Improved Image Compression using Autoencoder and Discrete Cosine Transformation

Sarthak Jindal
BML Munjal University
sarthak.jindal.21cse@bmu.edu.in
Gurgaon, India

Indradhar Paka
BML Munjal University
indradhar.paka.18mec@bmu.edu.in
Gurgaon, India

Esha Agarwal
BML Munjal University
esha.agarwal.20cse@bmu.edu.in
Gurgaon, India

Dr. Atul Mishra
BML Munjal University
atul.mishra@bmu.edu.in
Gurgaon, India

***Abstract*—This study proposes a method for improved image compression using a combination of discrete cosine transformation (DCT) and autoencoder. Images typically contain large amounts of data that require significant storage space, making them difficult to store and transmit. Compressing images is a practical solution to this issue as it reduces memory usage and enables faster transmission to the receiver. In this approach, we use DCT as a preprocessing step before training an autoencoder model to compress the image while retaining all essential information. The proposed method involves a convolutional neural network (CNN) that performs down-sampling and up-sampling operations on the input data processed by DCT. The performance of the proposed method is evaluated and compared with traditional image compression techniques such as JPEG, JPEG 2000, and BPG. Experimental results demonstrate that the proposed approach outperforms the traditional techniques in terms of compression ratio and image quality.**

Keywords— Compression, Autoencoder, Neural network, UX, Convolutional

I. Introduction

Images are a significant part of modern data and are used in various domains, such as multimedia communication, medical imaging, and remote sensing. However, the increasing need for effective image data management has resulted in the development of advanced image compression techniques. While User Experience (UX) is a critical aspect of software products, its evaluation is often neglected during software development due to misconceptions of resource intensiveness and automation challenges.

The study illustrates a new methodology, which uses discrete cosine transformation (DCT) in conjunction with autoencoders to provide a solution to the challenge of image storage and transmission. The method compresses images using an autoencoder model that retains essential image information while reducing the image size for storage and transmission.

Tools for remote testing have been developed to enable participants to test systems from their locations, making it more convenient for them. These tools automatically gather and store data about the tests and provide useful features for the tools of the past. For instance, eGLU-Box PA allows the detection of issues concerning UX, called "UX smells," using visualization techniques that show the paths followed by participants to carry out tasks on websites during a test.

Emotions play a significant role in UX, and some researchers are investigating ways to determine users' emotions by looking at how they interact with the system using machine learning algorithms. Since images play a crucial role in communicating with others, it is essential to develop effective image compression methods that can be incorporated into existing applications.

To decode an image with the least amount of loss, it must first undergo image compression. There are two main image compression techniques: lossy and lossless. This paper presents image compression with little input data loss, primarily used for medical applications. There are several methods for lossy image compression, including wavelet compression and discrete cosine transform. Thus, research on image compression techniques is a popular topic for various researchers in image processing.

The proposed autoencoder-based image compression technique uses a novel layered architecture based on CNN. The approach uses a convolution neural network to design an autoencoder, expected to achieve better compression performance than current image compression standards, including JPEG. It can be used to reduce the dimensionality and compress images with the least amount of loss.

The paper's remaining part is organized as follows: Section 2 explains related work, and Section 3 describes the proposed system. Section 4 describes the results, verification, and validation. Finally, section 5 concludes the work.

II. Ease of Use

This paper describes the use of autoencoders, which use convolutional neural networks, to perform image compression. We will contrast this method with JPEG image compression, which uses discrete cosine transformation, to determine how the effectiveness of neural network image compression. Table I presents a comparison of existing literature work on autoencoder (AE) models used in image compression. However, few AE-based compression methods support error-bounding features, which are critical in scientific applications.

