International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Strategies in JPEG Compression using Convolutional Neural Network (CNN)

Suman Kunwar [a]*, *Abayomi Simeon Alade* [b]

[a] Graduate School of Technology, Asia Pacific University of Technology and Innovation (APU), Kuala Lumpur, Malaysia
Email: *tp066707@mail.apu.edu.my*
[b] Physics and Electronics Department, Adekunle Ajasin University Akungba-Akoko (AAUA), Ondo State, Nigeria
Email: abayomy.alade@gmail.com

DOI: https://doi.org/10.55248/gengpi.4.723.48514

A B S T R A C T

Interests in digital image processing have been growing enormously in recent decades. As a result, different data compression techniques have been proposed which are concerned mostly with minimization of information used for the representation of images. With the advances in deep neural networks, image compression can be achieved to a higher degree. This paper begins with an overview of JPEG Compression and then discusses Discrete Fourier Transform (DFT) and Convolutional Neural Network (CNN) regarding image compression followed quality metrics to measure image compression performance. Later, it discusses the advancement of deep learning for image compression mostly focused on JPEG and suggests ways to improve the compression.

Keywords: JPEG Compression, Discrete Fourier Transform (DFT), Convolutional Neural Network (CNN), performance indicators

1. Introduction

JPEG is one of the most used compression techniques that has been widely accepted as a standard for lossy image compression. There are two types of image compression techniques: lossy and lossless compression (Wang et al., 2021). Lossy image compression techniques are non-reversible and can achieve a higher compression ratio whereas lossless methods provide the best visual experience. Lossy compression comes with a trade-off between file size and decomposed image quality. In practice, lossy compression schemes are often preferred on consumer devices because of their much higher compression ratio.

Convolutional Neural Networks are similar to Neural Networks. It consists of neurons that have learnable weights and biases. Each neuron receives some input, performs a dot product, and optionally follows it with a non-linearity (Kunwar,

2018). Deep Convolutional Neural Networks (ConvNets) are one of the essential tools for computer vision, used in image classification (Ren et al., 2016), object recognition (Long et al., 2015), and semantic segmentation (Cavigelli et al., 2015). It has also gained relevance for regression tasks in low-level image and video processing by computing saliency maps (Dosovitskiy et al., 2015), optical flow fields (Dong et al., 2014), and single-frame super-resolution images (Poor, 1988) with state-of-the-art performance.

This paper begins with the discussion of JPEG compression, Discrete Fourier Transform (DFT) and Convolutional Neural Networks (CNN). Then introduces the concept of the JPEG compression technique and Convolutional Neural Networks (CNN) and discusses some quality metrics used in image compression. Later talk about the advancement that has happened in JPEG compression using CNN. Finally, a conclusion section ends the chapter.

1.1 JPEG Compression

JPEG has different standards to compress an image such as JPEG, JPEG-LS, JPEG-2000. Two basic image compression algorithms (Wallace, 1991) have been developed by the Joint Photographic Experts Group (JPEG), one of which defines a combination of prediction method and entropy coding, and the other defines a hybrid compression method based on discrete cosine transform (DCT) (Rao & Ochoa-Dominguez, 2019) and entropy coding. The former method is a lossless compression technique based on the Differential Pulse Code Modulation (DPCM) and later is a lossy compression technique and is used mostly cause of the high compression ratio. Image coding algorithm focuses on reducing the correlation between pixels, quantization, and entropy coding (Kunwar, 2017). Constitution of image coding algorithm is shown in Figure 1.

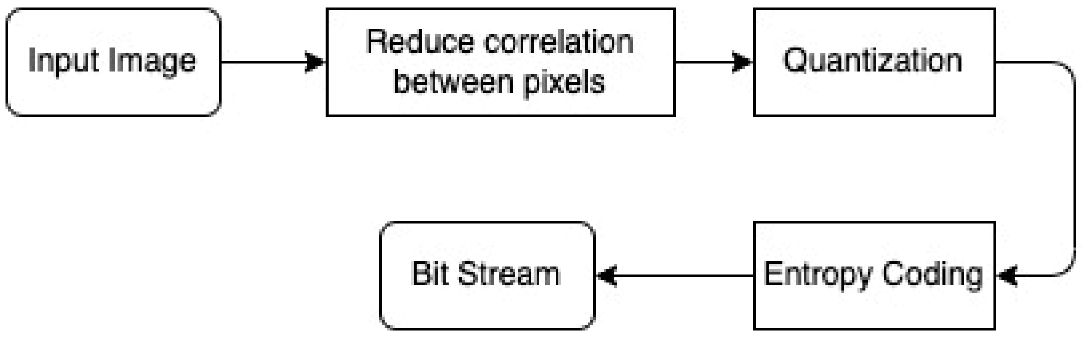

Fig. 1 - Constitution of image coding algorithm

Source Encoder

The source encoder aims to decorrelate the input signal by transforming its representation where the set of data values is sparse, thereby condensing the information content of the signal into the smaller number of coefficients.

