WWW.IJRPR.COM

International Journal of Research Publication and Reviews

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

Image Compression and Decompression Using Neural Networks

Neha Fatima [a], Muskan Bhate [a], Rohan Salgudi [a], Fahad Neshat [a], Shahla Sohail [a].

[a] Department of Electronics and Communication Engineering, Dayananda Sagar College of Engineering Bangalore, Karnataka, India.

ABSTRACT:

This paper focuses on developing a neural network-based image compression system specifically for biomedical applications. High-resolution medical images such as MRI, CT scans, and X-rays require large amounts of storage, which complicates their transmission. This project uses convolutional neural networks (CNN) and autoencoders to compress these images while maintaining critical diagnostic details. The goal is to create a model that can significantly reduce storage requirements and transmission times without losing vital information, enhancing the efficiency of medical data handling. Image compression using neural networks is an advanced technique aimed at reducing the storage and transmission requirements of digital images while maintaining visual quality. This method leverages deep learning architectures, such as convolutional neural networks (CNNs) and autoencoders, to learn efficient representations of image data by extracting meaningful features and removing redundancies. Neural networks optimize compression by balancing compression rates and reconstruction accuracy, often outperforming traditional algorithms like JPEG and PNG in terms of quality and compression efficiency. The approach provides scalable and adaptive compression solutions, making it particularly useful for modern applications requiring high-quality image transmission, such as multimedia streaming, medical imaging, and autonomous systems.

Keywords: Image Compression, Neural Networks, Autoencoders, Storage, Quality.

Introduction:

Image compression is a crucial aspect of modern digital communication, storage, and media transmission. With the growing use of high-resolution images in various fields like entertainment, medical imaging, and autonomous vehicles, the demand for efficient image compression techniques has increased significantly. Compression reduces the amount of data required to represent an image, making storage and transmission faster and more cost-effective. Traditional methods like JPEG, PNG, and GIF have been widely used to compress images, however, they often suffer from a trade-off between compression ratio and image quality, resulting in visible artifacts or quality degradation, especially at higher compression levels.

In recent years, deep learning techniques, specifically neural networks, have emerged as a powerful tool for solving complex problems in various domains, including image compression. Neural networks, especially convolutional neural networks (CNNs) and autoencoders, have shown great promise in learning highly efficient representations of images. Unlike traditional methods that rely on fixed transformations, neural networks can adaptively compress images based on their content, preserving essential features and discarding redundant information. This ability to learn from the data makes neural networks more flexible and efficient for image compression tasks, allowing for higher-quality reconstructions with fewer artifacts and better compression ratios.

What is Image Compression and Decompression?

Image Compression: Image compression is the process of reducing the size of a digital image file while preserving its visual quality to the greatest extent possible. The primary aim is to optimize storage and transmission by removing redundancies or unnecessary data. This process is vital in applications such as medical imaging, multimedia streaming, and web browsing, where storage and bandwidth are often limited. Compressed images occupy less space, making them easier to store and faster to transmit, which improves system efficiency and accessibility.

There are two main types of image compression: lossless and lossy. Lossless compression retains all the original image data, enabling exact reconstruction of the original image from the compressed file. Techniques such as Run-Length Encoding (RLE) and Huffman Encoding are commonly used for lossless compression. In contrast, lossy compression sacrifices some data to achieve greater compression rates, resulting in an image that closely resembles the original but isn't identical. Transform coding methods, like those used in JPEG, are examples of lossy compression and are widely used for applications where slight quality loss is acceptable.

Image Decompression: Image decompression is the reverse process of compression, where a compressed image file is converted back into a usable format. The goal of decompression is to reconstruct the image as accurately as possible, either perfectly (in the case of lossless compression) or approximately (in lossy compression). This is a crucial step for accessing, viewing, or analyzing compressed images.

The decompression process starts by decoding the compressed file, interpreting the encoded data to extract meaningful patterns. Finally, the image is reconstructed from the processed data.

