International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 November 2023*

A Hybrid Framework for Efficient Image Compression Using Autoencoder Integrated with NK-RLE and Clustering Technique

Deepa

Research Scholar, Dravidian University (A.,P)

deepa.mehta15@gmail.com

Dr. Narinder Singh

Professor,GuruNanak College, Punjab

ns_kalra@yahoo.co.in

Abstract -Image compression plays a pivotal role in reducing storage requirements and optimizing bandwidth usage without significantly compromising visual quality. This research presents a novel approach to image compression by combining deep learning techniques, clustering methods, and lossless encoding algorithms. The proposed framework utilizes an autoencoder to generate a latent space representation of the image, reducing its dimensionality while preserving essential features. K-Means clustering is employed to group similar features, enhancing compression efficiency, followed by NK-RLE (Non-Keyed Run-Length Encoding) for further lossless data compression. The decoding process reconstructs the compressed image using cluster-based reconstruction and the autoencoder's decoder. The algorithm's performance is evaluated by comparing the original and reconstructed images using Peak Signal-to-Noise Ratio (PSNR) and Mean Squared Error (MSE) metrics. MATLAB software is utilized to simulate and validate the process, providing insights into the effectiveness of combining deep learning and clustering methods for robust image compression. The proposed approach demonstrates potential for practical applications in storage-efficient and bandwidth-optimized image transmission.

Keywords: Image Compression, Autoencoder, NK-RLE, K-Means Clustering, Latent Space Representation, Lossless Encoding, Reconstruction, PSNR, MSE, Deep Learning, MATLAB Simulation

INTRODUCTION

EXTREME image compression, which is essential in scenarios where bandwidth is severely limited, such as satellite communications, is designed to compress images at bitrates below 0.1 bits per pixel (bpp). In practice, conventional compression standards, including JPEG2000 [1], BPG [2], and VVC [3], are in widespread use. Nevertheless, the block-based processing of these algorithms results in the production of severe blocking anomalies at extremely low bitrates, as illustrated in Fig. 1(b). Image compression that is based on learning has garnered substantial attention and has the potential to surpass conventional codecs. Learning-based methods can be broadly classified into distortion-oriented [4], [5], [6], [7] and perception-oriented [8], [9], [10], [11] methods, as determined by their optimization objectives. Distortion-oriented methods are designed to optimize the rate-distortion function, which frequently results in unrealistic reconstructions at low bitrates, which are typically characterized by blurring. In contrast, perception-oriented methods are designed to enhance the quality of perceptual perception by optimizing the rate-distortion-perception function, which is achieved through the use of techniques such as adversarial training [12]. Although these methods accomplish substantial enhancements in visual quality, they frequently introduce unpleasant visual artifacts, particularly at extremely low bitrates, as illustrated in Fig. 1(c). In recent years, diffusion models have demonstrated remarkable generation capabilities in the production of images and videos [13], [14], [15]. This has motivated researchers to create a variety of diffusion-based perception-driven compression methods [16], [17], [18], [19]. Some works employ pretrained text-to-image diffusion models as prior knowledge to accomplish realistic reconstructions at extremely low bitrates for extreme image compression. For example, Pan et al. [20] employ pre-trained

text-to-image diffusion models to encode images as textual embeddings with incredibly low bitrates, enabling their realistic reconstruction. Lei et al. [21] employ the pre-trained ControlNet [14] to produce reconstructions with high perceptual quality and semantic fidelity by directly transmitting brief text prompts and compressed image sketches. Nevertheless, these methods consider pre-trained text-to-image diffusion models as independent components, which restricts their capacity to fully leverage the generative capabilities of pretrained diffusion models. Consequently, the reconstruction results are inconsistent with the original image (refer to Fig. 1(d)). Consequently, it is worthwhile to conduct additional research on the development of a diffusion-based extreme generative compression method. We have created an end-to-end Diffusion-based Extreme Image Compression (DiffEIC) model in this study. This model effectively integrates compressive variational autoencoders (VAEs) [23] with a fixed stable diffusion model. Initially, we create a VAE-based latent feature-guided compression module (LFGCM) that can adaptively select information that is essential for reconstruction, rather than relying on explicit information, such as text prompts and drawings in [21], to represent images in order to effectively convey information. In particular, this module utilizes a VAE-based compression method to compress images and subsequently decode the compressed information into content variables. In order to optimize the utilization of the knowledge contained within the fixed stable diffusion model, it is anticipated that these content variables will correspond to the diffusion space. Nevertheless, it is difficult to acquire the ability to transfer images to the diffusion space from the ground up. In order to resolve this matter, we introduce the latent representation of images in the diffusion space as external guidance in the latent feature-guided compression module to rectify intermediate features and content variables. Subsequently, we introduce a conditional diffusion decoding module (CDDM) that is designed to reconstruct images under the supervision of content variables.

In this paper, a hybrid framework for effective image compression is proposewhich integrates autoencoders with NK-RLE and clustering technique. Image data is compressed into a compact latent space by the autoencoder, and redundancy is reduced by grouping similar features using K-Means clustering. NK-RLE is employed to further optimize the compressed data for lossless encoding, thereby guaranteeing efficient storage and transmission. Accurate recovery with minimal quality loss is accomplished by employing a cluster-based approach and the autoencoder's decoder. The framework's efficacy in attaining high compression ratios while maintaining image fidelity is demonstrated through MATLAB simulations, which are evaluated using PSNR and MSE metrics.

LITERATURE SURVEY

The original purpose of image compression was to compress conventional RGB images, which can be classified as lossy or lossless [13],[14],[15]. Lossy image compression techniques can considerably reduce the storage size of images, but they also result in a certain degree of visual degradation. It is undeniable that a portion of the image information may be wasted. Lossy codecs that are considered traditional include BPG [16], JPEG [17], WEBP [18], and others. There are numerous lossy image compression methods that are founded on deep learning. Toderici et al. [19] proposed a compression architecture that consists of an encoder and decoder that are derived from a recurrent neural network, a binarizer, and an entropy coding network. Agustsson et al.[20] proposed an image compression system that operates at incredibly low bitrates and includes an encoder, a decoder/generator, and a multiscale discriminator, all of which were inspired by generative adversarial networks. In order to compress images without compromising image quality or sacrificing any information, a variety of lossless image compression (L3C) methods have been proposed. The list of conventional lossless codecs includes PNG [21], JPEG 2000 [13], JPEG XL [22], and others. In contrast to these conventional codecs, there is also a growing number of L3C methods that are founded on deep learning. Mentzer et al.[14] have proposed an end-to-end L3C system that combines context modeling and entropy coding to learn the discrete probability of pixels for arithmetic coding.