Quantizer

A quantizer aims to reduce the number of bits needed to store the transformed coefficients by reducing the precision of these values. Quantization is performed on a per coefficient basis, i.e., Scalar Quantization (SQ) or Vector Quantization (VQ).

Entropy Coding

Entropy coding aims to eliminate redundancy by removing repeated bit patterns. The most common entropy coders are Huffman coding, arithmetic coding, run-length coding (RLE), and the Lempel-Ziv (LZ) algorithm.The basic process of JPEG compression along with entropy encoder is defined in Figure 2. The coding process can be summarized into four parts as shown in red font in Figure 2 (a), namely uniform partitioning, transform encoding (FDCT), quantizer, and entropy encoder (Figure 2 (b))

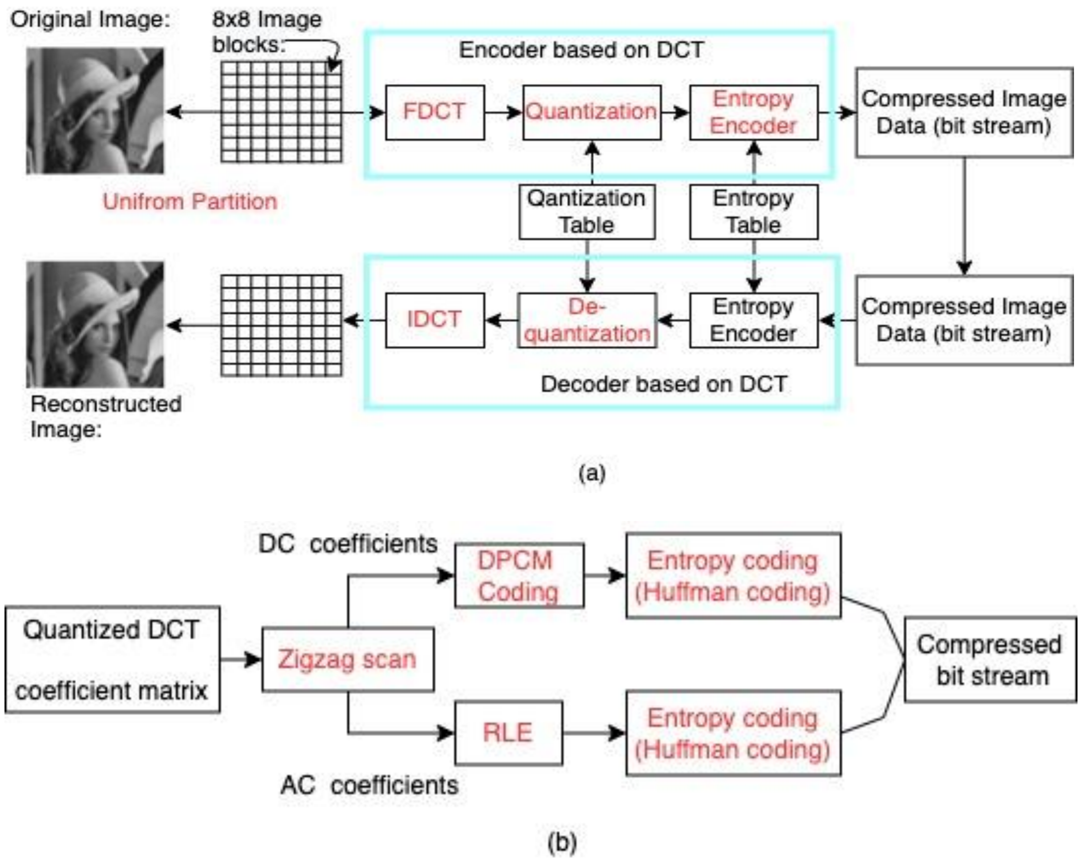

Fig. 2 - Block diagrams of JPEG and Entropy (Zhang et al., 2021)

The entropy coder consists of the Zig-Zag scan, Differential Pulse Code Modulation (DPCM) coding, run-length encoding (RLE) followed by Huffman coding. Zig-Zag scan groups lower frequency coefficients in top of vector and high frequency coefficients at the bottom, and maps 8 x 8 matrix to a 1 x

64 vector. Differential Pulse Code Modulation coding encodes the difference between the current and previous 8x8 block. Here, smaller numbers represent the fewer bits. The resultant vector obtained from DPCM contains lots of zeros as a result of a quantization table or due to higher entries in the vector capture higher frequency (DCT) where components tend to capture less of the content. RLE encodes series of zeros as skip value pair, where skip denotes number of zeros and value is the next non-zero component. Huffman coding follows the rule of assigning a shorter code to the most frequently used words and a longer code to the less frequently used words. Decoding is performed in the reverse order of encoding (i.e., entropy decoder, de-quantizer, and IDCT).

