Evaluating Lossy and Lossless Compression for DICOM Medical Files

Yizhe Yang
Clemson University
Clemson, South Carolina, USA
yizhey@g.clemson.edu

Carson D. Sisk
Clemson University
Clemson, South Carolina, USA
cdsisk@g.clemson.edu

Jon C. Calhoun
Clemson University
Clemson, South Carolina, USA
jonccal@clemson.edu

2024 IEEE International Conference on Big Data (BigData) | 979-8-3503-6248-0/24/$31.00 ©2024 IEEE | DOI: 10.1109/BigData62323.2024.10825078

***Abstract*—Digital Imaging and Communications in Medicine (DICOM) is a widely used standard for handling, storing, and sharing medical images. However, the large file sizes associated with DICOM data pose challenges for storage and data transfer. Data reduction helps mitigate these challenges by reducing the size of the data while maintaining its integrity. This paper examines various compression methods to reduce the size of DICOM files. We evaluate 5 lossless and 4 lossy compressors on DICOM data. This study aims to compare and evaluate the performance of these compressors. By analyzing each compressor's compression efficiency and produced image fidelity, this research seeks to determine the most effective compression strategy. Results show SZ3 is able to achieve $183.74\times$ with error bound $1e^{-7}$ and ZFP received compression bandwidth `303.82 MB/s` while error bound is $1e^{-7}$.**

***Index Terms*—DICOM, Medical Data, Lossy Compression, Lossless Compression, Data Visualization**

I. Introduction

DICOM (Digital Imaging and Communications in Medicine) files serve as data containers for medical images. Specifically, image files that are generally referred to as "DICOM format files" or simply "DICOM files" and are represented as ".dcm." [1]. What sets DICOM apart from other image formats is its ability to store not only images, but at the same time related metadata. This includes detailed patient data, study descriptions, and image acquisition parameters [2].

Medical imaging techniques generate data crucial for patient diagnosis and treatment planning, with the high level of intricacy in these images contributing to the substantial sizes of DICOM files.

Healthcare facilities globally face substantial challenges due to the rapid increase in medical imaging data. Healthcare providers face the need for substantial storage capacity and bandwidth to handle the vast amounts of data generated by high-resolution imaging modalities such as MRI and CT scans. However, many hospitals and clinics, particularly those in resource-limited settings, struggle to keep up with this data explosion. Our research aims to address these challenges by introducing an efficient compression technique specifically designed for DICOM images. By employing lossless compression on metadata and utilizing controlled lossy compression on image data, we enable significant reductions in file sizes without sacrificing crucial information or perceptible image quality. Implementing our compression algorithms in real-world healthcare systems can lead to more efficient data management, cost savings, and improved patient outcomes by enabling quicker and more reliable access to medical imaging data. Figure 1 shows the initial packets of a DICOM file form the "header", which includes patient demographics, acquisition parameters, image dimensions, and other information needed to display the image. This header is encoded with the pixel intensity data. The header is decoded and associated with the correct study and patient.

With the increasing amount of medical imaging data, costs associated with storage and transmission have also risen. For example, in 2013, transferring an uncompressed CT study of 1,542 images (around 930.17 MB) over a 12 Mbps connection requires approximately 10 minutes, resulting in increased electricity consumption and higher operational costs [3]. Employing both lossy and lossless compression algorithms help address these challenges by reducing file sizes, lowering storage demands, and decreasing data transfer times, which in turn reduces energy usage and improves the efficiency of medical workflows. [4] To address this challenge, our study explores the use of 4 lossy compression algorithms and 5 lossless compression algorithms applied to a dataset consisting of 2,635 DICOM files. Of these DICOM files, there are 3 different styles of a DICOM file.

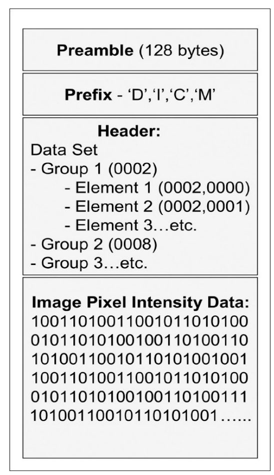


Fig. 1: DICOM File Structure

This paper uses DICOM medical data to determine which compressor and error bounds are most effective for accurately compressing the data. By evaluating different data compression methods, we analyze both lossless and lossy compression methods for DICOM medical data. The contributions of this paper are as follows:

- A comparative analysis of 5 lossless and 4 lossy advanced compressors to compress DICOM medical images.
- For lossless methods, ZLIB provides the best compression ratio of $18.27\times$ times, and LZ4 offers the best compression bandwidth of `1265.47 MB/s`.
- Among lossy methods, SZ provides the best compression ratio of $5530.82\times$ times on `Difusion` data set with $1e^{-9}$, and ZFP delivers the highest overall compression bandwidth of `303.82 MB/s`.

