregation-based transformer for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5536115, doi: 10.1109/TGRS.2022.3201145.

[53] B. Tu, W. He, Q. Li, Y. Peng, and A. Plaza, "A new context-aware framework for defending against adversarial attacks in hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 61, 2023, Art. no. 5505114.

[54] J. Choi, Z. Wang, S. Venkataramani, P. I.-J. Chuang, V. Srinivasan, and K. Gopalakrishnan, "PACT: Parameterized clipping activation for quantized neural networks," 2018, *arXiv:1805.06085*.

[55] R. Gong et al., "Differentiable soft quantization: Bridging full-precision and low-bit neural networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 4851–4860.

[56] Y. Fu, T. Zhang, Y. Zheng, D. Zhang, and H. Huang, "Hyperspectral image super-resolution with optimized RGB guidance," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 11653–11662.

[57] X. Wang, K. C. K. Chan, K. Yu, C. Dong, and C. C. Loy, "EDVR: Video restoration with enhanced deformable convolutional networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2019, pp. 1954–1963.

[58] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[59] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2261–2269.

[60] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," 2017, *arXiv:1710.05941*.

[61] Z. Lin et al., "Revisiting RCAN: Improved training for image super-resolution," 2022, *arXiv:2201.11279*.

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

[63] Y. Zhou et al., "Secchi depth estimation for optically-complex waters based on spectral angle mapping–derived water classification using Sentinel-2 data," *Int. J. Remote Sens.*, vol. 42, no. 8, pp. 3123–3145, Apr. 2021.

[64] C. Willmott and K. Matsuura, "Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance," *Climate Res.*, vol. 30, pp. 79–82, 2005.

[65] Q. Huynh-Thu and M. Ghanbari, "Scope of validity of PSNR in image/video quality assessment," *Electron. Lett.*, vol. 44, no. 13, pp. 800–801, 2008.

[66] M. Amrouche, H. Carfantan, J. Idier, and V. Martin, "Statistical destriping of pushbroom-type images based on an affine detector response," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5627014.

[67] J. P. Arroyo-Mora et al., "Assessing the impact of illumination on UAV pushbroom hyperspectral imagery collected under various cloud cover conditions," *Remote Sens. Environ.*, vol. 258, Jun. 2021, Art. no. 112396.

[68] W. He et al., "Non-local meets global: An iterative paradigm for hyperspectral image restoration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 4, pp. 2089–2107, Apr. 2022.

[69] E. Freire, L. Schnitman, W. Oliveira, and A. Duarte, "Evaluation of the Huffman encoding for memory optimization on hardware network intrusion detection," in *Proc. 3rd Brazilian Symp. Comput. Syst. Eng.*, Dec. 2013, pp. 131–136.

[70] D. M. Kate, "Hardware implementation of the Huffman encoder for data compression using altera DE2 board," *Int. J. Adv. Eng. Sci.*, vol. 2, pp. 11–15, 2012.

[71] S. Zhou, Y. Wu, Z. Ni, X. Zhou, H. Wen, and Y. Zou, "DoReFa-Net: Training low bitwidth convolutional neural networks with low bitwidth gradients," 2016, *arXiv:1606.06160*.

[72] S. K. Esser, J. L. McKinstry, D. Bablani, R. Appuswamy, and D. S. Modha, "Learned step size quantization," in *Proc. Int. Conf. Learn. Represent.*, 2020.