info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

A Comparative Study of Compression Techniques for Medical Images

Hadeel Talib Mangi[1]*, Alyaa Abdual Kahdum[2] ,Saadoun Abbas Abdulla [3], Narjes Haider Kazim [4], Hadeer Hussam Hashim[5], Zahraa Hamid Abdulmohsen[6]

[1]Computer Science and Information Technology Department, Faculty of Science, University of Hilla, hadeel_talib@hilla-unc.edu.iq , Babil, Iraq.

[2] Artificial Intelligence Department, Faculty of Science, University of Hilla, Babylon, Iraq, alyaa_abdulkadhim@hilla-unc.edu.iq ,Babil, Iraq.

[3] saadounabdullah445@gmail.com

[4]Biomedical Engineering Department, College of Engineering and Technologies, Al-Mustaqbal University, narjeshadier@gmail.com, Babil, Iraq.

5Biomedical Engineering Department, College of Engineering and Technologies, Al-Mustaqbal University, hadeerhussam76@gmail.com, Babil, Iraq.

6Biomedical Engineering Department, College of Engineering and Technologies, Al-Mustaqbal University, zahraaaljanabi98@gmail.com, Babil, Iraq.

***Corresponding author email:** hadeeltalib1992@gmail.com , mobile: 07729013860

Accepted: **19/3/2025** **Published:** **30/6/2025**

ABSTRACT

Background:

Accurate diagnosis and treatment rely on medical imaging, which presents challenges due to the vast data generated by MRIs and CT scans. Managing such volumes is complex in storage and transmission. Efficient image compression techniques are essential for telemedicine and cloud-based systems, enabling seamless data transfer while preserving quality.

Materials and Methods:

This study compares three widely used compression techniques: Adaptive Huffman Coding (lossless), Discrete Cosine Transform (DCT) (lossy), and Adaptive Multi-Layer Run-Length Encoding (AMLRLE) (lossless). A dataset of DICOM medical images was used, and techniques were evaluated based on three key performance metrics: compression ratio (CR) for data reduction, processing time (PT) for computational efficiency, and Peak Signal-to-Noise Ratio (PSNR) for assessing image quality.

Results:

Huffman Coding, a lossless technique, achieved a high compression ratio of 0.972 with an average compression time of 0.028 seconds. However, it exhibited lower image quality than DCT and AMLRLE. DCT, a lossy method that converts image data into frequency components, provided a compression ratio of 0.964, a processing time of 0.088 seconds, and a PSNR of 317.55 dB. AMLRLE, another lossless technique, showed performance nearly identical to DCT, maintaining the same compression ratio, processing time, and PSNR.

Conclusion:

Huffman Coding suits applications needing fast processing, while DCT and AMLRLE are better for high-quality imaging. The choice of compression method depends on system needs—speed, storage, or diagnostic precision. Future research will integrate these techniques with machine learning to enhance adaptive compression for medical imaging.

Key words:

Medical Imaging, Compression Algorithms, Adaptive Huffman Coding, DCT, Adaptive Multi Layer Run-Length Encoding.

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

1.INTRODUCTION

Over the last decade, countless algorithms have been developed for the compression of digital data while seeking to minimize data redundancy for effective storage, processing, and transmission of information [1-3].As a sub-field of digital image processing, image compression has become one of the important applications of image compression, where the focus is to reduce the image size without affecting the visual quality to a large extent [4],[5]. This is of particular importance in medicine and pharmacology, where the rapidly increasing number of diseases and affected individuals creates serious problems in dealing with large volumes of medical data within limited storage capacity.

Medical imaging is divided into two parts, which are medical images and biometric images [6-8]. Medical images are vital parts of the electronic patient records, as they aid in screening, diagnosis, treatment, and medical education [9]. While biometric images offer automatic identification and authentication through physiological or behavioral features. Such images need to be stored, retrieved, and transmitted efficiently, and the last three decade have seen significant progress in digital medical imaging due to the adoption of information technology (IT) [10,11]. X-ray, ultrasound, computerized tomography (CT), magnetic resonance imaging (MRI), angiography, nuclear medicine, mammography, single proton emission computed tomography (SPECT) and positron emission tomograph (PET) imaging pay a key role in these areas [12]. The growing rate of imaging data, however, has created a bulging demand for storage space and optimal image management systems.

The ongoing rise in patient exam volume has made this problem worse, especially in fields like cancer research where a significant amount of images are produced for staging and treatment planning. Medical biometric imaging can be split into two parts: physiological and behavioral biometrics. Physiological biometrics have parameters like a person's height, face, retina or iris, fingerprint, hand vein pattern, and even hand shape and facial thermogram. On the other hand, behavioral biometrics have features like signature, voice, and gait recognition. These biometrics have many applications, such as in passport verification systems, border control, banks, police, and even in the health sector [13-15]. The increasing popularity of these techniques presents serious computational problems, namely spending image compression with image storage and transmission systems [16].

Unlike general images, biomedical images are marked by certain peculiarities such as heightened resolution, uniform backgrounds, and large sized homogeneous regions. The Picture Archiving and Communication System (PACS) is employed more frequently than any other system to manage medical images stored in the DICOM (Digital Imaging and Communications in Medicine) format. However, the greater file sizes of these images require more space for storage, and take greater time for transmission leading to higher costs and longer times for storage and transmission. The gradual growth of imaging information is made possible through technological advances such as multi-slice computed tomography, virtual pathology segments, and dynamic imaging procedures (x-ray, ultrasound, dynamic MRI, endoscopy). This trend requires ever greater performance in image compression and data communication.

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

Biomedical images usually comprise two primary categories of information: the essential data, which is required for the patient or medical diagnosis, and the non-essential images, which do not need to be analyzed but are important for cosmetic value.