II. BACKGROUND

A. Data Sets

In our study, we used real-world DICOM files. They contain 3 different data sets (`CE (Contrast-Ehanced`, `Perfusion(ep2dperf)` and `Diffusion(ep2ddifMDDWIPAT)`).

- `CE (Contrast-Enhanced)`: CE images emphasize vascular structures and tissues by using a contrast agent that enhances differences in signal intensity.
- `Diffusion (ep2ddifMDDWIPAT)`: Diffusion imaging captures the movement of water molecules within tissues, which is sensitive to cellular density and structure. [5]
- `Perfusion (ep2dperf)`: Perfusion imaging measures blood flow, providing quantitative data that is crucial for assessing tissue viability. [6]

The differences between them stem from the varying ways in which they capture and represent medical data. This also leads to dissimilar results after compression with different compressors and different error bounds.

Medical imaging data stored in the DICOM format often consist of multi-dimensional arrays, with slices from modalities like CT and MRI scans commonly having dimensions of 512×512 pixels. These datasets can be substantial in size—sometimes reaching multiple gigabytes per study—due to the requirement for high-resolution images necessary for precise diagnosis [7]. In 2013, the global volume of healthcare data was approximately 153 exabytes (1 exabyte = 1 billion gigabytes), which surged to an estimated 2,314 exabytes by 2020 [8].

B. Lossless Compressors

We used 5 lossless compressors in this paper: BLOSCLZ [9], LZ4 [10], LZ4HC [10], Zstandard [11] and ZLIB [12]. Currently, it is common practice to use lossless compressors on patient DICOM files. This is because there is no risk of any important data being lost, and it is still better than not compressing the files at all. Thus, we also include 5 popular lossless compressors for our lossy compressors to compare against.

1) BLOSCLZ: BLOSCLZ [9] is a meta-compressor made to compress binary data. Similar to other compressors, it works by first breaking data down into individual blocks. Then, since it is a meta-compressor, it then selects from a variety of compressors a compressor for a specific block. It also has access to different methods to preprocess a block, allowing it to compress even better. Blosc utilizes multi-threading an efficient memory access, which is ideal for situations where a lot of data must be compressed quickly.
2) LZ4: LZ4 [10] is an LZ77-type compressor also focused on fast compress / decompress times. It separates blocks of data to be compressed into blocks. Each block is a sequence that starts with a token.
3) LZ4HC: LZ4HC [13] was also used, which optimizes for higher compression than the normal LZ4, but at the cost of relatively lower speeds.
4) Zstandard: Zstandard [11] is a compressor that builds a dictionary tailored to whatever data it needs to compress. It then splits the data into frames. Whenever a new frame is to be compressed, it references the dictionary, and creates prediction algorithm for the frame. It then uses Huffman encoding on the frames.
5) ZLIB: ZLIB [12] uses deflation, which is a compressed stream of data blocks. Each block starts with a short header, and then either the raw data, a static Huffman compressed section, or a dynamic Huffman compressed section

C. Lossy Compressors

Lossless compressors compress data without ever losing any data points, such that the decompressed data is indistinguishable from the original. However, you quickly hit a limit as to how much compressed using only lossless methods. Therefore, lossy compression is commonly used in situations where a higher compression ratio is needed. Lossy compression refers to the irreversible loss of some data during the compression process, and its main purpose is to significantly reduce file size while maintaining a level of quality suitable for the intended purpose. We evaluated 4 lossy compressors: JPEG [], SZ [14], SZ3 [15] and ZFP [16].

1) JPEG: Joint Photographic Experts Group [17] (JPEG) compression is a common method for reducing image file sizes, especially for photos, by discarding some image data, which slightly lowers quality but saves space. Developed in 1992, JPEG uses a process that breaks images into 8x8 pixel blocks, applies a mathematical transformation, and then compresses the data. This "lossy" compression allows adjustable quality levels: more compression means smaller files with more quality loss, while less compression preserves more detail. It's best for photos rather than sharp images like logos and is widely used with '.jpg' or '.jpeg' file extensions.
2) SZ: SZ [14] is floating point data compressor which has four main steps. Firstly, it divides the data to be compressed into blocks of data. For each of these blocks,

a separate prediction function is generated. Secondly, SZ quantizes with whichever error bound was specified. Thirdly, the quantization index is encoded via Huffman encoding. Finally, it is losslessly compressed further improving the CR. [18]