1.2. Discrete Fourier Transform (DFT)

Signal and Image processing is one of the emerging technology that touches most of the fields of engineering. Digital Signal is used in applications like data compression, filtering, image processing power spectral estimation, adaptive filtering, multirate system design and many more. The applications of the discrete Fourier transform are numerous once the fast Fourier transform (FFT) algorithms become available (Ifeachor & Jervis, 2002). Using the symmetry property, data compression is effectively performed by storing only half of the DFT response. To reconstruct the signal, a complete response is required. This can be achieved by considering the complex conjugates of the previous DFT values according to the property and can be checked with Matlab using some non-real-time data (Lim & Oppenheim, 1988).

DTFT response is continuous, making it difficult to perform calculations with a digital computer. Because of the infinite spectrum, this is even more difficult during multiplication. To make the computation easier, Discrete Fourier Transform (DFT) is introduced. In DFT, the sampling is done in the frequency domain, i.e., discrete, the corresponding data sequence must be periodic in the time domain (Mitra, 2006). Given N real numbers.

$$f_0, \dots, f_{N-1} \qquad (1)$$

we have to compute

$$c_k = \frac{1}{N}\sum_{j=0}^{N-1} f_j\, e^{-ijk2\pi/N}, \quad k = 0, \dots, N-1. \qquad (2)$$

The mapping of $f_j,\ j = 0, \dots, N-1,$ into $c_k = \frac{1}{N}\sum_{j=0}^{N-1} f_j\, e^{-ijk2\pi/N},\ k = 0, \dots, N-1,$ is called the *Discrete Fourier Transform* (DFT). *inverse discrete*

Fourier transform maps c into f and is given by

$$f_k = \sum_{j=0}^{N-1} c_j\, e^{ijk2\pi/N}, \quad k = 0, \dots, N-1.$$

In a straightforward implementation, the computations of matrix-vector products require N^2 multiplications. JPEG compression in JPEG through is shown in Figure 3.

In DFT transformation, DFT transforms the image symbols into certain coefficients, making it possible to compress by setting a threshold. These thresholds are applied later to the whole image or a certain part of the image part, usually in the form of squares. Here, irrelevant information or less important information is discarded. The selection of the size of the block where it will be applied is also relevant to the quality of the image (Pratt, 2006). The original and compressed image using DFT is shown in Figure 4.

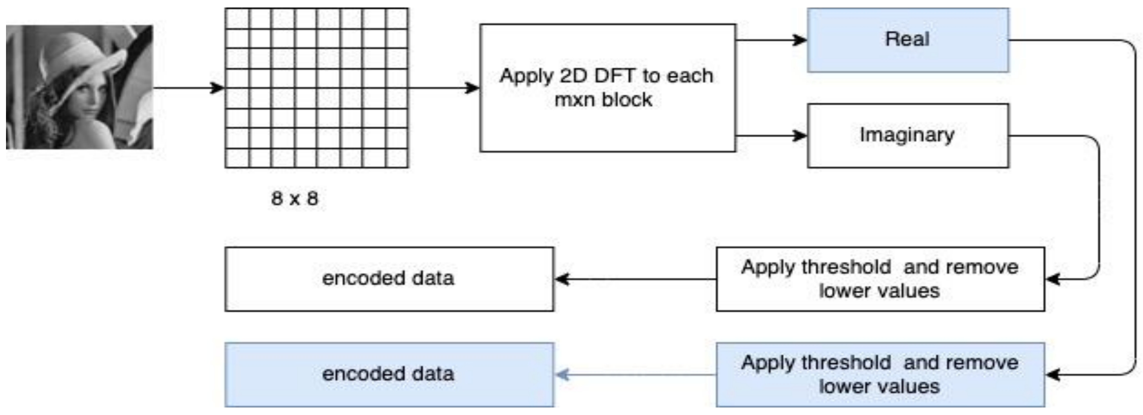
8 x 8
Apply 2D DFT to each mxn block
Real
Imaginary
Apply threshold and remove lower values
encoded data
Apply threshold and remove lower values
encoded data