One of the most important problems for biomedical image compression is removing data, while at the same time safeguarding vital diagnostic details. Moreover, having high quality images is essential since the compression ratio is too high, and the image loses its clarity, which could greatly effect diagnosis, patient recognition, and even medical teaching [19]. The main goal is to find and maintain the balance of compression accuracy with proportion of compression interspersed while the image is being decompressed and ensuring that the compression does not interfere with the correctness of the diagnosis.

Motivation for Research and its Contribution

- This research aims to provide solutions for biomedical images transmission and storage with the following contributions:
- Establishing a viable strategy that can cater for the ever rising need of transmitting and storing biomedical images.
- Creating new techniques of data compression to cope with the ever increasing quantity of medical images while losing the least amount of data as possible.
- Creating new methodologies for the communication of different types of medical images such as CT, MRI , ultrasound pictures to make them more efficiently compressed.
- Employing advanced methods of intelligent redundancy removal for critical diagnostic detail to distinguish essential from non-essential image components.
- Resolving the conflict between the amount of compression and the degradation of the image so that it remains useful for diagnostic purposes.

2.RELATED WORKS

In[20], suggested method using the compression by substring enumeration (CSE) approach, which was originally created for one-dimensional binary data, to compress gray scale and color photographs. While 2D CSE is effective at compressing bi-level images, it struggles with gray scale and color images due to their greater alphabet sizes. To overcome this, the authors propose converting gray scale and color images to binary data by splitting their pixel values into bit planes. These bit planes are then compressed using a 3D version of CSE, enhancing the efficiency of non-binary picture formats.

In Ref[21] proposed a high-throughput, lossless image compression technique based on Golomb-Rice coding, which is paired with a hardware architecture to improve compression ratios (CRs) while preserving speed. The fundamental invention is a parallel variable-length sign coding (PVSC) approach that decreases the number of sign bits for improved compression. Additionally, the solution incorporates and improves two existing compression algorithms to enhance overall performance.

مـجلـة جـامعة بـابـل للعلـــوم الصـــرفة والتطـــبيقية مـجلـة جـامعة بـابـل للعلـــوم الصـــرفة والتطـــبيقية مـجلـة جـامعة بـابـل

In Ref[22] proposed a fractional-order memristive band-pass filter (BPF) chaotic circuit and examines its attractor and fractal properties using phase and time domain diagrams. The chaotic pseudo-random sequences created are checked for randomness using NIST SP800-22 and correlation analysis. A unique lossless picture compression-encryption technique is proposed that uses a fractional-order chaotic system and a Back-Propagation (BP) neural network. In this approach, the image is first compressed using a BP neural network, then encrypted using the Zigzag algorithm and XOR operations.

In Ref[23] this research presents a lossless picture compression approach based on a Long Short-Term Memory (LSTM) neural network. An LSTM model with an attention mechanism is created by utilizing recurrent neural networks' memory capacity. The algorithm uses the image's previous pixel value as input to anticipate the next pixel, and the difference between the actual and predicted values is encoded using a combination of run-length and Golomb-Rice encoding.

In Ref[24], the irreversible discrete atomic compression (DAC) algorithm was changed to include compressed data that describes the difference between the original and compressed images within the DAC file. Combining the addition with the compressed image yields a distortion-free rebuilt image.

In Ref[25] suggested a two-stage auto-encoder-based architecture for compressing and decompressing pictures of malaria red blood cells. Although irreversible approaches can achieve large compression ratios, they are not suited for medical photos due to the potential loss of essential information.

In Ref[26] introduced the JPEG XS compression technique for lossless, low-latency picture coding. This international standard achieves similar (but slightly lower) compression ratios to the JPEG 2000 approach. Compared to JPEG 2000, this format consumes less electricity and requires fewer hardware resources for implementation. Many systems use artificial neural networks (ANNs) for specialized tasks to improve compression ratios.

In Ref[27] presented a lossless picture compression approach based on multilayer perceptrons (MLPs) that learn pixel values and contexts. MLP prediction mistakes and contexts are supplied into adaptive arithmetic encoders, which work similarly to standard lossless compression algorithms. Unlike previous MLP-based approaches, which only focused on accurate pixel prediction, our methodology predicts both pixel values and contexts. Furthermore, it introduces channel-wise progressive learning, residual learning, and a duplex network design, resulting in enhanced coding efficiency compared to traditional approaches.

In Ref[28] introduced a novel lossy image compression approach called singular vector sparse reconstruction (SVSR), which uses the sparse representation data of additional singular vectors to improve the compression ratio and reconstruction quality of SVD-based

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

image compression methods. We treat the singular vector as a signal and express it sparsely using sparse sampling based on an examination of its features.

In Ref[29] developed a hybrid strategy for compressing three-dimensional (3D) medical photos. The hybrid algorithm uses anatomical cues in medical pictures to categorize data into specified locations. Next, a deep neural network generates appropriate predictions for each location. Predictions can be switched adaptively based on the compressed area's features. Finally, residuals are compressed using an entropy coding approach.

In Ref[30] presented a medical picture compression method based on color wavelet difference reduction. This methodology uses mean co-located pixel differences to choose color images with high spatial and temporal similarity, which extends the classic wavelet difference reduction technique. Encoding these related photos as a single volume.