3) SZ3: SZ3 [15] SZ3 uses a modular approach to compressing the data, where the process has 5 distinct modules: preprocessing, prediction, quantization, encoding, and lossless compression. Preprocessing starts by shaping the data to where it is more easily compressed. The second module is prediction. SZ3 uses a Lorenzo predictor, a regression based predictor, to perform prediction for each data point based on the neighboring data points. Thirdly, the error produced by the predictor is quantized. Fourthly, the quantized error data is encoded, shrinking their size. Fifthly, the now encoded data is losslessly compressed, shrinking the size once more.
4) ZFP: ZFP [16], another floating point data compressor for 3D data, has 5 main steps. Before it starts, carves 4 x 4 x 4 blocks out of the data. These blocks are then individually compressed. The first step to compress a block is to align all of the data points in a block to a common exponent. Secondly, it converts the floating point data into fixed point data. Thirdly, it applies a transform to the data in the block to decorrelate the values from each other. Fourthly, it orders the transform coefficients by expected magnitude. Fifthly, it encodes the coefficients.

III. METHODS AND ANALYSIS

In this paper, DICOM medical image compression is evaluated on the extent of data reduction and its bandwidth, i.e., the speed of compressing and decompressing the data. The quality of the compressed image, as measured by the peak signal-to-noise ratio, is also evaluated. There are 3 different types in our data set: `CE`, `Diffusion`, and `Perfusion`, and each have their own purpose in the medical field.

To assess the effectiveness of a current image compression algorithm on DICOM files, we used JPEG compression. Since JPEG compression is typically applied to standard image formats, we first converted the DICOM files to JPEG format. This conversion enabled the application of JPEG compression directly. We selected a quality setting of 50 to balance file size reduction with image quality, as this level is commonly used to achieve significant compression while maintaining acceptable visual fidelity [19].

We apply lossless compression techniques specifically to the DICOM metadata. This approach ensures that all critical patient information and imaging parameters remain entirely intact and unaltered throughout the compression and decompression processes. This not only maintains compliance with medical data standards but also enhances interoperability between different healthcare information systems.

To apply lossy compression to the DICOM files, we first normalized the pixel values, originally stored as 16-bit unsigned integers (uint16), to a 0 to 1 range using the float32 data type. This normalization facilitates the compression process by allowing algorithms to operate more effectively on floating-point data. After decompression, we converted the pixel values back to uint16 to restore them to their original format. This ensures compatibility with standard DICOM viewers and preserves the high dynamic range necessary for accurate medical diagnosis. The compressors employed in our compression framework exhibit linear time complexity with respect to the size of the input data. This linearity implies that the computational time required for compression and decompression scales proportionally with the amount of data processed. Such efficiency is crucial when dealing with large-scale medical images, where computational resources and time are often constrained. Linear time algorithms enable real-time processing capabilities, making our solution practical for clinical settings where swift access to imaging data is essential for patient care.

Given that each compression algorithm inherently balances image quality and compression efficiency in distinct ways, we adopted a systematic approach to identify optimal performance thresholds. By testing error bounds from $1e^{-9}$ to $1e^{-1}$, we sought to reveal the conditions under which each algorithm could best fulfill its intended purpose, ultimately guiding a balanced selection that maximizes both data fidelity and compression efficacy.

For each compressed file, metrics were also recorded: compression ratio achieved, time spent compressing, and other relevant metrics (such as compression and decompression bandwidth). Each compressed file was then decompressed. We performed a quality assessment on these decompressed files to compare against the original DICOM file. One way was by calculating the peak signal-to-noise ratio (PSNR). The higher the PSNR, the compressed image's quality is closer to the original image. We also visually compared the post-decompressed DICOM images to original DICOM files to ensure our compression did not significantly alter the images.

A. Evaluation Metrics

In order to find which compressor produces the best data reduction, the compression ratio (CR) metric is used. The compression ratio is a measure of the effectiveness of a data compression algorithm. It is defined as the ratio of the original data size to the compressed data size. A higher compression ratio indicates that the data has been reduced more significantly.

$$\text{CompressionRatio} = \frac{\text{UncompressedSize}}{\text{CompressedSize}} \tag{1}$$

The transfer time of DICOM files is a critical factor in medical imaging workflows.When compared to lossy methods, lossless compression algorithms generally result in lower compression ratios despite their ability to preserve all original data without any degradation. This results in larger file sizes, which can lead to longer transmission times over networks. However, lossless compression is essential in scenarios where maintaining data integrity is paramount, as it ensures that no information is lost or altered during the compression process.