Table I. Comparison of Existing work on Image Compression

Limitations	Proposed Solution
Limited Automation in Remote Testing Tools:	The autoencoder-based approach may provide a more automated solution by efficiently compressing images, reducing the need for manual inspection of large datasets. This can streamline the remote

	testing process and minimize the time and effort required for analysis.
Detection of "UX Smells" in User Experience	y incorporating autoencoders into the testing process, the neural network can potentially identify and quantify these "UX smells" objectively through image compression techniques. It can capture subtle visual cues in user interactions, providing a data-driven approach to detecting UX issues.
Emotion Detection in User Experience	The autoencoder-based method can contribute to emotion detection by efficiently compressing images and videos, making it easier to process large volumes of visual data.
Image Compression Efficiency	The proposed autoencoder neural network can be tailored to optimize image compression specifically for the unique requirements of user experience testing.
Integration into Existing Applications	The autoencoder-based technique can be designed with integration in mind, offering a solution that is adaptable and compatible with a wide range of applications.

A. Discrete Cosine Transformation in JPEG compression

The template is used to format your paper and style the text. All margins, column widths, line spaces, and text fonts are prescribed; please do not alter them. You may note peculiarities. For example, the head margin in this template measures proportionately more than is customary. This measurement and others are deliberate, using specifications that anticipate your paper as one part of the entire proceedings, and not as an independent document. Please do not revise any of the current designations.

In JPEG compression, the entire data is first transformed into YCbCr (Intensity, colour blue, colour red) from RGB [9]. Down samples, the data and then applied DCT and, after that, finally quantization and entropy encoding. A brief explanation of JPEG compression:

Transform data from RGB to Y, Cb, and Cr and then sample the data into 8X8 blocks ---------F(I,f)

F(I,f) matrices have values 0 to 255; To compress the data, the values must be lowered.

Subtracting 128 from all the pixel values must obtain a matrix 8X8 of range -128 to 127--g(x,y).

Apply DCT on g(x,y) by the formula:

The $\alpha(u)\alpha(y)\sum_{x=0}^{7}\sum_{x=0}^{7} g(x,y)\cos\left[\frac{\pi}{8}\left(x+\frac{1}{2}\right)u\right]\cos\left[\frac{\pi}{8}\left(y+\frac{1}{2}\right)v\right]$---------A(j,k)

$\alpha_p(n) = \sqrt{\frac{1}{8}}$ if n=0; otherwise $\sqrt{\frac{2}{8}}$

Let a Luninance matrix (a standard matrix used in JPEG format) Q(j,k). Quantization of the data by formula:

$$\text{round}\left(\frac{A(j,k)}{Q(j,k)}\right) \text{-------B(j,k)}$$

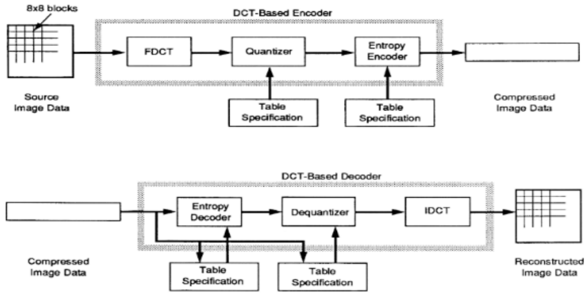


Fig. 1. Block diagram of JPEG compression [10]

To reduce redundancy, serialization is finally performed in a zigzag fashion into a 1-d array as output. And finally, using the inverse discrete transformation depicted in Fig. 1. The next section discusses the methodology in detail.

III. Methodology

This section provides a detailed methodology for the proposed image compression method. The methodology covers three main aspects: Dataset, Architecture, and Performance metrics.

The autoencoder framework is designed with three layers: Encoder, bottleneck, and decoder. The Encoder compresses the data into lower dimensions, and the output is then sent to the network, where it can be decoded using the Decoder.

To train and validate the autoencoder model, the Kodak dataset is used [11][14]. The dataset is divided into training and testing sets, and the model is tested using the testing set. GPU is preferred for faster computation.

For image compression, all images are first converted to YCbCr. Then, each image is resized to match the dataset's size, and the entire dataset is normalized within the range of 0-1. The architecture is trained to produce output with less loss..