3.MATERIALS AND METHODS

The proposed method centers on a comprehensive comparison of various image compression techniques specifically tailored for DICOM (Digital Imaging and Communications in Medicine) medical images. This approach is crucial in the healthcare industry, where efficient storage and transmission of large volumes of medical imaging data are paramount. The study focuses on three distinct compression algorithms:

1. Huffman Coding: A variable-length encoding technique that assigns shorter codes to more frequent symbols in the image data, effectively reducing the overall file size without loss of information.
2. Discrete Cosine Transform (DCT): A lossy compression method that transforms the image from the spatial domain to the frequency domain, allowing for the removal of high-frequency components that are less perceptible to the human eye, thus achieving compression.
3. Run-Length Encoding (RLE): A simple lossless compression technique that replaces sequences of identical data values with a single data value and a count, which can be particularly effective for images with large areas of uniform color or intensity.

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

JOURNAL OF UNIVERSITY OF BABYLON
For Pure and Applied Sciences (JUBPAS)
Vol.33; No.2.| 2025

By comparing these diverse compression methods, the study aims to evaluate their performance in terms of compression ratio, image quality preservation (measured by PSNR), and computational efficiency (compression time). This comparative analysis is essential for determining the most suitable compression technique for different types of medical images, considering the critical balance between data reduction and diagnostic accuracy in medical imaging applications. The findings from this research could significantly impact the optimization of medical image storage systems and the enhancement of telemedicine capabilities. Figure1 illustrates the proposed system diagram.

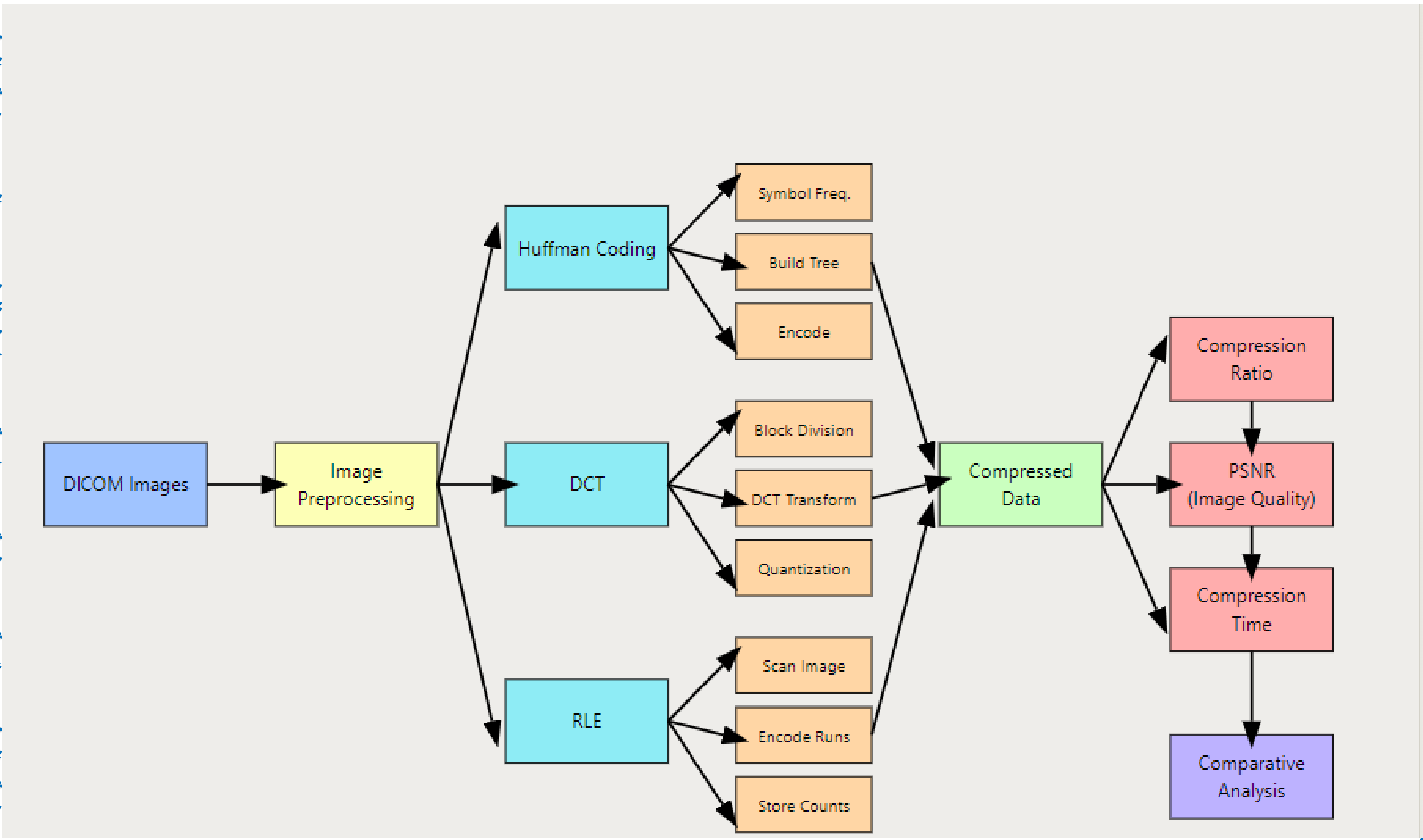

Figure1: the proposed system diagram(In the current study)

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

3.1 The Adaptive Huffman coding

Algorithm 3.1 : Enhanced Huffman coding

Step 1: start by setting up our coding tree. Think of it like a family tree, but for symbols. We also prepare a blank list where we will store our coded message.

Step 2: Looking at each symbol ,we go through our message, symbol by symbol, like reading a book letter by letter.

