Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 17144-17160
Received 25 April 2021; Accepted 08 May 2021.

An Efficient Lossless Medical Data Compression using LZW compression for Optimal Cloud Data Storage

[1]A. Phani Sridhar, [2]Dr.P.V.Lakshmi
[1]Department of CSE, Aditya Engineering College (A), Surampalem.
[2]Department of CSE, GITAM (Deemed to be university), Visakhapatnam
[1]Phani.addepalli@aec.edu.in, [2]vpanga@gitam.edu

Abstract

In the current digital era, the medical records of the patients are all processed and stored digitally. With the increasing population, the storage space requirements for the medical data are increasing exponentially. The medical and pharmaceutical companies spend a large amount of money on the storage spaces to store the patient records. Conventional lossy storage techniques cannot be used to store medical data as the data needs to recovered completely while decompression. This paper presents a novel lossless data compression technique which is fast and efficient to store the medical data. LZW compression is a dictionary based lossless compression technique which utilizes the redundancy and repetition in the data to compress it. The compressed data can fully be recovered very quickly whenever needed. The proposed method is compared with other lossless compression techniques. The proposed method is implemented using python and HADOOP is used to store the compressed data.

Keywords: Medical data, HADOOP, LZW compression, RLE, Huffman Coding, Compression and Decompression, Lossless, DTCT.

1. Introduction

The supporting medical services that utilize telecommunications are described by telemedicine. The "tele" prefix indicates the distance and is originated from ancient Greek language. The medical services over a certain distance are provided by telemedicine. The delivery of medical information between pairs of receiver and transmitter is included in the telecommunications utilization in medical applications [1]. To refine the obtained information from a human body, medical data can be provided as simple as a consultation of a doctor.

The delivery of required medical advice over long distances specifically in rural areas is ensured by telemedicine through the telecommunication usage economically. To provide the consultation, diagnosis, medical care, and treatment in addition to the medical information and education transmission, applications of telemedicine use telecommunication systems and multimedia equipment. The physicians who utilize store-and-forward consultations and interactive videos during the treatment of patients include in the telemedicine normally.

Based on the monitors and modified gadgets specifically, the direct communication with remote patients is made possible with medical specialists through the interactive videos. The physicians who can send X-rays, pictures, and patient information straight to the specialist's computer are involved in the store-and-forward methods. The received analysed information can send by the specialist to the local doctor who treats patients and ensures the availability of follow-up care [2].

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 17144-17160
Received 25 April 2021; Accepted 08 May 2021.

With the observation of a medical doctor, most of the testing is carried out in the countries as healthcare involves laboratory testing. Here, the testing is made for chronic diseases' therapy monitoring, prophylaxis, and risk analyzation. By sending the specimen of patients to the medical laboratory from the attending physician, this testing will perform in most cases [3].

The testing called as "point-of-care testing" (POCT) in which glucose self-testing is considered initiating based on the non-invasive urine tests instead of testing in interstitial fluid and from fingerprints. It was the frequent application utilized for humans themselves without the help of a physician for laboratory testing which is an open procedure whether the patient or health-care professionals can process. Here, self-testing doesn't involve the healthy subjects. The adoption of term DTCT (direct-to-consumer testing) was done in [4, 5]. For laboratory testing, it was the most frequent application for humans without the assistance of a physician. This testing is an open process whether it can perform by healthcare professionals or the patient himself and self-testing couldn't include in healthy subjects. In [4, 5], the term of DTCT (direct-to-consumer testing) was familiarized. The testing place either near to the consumer or patient doesn't demonstrate by DTCT that helps to share POCT features or central laboratories have been used to send the specimen (saliva, whole blood, and obtained cells by a mouth wash) through an email. Based on the internet and the measured data with some long-term storage using apps, a transmission of results includes in all of this DTCT. In the preanaytical processes like sample collection and test selection, whereas in the post-analytical processes such as counselling and data interpretation, the lack of healthcare professionals is the salient difference of DTCT by comparing with conventional testing based on the results.

Due to diversity of features, DTCT utilizes instead of real laboratory testing. 23 and Me [6] was offered a test with an illustrative example that helps to investigate the personal ancestry subsuming Neanderthalian heritage. The commercial interests may trigger other DTCT. For example, the tests have designed for cytochrome p450 polymorphism as a gift for depressive relatives [7]. Owing to the fear of discrimination, others may consider DTCT. For example, testing of sexually transmitted disease (STD) or human immunodeficiency virus (HIV) [8] is done due to the easier utilization like testing of lactate in the activities of sports or ensure the lower costs than the conventional process of testing [9].