A. Dataset

Kodak's dataset consists of lossless full-colour RGB images which were released by the Eastman Kodak Company for unrestricted usage. These images are commonly used for testing the reconstruction or compression of images. However, they can only be previewed upon downloading them over FTP. The release was lossless, and the PNG format has been integrated into all major browsers. PNG supports 24-bit lossless colour, which is not supported by GIF and JPEG. This makes it possible to offer browser-friendly access to these images.

B. Model Architecture

The architecture of the autoencoder model was created using a convolutional layer design. Before normalization, the data was first converted into YCbCr. The model creation involved the use of convolution layers and max pooling. Activation of neurons was achieved through rectified linear unit (ReLU) to overcome the vanishing gradient problem [13]. This model can also be referred to as a generative model because it can generate images by using image data in mathematical terminology, also known as conditional probability. The normalized data is sent to the input layer, which further downsamples the data, upsamples it, and decodes it using the decoder, as shown in Fig. 2.

For the model training, mean square error was used as the cost function. The study used several evaluation metrics, including MAE, RMSE, MAPE (Mean Absolute Percentage Error), and RMSLE (Root Mean Square Log Error). MSE was chosen because it has a squared order power equation, as it is a squared error. The other evaluation metrics have higher-order power equations. The reason behind selecting the squared order power function is that the optimizer (gradient descent) performs better with it.

The optimizer works to find the global minima and reduce the cost function, resulting in more accurate results. The distortion between the original image and the reconstructed image is calculated by PSNR [12].

The architecture begins by processing a single image into a 512x768 model of 3 channels (YCbCr). The input data is then processed by the convolution layer, which increases the number of channels from 3 to 32. Next, max pooling is applied, reducing the shape of the image produced by the convolution layer to 256x384 of 32 channels. After 6 layers, the shape of the data produced by the convolution layer becomes a compressed image of 128x192x128. By applying up-sampling layers and convolution layers, the image size generated returns to 512x768 of 3 channels. The reconstructed image is trained by the autoencoder model using updated weights and backpropagation. Pooling which reduces the shape of the image produced by the convolution layer into 256x384 of 32 channels.

After performing 6 layers, the shape of the data produced by the convolution layer is 128x192x128 compressed image. Now, by applying up sampling layers and convolution layers, the image size generated is 512x768 of channel 3. The reconstructed image is trained by the autoencoder model by updating those weights and backpropagation.

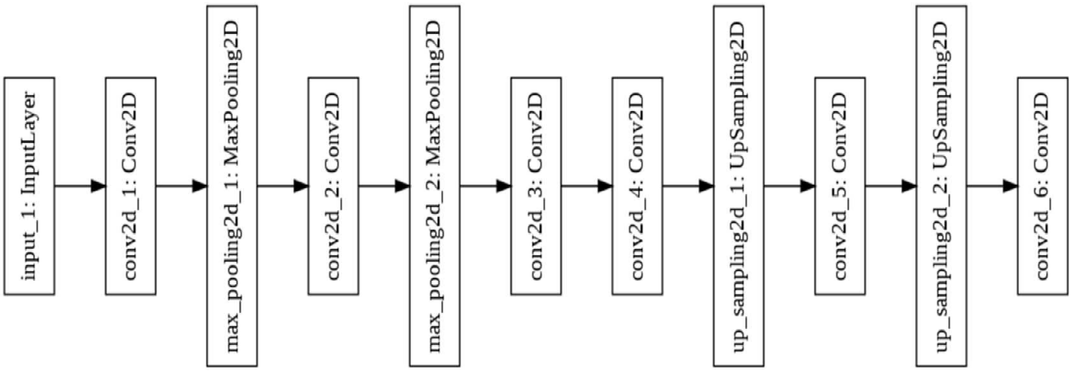


Fig. 2. Architecture of proposed autoencoder mode

C. *Performance Metrics*

1) Peak Signal to Noise ratio: The standard of a reconstructed image is measured using PSNR. The original data is the signal, and the compression error is the noise. When comparing compressed images, PSNR provides a reconstruction quality that is similar to that of a human [12]. Image quality is improved by a higher PSNR. The PSNR typically ranges from 60 to 80 dB for 16-bit data. For wireless propagation, values between 20 and 25 dB are acceptable [20, 21]. The mean squared error formula can be used to calculate PSNR.

$$PSNR = 20\cdot\log10(MAX) - 10\cdot\log10(MSE)$$

Where,

MAX is maximum value of the pixel in an image

MSE is Mean of Squared Error.