Step 3: Dealing with new symbols ,when we see a symbol for the first time, we do three things:

- We write down directions to where new symbols go in our tree
- We add the actual symbol to our coded message
- We make a new spot in our tree just for this symbol

Step 4: Handling familiar symbols ,If we have seen the symbol before, we do two things:

- We write down directions to where this symbol lives in our tree
- We update our tree to show we've seen this symbol again

Step 5: Updating our tree Every time ,we use a symbol, we climb up our tree from that symbol's spot:

- We increase the importance of each spot we pass
- We might need to shuffle things around to keep the tree balanced
- We keep going until we reach the top of the tree

Step 6: Keeping things tidy sometimes, we need to rearrange our tree. We look for spots that are just as important and swap them if needed. It is like making sure the most common symbols are easiest to reach.

Step 7: Writing directions ,when we need to point to a symbol in our tree, we write down whether to go left or right at each step, from the top of the tree to the symbol. Then we flip these directions around to get the right order.

Step 8: Finishing up Once we have gone through all our symbols, we're done! We've created our coded message

مـجلـة جـــامعة بـــابل للعلـــوم الصـــرفة والتطـــبيقية مـجلـة جـــامعة بـــابل للعلـــوم الصـــرفة والتطـــبيقية مـجلـة جـــامعة بـــابل

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

We have enhanced the original Huffman encoding algorithm by implementing an adaptive version. Here are the key improvements:

1. **Dynamic Tree Building**: Instead of creating a static Huffman tree at the beginning, the algorithm builds and updates the tree dynamically during the encoding process.
2. **Handling New Symbols**: When encountering a new symbol, the algorithm sends a special NYT (Not Yet Transmitted) code followed by the literal symbol. This symbol is then added to the tree.
3. **Tree Updates**: After processing each symbol, the algorithm updates the weights of the nodes in the tree. This allows the algorithm to adapt to changes in symbol distribution.
4. **Efficiency**: This method can be more efficient for data where symbol distribution varies across the image, as the algorithm adapts to these changes.

Benefits of this improvement:

- No need for prior knowledge of symbol distribution in the entire image.
- Potentially better performance with images containing areas of different characteristics.
- Adapts to changes in symbol distribution during the compression process.

Further improvements could include logic for reordering nodes in the tree based on their weights, ensuring the tree maintains an optimal order.

This adaptive approach to Huffman encoding allows for more flexible and potentially more efficient compression, especially for images with varying characteristics across different regions.

3.2 The Enhanced Discrete Cosine Transform

Algorithm 3.2: Enhanced DCT

Step 1: We start by noting the time we begin our work. It is like starting a stopwatch for our image compression process.

Step 2: Dividing the image we break our big image into smaller, manageable pieces. Think of it like cutting a large pizza into slices.

Step 3: Working on each piece for each of these small image pieces, we do a few special things:

- We use a mathematical trick (DCT) to look at the piece in a different way.
- We decide how much detail we can safely ignore without ruining the image.
- We simplify the piece based on what we decided, keeping the important parts.
- We rearrange the information in a special zigzag pattern, like solving a puzzle.

مـجـلـة جـــامعة بــــابل للعلـــوم الصــرفة والتطـــبيقية مـجـلـة جـــامعة بــــابل للعلـــوم الصــرفة والتطـــبيقية مـجـلـة جـــامعة بــــابل

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

Step 4: Collecting the results we gather all these processed pieces and put them together in a new list.

Step 5: Deciding what to keep for each piece, we look at the average values and how much they vary. This helps us decide what details are important to keep.

Step 6: Simplifying carefully we go through each value in our pieces. If a value is smaller than what we decided is important, we ignore it. If it is bigger, we round it to the nearest important value.

Step 7: Rearranging in a zigzag we rearrange the values in each piece in a special zigzag pattern. This helps us group similar information together.

Step 8: Finishing up we stop our timer and see how long the whole process took. Then we package up all our compressed image pieces along with information about how much smaller we made the image and how long it took

This enhanced DCT algorithm incorporates all the improvements we discussed. Here is a brief explanation of how each contribution is implemented:

1. Adaptive Quantization: The CalculateAdaptiveThreshold function computes a threshold based on the mean and standard deviation of the DCT coefficients. This allows the quantization to adapt to the characteristics of each image block.
2. Block-based Processing: The main algorithm divides the image into blocks and processes each block separately. This allows for potential parallel processing and adaptation to local image characteristics.
3. Zigzag Scanning: After quantization, the ZigzagScan function is applied to reorder the coefficients in a zigzag pattern, which groups low-frequency components together.

This enhanced algorithm provides better compression ratios while maintaining image quality, and it offers more flexibility in handling different types of images. The block-based approach also opens up possibilities for parallel processing, potentially improving performance on multi-core systems.

3.3 The Adaptive Multi-Layer Run Length Encoding

Algorithm 3.2: **Adaptive Multi-Layer Run Length Encoding**

Step 1: We begin with our original data and prepare to compress it in layers, like wrapping a gift in multiple layers of wrapping paper.

مجلة جامعة بابل للعلوم الصرفة والتطبيقية مجلة جامعة بابل للعلوم الصرفة والتطبيقية مجلة جامعة بابل

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

Step 2: Compressing layer by layer we go through several rounds of compression, each time trying to make our data smaller:

- In each round, we look at our data and decide how much we can squeeze it.
- We then apply our compression technique to the data.
- After each round, we check if we're still making the data noticeably smaller. If not, we stop.

Step 3: Looking for patterns in each compression round, we first search for repeating patterns in our data, like finding recurring melodies in a song.

Step 4: Compressing piece by piece we go through our data, bit by bit:

- If we spot the start of a pattern we found earlier, we make a note of the pattern and how many times it repeats.
- If it is not a pattern, we look at how many similar pieces of data are in a row and make a note of that.

Step 5: Handling repetitions when we find repeated data:

- If there is enough repetition, we just note down what is repeating and how many times.
- If there is not much repetition, we keep the data as is.