The term Compression Bandwidth (cBW) refers to the rate at which an algorithm compresses uncompressed data. A higher cBW indicates a faster compression process, which is advantageous in time-sensitive environments. Balancing compression speed (cBW), compression ratio, and data integrity is crucial when selecting an appropriate compression method for DICOM files.

$$cBW = \frac{\text{Uncompressed Size}}{\text{Compression Time}} \tag{2}$$

Decompression bandwidth (dBW) is the speed at which the compressed data is decompressed back to its original form. This timing does not include loading data.

$$dBW = \frac{\text{Uncompressed Size}}{\text{Decompression Time}} \tag{3}$$

Lossy compression achieves a much higher CR compared to lossless compression. However, to obtain this high CR, the quality of the lossy compressed DICOM files may degrade. Comparing the resulting image quality with that of lossless compression; this metric is referred to as accuracy and measured by PSNR.

Peak Signal-to-Noise Ratio (PSNR) is a metric used to measure the quality of a reconstructed image compared to its original version. This is the traditional method to measure quality after a compression has taken place. PSNR is expressed in decibels (dB) and is calculated using the Mean Squared Error (MSE) between the original and compressed images. A higher PSNR value typically indicates better image quality. PSNR is used to evaluate lossy compression algorithms. Typically, a PSNR value of above 40 dB is acceptable for medical images [20]. However, different use cases will require different DICOM precisions. Since lossless compressors do not distort the original data, as they don't add noise in the compression process, it is not necessary to measure them in this way.

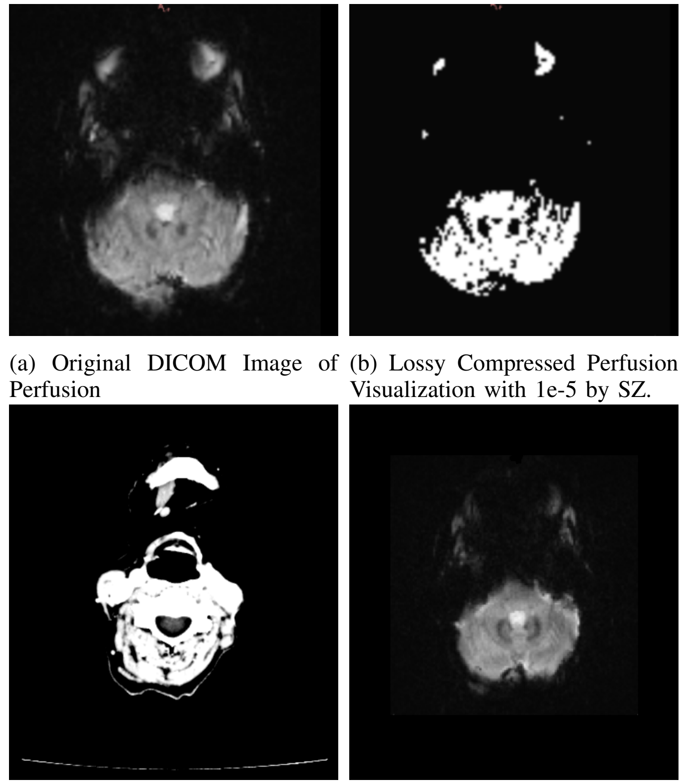
$$\text{PSNR} = 10 \cdot \log_{10}\left(\frac{\text{MAX}^2}{\text{MSE}}\right) \tag{4}$$

In this case, MAX is the maximum possible pixel value of the image (e.g., 255 for an 8-bit image). The Mean Squared Error (MSE) is the Mean Squared Error between the original and compressed images. MSE is defined as:

$$\text{MSE} = \frac{1}{MN}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\left[I(i,j) - K(i,j)\right]^2 \tag{5}$$

Where $I(i,j)$ is the pixel value of the original image at position (i,j), $K(i,j)$ is the pixel value of the compressed image at position (i,j), M and N are the dimensions of the image.

PSNR provides insight into the extent of changes in pixel values, which is particularly important for the accuracy and reliability of DICOM-format medical images.



(a) Original DICOM Image of Perfusion

(b) Lossy Compressed Perfusion Visualization with 1e-5 by SZ.

(c) Original DICOM Image of CE

(d) Converted DICOM JPEG Image

Fig. 2: Example DICOM Images with Various Standards

B. Visualization

Figure 2a shows an original DICOM image with `Perfusion`. Figure 2b displays the image after being processed with SZ lossy compression; the differences between the original and compressed images are substantial, even with an acceptable error bound. Figure 2c is an example of an original DICOM file of `CE`. Converting DICOM files into jpg files results in images similar to Figure 2d.