2) Mean Squared Error - The MSE can be calculated in a useful manner by adding the estimator's variance and squared bias, which suggests that for unbiased estimators, the

MSE and variance are equivalent [21]. The following is the definition of an estimator's MSE with respect to an unknown parameter:

$$MSE(\hat{\theta}) = E\_\theta\ [(\hat{\theta}-\theta)^2\ ]$$

The MSE can be written as the sum of the variance of the estimator and the squared bias of the estimator, providing a useful way to calculate the MSE and implying that in the case of unbiased estimators, the MSE and variance are equivalents.

$$MSE(\hat{\theta}) = Var\_\theta\ (\hat{\theta}) + Bias(\hat{\theta},\theta)^2$$

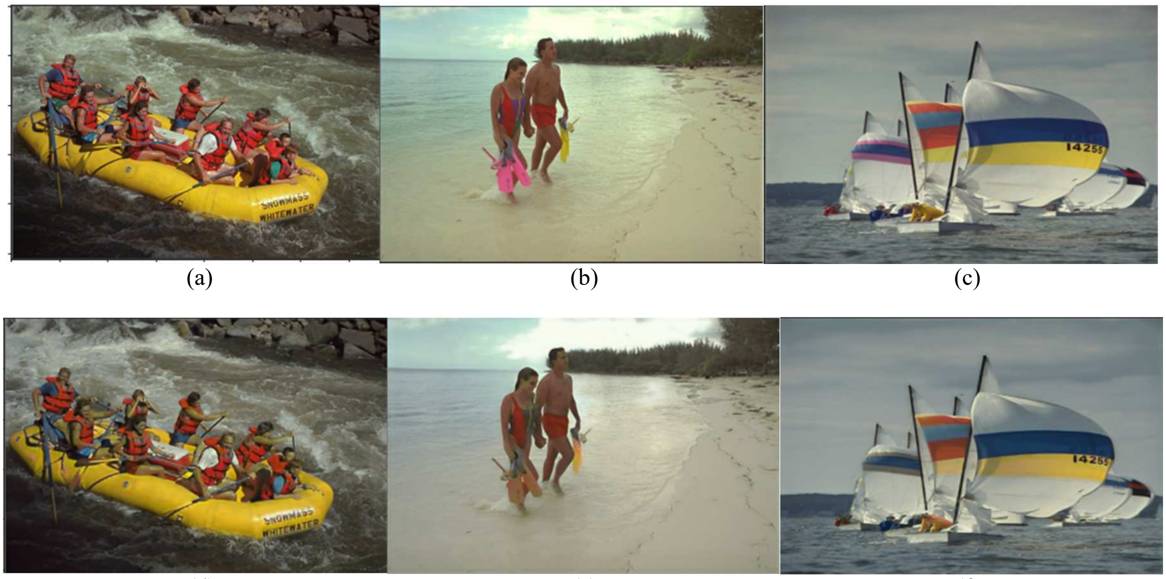
IV. RESULTS

To determine how well the proposed model fits the system, the results are assessed twice. The study determines whether the model correctly applies the assumptions (model verification). Then, the study determined whether the made assumptions are reasonable in light of the actual system (model validation).

A. *Verification*

The Kodak dataset was used for both training and validation of the entire model [11]. It performed better than the JPEG image compression technique, JPEG2000, and BPG after being put into practice, producing a reconstructed compressed image with less loss. Since all of the layers were activated using rectified linear units, mean squared error was used as the loss function during model training, and the vanishing gradient problem was avoided.

Fig. 3. demonstrates the original (Fig. 3 (a), (b), and (c)) and related compressed images. The difference between the final images and the original images is shown in Fig. 3 (d), (e), and (f).