In DTCT, apps and fitness trackers are the complementary for biochemical tests and they are available for reasonable costs. The storage and recording of diverse physiological data including blood pressure, heart rate, body weight, or physical activity is allowed by the devices known as "wearable" (for "wearable computers"). In a diabetes diary app [3], the attempts have been made in a way that the numerical biochemical and physiological data is integrated. By using biochemical tests or wearables, numerical results are obtained. These tests are calculate the indices (usually proprietary) for personal fitness automatically or assess the risk in adverse medical conditions [10].

The DTCT test results will be utilized for healthcare services due to the uncertain differentiation between POCT, healthcare testing, and DTCT. The addressing of challenges in long-term data storage challenges should also be done for the information of DTCT. For

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 17144-17160
Received 25 April 2021; Accepted 08 May 2021.

obtaining the medical data with long-term storage, the European Commission is put strong efforts currently in the “Horizon 2020 program” either for attending physicians, the benefit of patients, the costs reduction or allowing scientific investigations [11]. For little regulation and variant uses of DTCT data, data security, quality assurance, or the data standardization, various contexts have been used collaboratively to characterize the current situation.

2. Lossless data Compression Techniques

This section presents the description for lossless compress techniques namely Huffman coding and run length coding.

2.1 Huffman coding

Huffman encoding is a statistical data compression method that allows reduce the encoding length of an alphabet. The Huffman code (1952) is a length code optimal variable, that is to say such that the average length of a coded text is minimal. We observe thus size reductions of the order of 20 to 90%.

2.1.1 The principle

The recording of encountered bytes in a set source data with the values of variable binary length is the principle of the Huffman algorithm. The processing unit decreases to the bit. For the purpose of recoding of data, Huffman proposes that includes a very low occurrence over a binary length greater than the average, and the data recodes frequently over a very short binary length.

So, for sparse data, we lose some regained bits for repetitive data. Through example, in an ASCII file the "w" appearing 10 times will have a very long code: 0101000001000.Here the loss is 40 bits (10 x 4 bits), because without compression, it would be encoded on 8 bits instead of 12. On the other hand, the most frequent character like the "e" with 200 appearances will be coded by 1. The gain will be 1400 bits (7 x 200 bits). We understand the interest of such a method.

In addition, the Huffman encoding has a prefix property: a binary sequence can never be both representative of a coded element and constituting the beginning of the code of another element. If the binary combination 100 represents a character, the combination of 10001 can’t be the code for any other data. 5 bits would interpret by the decoding algorithm such as a succession of coded character 100 then it is combined into the coded character of 01. This characteristic Huffman coding allows coding using a binary tree structure.

2.1.2 The Huffman compression algorithm

This algorithm is "non-destructive", that is to say that it compresses the data without introducing any loss information, so that the uncompressed file is a true copy of the original. The coding used is said to be "entropic", that is to say that it is based on the statistics of

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 17144-17160
Received 25 April 2021; Accepted 08 May 2021.

appearance of different bytes of the file, and "of variable size", that is to say that each byte is coded according to a sequence of bits, the size of which differs depending on the byte.

Based on a principle of storing information in binary, this presentation is started for understanding how a file compression allows by an encoding of "entropy".

Binary encoding convention

Based on a sequence of 8 binary digits ("bits" in English, equal to 0 or 1), an integer between 0 and 255 is coded, also termed as byte. A priori, it suffices for this to give yourself a correspondence table like:

Table 1: Priority in binary encoding convention

…	…
138	10001010
139	10001011
…	…

As long as each integer between 0 and 255, the correspondence can be any and is assigned a binary code. Consider it as convention after a match is fixed. The convention was selected is that of the representation in base 2 practically and it has the merit to be logical and natural. The base 2 representation of 138 is 10001010 in the above table. This is adopted as the process of noting the numbers easily. The basic unit of data storage is the defined byte based on this convention. A series of bytes which are arranged in a well-defined order is a computing of any file. The constituted number of bytes is simply the size of a file. 1024 bytes (and not 1000) corresponds to the kilobyte (KB) and 1024 * 1024 bytes to megabyte (MB).

Fixed size, variable size coding

Based on ANOTHER way of coding the digits but less logical but more judicious, the problem of "entropic" data compression is made. So, the new convention utilizes to recode the same file size that would be LESS LARGE. In long texts, letters do not appear with the same frequency. These frequencies vary depending on the language used. In French, the most common letters are in order:

ESAINTRULODCPMVQGFHBX JYZKW

with the frequencies (often close and depending on the sample used):

Table 2: Byte content of the WordPad

E	S	A	I	N	T	R	U	L	O	D
14.69%	8.01%	7.54%	7.18%	6.89%	6.88%	6.49%	6.12%	5.63%	5.29%	3.66%

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 17144-17160
Received 25 April 2021; Accepted 08 May 2021.