IV. RESULTS

A. Software Environment

Testing was performed on the Palmetto Cluster 2 at Clemson University. The node requested for the experiment contains 2× 20-core Intel(R) Xeon(R) Gold 6258R CPUs with a clock frequency of 2.70GHz and 384 GB of RAM. All of these experiments were ran single-threaded. Using multi-threading and more gpus may receive better results. The compressors and environment softwares are shown in table I.

B. JEPG Compression

DICOM files differ from standard image formats by containing patient metadata and the capability to store multiple images within a single file. [22] To apply JPEG compression, we first extracted individual images from the DICOM files and converted them to JPEG format. The table below presents the median compression metrics for JPEG compression across

TABLE I: Software and Libraries Used in the Experiment

Software/Library	Version
LibPressio [21]	0.99.2
GCC	12.1.0
Python	3.9.2
ZLIB	1.2.13
ZFP	1.0.0
SZ-master	3.2
SZ3	3.1.7
BLOSC	1.11.2
LZ4	4.3.2
ZSTD	1,5,2
Python	3.11

different DICOM categories: This approach revealed two sig-

Categories	CR	cBW	dBW
CE	58.71	10.10	3.81
Perfusion	24.27	0.82	0.82
Diffusion	72.55	21.77	5.66

TABLE II: Median Compression Metrics for JPEG Compression by DICOM Category

nificant limitations: a low compression ratio and substantial data loss. These findings emphasized the shortcomings of the standard JPEG compression technique for DICOM files and underscored the need to explore alternative compression methods that can achieve higher compression ratios while retaining essential diagnostic information.

C. Lossless Compression

All the lossless compressors were configured to compress at their highest capability. Figure 3 compares the CRs achieved by the same set of algorithms under varying bound constraints.

ZLIB achieved the highest max compression ratio across most categories (such as $18.17\times$ for `CE`), indicating its strength in reducing file sizes the most. However, its lower bandwidth suggests it is slower than alternatives like LZ4. The different categories of DICOM files in lossless compression `CE`, `Perfusion` and `Diffusion` have little effect on the overall trend of CR. In analyzing the lossless compression results, ZSTD also has results competitive with ZLIB across all categories. Specifically, ZLIB achieves the highest maximum CR of $18.17\times$ in the `CE` category, while ZSTD consistently shows strong performance across all categories with a notable median CR of $4.97\times$ in the `Diffusion` category. This indicates that both compressors effectively reduce file size, making them suitable for applications where maximizing compression is crucial.

Conversely, compressors like LZ4 and BLOSCLZ demonstrate significantly higher bandwidth capabilities in some categories, which can translate into faster compression and decompression speeds. For instance, LZ4 achieves a maximum decompression bandwidth of 3768.40 MB/s in the `CE` category.

Figure 4 and 5 shows the compression and decompression bandwidth for various lossless algorithms (ZSTD, BLOSCLZ, LZ4, LZ4HC, ZLIB).

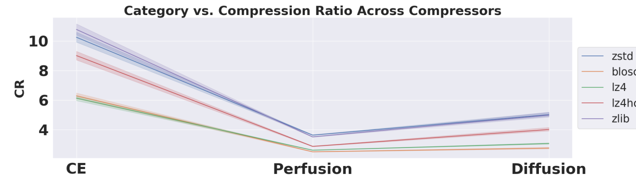


Fig. 3: Categories vs Compression Ratio with Different Lossless Compressors.

LZ4 is the fastest in terms of compression bandwidth, which makes it the first choice for applications that require fast compression. It achieves 883.04 MB/s on `CE`. With LZ4 and BLOSC having high bandwidth for compression and decompression, these compressors are suited for scenarios where speed is more critical than achieving the maximum compression ratio. This makes them ideal for applications requiring rapid access to data, such as real-time diagnostic imaging. Compressors with high compression bandwidth (e.g., LZ4 for the CE category and BLOSC for Perfusion) can accelerate the processing time required to store and retrieve images. For systems with limited computational resources, compressors that balance compression ratio and bandwidth, like ZSTD, help manage large datasets without heavily taxing the system.

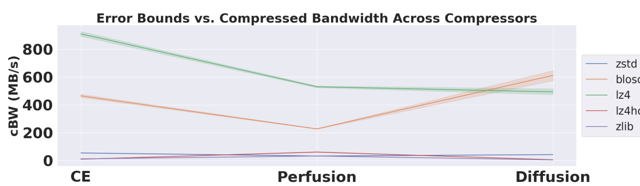


Fig. 4: Compression Bandwidth with Different Lossless Compressors.

