IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

Using Multi-Layer Percepton techniques for lossless multi-spectral satellite image compression

[1]Abhinav Jain, [1]Adarsh Kumar, [1]Ashish Kumar, [2]Gyan Singh Yadav

[1]Undergraduate, [2]Assistant Professor

Computer Science Department

Indian Institute of Information and Technology Kota

Abstract : This research investigates the utilization of Multilayer Perceptrons (MLPs) in image compression. We explore the efficacy of MLPs in reducing image file sizes while pre- serving visual quality. By training MLPs on diverse image datasets, we assess their compression performance across various configurations. Our findings reveal the poten- tial of MLP-based image compression to revolutionize data storage and transmission in resource-constrained environments, offering a promising alternative to traditional com- pression algorithms. This study contributes to the development of neural network-based compression techniques and their practical applications in fields such as remote sensing, satellite imaging, and video streaming. Our methodology involves training MLPs on a diverse range of image datasets, varying network architectures, training strategies, and hyperparameters. We meticulously assess the trade-offs between compression ratios and image fidelity. By doing so, we aim to establish MLP-based image compression as a viable alternative to traditional methods.

1. INTRODUCTION

In various domains, such as climate research, agriculture, ur- ban planning, and natural resource management, satellite imagery has become an essential tool. Advanced sensors have enabled the collection of vast amounts of data through remote sensing systems, posing significant challenges in terms of data storage, transmission, and processing. Efficiently compressing satellite images is vital to address these challenges and facilitate the analysis of extensive datasets.
One approach gaining traction for lossless image compression is the utilization of Multilayer Perceptrons (MLPs)[1]. MLPs have garnered attention for their ability to compress images without any loss of information, particularly in satellite image compression. Nevertheless, achieving optimal compression ratios for multi-spectral satellite images with highly correlated bands remains a challenge.

Lossless image compression techniques have become increasingly popular in recent years due to their ability to compress images without losing any information. One such method that has recently been widely utilized in the lossless compression of satellite photos is the Multilayer Perceptrons(MLPs)[2]. We propose a novel method for lossless compression of multispectral satellite images, focusing on the strength of MLPs. This approach seeks to maintain image quality while outperforming existing strategies in terms of compression ratios and image fidelity. Evaluation using a dataset of multi-spectral satellite photos demonstrates its superior performance compared to current compression algorithms.
The suggested method tries to retain the quality of the compressed pictures while achieving greater compression performance than existing strategies. The suggested method is evaluated on a dataset[3] of multi-spectral satellite photos, and the findings demonstrate that it performs better in terms of compression ratio and image quality than state-of-the-art compression algorithms.
Compression ratios can exhibit a significant increase compared to those achievable through lossless methods. Various techniques have been employed for data compression, each with its own set of advantages and drawbacks. Many image compression methods rooted in neural networks have been introduced[4], classifiable as either lossless or lossy image compression techniques.

2. NEED OF THE STUDY.

In a variety of disciplines, including climate research, agriculture, urban planning, and natural resource management, among others, satellite imagery has emerged as a crucial instrument. High-resolution sensors have enabled the collection of massive amounts of data produced by remote sensing systems, leading to significant challenges in data storage, transmission, and processing. Efficient compression of satellite images is, therefore, crucial to overcome these challenges and facilitate
the analysis of large datasets.

3. RESEARCH METHODOLOGY

The Perceptron Learning algorithm[7] is founded on the pre- viously explained back-propagation rule. This algorithm is implementable in any programming language, and for this tutorial, we utilize Java for applets. In this context, we assume the utilization of the previously described sigmoid function f(net) due to its straightforward derivative.

4. ACTIVATION FUNCTION OF MLP ALGORITHM

If we consider a multilayer perceptron (MLP) [8] with a linear activation function in all its neurons, this implies a binary activation mechanism that determines whether a neuron fires or not. In linear algebra, it can be demonstrated that any number of layers in such a network can be reduced to the standard two-layer input-output model.

What distinguishes a multilayer perceptron [9]is the utilization of nonlinear activation functions, which are designed to model the firing frequency of biological neurons in the brain, known as action potentials. These functions take various forms but must always exhibit properties of normalizability and differentiability.

In contemporary applications, the two primary activation functions employed are both sigmoid functions. One is de- scribed by the hyperbolic tangent, which has a range from -1 to 1, while the other, similar in shape, ranges from 0 to 1. In these functions, "yi" represents the output of the ith neuron, and "vi" signifies the weighted sum of the input synapses.