Fig. 3 - JPEG image compression through DFT

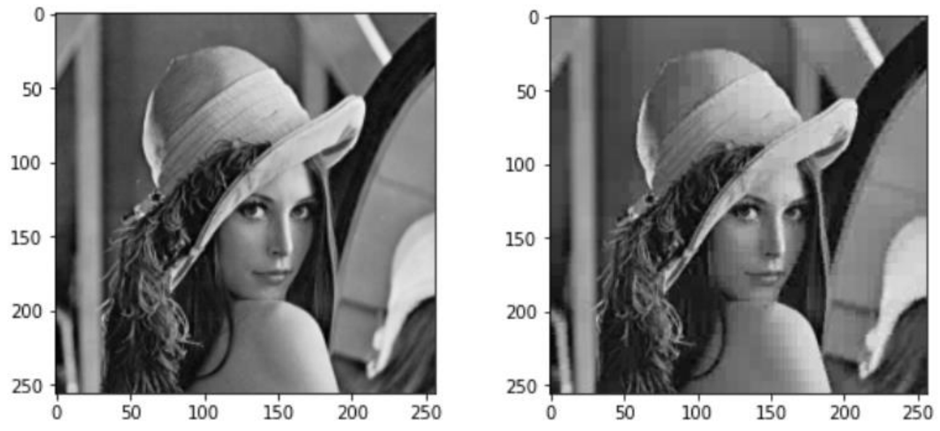

Fig. 4 – (a) Original image ; (b) Compressed image

1.3 Convolutional Neural Networks (CNN)

Usage of Convolutional Neural Networks (CNNs) for classification and computer vision tasks are gradually increasing and have achieved great success in image recognition (Krizhevsky et al., 2017). It provides a more scalable approach to object detection and image classification tasks. Supervised Learning and Unsupervised Learning are the two main learning paradigms in image processing tasks (O'Shea & Nash, 2015). In supervised learning pre-labeled inputs act as targets.

Supervised learning is best suited for those problems which have a reference point to train the algorithm. The expected result is known upfront. Unsupervised learning, in contrast, does not contain labels in the training set. Algorithms are left on their own to discover and display the interesting structures in the data. CNN consists of three layers: convolutional layers, pooling layers, and fully connected layers as shown in Figure 5.

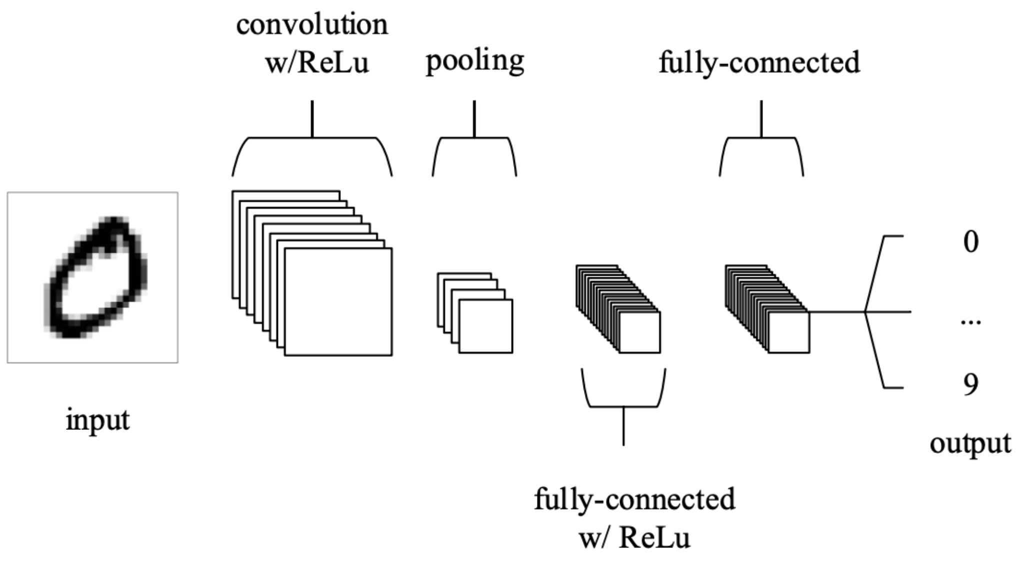

Fig. 5 - CNN Architecture, comprised of just five layers (O'Shea & Nash, 2015)

Here, input layers store the pixel values, convolution layers determine the output of the neurons associated with local regions done by computing the scalar product between their weights and the region associated with the input volume. Rectified Linear unit (ReLu) adds the activation function to the output generated by the previous layer.

For an z input value. The ReLu function is represented as:

$relu(\boldsymbol{z}) = max(0, \boldsymbol{z})$ (3)

As per equation 3, the output of ReLu is maximum between zero and input value. It can be re-written as

$\nabla(relu(\boldsymbol{z})) = \begin{cases} \boldsymbol{z}, & if\, \boldsymbol{z} \geq 0, \\ 0, & if\, \boldsymbol{z} < 0 \end{cases}$ (4)

The pooling layer down samples the spatial dimensionality of the given input and the fully connected layers generate class values from the activations, which are used for classification.

1.4 Convolutional Layer

Convolutional layer focuses on the use of learnable kernels/filters that perform convolution operations as it is scanning the input I with respect to its dimensions giving resulting output O is called activation map or feature map. A visual representation of a convolutional layer is shown in Figure 6.