(a) (b) (c)

(d) (e) (f)

Fig. 3. (a)(b)(c) are original images, (d)(e)(f) are compressed images of the original images

B. Validation

In the validation, the results are compared with JPEG compression using the same dataset [9][11]. PSNR values are calculated. The results show that the proposed model outperformed JPEG by getting higher values for PSNR (in dB) compared to JPEG, as show in Fig. 4. Even though we have currently validated our model on kodak dataset, but it can be used with any image dataset generated out of a UX evaluation task.

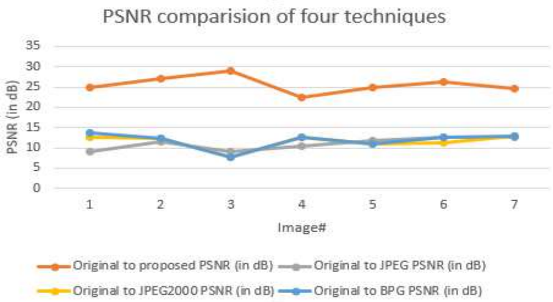


Fig. 4. PSNR curves for all images over four techniques

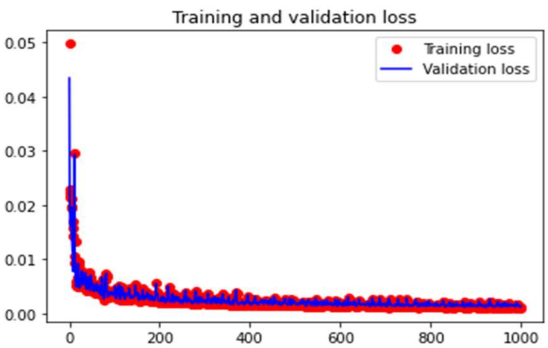


Fig. 5. Plot for the loss function values over each iteration for the Training and Validation set

The study also concludes that the cost function's loss for the training and validation sets is almost zero, as shown in Fig. 5. This indicates that the model was trained correctly and can reconstruct the testing data with little error.

Based on the PSNR calculation presented in Table II and Fig. 6, it can be concluded that the proposed technique outperforms JPEG, JPEG 2000, and BPG in terms of image quality. As mentioned in section 3, "Higher PSNR indicates better image quality", and the PSNR value obtained using the proposed method is higher for all tested images compared to other compression techniques.

Table II. Calculated PSNR for proposed, JPEG, JPEG2000, BPG compressed techniques

Image	Proposed	JPEG	JPEG2000	BPG
	PSNR (in dB)	PSNR (in dB)	PSNR (in dB)	PSNR (in dB)
1	24.97723	9.003919	12.65672	13.656495
2	27.05327	11.538549	12.35486	12.408018
3	28.8848	9.139331	7.664106	7.667928
4	22.46803	10.525949	12.62793	12.643165
5	24.89617	11.688374	11.03956	11.067859
6	26.27813	12.680623	11.39313	12.656573
7	24.62789	12.724757	12.89527	12.916949

(a) Proposed Compression Technique

(b) JPEG Compression Technique

(c) JPEG2000 Compression Technique

(d) BPG Compression Technique

Fig. 6. Sample images compressed using (a) the proposed method, (b) JPEG, (c) JPEG 2000, and (d) BPG compression techniques

V. CONCLUSION

The study concluded that media images are complex and can be challenging to process on a large scale. This is because data is distributed over long-range networks, resulting in slower transmission times, especially for large amounts of information. However, the study proposes a new method of using autoencoder training to compress images. This method was compared to the well-known JPEG and BPG algorithms and showed higher PSNR values, demonstrating that it outperforms standard compression methods. The researchers also suggest that Huffman coding could be added to the process to improve the level of compression further.

REFERENCES

[1] Jackson, P. J., & Hwang, A. S. (2021). Design Tools: Transforming Ideas into Products. In Knowledge, Innovation, and Impact (pp. 211-219). Springer, Cham.

[2] "User Research Userlytics." Userlytics Corporation, www.userlytics.com. Accessed 5 Nov. 2022.