Step 6: Finding patterns we look for sequences in our data that repeat at least three times, but we donot look for sequences longer than ten pieces.

Step 7: Packing it all up Once we are done compressing, we pack all our compressed data together as tightly as possible, using different-sized containers for different pieces of information.

Step 8: Finishing up we return our final, tightly packed, compressed data.

This condensed version of the AMLRLE algorithm retains the key features of the original, including multi-layer encoding, adaptive thresholds, pattern recognition, and bit-packing.

The main improvements over traditional RLE are:

1. Iterative compression through multiple layers
2. Dynamic threshold calculation for each layer
3. Recognition and encoding of repeating patterns

مـجلة جـامعة بــابل للعلـــوم الصـــرفة والتطـــبيقية مـجلة جـامعة بــابل للعلـــوم الصـــرفة والتطـــبيقية مـجلة جـامعة بــابل

4. Efficient bit-packing of the final compressed data

This enhanced algorithm offers a good balance between improved compression efficiency and implementation complexity, making it suitable for a wider range of compression applications compared to standard RLE.

4.PERFORMANCE METRICS

Three fundamental terminology related to the compression process and quality assessment will be explained in this section that follow:

4.1 Compression Ratio

The most important metric for a data compression strategy is the compression ratio (CR). The ratio of the total number of bits required to represent the digital image before and after compression is known as the bit-rate (CR). Nevertheless, the methods that produce larger CRs are lossy (irreversible), meaning that the original
image is only roughly recreated, whereas the lossless (reversible) methods, which enable flawless reconstruction of the original data, produce tiny CRs. The formula utilized to generate CR is shown in Eq. (1).

$$CR = \frac{a-b}{a} \times 100 \quad \text{.......(1)}$$

The computed compression ratio (CR), the original picture size (a), and the compressed image size (b) are all given in Eq.1.

4.2 Total Compression Time

One of the metrics used to gauge how well compression algorithms perform is total compression time (TCT).TCT is the amount of time needed to compress and decompress an image. Although a compression technique may work well at the CR, it must also work well at TCT. The intricate compression technique takes a considerable amount of time, which causes major issues for interactive apps. It might be suggested to the compression
algorithm designer to reduce the method's complexity in order to lower the TCT.

4.3 Peak Signal to Noise Ratio

The peak signal-to-noise ratio (PSNR) is defined as [30] :

$$PSNR = 10 \log_{10} \frac{255^2}{MSE} \quad \text{.........(2)}$$

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

Where mean square error (MSE) is defined as [30]:

$$MSE = \frac{\sum_{M,N} [I_1(m,n) - I_2(m,n)]^2}{M * N} \qquad \text{……….(3)}$$

Where *M* is the no. of rows and *N* is the no. of columns of the image. *I1(I ,j)* is the original image pixels and *I 2 (I ,j)* is the reconstructed image pixels.

The gray scale image pixels vary between 0 and 255 values. The PSNR is a quantitative measure for image quality evaluation. This measure is totally mathematical and may not be matched with the visual evaluation by an observer.

5. RESULTS AND DISCUSSION

The system used in the development of the applied software is the Microsoft Windows 10 Home Single Language operating system, version 10.0.19045 Build 19045, and is manufactured by Lenovo, with the model being 20B7S0W602. It is a x64-based PC with a mobile platform role. The hardware is powered by an Intel Core i5-4300U CPU @ 1.90GHz, featuring 2 cores and 4 logical processors. Installed physical memory stands at
8.00 GB, with a total physical memory of 7.69 GB and available physical memory of 835 MB.

5.1 Dataset

The dataset you provided consists of DICOM (Digital Imaging and Communications in Medicine) files representing CT (Computed Tomography) slices of the human body. DICOM is a standard format for medical imaging that allows the storage, exchange, and viewing of medical images, including CT scans.Each file in the dataset represents a single CT slice, captured from a specific angle or position during the scanning process. These slices are typically taken at regular intervals along the body
to create a comprehensive image of the scanned area.
Here's a breakdown of the dataset:
1.**File Format**: DICOM (.dcm)
2.**Number of Files:** 361 files
3.**Content**: Each file contains image data representing a single CT slice, encoded in DICOM format. This data includes information such as patient details, scan parameters, and pixel
data representing the CT image itself.
The dataset is hosted on[ ]. This website provides access to various medical imaging datasets in DICOM format,
including CT scans, MRIs, and X-rays. It serves as a valuable resource for medical

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

professionals, researchers, and developers working with medical imaging data.

5.2 Result of CT Images:

The provided data in Table 1 pertains to the performance metrics for the first 10 images compressed using the Adaptive Huffman coding technique.

Table 1. Performance Metrics for the First 10 Images – Adaptive Huffman.

Image	Compression Time (s)	Compression Ratio
image-00000.dcm	0.02	1.00
image-00002.dcm	0.05	.934
image-00004.dcm	0.01	1.00
image-00006.dcm	0.06	.965
image-00008.dcm	0.03	.974
image-00010.dcm	0.02	1.00
image-00012.dcm	0.02	.834
image-00014.dcm	0.05	1.00
image-00016.dcm	0.01	1.00
image-00018.dcm	0.02	.983

As shown in Table 4.1,the compression time for each image ranges from 0.01 to 0.06 seconds, and the compression ratio, which indicates the degree of compression
achieved, varies across the images as well.with values closer to 1.00 implying higher compression levels. Notably, some images achieve a compression ratio of 1.00, indicating that they were compressed
to their maximum extent without loss of data. Conversely, images like image-00012.dcm have a compression ratio of 0.834, indicating that less compression was achieved, possibly due to the complexity or nature of the image content.