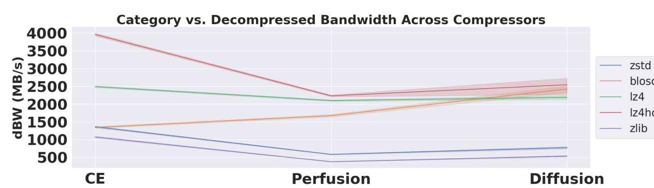


Fig. 5: Decompression Bandwidth with Different Lossless Compressors.

The ZSTD compressor demonstrates a well-rounded performance, balancing a moderate to high CR with competitive bandwidth across various categories. For instance, in the CE category, ZSTD achieves a maximum CR of $16.65\times$ while maintaining a cBW of 71.43 MB/s and a dBW of 1,720.09 MB/s. This indicates ZSTD would suit an environment requiring efficient data reduction without sacrificing speed.

In the `Perfusion` and `Diffusion` categories, ZSTD also exhibits notable performance, with a CR of $4.07\times$ and $6.07\times$, respectively. Its maximum cBW in `Perfusion` is

42.11 MB/s, and in `Diffusion`, it's 46.93 MB/s, providing a consistent transfer speed that outperforms other compressors in terms of dBW. These results highlight ZSTD's versatility for diverse medical imaging scenarios, as it offers both substantial compression and fast data handling capabilities, which are essential for managing large DICOM files efficiently.

D. Lossy Compression

Each lossy compressor is evaluated within a set error bound. The error bound is used to set the accuracy of the data so that its effect on the CR, cBW, dBW and PSNR is analyzed. The error bound varies between $1e^{-9}$ and $1e^{-1}$.

Figure 6 compares the CR for different compressors — SZ, SZ3, and ZFP—across various error bounds, on our 3 different types of DICOM images - `CE`, `Perfusion` and `Diffusion`.

Figure 6 shows the CR for each of the three floating point compressors. All three categories have similar CR across the different error bounds. This indicates that the maximum CR was already met without having to sacrifice data by increasing the error bounds.

SZ3 received the highest median CR of $3439\times$ on the `Perfusion` type DICOM file with the error bound of $1e^{-5}$, which is $1.5\times$ higher than SZ. On the `Diffusion` dataset, SZ3 achieves an median CR of $42.02\times$ and a maximum CR of $183.74\times$ at the highest error bound of $1e^{-7}$. In comparison, SZ achieves an median CR of $2062\times$ with a maximum of $5530\times$ at the error bound $1e^{-9}$, while ZFP, which has the lowest CR among the three, achieves a median CR of $236.54\times$ and a maximum of $249.6\times$.
For `CE`, SZ3 has $169.39\times$ CR which is 1.3 times more than SZ, and 8.79 times higher than ZFP. This shows that classes of DICOM images respond differently to the different error bounds of compression. For example, the CR of `Diffusion` increases significantly with increasing error bounds. `Perfusion` and `CE` maintain consistently lower CR even as the error bounds increase.

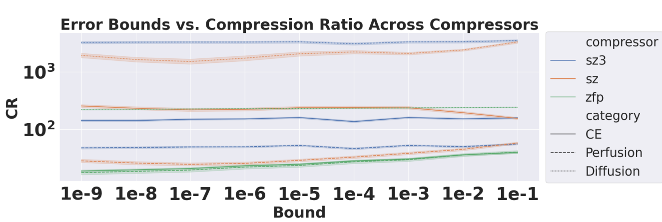


Fig. 6: Error Bounds vs Compression Ratio with Different Lossy Compressors.

Figure 7 evaluates the PSNR as a function of error bound for the same compressors. The PSNR of all the compressors typically decrease as the error margin increases. This suggests that as tolerable error increases, the image quality also decreases.
ZFP compresses without any data distortion in the error bounds of $1e^{-9}$ to $1e^{-2}$. It also has a relatively stable PSNR around 67 in other error bounds, especially in some categories, suggesting that it may be less sensitive to variations in error ranges than SZ3 and SZ, and that ZFP are achieved. `CE` shows higher PSNR at low error bounds, but PSNR decreases significantly as the error bounds increase, especially for SZ3 and SZ. And `Diffusion` show more stable PSNR values across different error bounds, particularly when using ZFP compression.

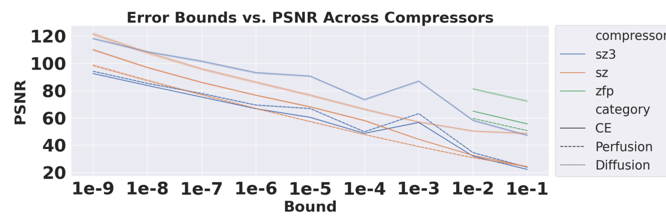


Fig. 7: Error Bounds vs PSNR with Different Lossy Compressors.