A byte content of the Wordpad.exe file with an analysis is considered as an example. The possible values of a given byte (0… 255) are provided by both the ordinate and the abscissa in this table. The number of bytes in the file is proportional to the circle size with this value on the abscissa, on the diagonal and corresponding ordinate.

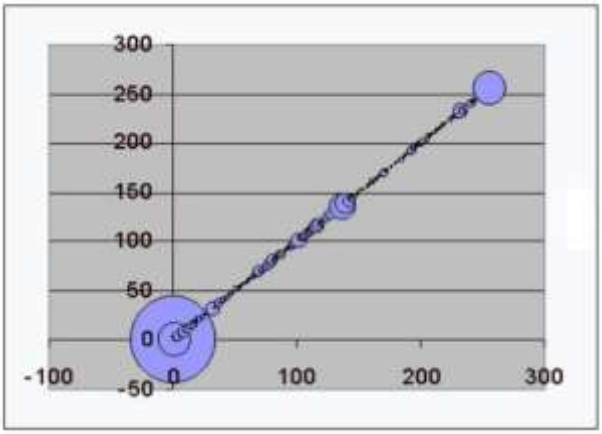


Figure 1: The frequency of number of bytes represented in circular size

Table 3: Frequent values used code word

Value	Number	Frequency
0	71891	34.4%
2	1119	0.53%
128	1968	0.94%
130	79	0.038%
255	10422	4.99%

The values 0, 128 and 255 can view clearly and are much more frequent than the others!

The general idea of entropy coding is as follows:

Based on a coding convention of "variable size", the decision will take and it represents a frequent value BY A SMALL NUMBER OF BITS, and an infrequent value with a large number of bits. The sequence "11" (formerly "00000000") will represent with 0 and "1011010" (formerly "10000000") represents with 128, and "0100" (formerly "11111111") indicates 255, etc. The considerable space saves by coding on two bits rather than eight since 0 represents one third of the file and similar process is for other frequent values. From a series of values with frequency table, a variable-size prefix code generates by Huffman's algorithm.

Binary trees

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 17144-17160
Received 25 April 2021; Accepted 08 May 2021.

We call a binary tree a collection of elements (the "nodes") in which each node is connected to 0 or 2 other nodes. With each node we can associate one or more values. Here is an example of a binary tree:

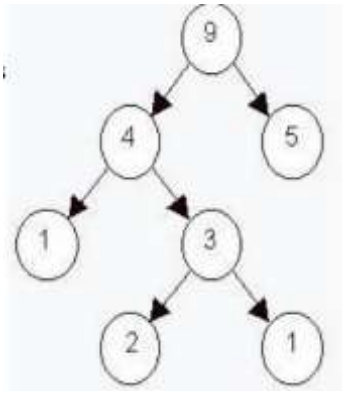


Figure 2: Binary tree

The top element of the tree (the "9") is the root; while the terminal nodes are the leaves.

2.1.3 Implementation of the Huffman algorithm

A file to compress first, draw up the table of frequencies (Key / Value) below against. In this table, Frequency (i) denotes the number of bytes of the file with the value i.

Table 4: Key Vs Value in Huffman coding

Key	Value
0	Frequency (0)
1	Frequency (1)
…	
255	Frequency (255)

For each Key associated with a non-zero frequency, create a terminal node, and assign it the couple (Key, Value). Each terminal node created represents as many binary trees to element unique.

Iterate:

1. Classify each binary tree by increasing value;

2. Remove from the list the two binary trees with the lowest values. Insert, at the place, a new binary tree whose root points to the roots of the two binary trees deleted. To this new binary tree, assign for Value, the sum of the Values of two previous trees. No need to assign him a Key.

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 17144-17160
Received 25 April 2021; Accepted 08 May 2021.

3. Repeat at 1 until there is only one single binary tree left.

We thus build a binary tree in which the terminal nodes are the elements of our table frequencies. To find out the Huffman Code associated with each Key, simply go down the tree, starting from the root, to reach the desired Key. At each junction, we count "0" if you have gone to the left, and "1" if you have gone to the right.