[3] Ekşioğlu, M., Kiris, E., Çapar, B., Selçuk, M. N., & Ouzeir, S. (2011, July). Heuristic evaluation and usability testing: case study. In International Conference on Internationalization, Design and Global Development (pp. 143-151). Springer, Berlin, Heidelberg.

[4] Federici, S., Mele, M. L., Lanzilotti, R., Desolda, G., Bracalenti, M., Buttafuoco, A., ... & Simonetti, E. (2019, July). Heuristic evaluation of eGLU-Box: a semi-automatic usability evaluation tool for public administrations. In International Conference on Human-Computer Interaction (pp. 75-86). Springer, Cham.

[5] Buono, P., Caivano, D., Costabile, M. F., Desolda, G., & Lanzilotti, R. (2019). Towards the detection of UX smells: the support of visualizations. IEEE Access, 8, 6901-6914.

[6] Desolda, Giuseppe, et al. "Detecting Emotions Through Machine Learning for Automatic UX Evaluation." IFIP Conference on Human-Computer Interaction. Springer, Cham, 2021.

[7] Rabbani M. JPEG2000: Image compression fundamentals, standards and practice. Journal of Electronic Imaging. 2002 Apr;11(2):286.

[8] Wu L, Huang K, Shen H. A GAN-based Tunable Image Compression System. InThe IEEE Winter Conference on Applications of Computer Vision 2020 (pp. 2334-2342).

[9] G. K. Wallace, "The JPEG still picture compression standard," in IEEE Transactions on Consumer Electronics, vol. 38, no. 1, pp. xviii-xxxiv, Feb. 1992.

[10] N. Ahmed, T. Natarajan and K. R. Rao, "Discrete Cosine Transform," in IEEE Transactions on Computers, vol. C-23, no. 1, pp. 90-93, Jan. 1974.

[11] Li, X., Gunturk, B., & Zhang, L. (2008, January). Image demosaicing: A systematic survey. In Visual Communications and Image Processing 2008 (Vol. 6822, pp. 489-503). SPIE.

[12] Huynh-Thu Q, Ghanbari M. Scope of validity of PSNR in image/video quality assessment. Electronics letters. 2008 Jun 19;44(13):800-1.

[13] Nair V, Hinton GE. Rectified linear units improve restricted boltzmann machines. InICML 2010 Jan 1.

[14] Li, X., Meng, L., Tan, Y., Zhang, J., Wan, W., & Zhang, H. (2022). Multiple description coding network based on semantic segmentation. Multimedia Tools and Applications, 1-17.

[15] Cheng, Z., Sun, H., Takeuchi, M., & Katto, J. (2018, June). Deep convolutional autoencoder-based lossy image compression. In 2018 Picture Coding Symposium (PCS) (pp. 253-257). IEEE

[16] Dumas, T., Roumy, A., & Guillemot, C. (2017, March). Image compression with stochastic winner-take-all auto-encoder. In 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 1512-1516). IEEE.

[17] Gaidhane, V. H., Singh, V., Hote, Y. V., & Kumar, M. (2011). New approaches for image compression using neural network. Journal of Intelligent Learning Systems and Applications, 3(04), 220.

[18] Kenyon, G. (2022). Image Compression: Sparse Coding vs. Bottleneck Autoencoders (No. LA-UR-17-30743). Los Alamos National Lab.(LANL), Los Alamos, NM (United States).

[19] Duan, Z., Lu, M., Ma, Z., & Zhu, F. (2022). Lossy Image Compression with Quantized Hierarchical VAEs. arXiv preprint arXiv:2208.13056.

[20] Nortje, A., Brink, W., Engelbrecht, H. A., & Kamper, H. (2021). BINet: a binary inpainting network for deep patch-based image compression. Signal Processing: Image Communication, 92, 116119.

[21] Wackerly, Dennis; Mendenhall, William; Scheaffer, Richard L. (2008). Mathematical Statistics with Applications (7 ed.). Belmont, CA, USA: Thomson Higher Education. ISBN 978-0-495-38508-0.