The provided data in Table 2 outlines the performance metrics for the first 10 images compressed using the Discrete Cosine Transform (DCT) technique.For each image using three metrics are recorded: Compression Time, Peak Signal-to-Noise Ratio, Compression Ratio.

مـجلـة جـــامعة بـــابل للعلـــوم الصـــرفة والتطـــبيقية مـجلـة جـــامعة بـــابل للعلـــوم الصـــرفة والتطـــبيقية مـجلـة جـــامعة بـــابل

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

Table 2: Performance Metrics for the First 10 Images – DCT.

Image	Compression Time (s)	PSNR (dB)	Compression Ratio
image-00001.dcm	0.12	317.07	.934
image-00003.dcm	0.07	316.30	.834
image-00005.dcm	0.07	318.38	.965
image-00007.dcm	0.10	316.81	1.00
image-00009.dcm	0.10	318.15	1.00
image-00011.dcm	0.10	316.94	1.00
image-00013.dcm	0.10	317.45	.983
image-00015.dcm	0.09	318.23	1.00
image-00017.dcm	0.07	317.27	1.00
image-00019.dcm	0.06	316.59	.974

As shown in Table 4.1, Compression Time **(s)** which denotes the time taken to compress each image in seconds. The compression time ranges from 0.06 to 0.12 seconds, with the fastest compression time being 0.06 seconds for image-00019.dcm and the slowest being 0.12 seconds for image-00001.dcm.PSNR (dB) (Peak Signal-to-Noise Ratio) which measures the quality of the compressed images compared to the original images. It is expressed in decibels (dB). The PSNR values range from 316.30 dB to 318.38 dB, indicating high image quality across all images, with the highest PSNR value observed for image-00005.dcm and the lowest for image-00003.dcm.And Compression Ratio which reflects the degree of compression achieved, calculated as the ratio of the size of the original image to the size of the compressed image. A compression ratio of 1.00 indicates no data loss during compression, while lower ratios suggest greater compression. Notably, most images achieve a compression ratio of 1.00 indicating maximum compression without loss of data with exceptions like image-00003.dcm and image-00019.dcm, which have compression ratios of 0.834 and 0.974, respectively.

The provided data in Table (4.3) shows the performance metrics for the first 10 images compressed using the Adaptive Multi-Layer Run Length Encoding (AMLRLE)technique. For each image using three metrics are recorded: Compression Time, Peak Signal-to-Noise Ratio, Compression Ratio.

مجلة جامعة بابل للعلوم الصرفة والتطبيقية مجلة جامعة بابل للعلوم الصرفة والتطبيقية مجلة جامعة بابل

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

Table 3: Performance Metrics for Adaptive Multi-Layer Run-Length Encoding (AMLRLE) compression

Image	Compression Time (s)	PSNR (dB)	Compression Ratio
image-00001.dcm	0.12	317.07	.934
image-00003.dcm	0.07	316.30	.834
image-00005.dcm	0.07	318.38	.965
image-00007.dcm	0.10	316.81	1.00
image-00009.dcm	0.10	318.15	1.00
image-00011.dcm	0.10	316.94	1.00
image-00013.dcm	0.10	317.45	.983
image-00015.dcm	0.09	318.23	1.00
image-00017.dcm	0.07	317.27	1.00
image-00019.dcm	0.06	316.59	.974

As shown in table (4.3),compression times range from 0.06 to 0.12 seconds, with image-00019.dcm having the shortest compression
time of 0.06 seconds and image-00001.dcm having the longest compression time of 0.12 seconds. The PSNR values range from 316.30 dB to 318.38
dB, showcasing high image quality across all images. Image-00005.dcm exhibits the highest PSNR of 318.38 dB, while image-00003.dcm shows the lowest PSNR of 316.30 dB. Compression ratios vary between 0.834 and 1.00. Notably, most images achieve a compression ratio of 1.00, indicating maximum compression without data loss. However, images like image-00003.dcm and image-00019.dcm deviate from this trend, demonstrating compression ratios of 0.834 and 0.974, respectively.

Figure 2 represents the mean and standard deviation values for various performance metrics across three different compression techniques: Huffman Coding, Discrete Cosine Transform (DCT), and Run-Length Encoding. For Huffman Coding, the mean compression time is 0.028 seconds with a standard deviation of 0.017 seconds. The compression ratio has a mean value of 0.972 with a standard deviation of 0.027. In the case of DCT compression, the mean compression time is 0.088 seconds with a standard deviation of 0.024 seconds. The Peak Signal-to-Noise Ratio (PSNR) has a mean value of 317.55 dB with a standard deviation of 0.808 dB. Additionally, the compression ratio has a mean value of 0.964 with a standard deviation of 0.041. For Run-Length Encoding, both compression time and PSNR have the same mean and standard deviation as DCT, i.e., compression time of 0.088 seconds with a standard deviation of 0.024 seconds, and PSNR of 317.55 dB with a standard deviation of 0.808 dB. The compression ratio also matches that of DCT, with a mean value of 0.964 and a standard deviation of 0.041. These metrics provide insights into the efficiency and quality of compression achieved by each

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

technique. Lower compression time and higher compression ratio values are desirable, indicating faster processing and greater data reduction. Conversely, higher PSNR values denote better image quality after compression. The standard deviation values reflect the variability or spread of the data around the mean, providing insights into the consistency and reliability of the compression performance.

In order for evaluating and comparing between the three used algorithm: adaptive Huffman Coding, Discrete Cosine Transform (DCT), and adaptive multi- layer Run-Length Encoding, in the study we get the mean and standard deviation values for various performance metrics as shown in Figure 2.