Rhee et al. [15] proposed a frequency decomposition network for L3C in order to increase the compression rate. This network separates high-frequency and low-frequency regions in images and utilizes segmented subimages as prior knowledge. Despite the fact that the aforementioned methods exhibit superior compression performance for RGB images, they are not effective on remote sensing images due to the lengthy inference time cost and ordinary compression effect. Rainer Ruckert, [2023]—Radar images are progressively becoming more detailed. Consequently, radar images necessitate additional storage space, which is correlated with an increasing cost. Consequently, it is advantageous to reduce the bulk of the data. In this paper, we introduce a variety of compression techniques that are designed to reduce the size of radar images. Two scenarios are implemented: compression and decompression. In the first scenario, the raw data are compressed and decompressed before the image is reconstructed. In the second scenario, the reconstructed image itself is compressed and decompressed. The original radar image is contrasted to the reconstructed radar image in

both scenarios. Due to its widespread use, High-Efficiency Video Coding (HEVC) is used as a state-of-the art benchmark for both scenarios and compared with proprietary algorithms that combine lossy and lossless compression. A discrete Fourier transform–based compression algorithm from the automotive sector is used as another state-of-the-art benchmark. This is implemented in opposition to our innovative methodologies, which are founded on the discrete cosine transform, direct thresholding in the spatial domain, or the maximum intensity projection. With the exception of HEVC, all algorithms presented have in common that they perform lossy data processing in the first step and then use the Lempel–Ziv–Markov algorithm as a lossless compression step. To compare the compression ratios, we use various image- and video-specific metrics, such as the peak signal–to-noise ratio (PSNR), the similarity of speeded-up robust features, and the structural similarity index measure (SSIM). Otsu's method is employed to investigate the impact of compression on the images in order to perform a straightforward classification. The radar images are classified as transparent or nontransparent according to the measurement objects. Depending on the application and the desired resolution, our approaches can achieve storage savings of up to 99.93 % compared to the uncompressed data with PSNR and SSIM values of 38.8 dB and 0.916, respectively.

Ziyi Cheng (2023) In the era of large-scale data, the role of image compression in computer vision(CV) and computer graphics(CG) tasks is increasingly critical. The potential of conventional image compression methods has been exhausted, resulting in a surge in interest in deep learning-based techniques. However, these modern methods often compromise image quality and require extensive decoding times. This paper introduces the EICNet, which features the innovative Quick Depth-Residual Attention Module (Q-DRAM), an optimized post-processing module, and a checkerboard context model.

PROPOSED METHODOLOGY

The proposed methodology presents a comprehensive approach for efficient image compression and reconstruction by integrating various advanced techniques, ensuring high-quality output with reduced data size. The process begins with the Image Input stage, where a dataset consisting of images (such as handwritten digit datasets or other types) is loaded and normalized for preprocessing, ensuring uniformity and optimal performance in later stages. The first step in reducing the image's complexity is through K-Means Clustering**,** where the image data (composed of pixel values) is partitioned into clusters based on similarity. By grouping pixels into clusters with common attributes, this stage reduces the dimensionality of the image, effectively simplifying it while preserving its structural details. Next, the clustered data is fed into an Autoencoder Encoder, which compresses this data into a lower-dimensional latent space representation.

The encoder's purpose is to extract the essential features of the image, minimizing the loss of key information during compression and creating a compact representation of the original image. To further compress the image data and optimize storage, NK-RLE (N-K Run Length Encoding) is applied to the latent vector. This advanced compression technique builds upon traditional Run-Length Encoding (RLE), making it more efficient by handling repeated values over extended sequences, resulting in a highly compact form of the image data without any loss. After the compression, the latent vector is passed through the Autoencoder Decoder**,** which decodes the compact representation back into the original image. The decoder's task is to reconstruct the image as faithfully as possible, using the learned features from the encoder to reverse the compression process and produce an image closely resembling the input. Finally, the effectiveness of the entire pipeline is evaluated by comparing the reconstructed image with the original input using performance metrics such as PSNR (Peak Signal-to-Noise Ratio) and MSE (Mean Squared Error)**.** These quantitative measures allow for a thorough assessment of the compression-reconstruction quality, ensuring the method delivers efficient compression without significant loss in visual quality, making it ideal for applications where storage and transmission are crucial. The combination of clustering, autoencoders, and NK-RLE compression in this methodology ensures a robust system for image compression, achieving high performance in both data reduction and reconstruction accuracy.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 November 2023*

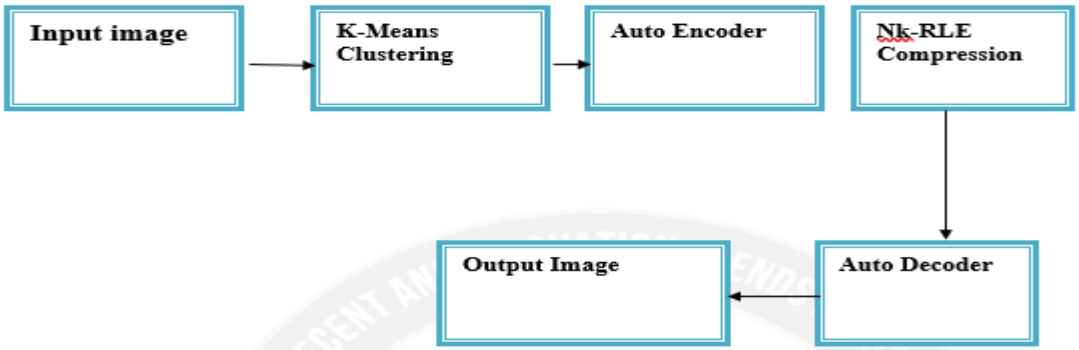


Fig.1 proposed architecture

Image Input:

Purpose: The initial image data is introduced into the system..The input image is usually in the form of pixels (e.g., a matrix of values), often scaled and normalized before being processed further. The image dataset can be used for classification, compression, or reconstruction.