Additionally, more specialized activation functions are avail- able, such as radial basis functions, which find use in another class of supervised neural network models. These functions expand the toolbox for modeling and capturing complex relationships in data.

Most common activation functions are the logistic and hyperbolic tangent sigmoid functions. The project uses the hyperbolic tangent function:

$$f(x) = \frac{2}{1 + e^{-\lambda x}} - 1$$

and its derivative:

$$f'(x) = f(x)(1 - f(x))$$

Weight Adjustment

The network's weights need to be adjusted in order to minimize the difference or the error between the output and the expected output. This is explained in the equations below.

The error signal at the output layer of the ith neuron at iteration n is given by:

$$e_i(n) = X_i(n) - X'(n)$$

Where Xi represents the desired output and X′ represents the actual output. The error function over all neurons in the output layer is given by:

$$E_l(n) = \sum e^2(n)$$

Algorithm 1 Compression Algorithm: MLP for Multispectral Images

1: x ← input training vector
2: t ← Output target vector
3: δk ← portion of error correction weight for wjk that is due to an error at output unit Yk; also the information about the
4: δj ← portion of error correction weight for vjk that is due to the backpropagation of error information from the output layer to the hidden unit Zj
5: α ← learning rate
6: vo j ← bias on hidden unit j
7: wok ← bias on output unit k
8: ***procedure*** *Algorithm:*
9: Initialize weights (set to small random values).
10: ***while*** *stopping condition is false do*
11: ***for*** *each training pair do*
12: Feed forward:
13: ***for*** *each input unit (Xi, i = 1, . . . , n)* ***do***
14: Receive input signal Xi and broadcast this signal to all units in the layer (the hidden units).
15: ***end for***
16: ***for*** *each hidden unit (Zj, j = 1, . . . , p)* ***do***
17: Sum its weighted input signals, apply its activation function to compute its output signal

Zj = f'(zin j)

18: ***end for***
19: ***for*** *each output unit (Yk, k = 1, . . . , m)* ***do***
20: *Sum its weighted input signals.*
21: ***end for***
22: ***for*** *each output unit (Yk, k = 1, . . . , m)* ***do***
23: *Receive target pattern corresponding to the input training pattern, compute its error . . information term:*
24: *δk = (tk − yk)f'(yink)*
25: *Calculate its weight correction term (used for later update).*
26: *Calculate its bias correction term (used for later update) and send to units in the layer below.*
27: ***end*** *for*
28: ***for*** *each hidden unit (Zj, j = 1, . . . , p)* ***do***
29: *Delta inputs, multiplied by the derivative of its activation function to compute its error . . information term:*
30: *δj = δin j f'(zin j)*
31: *Calculate its weight correction term (used for later update):*
32: *Δvi j = αδjxi*
33: *Calculate its bias correction term (used for later update).*
34: ***end for***
35: *Update weights and biases:*
36: ***for*** *each output unit (Yk, k = 1, . . . , m) do*
37: *Update its bias and weight (j = 0, . . . , p).*
38: ***end*** *for*
39: ***for*** *each hidden unit (Zj, j = 1, . . . , p)* ***do***
40: *Update its bias and weights (i = 0, . . . , n).*
41: ***end for***
42: *Test Stopping Condition.*
43: ***end for***
44: ***end while***

The error function over all input vectors in the training image is:

$$E = \sum E_l, \qquad E_l = (X', w)$$

where l indexes the image blocks (input vectors), X′ is the vector of outputs, and w is the vector of all weights. In order to minimize the error function with respect to the weight vector (w), it is necessary to find an optimal solution ($w*$) that satisfies the condition:

$$E(w^*) \leq E(w)$$

The necessary condition for optimality is:

$$\Delta E(w) = 0$$

where Δ is the gradient operator:

$$\Delta = \frac{\partial}{\partial w}$$

and ΔE(w) is the gradient vector (g) of the error function, defined as follows:

$$\Delta E(w) = \frac{\partial E}{\partial w}$$

The solution can be obtained using a class of unconstrained optimization methods based on the idea of local iterative descent. Starting with an initial guess denoted w(0), generate a sequence of weight vectors w(1),w(2), . . . such that the error function is reduced for each iteration:

$$E(w_n+1) \leq E(w_n)$$

Algorithm 2 Huffman Encoding Algorithm

```
function Huffman(input)
        itemqueue ← CreateNodeList(input)
        while Length(itemqueue) > 1 do
                l ← Heappop(itemqueue)
                r ← Heappop(itemqueue)
                n ← CreateIntervalNode(l, r)
                Heappush(itemqueue, n)
        end while
        codes ← {}
        CodeIt("", root of Huffman tree)
        encoded input ← EncodeInput(input, codes)
        return codes, encoded input
end function
```