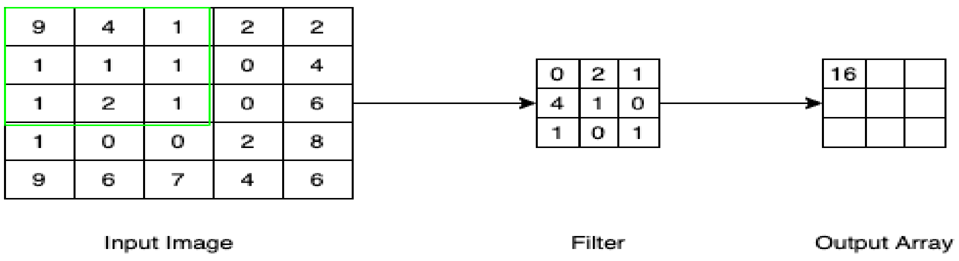

Fig. 6 - Visual representation of a convolutional layer.

As seen in Figure 6, filter map value is connected to the receptive field, where filter is being applied. The hyperparameters that affect the resulting output size are Stride, Number of filters, and Zero-padding.

Stride

It is the distance, moved by the kernel across the input matrix. While stride values of two or more are rare, a larger stride results in a smaller output.

Number of filters

The number of filters affects the depth of the output as each individual filter yields different feature maps of its own depth.

Zero-padding

It is mostly used when the filters do not fit the input image. In this case, all elements that fall outside the input matrix are set to zero, resulting in an output that is larger or the same size. Valid padding, Full padding and Same padding are three different types of padding that can be used.

Pooling layer

Pooling layer aims to gradually reduce the spatial size of the representation to reduce the number of parameters and computations in the network. The pooling layer works with each feature map independently. It maps the input and scales its dimensionality with the function "MAX". The most common approach to pooling is max pooling.

Fully connected layer

A fully connected layer contains neurons that are directly connected to neurons in the two adjacent layers, without being connected to any of the layers in them. This is consistent with the way neurons are arranged in traditional forms of ANN shown in Figure 7.

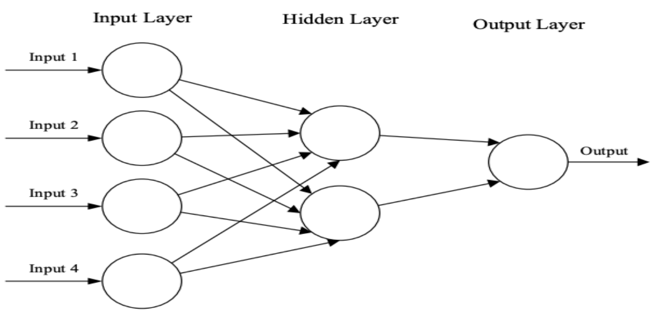

Fig. 1 - Simple three-layer feedforward neural network (FNN) consisting of an input layer, a hidden layer, and an output layer (O’Shea & Nash, 2015).

1.5 Performance Indicators

Different quality metrics such as Mean Square Error (MSE), Peak Signal to Noise Ratio (PSNR), Structural Similarity Index (SSMI) are used to measure the quality of compressed results (Deshmukh, 2019). Metrics like MSE and PSNR are easy to calculate and applicable in most cases, but they sometimes do not correspond to perceived quality and are not normalized in presentation. The structured similarity indexing method (SSIM) and Feature similarity indexing method (FSIM) come into place to solve this problem (Sara et al., 2019).

Mean Square Error (MSE)

Mean Square Error (MSE) is one of the most common image error metrics used to compare image compression quality. It represents the cumulative squared error between the compressed and the original image. The lower the MSE, the lower the error. The Mean Square Error can be calculated with the following expression:

$$MSE = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\left(x_{ij} - y_{ij}\right)^2 \qquad (5)$$

Peak Signal to Noise Ratio (PSNR)

PSNR stands for peak signal to noise ratio. This ratio is often used as a quality measure between the original and a compressed image. In the quality degradation of image and video compression, the PSNR value varies between 30 and 50 dB for the representation of 8-bit data and between 60 and 80 dB for 16-bit data. For wireless transmission, the accepted range of quality loss is about 20 - 25 dB (Sara et al., 2019). The higher the PSNR value, the better the quality of the compressed or reconstructed image. The PSNR can be calculated with the following expression:.

$$PSNR(x, y) = \frac{10log_{10}\left[max\left(max(x), max(y)\right)\right]^2}{(x-y)^2} \qquad (6)$$

Structural Similarity Index (SSMI)

Structural Similarity Index is a perception-based model that measures the structural similarity between images. Here, image degradation is considered as a change in the perception of structural information. The SMI can be calculated with the following expression:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \qquad (7)$$

2. Literature Survey

In the last decades, deep learning has been recognized as one of the most effective methods for managing large amounts of data and is widely used in literature. Interest in more hidden layers beyond classical algorithms has recently developed in image coding and compression. In this section, previous work on the use of Deep Learning for image compression is summarized:

Experiments with two image databases, the MNIST database, and the CIFAR-10 database, show that the complexity of the learning task of NN models can be reduced when working with partially decompressed images with a small loss of accuracy of the model, while the loss of accuracy of the CNN model depends on the JPEG data. For a quality factor of 80, the gain in computational complexity with decompression is 45% with an accuracy loss of 13% and illustrates the need for models adapted to compressed data (Pistono et al., 2020).