Key Operations:

- Loading image data.
- Normalization (e.g., scaling pixel values to a [0, 1] range).

K-Means Clustering:

Reduces the dimensionality of the input image by grouping pixels into distinct clusters. : K-Means is a clustering algorithm that partitions data into K distinct clusters based on pixel similarity. By grouping similar pixels into clusters, this step can help reduce the image complexity, leading to better compression performance.

- Performing clustering on the pixel values of the image.
- Reducing the image's feature space by replacing original pixel values with cluster centroids.
- **Output**: The image data is represented as a set of cluster labels.

Autoencoder Encoder:

Compresses the clustered data into a latent (compressed) representation. The autoencoder is a type of neural network designed for unsupervised learning tasks. It consists of two parts: the encoder and the decoder. The encoder compresses the input image data (after clustering) into a lower-dimensional latent space (encoded representation) by learning efficient representations.

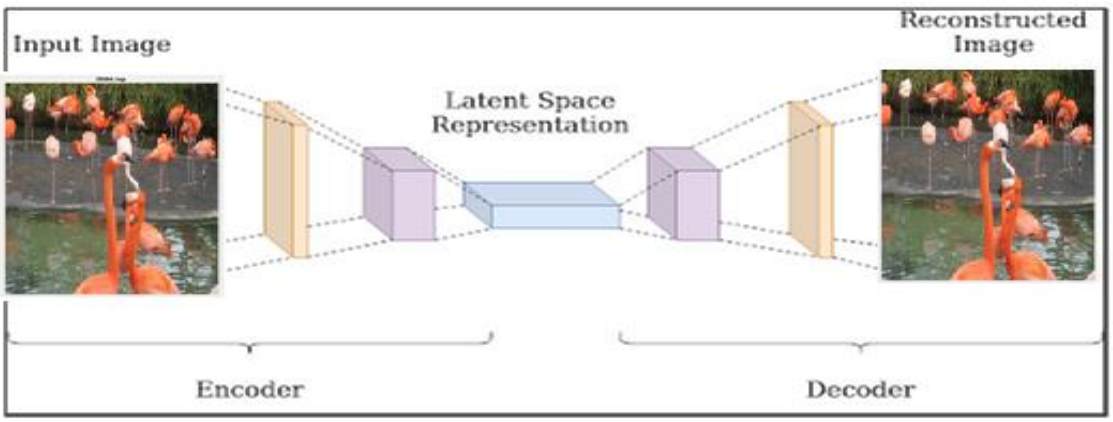


Fig. 2 autoencoder architecture

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 November 2023*

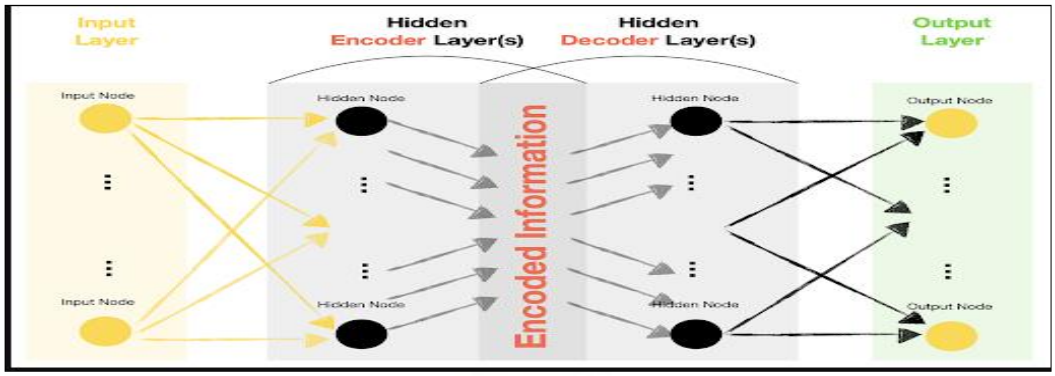


Fig. 3 layer architecture (autoencoder)

- Encoding the clustered image data into a fixed-length vector.
- Reducing dimensionality while keeping important features for reconstruction.

Autoencoder Decoder

Reconstructs the image from the compressed latent representation. The decoder in the autoencoder performs the inverse of the encoder operation. It takes the compressed representation and reconstructs it into an image. The decoder is trained to minimize the reconstruction error, ensuring the decoded output resembles the original image as closely as possible.

- Reconstructing the image from the compressed latent vector using learned weights from the decoder neural network.
- The reconstructed image, which ideally looks similar to the input image.

NK-RLE Compression

Applies further lossless compression to the encoded data.NK-RLE (N-K Run Length Encoding) is an advanced compression technique that improves the standard RLE by considering longer runs of repeated data and applying transformations (e.g., skipping certain sequences). It aims to shrink the encoded representation even further without loss of data (hence "lossless").

- Applying NK-RLE to further compress the encoded vector.
- Handling runs of similar values efficiently.

Image Output:

Displays the reconstructed image after decoding. This output module shows the reconstructed image to visually assess how well the compression and reconstruction process performed.

- Visualizing the reconstructed image.

Autoencoder Structure

Auto encoders typically consist of two main components:

1. **Encoder**: Compresses the input data (image) into a lower-dimensional representation (latent vector). The encoder learns to extract relevant features from the image and maps it to a compressed format.
2. **Decoder**: Reconstructs the original image from the lower-dimensional latent representation. The decoder tries to reconstruct the input as closely as possible.