Figure 8 and 9 presents the compression and decompression bandwidth for lossy compression algorithms, which is essential for evaluating their efficiency and suitability for handling large medical imaging datasets.

For cBW, ZFP consistently shows the highest median encoding bandwidth 270.86 MB/s across all error bounds, which means it is the most efficient in terms of compression bandwidth. The `CE` and `Perfusion` are similar, but slightly lower median encoding performance than `Diffusion`. SZ3 has higher compression bandwidth than SZ across all error bounds and different categories on DICOM files. SZ3 recieved 49.73 MB/s especially on `Perfusion` which is $12\times$ faster than SZ.

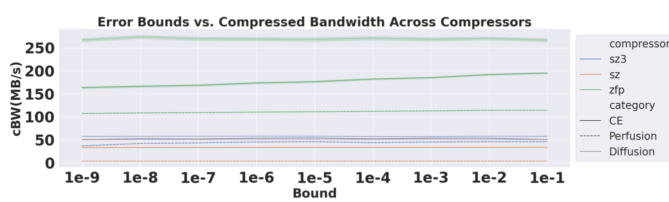


Fig. 8: Lossy Compressors Compression Bandwidth.

For dBW, the decompression bandwidth fluctuates very little as the error bounds increase, especially for ZFP, which shows that ZFP maintains a stable compression speed regardless of the error bounds. ZFP consistently shows the highest decompression bandwidth of 150 MB/s across all error bounds. ZFP's compression bandwidth and decompression bandwidth are higher compared to the other two compressors, indicating ZFP has the fastest files transfer speed. Higher bandwidth values reflect a greater data processing rate, meaning ZFP can compress and decompress data more quickly, facilitating faster file transfer [23]. In all, ZFP has the fastest DICOM file transfer speeds.

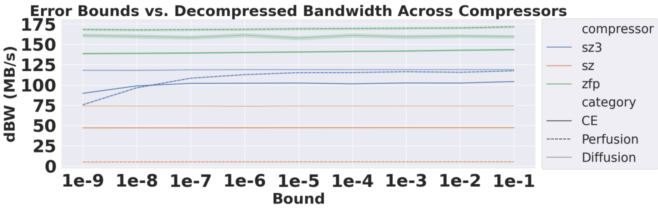


Fig. 9: Lossy Compressors Decompression Bandwidth.

V. Conclusion

As technological innovations progress, the size of DICOM files continues to increase as they have greater fidelity. This requires finding an optimal balance between compression ratio and visual quality using existing compressors. This paper focuses on demonstrating the maximum achievable compression ratio with lossy compression while maintaining acceptable visualization, and analyzing the results of some lossless compressors on DICOM files. Lossless compression continues to be an important method of compressing DICOM files while preserving high data integrity. Unlike lossy compression, which sacrifices some data for a higher compression rate, lossless compression ensures that the original image is perfectly reconstructed. Therefore, while this paper focuses on lossy compression, lossless compression methods are still indispensable in situations where no data loss in acceptable. The analysis reveals that for lossy compression of DICOM files, ZFP excels in terms of speed, making it the optimal choice for scenarios requiring rapid compression and decompression. When the primary objective is file size reduction, SZ3 outperforms other methods by achieving higher CR. For PSNR, which measures image quality retention, ZFP provides the best results, indicating its ability to preserve image fidelity despite a lower CR.
For lossless compression, LZ4 is the fastest compressor. When maximizing size reduction, ZSTD and ZLIB emerge as the best performers. This combination of results offers valuable insights for selecting the most suitable compression method based on the specific requirements of speed, storage efficiency, and image quality preservation.

VI. Acknowledgments

Clemson University is acknowledged for their generous allotment of compute time on the Palmetto Cluster. This work is sponsored by the United States Defense Advanced Research Projects Agency under agreement HR00112320008. The content of the information does not necessarily reflect the position or the policy of the US Government, and no official endorsement should be inferred. This material is based upon work supported by the National Science Foundation under Grant No. SHF-1943114 and SHF-1910197. We would like to thank Dr. Gerald Hefferman (Brigham and Women's Hospital) for providing the DICOM files.

References

[1] W. D. Bidgood Jr, S. C. Horii, F. W. Prior, and D. E. Van Syckle, "Understanding and using dicom, the data interchange standard for biomedical imaging," *Journal of the American Medical Informatics Association*, vol. 4, no. 3, pp. 199–212, 1997.