Methodology:

The development of the image compression and decompression system begins with data collection and preparation. A diverse dataset of images is essential to ensure robust model performance across different scenarios. The Dataset used in this paper is from Kaggle. The images are pre-processed by resizing them to a uniform resolution (e.g., 128x128 or 256x256) to ensure compatibility with the neural network model. Additionally, pixel values are normalized to a standardized range (e.g., [0, 1] or [-1, 1]) to facilitate faster convergence during training. The dataset is then split into training, validation, and test sets to support model evaluation and fine-training.

The next step involves neural network model development. Autoencoders and Convolutional Neural Networks (CNNs) are both deep learning architectures but serve different purposes and are designed for distinct types of tasks. We have used both Autoencoders and CNN. An autoencoder-based architecture is designed, comprising an encoder and a decoder. The encoder compresses the input image into a compact latent representation by extracting essential features, while the decoder reconstructs the input image from the compressed image. CNN architecture is composed of multiple convolutional layers followed by pooling, fully connected layers, and sometimes normalization layers. CNN is a supervised learning model, designed specifically for tasks involving spatial hierarchies in data, such as images. Extracts spatial and hierarchical features (edges, textures, etc.) for tasks like classification, segmentation, and object detection.

The model training process employs a reconstruction loss function, such as Mean Squared Error, to minimize the discrepancy between the input and reconstructed images. Optimizers like Adam employed with dynamic learning rate schedules to ensure stable training. Metrics such as Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Matrix (SSIM), and compression ratio are used to evaluate the model's performance.

After training, the model undergoes evaluation and testing. The system's performance is assessed on the test dataset using predefined metrics, and its results are analysed. Visual comparisons of reconstructed images help analyse and validate the quality of the compression.

To make the system user-accessible, a Flask-based web application is developed. This interface allows users to upload images, compress them using the trained model, and view the compressed image and also decompress them back to the uploaded image. The backend integrates the trained neural network model, while SQLite3 is used to store metadata, such as compression ratios and user information.

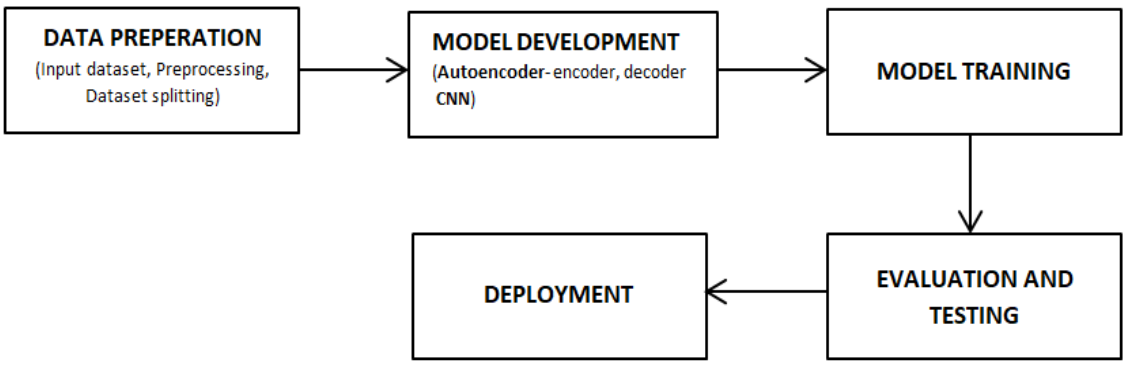


Figure 1 : Block Diagram.

Evaluation metrics

Compression Ratio: One of the key metrics in image compression is the compression ratio, which is the ratio of the size of the original image to the size of the compressed image. The compression ratio is given by:

$$\text{Compression Ratio} = \frac{\text{Original Image Size}}{\text{Compressed Image Size}}$$

Mean Squared Error (MSE):

The formula for MSE is:

$$\text{MSE} = \frac{1}{m \cdot n} \sum_{i=1}^{m} \sum_{j=1}^{n} (I(i,j) - K(i,j))^2$$

Peak Signal-to-Noise Ratio (PSNR):

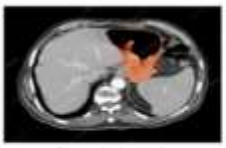
The formula for PSNR is:

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{\text{MAX}^2}{\text{MSE}} \right)$$

Structural Similarity Index (SSIM):

The formula for SSIM is:

$$\text{SSIM}(I, K) = \frac{(2\mu_I\mu_K + C_1)(2\sigma_{IK} + C_2)}{(\mu_I^2 + \mu_K^2 + C_1)(\sigma_I^2 + \sigma_K^2 + C_2)}$$

Filter Size – 2 X 2			
Parameters	Original Image	Compressed Image	Decompressed Image
Image	 Size - 176 kb	 Size - 48.9 kb	 Size - 160 kb
CR	3.6		
MSE	0.000858		
PSNR (dB)	78.8		
SSIM	0.929		
Filter Size – 3 X 3			
Image	 Size - 176 kb	 Size - 54.5 kb	 Size - 170 kb
CR	3.23		
MSE	0.15625		
PSNR (dB)	66.2		
SSIM	0.892		

Results

The results of the image compression and decompression project using Convolutional Neural Networks (CNNs) and autoencoders demonstrate significant improvements over traditional compression methods such as JPEG and PNG. The trained model achieved high compression ratios, reducing image sizes significantly while maintaining critical visual details. Evaluation metrics like Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) showed that the reconstructed images were of comparable or superior quality to those compressed with traditional methods.

The compression efficiency and image quality retention were further validated through visual assessments, where the decompressed images retained their essential features with minimal degradation. The system performed well under various test conditions, with real-time compression and decompression achieved in many scenarios. Additionally, the Flask-based web application provided a user-friendly interface for uploading and processing images, making the system accessible and efficient for practical use. These results indicate that CNN and autoencoder based model is a promising approach for applications requiring high-quality image compression and decompression, such as multimedia streaming, medical imaging, and autonomous systems.

Conclusion

This project explores the use of Convolutional Neural Networks (CNNs) and autoencoders for image compression and decompression, achieving high compression ratios with minimal loss in visual quality. Autoencoders efficiently compress and reconstruct images while preserving key features, making them ideal for applications where storage, bandwidth, and fast transmission are crucial. Compared to traditional methods like JPEG and PNG, CNN-based autoencoders offer better compression efficiency and superior image quality retention during both compression and decompression.

By leveraging deep learning, CNNs enable end-to-end learning, allowing the network to automatically determine the most efficient image representation. This reduces computational load and bandwidth usage without compromising image quality. CNN-based autoencoders are particularly beneficial for fields such as multimedia streaming, medical imaging, and autonomous systems, where high-quality image transmission and decompression are essential. The adaptability of this approach makes it a promising solution for industries relying on visual data, with potential for real-time compression and decompression in future technologies.

References:

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778, 2016.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," arXiv preprint arXiv:1810.04805, 2018.

[3] E. Battenberg, J. Chen, R. Child, A. Coates, Y. Gaur, Y. Li, H. Liu, S. Satheesh, D. Seetapun, A. Sriram, and Z. Zhu, "Exploring neural transducers for end-to-end speech recognition," arXiv preprint arXiv:1707.07413, 2017.

[4] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualizing image classification models and saliency maps," arXiv preprint arXiv:1312.6034, 2013.

[5] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in European conference on computer vision (ECCV), 2014.

[6] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," PloS one, vol. 10, no. 7, 2015.

[7] C. Gan, N. Wang, Y. Yang, D.-Y. Yeung, and A. G. Hauptmann, "Devnet: A deep event network for multimedia event detection and evidence recounting," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2568–2577, 2015.

[8] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should I trust you? : Explaining the predictions of any classifier," in Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp. 1135–1144, 2016.

[9] P. Dabkowski and Y. Gal, "Real time image saliency for black box classifiers," in NIPS, 2017.

[10] R. C. Fong and A. Vedaldi, "Interpretable explanations of black boxes by meaningful perturbation," in IEEE International Conference on Computer Vision (CVPR), pp. 3429–3437, 2017.

[11] J. Li, W. Monroe, and D. Jurafsky, "Understanding neural networks through representation erasure," arXiv preprint arXiv:1612.08220, 2016.

[12] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in NIPS, 2017.