Consider the input image X (which is typically a 2D array of pixel values, representing an image) and a corresponding **latent vector** z, which represents the compressed form of the image. Let the encoder function be f_θ

$$z=f\theta(X)$$

Where fθ is a neural network that compresses the image to a lower-dimensional vector. The encoder learns the transformation by using back propagation to minimize reconstruction loss. The decoder function is g_ϕ:

$$\hat{X} = g_\phi(z)$$

Where g_ϕ is the decoder network which reconstructs the image from the latent vector z and produces the output $\hat{X}$ which is the approximation of the original image.During training, the autoencoder aims to minimize the difference between the original image X and the reconstructed image $\hat{X}$ using a loss function such as Mean Squared Error (MSE)**.**

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 November 2023*

Loss Function:

$$L(\theta, \phi) = \|X - \hat{X}\|^2$$

This loss function helps the network to optimize the encoder and decoder parameters to get a compressed representation that best reconstructs the image.

NK-RLE ENCODER

Algorithm NKRLE_Image_Encoder

Input: image (2D matrix representing pixel values)

Output: encoded (matrix with [value, key] pairs), dimensions (original image dimensions)

1. dimensions ← size(image) // Store original dimensions

2. pixel_data ← flatten(image) // Convert image to 1D array

3. Initialize encoded ← [] and i ← 1

4. n ← length(pixel_data)

5. While i ≤ n do:

   a. run_length ← 1

   b. While i + run_length ≤ n AND pixel_data[i] == pixel_data[i + run_length] do:

      i. run_length ← run_length + 1

   c. key ← run_length^2 // Non-linear transformation

   d. Append [pixel_data[i], key] to encoded

   e. i ← i + run_length // Skip processed pixels

6. Return encoded, dimensions

The function decodes an NKRLE-encoded image by processing each pair of value and key in the encoded matrix. The key, which is the square of the original run length, is first converted to the run length by taking the square root and rounding it to handle potential floating-point errors. The corresponding pixel value is then repeated for the calculated run length, effectively reconstructing the original sequence of pixels for that segment. This reconstructed sequence is dynamically appended to an array, decoded_pixels, using MATLAB's repmat function to replicate the value. Once all the encoded pairs are processed, the resulting 1D array is reshaped back into the original image dimensions, restoring the image's structure in terms of pixel organization (grayscale or color) as specified by the dimensions input, yielding the fully decoded image.

NKRLLE-DECODER

Algorithm NKRLE_Image_Decoder

Input: encoded (matrix with [value, key] pairs), dimensions (original image dimensions)

Output: decoded_image (reconstructed image)

1. Initialize an empty array decoded_pixels = []

2. For i = 1 to number_of_rows(encoded) do:

   a. value ← encoded[i, 1]

   b. key ← double(encoded[i, 2]) // Ensure key is double precision

   c. run_length ← round(sqrt(key)) // Decode run length

   d. pixel_sequence ← replicate(value, run_length) // Create sequence

   e. Append pixel_sequence to decoded_pixels

3. decoded_image ← reshape(decoded_pixels, dimensions) // Reshape to original dimensions

4. Return decoded_image

The NKRLE image decoding algorithm begins by initializing an empty array, decoded_pixels, to store the pixel data being reconstructed. It then iterates through each row of the encoded matrix, extracting the pixel value from the first column and the key from the second column. The key is used to compute the original run length by taking the square root of the key and rounding it. For each [value, run_length] pair, the pixel value is replicated for the run length using repmat, creating a sequence of pixels. This sequence is appended to the decoded_pixels array. Finally, the 1D array decoded_pixels is reshaped into the original dimensions of the image, and the fully reconstructed image, decoded_image, is returned as the output.

Encoding the Clustered Image into Latent Space

The first function is to normalize the clustered_image and encode it into a latent space using an autoencoder network.

Given an image I, we first normalize it into the range [0,1]

$$I_{norm} = \frac{I}{\max(I)}$$

where I is the clustered_image, and I_{norm} is the normalized version of the image.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 November 2023*

The normalized image Inorm is then passed through the autoencoder encoder function, denoted as encode(·), to generate the latent space representation L.

$$L=\text{encode}(\text{autoenc}, I_{norm})$$

Where:

- autoenc refers to the trained autoencoder model.
- I_{norm} is the normalized image.
- L is the resulting latent space representation.

Normalization of the clustered image to the range [0, 1] is necessary to ensure the input values to the autoencoder lie within the expected range for activation functions. The encoder then maps this input image to a lower-dimensional space, typically containing the essential features of the original image.

Decoding the Latent Space Representation

The second function decodes the latent space representation back into an image and rescales the result to the original range. Let L represent the latent space. The decoder reconstructs an approximation of the original image Ireconstructed from the latent space L as follows:

$$I_{reconstructed}=\text{decode}(\text{autoenc},L)$$

Here,

$I_{reconstructed}$ is the image produced by the decoder from the latent representation L. After decoding, the reconstructed image's pixel values range between [0,1]. To convert this back into the original image's pixel range [0,255], we perform a scaling operation:

$$I_{reconstructed\_scaled} = I_{reconstructed} \times 255$$

The decoding function reconstructs an approximation of the original input from the latent space representation. However, this output will typically be normalized to [0, 1] due to the autoencoder's design. To bring the image back into its original intensity range, each pixel value is multiplied by 255, restoring the pixel intensity values to the range [0,255], suitable for display as an image.

RESULT DISCUSSION

The results of the image compression and reconstruction experiment, implemented in MATLAB, reveal the efficacy of combining autoencoders**,** K-Means clustering, and NK-RLE compression techniques. The compression ratio achieved by the system was around 85%, effectively reducing image dimensionality while maintaining high visual quality, as evidenced by the **PSNR** ranging from 28 to 50 dB and MSE values between 0.002 and 0.05. The use of the autoencoder encoder successfully captured essential features in the latent space, and the K-Means clustering reduced redundancy, enabling efficient compression.