[2] MiNNOVAA, "Unpacking DICOM File: Understanding the Rich Data It Contain - MiNNOVAA — minnovaa.com," https://minnovaa.com/dicom-file/, [Accessed 19-09-2024].

[3] N. Mansurov, "Jpeg compression levels in photoshop and lightroom," 2018, accessed: 2024-11-09. [Online]. Available: https://photographylife.com/jpeg-compression-levels-in-photoshop-and-lightroom

[4] L. Jiang, S. Parekh, and J. Walrand, "Time-dependent network pricing and bandwidth trading," in *NOMS Workshops 2008-IEEE Network Operations and Management Symposium Workshops*. IEEE, 2008, pp. 193–200.

[5] V. Valentini, S. Gaudino, P. Spagnolo, S. Armenise, T. Tartaglione, and P. Marano, "Diffusion and perfusion mr imaging," *Rays*, vol. 28, no. 1, pp. 29–43, Jan-Mar 2003.

[6] R. Luypaert, S. Boujraf, S. Sourbron, and M. Osteaux, "Diffusion and perfusion mri: Basic physics," *European Journal of Radiology*, vol. 38, no. 1, pp. 19–27, Apr 2001.

[7] postDICOM, "A Crash Course on Handling DICOM Medical Imaging Data — PostDICOM — postdicom.com," https://www.postdicom.com/en/blog/handling-dicom-medical-imaging-data, [Accessed 19-09-2024].

[8] M. Aiello, C. Cavaliere, A. D'Albore, and M. Salvatore, "The challenges of diagnostic imaging in the era of big data," in *Proceedings of the Conference on Clinical Medicine*, 2019, pp. 1–2.

[9] F. Alted, "Blosc, an extremely fast, multi-threaded, meta-compressor library," 2017.

[10] Y. Collet, "Lz4 lossless compression algorithm." [Online]. Available: https://lz4.org/

[11] Y. Collet and M. Kucherawy, "Zstandard compression and the application/zstd media type," Tech. Rep., 2018.

[12] P. Deutsch and J.-L. Gailly, "Zlib compressed data format specification version 3.3," Tech. Rep., 1996.

[13] "GitHub - lz4/lz4: Extremely Fast Compression algorithm — github.com," https://github.com/lz4/lz4, [Accessed 19-08-2024].

[14] X. Liang, S. Di, D. Tao, S. Li, S. Li, H. Guo, Z. Chen, and F. Cappello, "Error-controlled lossy compression optimized for high compression ratios of scientific datasets," in *2018 IEEE International Conference on Big Data (Big Data)*. IEEE, 2018, pp. 438–447.

[15] X. Liang, K. Zhao, S. Di, S. Li, R. Underwood, A. M. Gok, J. Tian, J. Deng, J. C. Calhoun, D. Tao *et al.*, "Sz3: A modular framework for composing prediction-based error-bounded lossy compressors," *IEEE Transactions on Big Data*, vol. 9, no. 2, pp. 485–498, 2022.

[16] P. Lindstrom, "Fixed-rate compressed floating-point arrays," *IEEE transactions on visualization and computer graphics*, vol. 20, no. 12, pp. 2674–2683, 2014.

[17] G. K. Wallace, "The jpeg still picture compression standard," *IEEE Transactions on Consumer Electronics*, vol. 38, no. 1, pp. xviii–xxxiv, Feb 1992.

[18] M. H. Faykus, J. Calhoun, and M. Smith, "Lossy and lossless compression for biofilm optical coherence tomography (oct)," in *Proceedings of the SC'23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis*, 2023, pp. 281–288.

[19] N. Mansurov, "Jpeg compression levels in photoshop and lightroom," 2018, accessed: 2024-11-09. [Online]. Available: https://photographylife.com/jpeg-compression-levels-in-photoshop-and-lightroom

[20] S. G. . S. A. Parah, "Advances in medical image watermarking: a state of the art review," *Multimedia Tools and Applications*, vol. 67, no. 1, p. 1440, May 2023.

[21] R. Underwood, V. Malvoso, J. C. Calhoun, S. Di, and F. Cappello, "Productive and performant generic lossy data compression with libpressio," in *2021 7th International Workshop on Data Analysis and Reduction for Big Scientific Data (DRBSD-7)*. IEEE, 2021, pp. 1–10.

[22] J. Miano, *Compressed Image File Formats: JPEG, PNG, GIF, XBM, BMP*. Addison-Wesley Professional, 1999.

[23] K. Sayood, *Introduction to data compression*. Morgan Kaufmann, 2017.