3. RLE (Run Length Encoding)

As early as 1967, the analog television signals transmit using the schemes of Run Length Coding (RLE). Hitachi was received the patent rights of track length coding in 1983. For palette-based bitmaps like computer icons, RLE is well-suited specifically. In prior to the advent of more sophisticated formats such as GIF, it was a popular method of image compression on early online services like CompuServe. Although JPEG utilizes on the coefficients which remain after the picture blocks quantization and transformation, it doesn't carry out well on continuous tone images such as photographs. For run-length encoded data, common formats have been included ILBM, PCX, Pack Bits, and True vision TGA. To encode print length color for facsimile machines called as T.45, a standard describes by the International Telecommunication Union. In modified Huffman encoding, the standard integrates with other techniques and is efficient as most faxed documents are white space typically.

Run Length Encoding (RLE) or Run Length Coding (RLC) is a lossless compression where the coded digits, characters or letters have a different number of bits. Since the length of the coded characters is different, the coding is called run-length coding.

In run-length coding, a sequence of identical numbers, characters or letters is replaced by a single symbol and an indication of the number of identical symbols. In this context one speaks of a run, which is a sequence of identical characters and the length of the respective sequence, which is stored in a run counter. The run length coding eliminates redundancies in the form of repetitions and is particularly suitable for graphics and image files with few colors and large areas of color. The more detailed the color image, the less suitable it is for RLE coding. In computer graphics, the RLE method is used for memory-intensive Raster graphics used and are most efficient when the graphics are simple with few colors and large areas of color. To increase the efficiency of the method, the Algorithm can optimize the route by choosing the most effective route, which can be line-sequential, zigzag or meandering.

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 17144-17160
Received 25 April 2021; Accepted 08 May 2021.

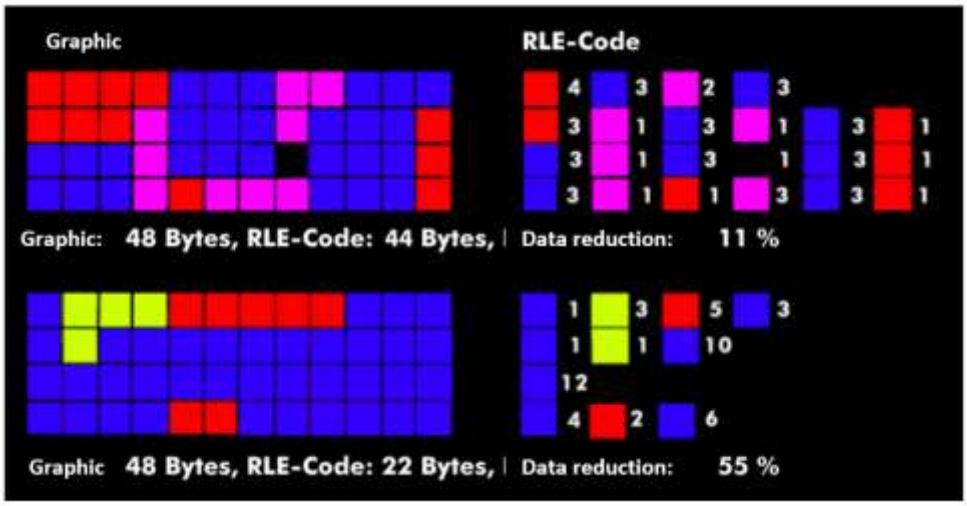


Figure 3: RLE compression Examples with lower and higher efficiency

With Run Length Encoding, for example, the sequence of numbers aaaaaa is replaced by 2 bytes. : one byte for the letter "a", the other for the number 6. If the letters, digits and characters are in binary form, the run-length coding can work even more efficiently.

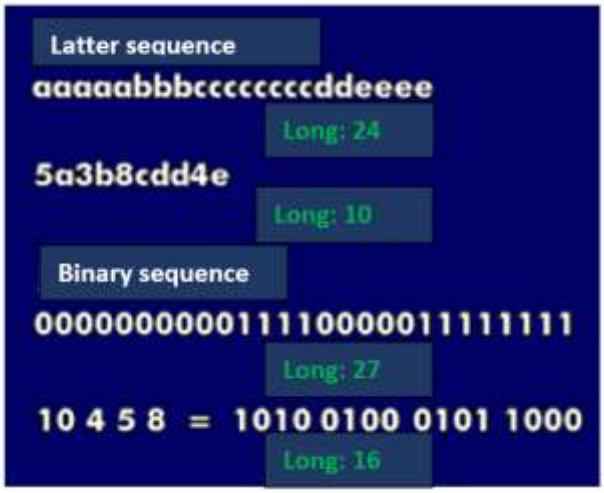


Figure 4: Run length coding using letters and binary data as an example

Run-length coding is characterized by its simplicity and speed. With compression, the same values are read in until the value changes. The value and the number of the same values are recorded. During decompression, only the value is read out and the corresponding number of bytes output.