Transmitting much less data (60%) of an image at the sender side in transform-based compression standards such as JPEG and use of a deep residual learning model on the receiver's side to recover the original data is possible with a maximum signal-to-noise ratio of over 31 dB, as in cloud servers (Qiu et al., 2021).

An algorithm for image compression is developed based on a back-propagation (BP) network after preprocessing the image. The proposed scheme is implemented, transfer functions influences and compression ratios within the scheme are investigated and found out that signal-to-noise ratio (PSNR) remains the same for all compression ratios, while the mean square error (MSE) varies (Rao et al., 2010).

The relationship between pixels is highly nonlinear and unpredictable without prior knowledge of the image itself. An algorithm was created using Artificial neural networks, that consider the psycho-visual features When the algorithm is applied to the image data, most of the features of the data are preserved while it is lossy and maximizes the compression performance (Dutta et al., 2009).

Comparison of six different image compression algorithms with a subjective quality test using high-resolution images shows that the learned compression optimized by MS-SSIM gives competitive results close to the efficiency of state-of-the-art compression (Cheng et al., 2019).

With the face representation model, face images are decomposed into shape and texture components, and different compression methods are applied to these components depending on the degree of redundancy. A convolutional neural network (CNN) is proposed to process the texture components. The texture components are optimized, and it was found that the proposed method achieved better performance in terms of PSNR and SSIM values at low bit rates in comparison to the traditional JPEG and JPEG2000 compression methods (Hu et al., 2020).

The study found that when a certain generative adversarial network (GAN) is used, the compression performance of color images is better than that of gray images. To improve the compression effect of gray images, a post-processing method is proposed in this paper (Ruihua et al., 2019).

Two recent deep-learning-based image compression algorithms are subjectively evaluated and compared with JPEG 2000 and the new BPG image codec based on HEVC Intra. The research found that compression approaches based on deep auto-encoders can achieve higher coding performance than JPEG 2000 and sometimes even as good as BPG and suggests shows that deep generative models are likely to bring tremendous innovations in video coding in the coming years (Valenzise et al., 2018).

Multilayer perceptron for image compression is presented using neural networks and used for transformation coding. Experiments are done on different images segmented into blocks of varied sizes for the compression process. The reconstructed image is compared with the original image based on the signal-to-noise ratio and the number of bits per pixel. The results show that it is possible to use multilayer perceptron's for image compression (Vilovic, 2006).

The use of deep neural network models on embedded or mobile devices is problematic for storage cause of modal size and large memory usage. A compression method for deep neural networks to reduce resource consumption for efficient visual inference is developed. Four modules were introduced here: approximation module to reduce the number of weights in each linked layer, quantization module for analysis and representation of distributed weights in each layer, pruning module to suppresses small weights resulting in a reduced number of parameters, and the end coding module that represents and optimize the sparse structure of the pruned weights with relative index by Huffman coding. The proposed framework, together with the model compression and memory allocation algorithms found effective in representative models, AlexNet and VGG-16, for object recognition and face verification tasks (Ge et al., 2017).

An effective machine learning-based method to distinguish between double and single compressed JPEG images was introduced. Difference of JPEG 2D arrays is used to improve the compression artifacts followed by applied Markov random process to the modeling of the 2D difference arrays to utilize second-order statistics along with thresholding technique to reduce the size of the transition probability matrices. Elements of these matrices are collected as features for double JPEG compression detection and a support vector machine is used as the classifier. Experiments have shown that the proposed system outperforms the previous methods (Chen et al., 2008).

A reduction in the time required to train machine learning models can translate into an accuracy improvement. For larger models, offloading of temporary data from limited GPU memory to CPU (Central Processing Unit) memory results in performance degradation. A new model JPEG for ACTivations (JPEGACT), was introduced that discards redundant spatial information. It adapts the well-known JPEG algorithm from 2D image compression to activation compression. This method achieves 2.4× higher training performance offload accelerators, 1.6× compression methods and consumes less than 1 percent power and area of modern CPU (Evans et al., 2020).

An efficient lossy image compression method based on asymmetric autoencoders, and decoder pruning was introduced through deep learning and the results show the effectiveness of the methods (Kim et al., 2020).