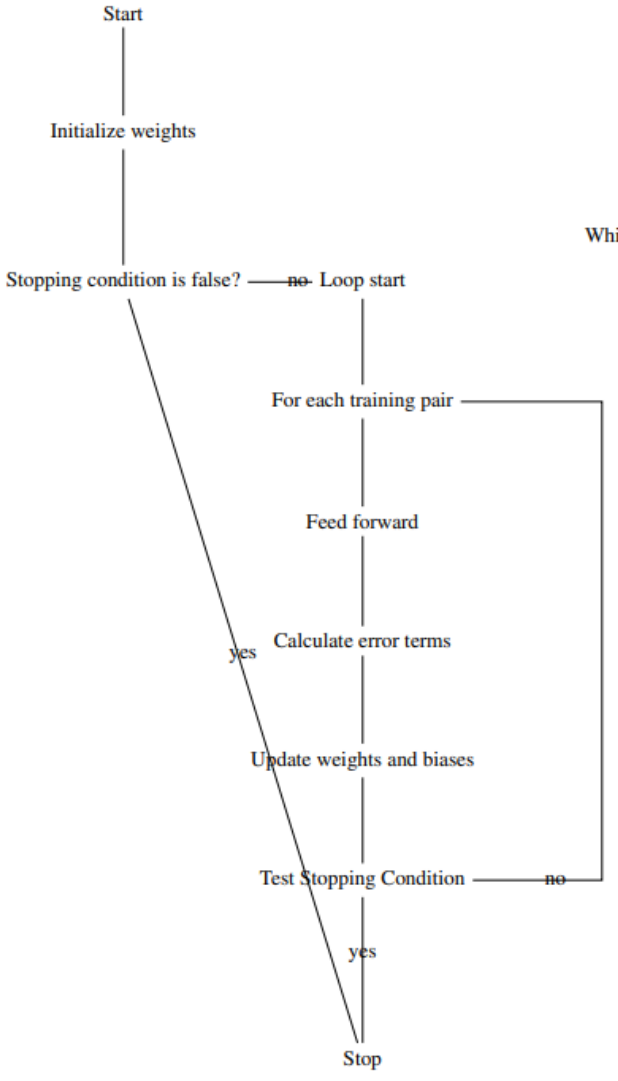


Fig. 1. Flowchart for the Compression Algorithm