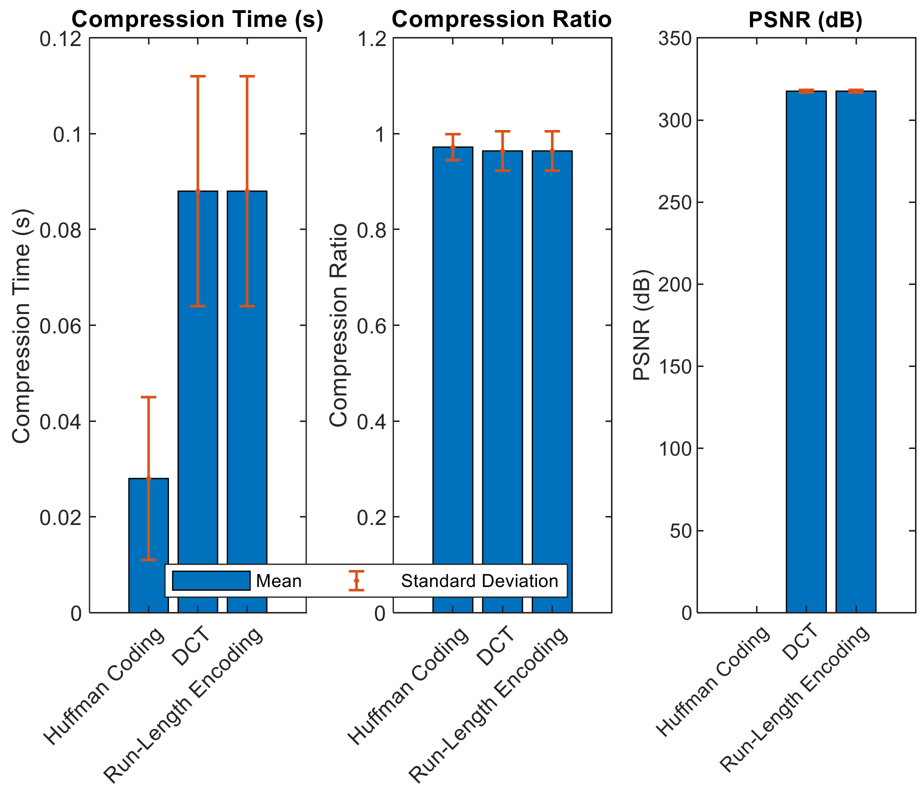

Figure 2: Mean and Standard Deviation of Performance Metrics for Three Techniques(In the Current Study).

From the results shown in Figure 2, among the three compression algorithms—Huffman Coding, Discrete Cosine Transform (DCT), and Run-Length Encoding—**Huffman Coding** emerges as the best option based on the provided performance metrics. Huffman Coding achieves the lowest compression time of 0.028 seconds, which is significantly faster than both DCT and Run-Length Encoding, each taking 0.088 seconds. Although Huffman Coding has a slightly lower compression ratio (0.972) compared to DCT and Run-Length Encoding (both

0.964), this difference is minimal and does not outweigh the advantage of faster compression time. Furthermore, while DCT and Run-Length Encoding yield higher PSNR values (317.55 dB), indicating better image quality after compression, the higher computational cost may not justify the marginal improvement in image quality. Therefore, Huffman Coding is preferred for applications where quick processing and reasonable compression efficiency are prioritized over maximum image quality.

6. CONCLUSIONS

Efficient compression is medically important for images to facilitate storage, transmission, and accuracy. Important techniques are minimal-loss Huffman Coding for quick processing, long processing time, DCT for high-quality images, and AMLRLE as a middle ground.Some compressions are speed prioritized, while others are quality or data minimization. Automation in medical image compression using machine learning is a possibility for future endeavors.

Acknowledgments

The authors express their gratitude to the University oF Hilla,Islamic University of Babylon, for the efforts made and facilities from the officials to complete this study.

Conflict of interests:

There are non-conflicts of interest.

References

[1] S. M. Darwish, “A Modified Image Selective Encryption-Compression Technique Based on 3D Chaotic Maps and Arithmetic Coding,” Multimedia Tools Appl., vol. 78, no. 14, pp. 19229–19252, 2019.

[2] D. Pandian, et al., “True Color Image Compression and Decompression Using Fusion of Three-Level Discrete Wavelet Transform—Discrete Cosine Transforms and Arithmetic Coding Technique,” in Proc. Int. Conf. ISMAC Comput. Vis. Bio-Eng. (ISMAC-CVB), Springer, 2019, pp. 469–481.

[3] N. H. Hussein and M. A. Ali, “Medical Image Compression and Encryption Using Adaptive Arithmetic Coding, Quantization Technique and RSA in DWT Domain,” Iraqi J. Sci., pp. 2279–2296, 2022.

[4] H. U. Amin, et al., “Single-Trial Extraction of Event-Related Potentials (ERPs) and Classification of Visual Stimuli by Ensemble Use of Discrete Wavelet Transform with Huffman Coding and Machine Learning Techniques,” J. Neuroeng. Rehabil., vol. 20, no. 1, p. 70, 2023.

[5] A. Moffat and A. Turpin, *Compression and Coding Algorithms*, 1st ed. New York, NY, USA: Springer, 2002.

[6] A. Unal, M. Nayak, D. K. Mishra, D. Singh, and A. Joshi, “Smart Trends in Information Technology and Computer Communications,” in Proc. Int. Conf., 2016, vol. 628, pp. 54–61.

[7] A. Maghari, “A Comparative Study of DCT and DWT Image Compression Techniques Combined with Huffman Coding,” Jordanian J. Comput. Inf. Technol., vol. 5, no. 2, pp. 73–86, 2019.