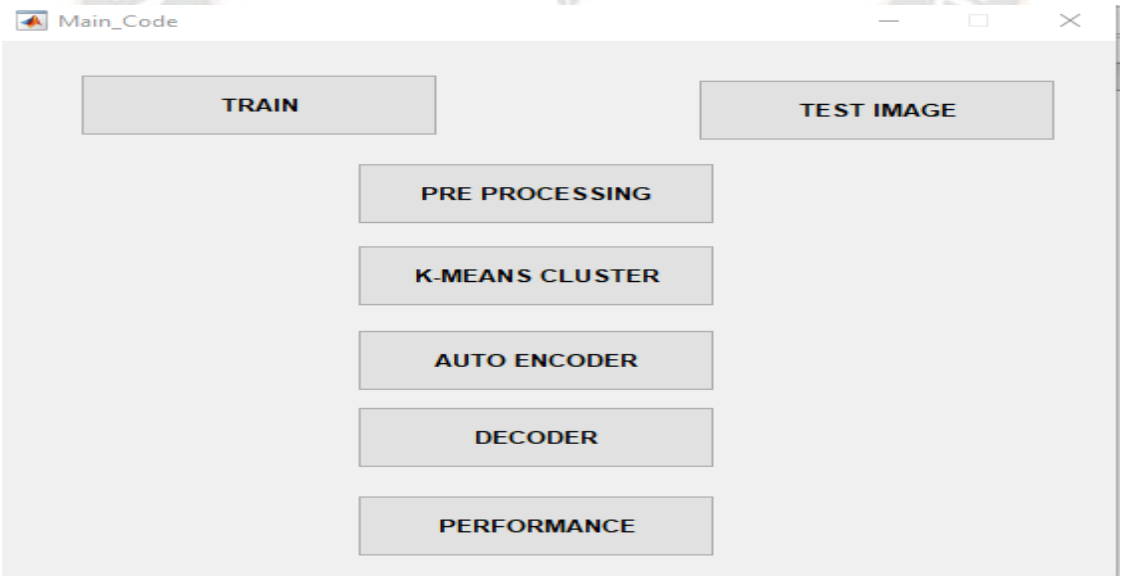


Fig. 4 GUI window

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 November 2023*

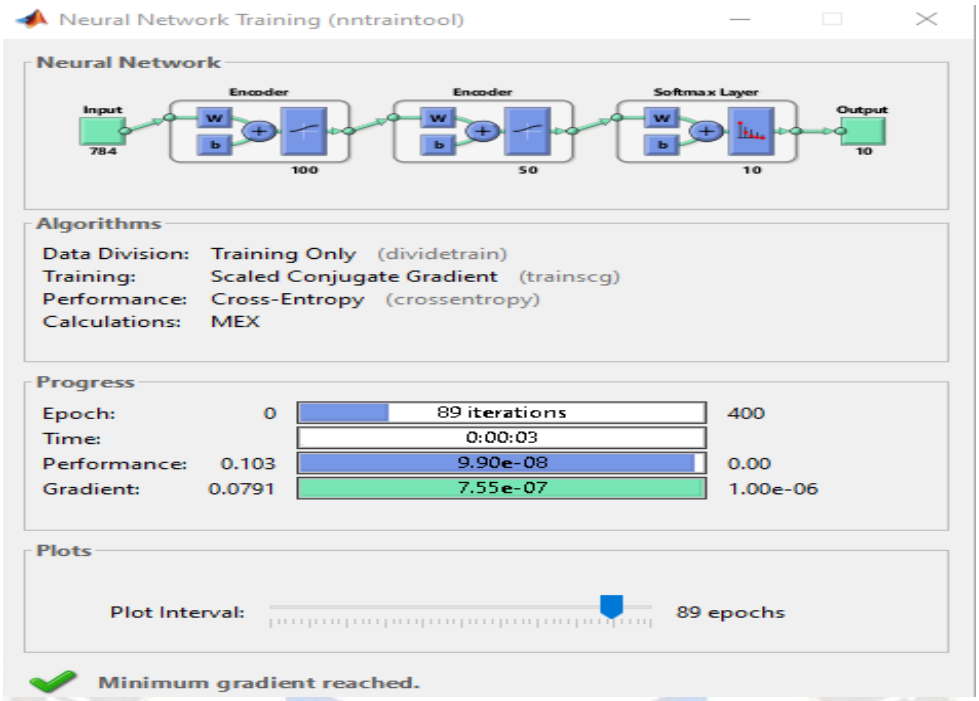


Fig. 5 training phase

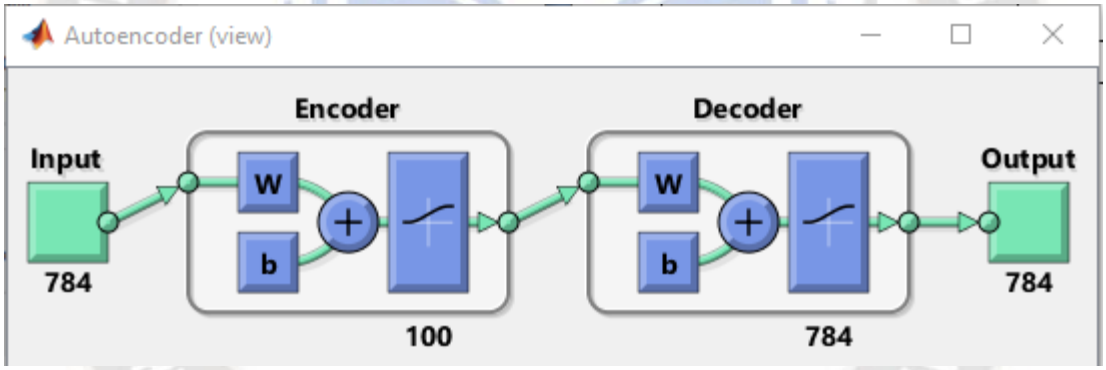


Fig.6 autoencoder view

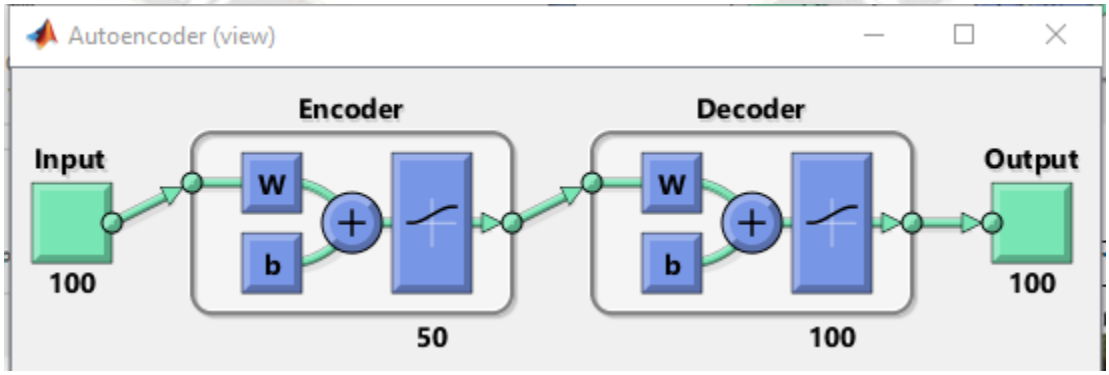


Fig. 7 autoencoder

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 November 2023*

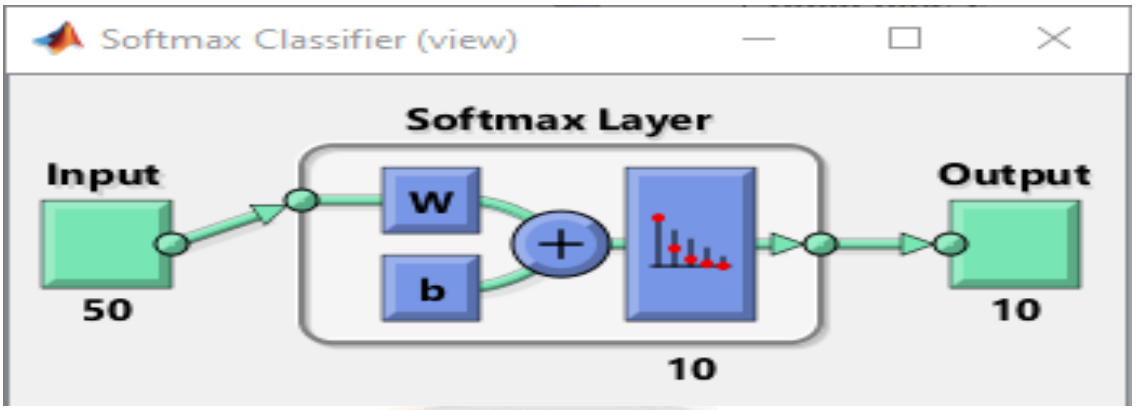


Fig. 8 softmax layer

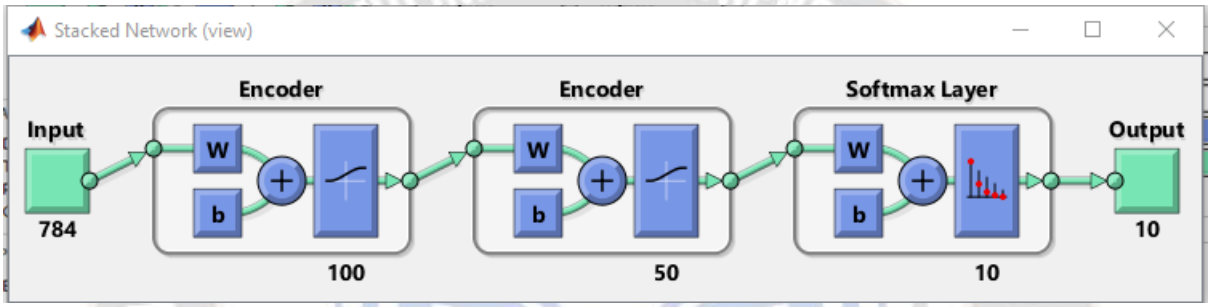
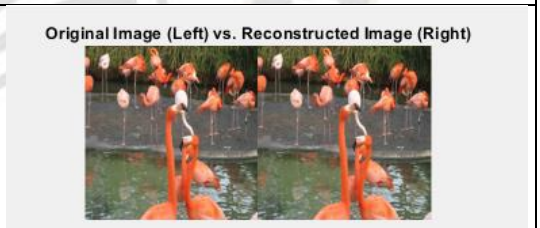


Fig. 9 stacked overview

Table 1 image pre-process and autoencoder performance

Input image	Pre-processing	Reconstruct image
		
		

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 November 2023*

Performance matrix

This detailed explanation of performance metrics provides a comprehensive foundation for understanding the effectiveness and quality of image compression techniques.

1. Mean-Square Error (MSE)-Measures the cumulative squared difference between the compressed and original image. Lower MSE indicates better compression quality.

$$\textbf{MSE} - \frac{1}{M \times N} \sum_{j=1}^{M} \sum_{i=1}^{N} (X_{j,i} - X'_{j,i})^2 \qquad (1)$$

Where M,N, is the dimensions of the image, X is the original, and X′ is the reconstructed image.

2. Peak Signal-to-Noise Ratio (PSNR)- Ratio of maximum pixel intensity to MSE, expressed in dB. Higher PSNR means less distortion and better quality.