To minimize the feature distortions caused by the JPEG artifacts a new training method named Feature consistency Training was introduced. Here, in each iteration, the raw image and its compressed version of randomly sampled quality were introduced to the training model. With the addition of feature consistency constraints to the objective function, feature distortion is minimized in the representation space to learn robust filters (Wan et al., 2020).

Images are compressed either to save storage space or for fast transmission requiring a time-consuming decompression step in deep modals. To address this, an architecture was purposed for object detection based on the block-wise discrete cosine transform (DCT) coefficients output from the JPEG compression algorithm in a neural network. Single shot multibox detector (SSD) is replaced by its first layers with a convolutional layer dedicated to the processing of the DCT inputs. Evaluations on PASCAL VOC and an industrial dataset of road traffic surveillance images show that the model is about 2× faster than the regular SSD and has promising detection performance (Deguerre et al., 2019).

3. Discussion

The research reveals that over time, there is an advancement of deep learning in image compression. JPEG being mostly being used on the internet provides flexibility in compression with a trade-off between image quality and size. With the adaptability of neural network content, valuable information is extracted and used as a model. The use of the texture and features in neural networks improves image compression and, in some cases, post-processing is also needed. The need for model adaptation for a better compression using CNN is also pointed out here. Most of the research shows promising results and pointed to the need for further research.

4. Conclusion

JPEG is a widely used image format on the internet and provides a high compression ratio. The use of deep learning provides flexibility for experiments and also eliminates the necessity of highly complex modeling methods. It has provided promising results in the areas like image compression, image classification, image segmentation, and many more. In this review, the working mechanism of JPEG, DFT in relation to JPEG and CNN is discussed with performance indicators, different deep learning-based image compression techniques were pointed out for new studies based on JPEG compression techniques and deep learning.

References

Cavigelli, L., Magno, M., & Benini, L. (2015). Accelerating real-time embedded scene labeling with convolutional networks. Proceedings of the 52nd Annual Design Automation Conference, 1–6. https://doi.org/10.1145/2744769.2744788

Chen, C., Shi, Y. Q., & Su, W. (2008). A machine learning based scheme for double jpeg compression detection. 2008 19th International Conference on Pattern Recognition, 1–4. https://doi.org/10.1109/ICPR.2008. 4761645

Cheng, Z., Akyazi, P., Sun, H., Katto, J., & Ebrahimi, T. (2019). Perceptual quality study on deep learning based image compression. 2019 IEEE International Conference on Image Processing (ICIP), 719–723. https://doi.org/10.1109/ICIP.2019.8803824

Deguerre, B., Chatelain, C., & Gasso, G. (2019). Fast object detection in compressed jpeg images. 2019 IEEE In- telligent Transportation Systems Conference (ITSC), 333–338. https://doi.org/10.1109/ITSC.2019.8916937

Deshmukh, K. R. (2019). Image compression using neural net- works (Master of Science). San Jose State University. San Jose, CA, USA. https://doi.org/10.31979/etd.h8mt-65ct

Dong, C., Loy, C. C., He, K., & Tang, X. (2014). Learn- ing a deep convolutional network for image super- resolution. In D. Fleet, T. Pajdla, B. Schiele, & T. Tuytelaars (Eds.), Computer vision – eccv 2014 (pp. 184–199). Springer International Publishing.

Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Smagt, P. v. d., Cremers, D., & Brox, T. (2015). Flownet: Learning optical flow with convolu- tional networks. 2015 IEEE International Conference on Computer Vision (ICCV), 2758–2766. https://doi. org/10.1109/ICCV.2015.316

Dutta, D. P., Choudhury, S. D., Hussain, M. A., & Ma- jumder, S. (2009). Digital image compression using neural networks. 2009 International Conference on Advances in Computing, Control, and Telecommuni- cation Technologies, 116–120. https://doi.org/10.1109/ACT.2009.38

Evans, R. D., Liu, L., & Aamodt, T. M. (2020). Jpeg-act: Accelerating deep learning via transform-based lossy compression. 2020 ACM/IEEE 47th Annual Interna- tional Symposium on Computer Architecture (ISCA), 860–873. https://doi.org/10.1109/ISCA45697.2020.00075

Ge, S., Luo, Z., Zhao, S., Jin, X., & Zhang, X.-Y. (2017). Compressing deep neural networks for efficient visual inference. 2017 IEEE International Conference on Multimedia and Expo (ICME), 667–672. https://doi. org/10.1109/ICME.2017.8019465

Hu, S., Duan, Y., Tao, X., Liu, Y., Zhang, X., & Lu, J. (2020). Content-aware facial image compression with deep learning method. 2020 International Conference on Wireless Communications and Signal Processing (WCSP), 516–521. https://doi.org/10. 1109/ WCSP49889.2020.9299680

Ifeachor, E. C., & Jervis, B. W. (2002). Digital signal process- ing: A practical approach (2nd ed). Prentice Hall.