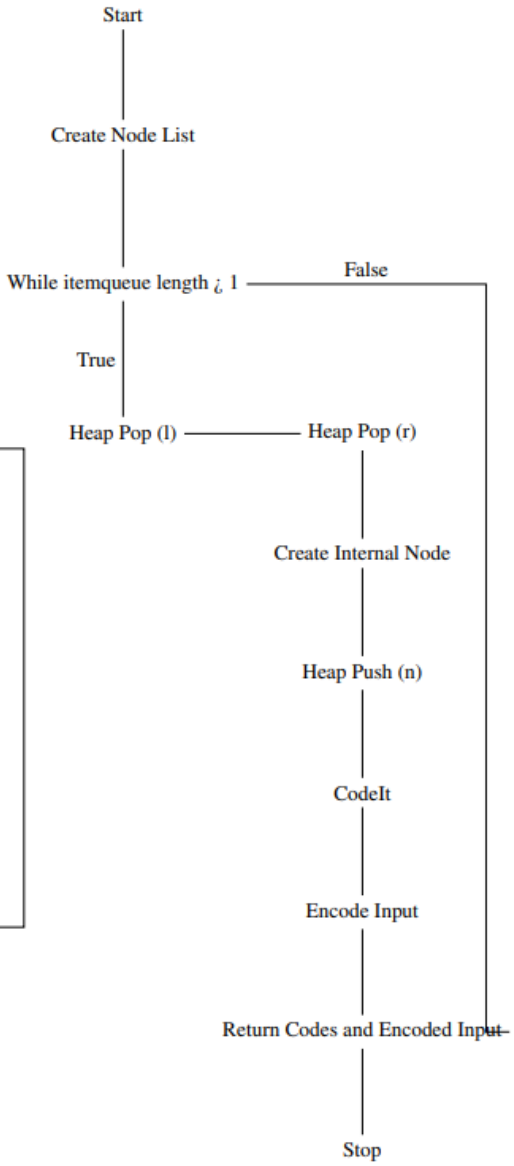
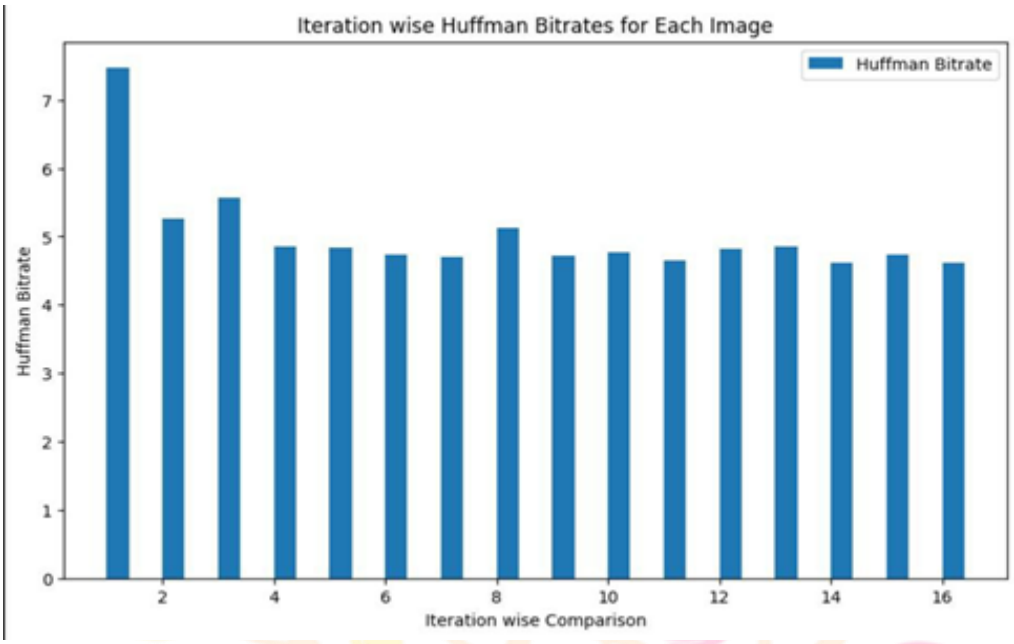
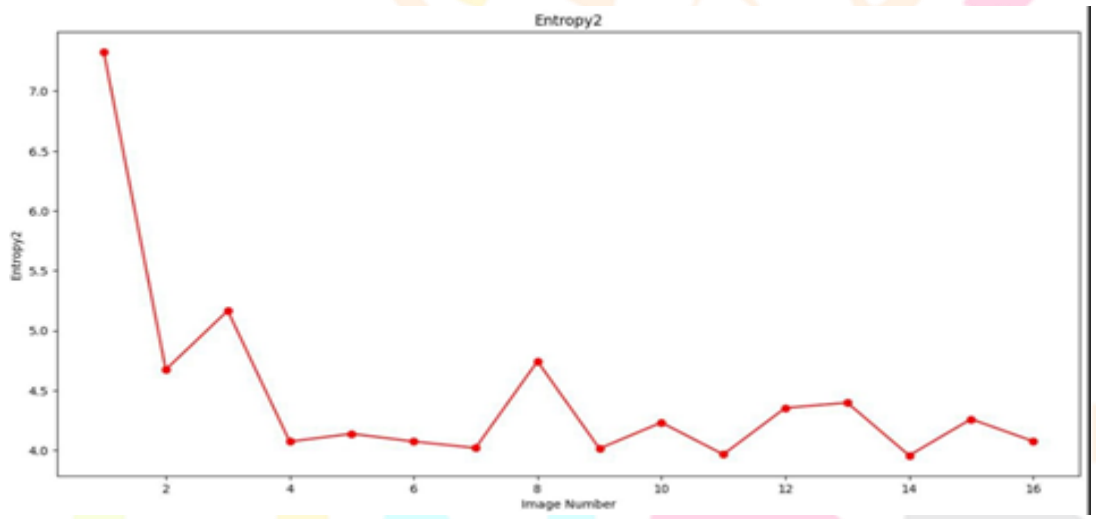