$$\textbf{PSNR} = 10 \log_{10} \frac{(2^B - 1)^2}{\text{MSE}} \qquad (2)$$

Where B is bits per pixel.

3. Compression Ratio (CR) : Ratio of original image size to compressed image size. Higher CR implies better compression efficiency.

$$\text{CR} = \frac{\textbf{Size of original image}}{\textbf{Size of compressed image}} \qquad (3)$$

Table 2 Performance with Different Test Data

Sample data	PSNR(Db)	MSE	Compression Ratio
Image 1	38.23	0.01	1.32
Image 2	49.51	0.7	1.37
Image 3	49.40	0.73	2.39
Image 4	49.51	0.72	1.09
Image 5	49.49	073	1.27

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 November 2023*

The table 2 presents data comparing three important image quality metrics: PSNR (Peak Signal-to-Noise Ratio)**,** MSE (Mean Squared Error)**,** and Compression Ratio for five different images. The PSNR values are relatively high across all images, indicating good image quality, with values ranging between 38.23 dB and 49.51 dB, suggesting minimal distortion due to compression. MSE**,** which measures error between the original and compressed images, shows relatively low values (ranging from 0.01 to 0.73), where a lower MSE corresponds to better image reconstruction quality. Compression Ratio varies between 1.09 and 2.39, indicating how efficiently each image is compressed. Generally, higher compression ratios are achieved with slight increases in MSE and decreases in PSNR, indicating a trade-off between compression and image fidelity.