Kim, J.-H., Choi, J.-H., Chang, J., & Lee, J.-S. (2020). Effi- cient deep learning-based lossy image compression via asymmetric autoencoder and pruning. ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2063–2067. https://doi.org/10.1109/ICASSP40776. 2020.9053102

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). Im- ageNet classification with deep convolutional neural networks. Communications of the ACM, 60(6), 84–90. https://doi.org/10.1145/3065386

Kunwar, S. (2017). Image Compression Algorithm and JPEG Standard. International Journal of Scientific and Research Publications, 7(12), 150–157. http://www. ijsrp.org/research-paper-1217/ijsrp-p7224.pdf

Kunwar, S. (2018). Jpeg image compression using cnn, 1. https://doi.org/10.13140/RG.2.2.25600.53762

Lim, J., S, & Oppenheim, A. V. (1988). Advanced topics in signal processing [OCLC: 857116358]. Prentice-Hall-International.

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 3431–3440. https://doi.org/10. 1109/CVPR.2015.7298965

Mitra, S. K. (2006). Digital signal processing: A computer based approach (3rd ed) [OCLC: ocm56011040]. McGraw-Hill Higher Education.

O'Shea, K., & Nash, R. (2015). An introduction to convolutional neural networks.

Pistono, M., Coatrieux, G., Nunes, J.-C., & Cozic, M. (2020). Training machine learning on jpeg compressed im- ages. 2020 Data Compression Conference (DCC), 388–388. https://doi.org/10.1109/DCC47342.2020.00070

Poor, H. V. (1988). An introduction to signal detection and estimation (J. B. Thomas, Ed.). Springer New York. https://doi.org/10.1007/978-1-4757-3863-6

Pratt, W. (2006). Digital image processing. John Wiley & Sons, Ltd.

Qiu, H., Zheng, Q., Memmi, G., Lu, J., Qiu, M., & Thu- raisingham, B. (2021). Deep residual learning-based enhanced jpeg compression in the internet of things. IEEE Transactions on Industrial Informatics, 17(3), 2124–2133. https://doi.org/10.1109/TII.2020.2994743

Rao, K. R., & Ochoa-Dominguez, H. (2019). Discrete cosine transform (Second edition). Taylor & Francis Group, CRC Press.

Rao, P. V., Madhusudana, S., Nachiketh, S., & Keerthi, K. (2010). Image compression using artificial neural networks. 2010 Second International Conference on Machine Learning and Computing, 121–124. https: //doi.org/10.1109/ICMLC.2010.33

Ren, S., He, K., Girshick, R., & Sun, J. (2016). Faster r-cnn: Towards real-time object detection with region proposal networks [arXiv: 1506.01497]. arXiv:1506.01497 [cs]. Retrieved September 25, 2021, from http://arxiv.org/abs/1506.01497

Ruihua, L., Quan, Z., & Huachao, X. (2019). An image com- pression processing method based on deep learning. 2019 IEEE 2nd International Conference on Information Communication and Signal Processing (ICICSP), 342–346. https://doi.org/10.1109/ICICSP48821.2019.8958605

Sara, U., Akter, M., & Uddin, M. S. (2019). Image quality assessment through fsim, ssim, mse and psnr—a comparative study. Journal of Computer and Com- munications, 07(03), 8–18. https://doi.org/10.4236/ jcc.2019.73002

Valenzise, G., Purica, A., Hulusic, V., & Cagnazzo, M. (2018). Quality assessment of deep-learning-based image compression. 2018 IEEE 20th International Workshop on Multimedia Signal Processing (MMSP), 1–6. https://doi.org/10.1109/MMSP.2018.8547064

Vilovic, I. (2006). An experience in image compression using neural networks. Proceedings ELMAR 2006, 95–98. https://doi.org/10.1109/ELMAR.2006.329523

Wallace, G. K. (1991). The JPEG still picture compression standard. Communications of the ACM, 34(4), 30–44. https://doi.org/10.1145/103085.103089

Wan, S., Wu, T.-Y., Hsu, H.-W., Wong, W. H., & Lee, C.-Y. (2020). Feature consistency training with jpeg com- pressed images. IEEE Transactions on Circuits and Systems for Video Technology, 30(12), 4769–4780. https://doi.org/10.1109/TCSVT.2019.2959815

Wang, Z., Bovik, A., & Lu, L. (2021). Why is image quality assessment so difficult? [online] Live.ece.utexas.edu. Available at:]. https://live.ece.utexas.edu/publications/ 2002/zw icassp2002 whyqa.pdf

Zhang, Y., Cai, Z., & Xiong, G. (2021). A new image com- pression algorithm based on non-uniform partition and u-system. IEEE Transactions on Multimedia, 23, 1069–1082. https://doi.org/10.1109/TMM.2020.2992 940