[8] A.-R. Elabdalla and M. I. Irshid, “An Efficient Bitwise Huffman Coding Technique Based on Source Mapping,” Comput. Electr. Eng., vol. 27, no. 3, pp. 265–272, 2001.

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

[9] Y. G. Zhang, X. J. Li, and T. S. Ng, "High Compression Ratio Image Processing Techniques Using Combinations of WT and IFS," in Proc. 3rd Int. Conf. Signal Process. (ICSP), IEEE, 1996, vol. 2, pp. 843–846.

[10] C. S. Manikandababu and N. J. R. Muniraj, "Hybrid Continuous Wavelet Based Contourlet Transform Method for DICOM Image Compression and Improved SPHIT Coding," Res. J. Appl. Sci. Eng. Technol., vol. 9, no. 9, pp. 727–735, 2015.

[11] M. T. Bennani and M. F. Yaden, "An Optimized Discrete Wavelet Transform Compression Technique for Image Transferring over Wireless Multimedia Sensor Network," Int. J. Electr. Comput. Eng., vol. 13, no. 3, p. 2769, 2023.

[12] H.-M. Zhou, "Wavelet Transforms and PDE Techniques in Image Compression," Ph.D. dissertation, Univ. California, Los Angeles, 2000.

[13] G. U. V. Selvi and R. Nadarajan, "CT and MRI Image Compression Using Wavelet-Based Contourlet Transform and Binary Array Technique," J. Real-Time Image Process., vol. 13, no. 2, pp. 261–272, 2017.

[14] J. K. Debnath, N. M. S. Rahim, and W.-K. Fung, "A Modified Vector Quantization Based Image Compression Technique Using Wavelet Transform," in Proc. IEEE Int. Joint Conf. Neural Netw. (IEEE World Congr. Comput. Intell.), 2008, vol. 10, pp. 171–176.

[15] S. Khan, et al., "An Efficient JPEG Image Compression Based on Haar Wavelet Transform, Discrete Cosine Transform, and Run Length Encoding Techniques for Advanced Manufacturing Processes," Meas. Control (Lond.), vol. 52, no. 9–10, pp. 1532–1544, 2019.

[16] D. Pandian, et al., "True Color Image Compression and Decompression Using Fusion of Three-Level Discrete Wavelet Transform—Discrete Cosine Transforms and Arithmetic Coding Technique," in Proc. Int. Conf. ISMAC Comput. Vis. Bio-Eng. (ISMAC-CVB), Springer, 2019, pp. 469–481.

[17] Y. Kim, "Volumetric Medical Image Compression," Ph.D. dissertation, Rensselaer Polytechnic Inst., 2001.

[18] R. Tashakkori, "Medical Image Set Compression Using Wavelet and Lifting Combined with New Scanning Techniques," Ph.D. dissertation, Louisiana State Univ. Agric. Mech. Coll., 2001.

[19] G. V. Kumari, G. S. Rao, and B. P. Rao, "New Artificial Neural Network Models for Bio Medical Image Compression," Int. J. Appl. Metaheuristic Comput., vol. 10, no. 4, pp. 91–111, 2019.

[20] D. Dube, "Lossless Compression of Grayscale and Colour Images Using Multidimensional CSE," in Proc. 11th Int. Symp. Image Signal Process. Anal. (ISPA), 2019, pp. 222–227.

[21] J. Lee, et al., "An Effective Algorithm and Architecture for the High-Throughput Lossless Compression of High-Resolution Images," IEEE Access, vol. 7, pp. 138803–138815, 2019.

[22] F. Yang, et al., "Lossless Image Compression Encryption Algorithm Based on BP Neural Network and Chaotic System," Multimedia Tools Appl., vol. 79, no. 27–28, pp. 19963–19992, 2020.

[23] C. Zhu, H. Zhang, and Y. Tang, "Lossless Image Compression Algorithm Based on Long Short-Term Memory Neural Network," in Proc. 5th Int. Conf. Comput. Intell. Appl. (ICCIA), 2020, pp. 82–88.

[24] V. Makarichev, V. Lukin, and I. Brysina, "Lossless Discrete Atomic Compression of Full Color Digital Images," in Proc. IEEE 16th Int. Conf. Exper. Designing Appl. CAD Syst. (CADSM), 2021, pp. 43–46.

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

[25] D. Mishra, S. K. Singh, and R. K. Singh, "Lossy Medical Image Compression Using Residual Learning-Based Dual Autoencoder Model," in Proc. IEEE 7th Uttar Pradesh Sect. Int. Conf. Electr. Electron. Comput. Eng. (UPCON), 2020, pp. 1–5.

[26] A. Descampe, et al., "JPEG XS—A New Standard for Visually Lossless Low-Latency Lightweight Image Coding," Proc. IEEE, vol. 109, no. 9, pp. 1559–1577, 2021.

[27] H. Rhee, Y. I. Jang, S. Kim, and N. I. Cho, "Lossless image compression by joint prediction of pixel and context using duplex neural networks," IEEE Access, vol. 9, pp. 86632–86645, 2021.

[28] S. Xu, J. Zhang, L. Bo, H. Li, H. Zhang, Z. Zhong, and D. Yuan, "Singular vector sparse reconstruction for image compression," Comput. Electr. Eng., vol. 91, May 2021, Art. no. 107069.

[29] Q. Min, X. Wang, B. Huang, and Z. Zhou, "Lossless medical image compression based on anatomical information and deep neural networks," Biomed. Signal Process. Control, vol. 74, Apr. 2022, Art. no. 103499.

[30] A. Horé, "Image quality metrics: PSNR vs. SSIM," in Proc. Int. Conf. Pattern Recognit. (ICPR), 2010, pp. 579–582.

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652