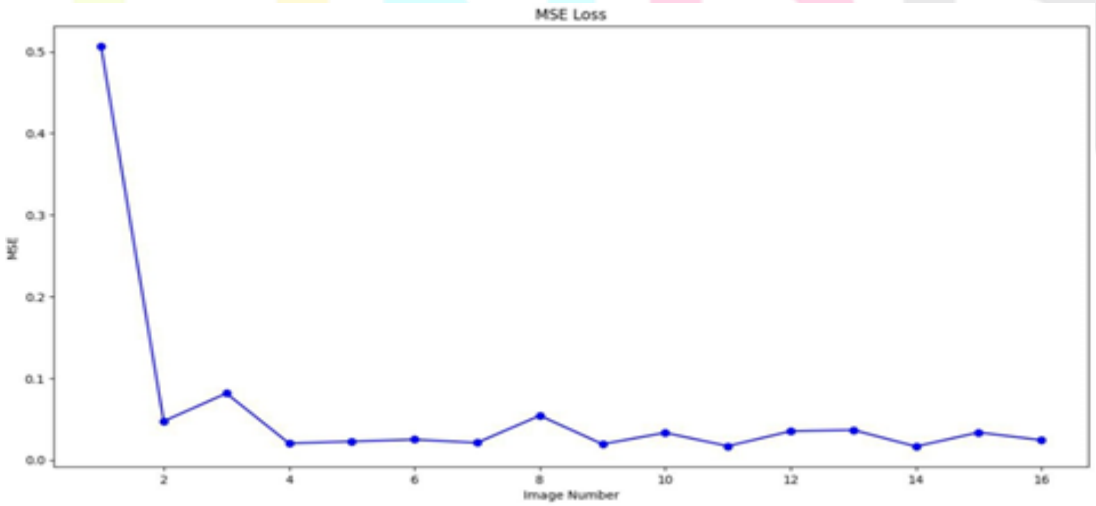
Fig. 2. Flowchart for the Activation Algorithm

5. Experimental Results & Comparison

The evaluation included key metrics such as Mean Squared Error (MSE) loss, entropy measures (Entropy 1 and Entropy 2), and bitrate. Through comprehensive experiments, the aim was to demonstrate the efficacy of employing multi-layer perceptron techniques in comparison to existing methods.