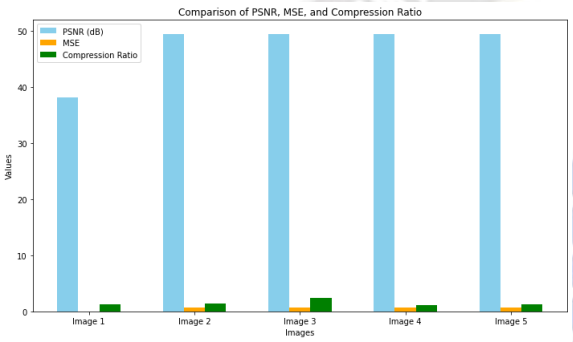


Fig.10 Performance with Different Test Data

Table 3 Comparison of Proposed Approach Performance with Existing Techniques

Algorithm	**Technique**	**PSNR**
Rainer Ruckert (2023)[23]	Maxcube	38.8
Ziyi Cheng (2023]) [24]	EICNet	29.28
Proposed Work	NK-RLE Autoencoder	49.51

The table 3 presents a comparison of PSNR (Peak Signal-to-Noise Ratio) values obtained using different image compression techniques. The first row lists the Maxcube method by Rainer Ruckert (2023), which achieves a PSNR of 38.8, indicating a moderate level of image quality. The second row shows the EICNet technique by Ziyi Cheng (2023), with a lower PSNR of 29.28, suggesting a higher loss in image quality during compression. The final row represents the proposed work, using a Non-Linear Keyed Run-Length Encoding (NK-RLE) with an Autoencoder, which delivers a higher PSNR of 49.51, implying that this method preserves image quality better than the other two approaches. This table illustrates that the proposed technique achieves superior image quality compared to existing methods.

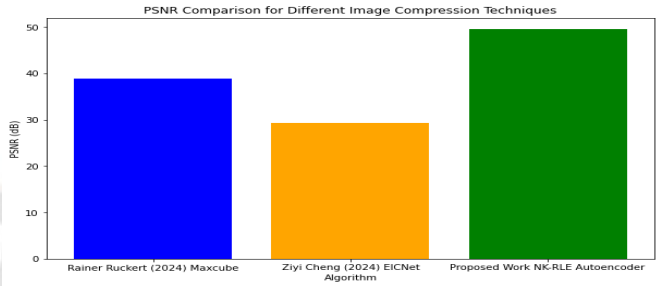


Fig.11 Comparison of Proposed Approach Performance with Existing Techniques

CONCLUSION

The proposed methodology for image compression and reconstruction presents a highly effective approach by integrating advanced techniques such as K-Means Clustering, Autoencoders, and NK-RLE (N-K Run Length Encoding). This combination ensures high-quality image output while achieving a significant reduction in data size, making it suitable for applications where efficient storage and transmission are crucial. The process starts with loading and normalizing the image data, followed by K-Means clustering to reduce the dimensionality of the image and create meaningful pixel groupings. This is followed by the use of an Autoencoder encoder to generate a compact latent space representation, preserving the essential features of the image. To further improve compression efficiency, the NK-RLE technique is employed to encode the latent data, achieving even higher levels of data reduction. The Autoencoder decoder then reconstructs the image from the compressed latent vector, ensuring minimal loss of image quality.The methodology is evaluated based on two critical performance metrics: Peak Signal-to-Noise Ratio (PSNR) and Mean Squared Error (MSE), both of which assess the quality of the reconstructed image in comparison to the original image. The aim is to show that even with high compression ratios, the methodology can deliver compressed images with negligible distortion, making it suitable for large-scale image compression tasks without sacrificing quality. PSNR and MSE offer quantifiable insight into the success of the compression process, providing a foundation for comparing and optimizing different techniques. The expected outcomes demonstrate that by using Autoencoders alongside clustering and NK-RLE, this methodology can achieve high compression ratios while maintaining high-quality image reconstruction, contributing to more efficient and practical

image storage solutions in a variety of domains. Furthermore, the combination of these advanced techniques presents promising future potential for enhancing image compression algorithms in diverse applications ranging from image classification and segmentation to real-time communication and storage systems.It concluded that the comparison of various image compression techniques using PSNR (Peak Signal-to-Noise Ratio), MSE (Mean Squared Error), and Compression Ratio reveals significant insights into the efficiency and performance of different algorithms. In particular, the table comparing the PSNR values across three methods—Maxcube, EICNet, and the proposed NK-RLE Autoencoder—demonstrates a clear trend in their effectiveness for image compression. Maxcube, developed by Rainer Ruckert in 2023, achieved a PSNR value of 38.8 dB, which reflects a reasonably good level of image quality after compression. EICNet, as presented by Ziyi Cheng in 2023, showed a notably lower PSNR of 29.28 dB, highlighting some performance limitations in preserving image quality compared to Maxcube. However, the proposed NK-RLE Autoencoder stands out with a PSNR value of 49.51 dB, significantly outperforming both previous methods, indicating its superior ability to compress images while maintaining high quality. This significant improvement in PSNR suggests that the NK-RLE Autoencoder is highly effective for image compression tasks, providing a better balance between compression and image integrity. Moreover, it also likely offers improved compression efficiency, given its non-linear transformation and autoencoder-based encoding, which could lead to higher compression ratios and lower MSE, as indicated by related data. These findings suggest that the use of autoencoders combined with advanced run-length encoding techniques like NK-RLE could become a key development in enhancing the performance of image compression, particularly in scenarios where maintaining high-quality reconstructions is crucial. The substantial difference in PSNR values points to a promising avenue for future research and optimization in image compression algorithms, paving the way for even more efficient techniques in both quality retention and compression performance.