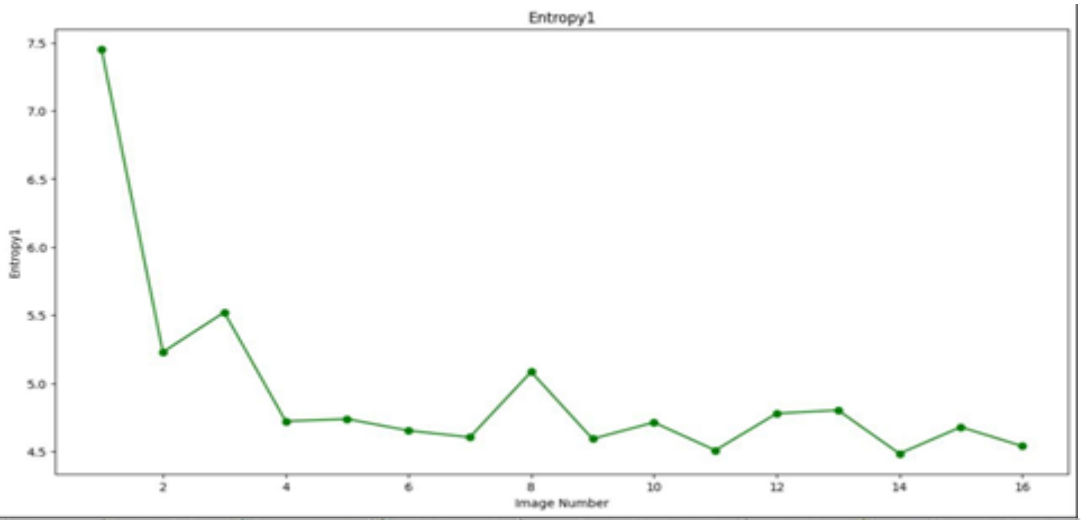


Image	MSE Loss	Entropy1	Entropy2	Original Bitrate	Rate	Huffman Bitrate
1	0.50677	7.45263	7.32569	7.29168	0.924444	7.476
2	0.04729	5.22911	4.67382	7.18633	0.836717	5.268
3	0.08145	5.52066	5.16547	7.07253	0.85241	5.565
4	0.02027	4.72135	4.07261	6.96458	0.8053	4.861
5	0.02259	4.73773	4.13769	7.44864	0.806402	4.841
6	0.0249	4.65331	4.07346	6.91452	0.800654	4.734
7	0.02099	4.60366	4.01881	7.61346	0.797194	4.703
8	0.05419	5.08388	4.74297	6.93156	0.828288	5.123
9	0.0193	4.5922	4.01412	8.17473	0.796387	4.711
10	0.0332	4.71297	4.23171	7.85783	0.804734	4.772
11	0.0169	4.50821	3.96526	7.64855	0.790373	4.65
12	0.03507	4.77868	4.35336	8.32575	0.80913	4.828
13	0.03658	4.80248	4.39705	7.31514	0.810698	4.848
14	0.01645	4.48403	3.9564	7.33509	0.788609	4.625
15	0.03358	4.67867	4.26034	7.28213	0.802399	4.735
16	0.02423	4.53929	4.07556	7.93588	0.792619	4.621

6. Conclusion

In this study, we applied Multilayer Perceptron (MLP) for image compression and compared the results with the original data across various metrics. The following key findings and conclusions can be drawn:

Mean Squared Error (MSE): The MSE value for the MLP- compressed images was lower compared to the original im- ages, indicating that the MLP compression method effectively reduced the reconstruction error.

Entropy1 and Entropy2: The entropy values, which mea- sure the information content of the images, were observed to be slightly different between the MLP-compressed images and the original images. These differences can be attributed to the compression process and its impact on the distribution of pixel values.

Huffman Bitrate: The Huffman bitrate for the MLP- compressed images was significantly lower than the original bitrate. This demonstrates the efficiency of the MLP-based com- pression technique in reducing the storage and transmission requirements for images.

In summary, the MLP[10] compression method was success- ful in reducing the MSE and Huffman bitrate, indicating its effectiveness in compressing images while maintaining visual quality. The differences in entropy values suggest that some in- formation may be lost during the compression process, but this is a common trade-off in lossy compression methods. Overall, the MLP approach offers a promising solution for image com- pression, particularly when reducing data storage and transmis- sion requirements is a priority. Further research and fine-tuning of the MLP model may yield even more efficient compression techniques.

References

[1] D. Batra, Comparison between levenberg-marquardt and scaled conjugate gradient training algorithms for image compression using mlp, Interna- tional Journal of Image Processing (IJIP) 8 (2014) 412–422.
[2] X. Wu, N. Memon, Context-based lossless interband compression- extending calic, IEEE Transactions on Image Processing 9 (2000) 994– 1001.
[3] Copernicus open access hub, https://scihub.copernicus.eu/ dhus/#/home/, Accessed September 30, 2022.
[4] Z. Li, C. Phillips, T. Chen, Efficient multi-band mlp based intra-prediction for lenslet image compression, in: 2023 IEEE International Conference on Consumer Electronics (ICCE), IEEE, 2023, pp. 1–4.
[5] C. Ren, X. He, T. Q. Nguyen, Single image super-resolution via adaptive high-dimensional non-local total variation and adaptive geometric feature, IEEE Transactions on Image Processing 26 (2016) 90–106.
[6] A. Adate, B. Tripathy, Deep learning techniques for image processing, Machine learning for big data analysis (2018) 69–90.
[7] O. A. Shawky, A. Hagag, E.-S. A. El-Dahshan, M. A. Ismail, Remote sensing image scene classification using cnn-mlp with data augmentation, Optik 221 (2020) 165356.
[8] S. Rezasoltani, F. Z. Qureshi, Hyperspectral image compression using implicit neural representation, arXiv preprint arXiv:2302.04129 (2023).
[9] G. Cazenavette, M. L. De Guevara, Mixergan: An mlp-based architecture for unpaired image-to-image translation, arXiv preprint arXiv:2105.14110 (2021).
[10] S. Ma, X. Zhang, C. Jia, Z. Zhao, S. Wang, S. Wang, Image and video compression with neural networks: A review, IEEE Transactions on Circuits and Systems for Video Technology 30 (2019) 1683–1698.