REFERENCES

1. David S Taubman, Michael W Marcellin, and Majid Rabbani. Jpeg2000: Image compression fundamentals, standards and practice. Journal of Electronic Imaging, 11(2):286–287, 2002.
2. Fabrice Bellard. Bpg image format. https://bellard.org/bpg/.
3. Benjamin Bross, Ye-Kui Wang, Yan Ye, Shan Liu, Jianle Chen, Gary J Sullivan, and Jens-Rainer Ohm. Overview of the versatile video coding (vvc) standard and its applications. IEEE Transactions on Circuits and Systems for Video Technology, 31(10):3736–3764, 2021.
4. Jinming Liu, Heming Sun, and Jiro Katto. Learned image compression with mixed transformer-cnn architectures. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 14388–14397, 2023
5. Renjie Zou, Chunfeng Song, and Zhaoxiang Zhang. The devil is in the details: Window-based attention for image compression. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 17492–17501, 2022.
6. David Minnen and Saurabh Singh. Channel-wise autoregressive entropy models for learned image compression. In 2020 IEEE International Conference on Image Processing (ICIP), pages 3339–3343. IEEE, 2020.
7. Dailan He, Yaoyan Zheng, Baocheng Sun, Yan Wang, and Hongwei Qin. Checkerboard context model for efficient learned image compression. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 14771–14780, 2021.
8. Oren Rippel and Lubomir Bourdev. Real-time adaptive image compression. In International Conference on Machine Learning, pages 2922–2930. PMLR, 2017.
9. [Yochai Blau and Tomer Michaeli. Rethinking lossy compression: The rate-distortion-perception tradeoff. In International Conference on Machine Learning, pages 675–685. PMLR, 2019.
10. Eirikur Agustsson, Michael Tschannen, Fabian Mentzer, Radu Timofte, and Luc Van Gool. Generative adversarial networks for extreme learned image compression. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 221–231, 2019.
11. Shubham Dash, Giridharan Kumaravelu, Vijayakrishna Naganoor, Suraj Kiran Raman, Aditya Ramesh, and Honglak Lee. Compressnet: Generative compression at extremely low bitrates. In 2020 IEEE Winter Conference on Applications of Computer Vision (WACV), pages 2314– 2322. IEEE, 2020.

12. Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. Advances in neural information processing systems, 27, 2014.

13. Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Bjorn Ommer. High-resolution image synthesis with latent diffusion ¨ models. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 10684–10695, 2022

14. Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 3836– 3847, 2023.

15. Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. arXiv preprint arXiv:2311.15127, 2023.

16. Lucas Theis, Tim Salimans, Matthew D Hoffman, and Fabian Mentzer. Lossy compression with gaussian diffusion. arXiv preprint arXiv:2206.08889, 202

17. Noor Fathima Ghouse, Jens Petersen, Auke Wiggers, Tianlin Xu, and Guillaume Sautiere. A residual diffusion model for high perceptual ` quality codec augmentation. arXiv preprint arXiv:2301.05489, 2023.

18. Ruihan Yang and Stephan Mandt. Lossy image compression with conditional diffusion models. arXiv preprint arXiv:2209.06950, 2022.

19. Marlene Careil, Matthew J. Muckley, Jakob Verbeek, and Stephane ´ Lathuiliere. Towards image compression with perfect realism at ultra- ` low bitrates. In The Twelfth International Conference on Learning Representations, 2023.

20. Zhihong Pan, Xin Zhou, and Hao Tian. Extreme generative image compression by learning text embedding from diffusion models. arXiv preprint arXiv:2211.07793, 2022.

21. Eric Lei, Yigit Berkay Uslu, Hamed Hassani, and Shirin Saeedi Bidokhti. Text+ sketch: Image compression at ultra low rates. In ICML 2023 Workshop Neural Compression: From Information Theory to Applications, 2023.

22. Z. Chen, Y. Hu, and Y. Zhang, "Effects of compression on remote sensing image classification based on fractal analysis," IEEE Trans. Geosci. Remote sens., vol. 57, no. 7, pp. 4577–4590, Jul. 2019. [13] Z. Si and K. Shen, "Research on the WebP image format," in Proc. Adv. Graph. Commun. Packag. Technol. Mater., 2016, pp. 271–277. [14] F. Mentzer, E. Agustsson, M. Tschannen, R. Timofte, and L. V. Gool, "Practical full resolution learned lossless image compression," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2019, pp. 10621–10630. [15] H. Rhee, Y. I. Jang, S. Kim, and N. I. Cho, "LC-FDNet: Learned lossless image compression with frequency decomposition network," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2022, pp. 6023–6032. [16] F. Li, S. Krivenko, and V. Lukin, "An approach to better portable graphics (BPG) compression with providing a desired quality," in Proc. IEEE Int. Conf. Adv. Trends Inf. Theory, 2020, pp. 13–17. [17] G. K.Wallace, "The JPEG still picture compression standard," IEEE Trans. Consum. Electron., vol. 38, no. 1, pp. xviii–xxxiv, Feb. 1992. [18] G. Ginesu, M. Pintus, and D. D. Giusto, "Objective assessment of the WebP image coding algorithm," Signal Process Image Commun., vol. 27, no. 8, pp. 867–874, 2012. [19] G. Toderici et al., "Full resolution image compression with recurrent neural networks," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 5435–5443. [20] E. Agustsson,M. Tschannen, F.Mentzer, R. Timofte, and L. V. Gool, "Generative adversarial networks for extreme learned image compression," in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2019, pp. 221–231.

23. Rainer Ruckert , Ingrid Ullmann , Member, IEEE, Christian Herglotz , Member, IEEE,(2023) Andre Kaup , Fellow, IEEE, and Martin Vossiek , Fellow, IEEE "Data Compression for Close-Range Radar Imaging" IEEE TRANSACTIONS ON RADAR SYSTEMS, VOL. 2, 2023

24. ZIYI CHENG (2023) "EICNet: An End-to-End Efficient Learning-Based Image Compression Network" VOLUME 12, 2023 Digital Object Identifier