Run length coding is used for various graphic file format such as the Tagged Image file Format (TIFF), the bitmap and TGA file format, but also for fax transmissions. Another method with variable length is Variable Length Coding (VLC).

4. Proposed LZW compression with HADOOP

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 17144-17160
Received 25 April 2021; Accepted 08 May 2021.

The LZW algorithm was proposed by Welch in 1984, and is an improvement of an algorithm originally proposed by Lempel and Ziv (LZ78). The general idea of the algorithms of this family is to take advantage of repeated patterns in the file to be compressed. Such a pattern is assigned a code particular. Subsequently, any new appearance of this reason is replaced by the corresponding code (normally chosen to be shorter than the pattern), which can lead to substantial space savings. We then end up with a dictionary between patterns and codes.

The dictionary is created dynamically and it is not useful to store it at the beginning of the file compress. Reading the stream of codes corresponding to the compressed file makes it possible to reconstitute as and measures the dictionary used during compression.

At a practical level, we encounter LZW compression in GIF images, in the ancient format Unix compress / uncompress, as well as possible compression of some old PDFs. In the United States, a software patent by Unisys on LZW has long been problematic for developers wanting to use this algorithm. This patent expired a few years ago.

To finish the overview of compression methods, there are more recent algorithms providing better compression rates, often at the cost of higher computational costs. We can cite in particular bzip2, based on more complex mathematics, namely the transform of Burrows-Wheeler.

4.1 The compression

Compression is done using the following elements:

- the string of "characters" to compress, which will be traversed linearly,
- a buffer in which characters awaiting coding can be placed,
- adictionary associating character strings with numerical codes. Initially, thisdictionary associates its ASCII code with each character.

The progress of the compression can then be summed up in two formulas: "as long as we win, we play again" and

"never make the same mistake twice".

More precisely:

- As long as the buffer contains a word found in the dictionary, we try to lengthen this wordusing characters taken from the string to compress,
- When the character read in the string forms with the buffer a word unknown in the dictionary,we send the code corresponding to the word of the current buffer, we enrich the dictionary with an entryfor the unknown word, in case it occurs again, and finally we start again with an empty buffercontaining just the last character read.

Table 5 shows how the algorithm works on the word "repetition":

Table 5: Word compression in LZW compression

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 17144-17160
Received 25 April 2021; Accepted 08 May 2021.

Step	Buffer	Red character	Code issued	Add to dictionary
0		r		
1	r	e	114 (r)	re: 256
2	e	p	101(e)	ep: 257
3	p	e	112(p)	pe: 258
4	e	t	101(e)	and: 259
5	t	i	116(t)	ti: 260
6	i	t	105(i)	it: 261
7	t	i		
8	ti	o	260(ti)	tio: 262
9	o	not	111(o)	on: 263
10	not	EOF	110(n)	

The size of the dictionary is an important point to manage. There are two possible approaches:

- Or its size is limited to k codes (for example, 1024). The codes will therefore be on (log k) bits(10 bits). If the dictionary is full, you must either keep it or use it (it will not change any more) or empty it. If the dictionary is emptied, you must indicate (for decompression) in the file compressed that it was emptied there.
- Either its size is variable. In this case, when the codes grow, it is necessary to indicate (for the decompression) in the compressed file that the following codes use an extra bit.

To indicate that the dictionary has been emptied or that the codes are on an additional bit, we use a control code. Usually the control codes are 256 to 259. The reason codes start at 260.

4.2 Decompression

The objective is now reversed: we receive a list of codes, and it is a question of finding the chainoriginal. Again, we have a dictionary, initially filled with the characters and their ASCII codes.Let's try on the coding [114; 101; 112; 101; 116; 105; 260; 111; 110] of the word "repetition":

- it is easy to convince yourself that the first code is necessarily that of a character, which is goodpresent in our dictionary. Here, 114 = r. Moreover, when this 114 was

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 17144-17160
Received 25 April 2021; Accepted 08 May 2021.

issued, the dictionary was enriched with an entry (r?: 256). Unfortunately, behind the "?" hides a character that we do not cannot determine for now.

- 101 = e is in our dictionary and does not involve the unknown entry number 256. We are besides now able to know what is hidden in 256: it is "e" which formed with "r" the word unknown "re" when creating entry 256. We are now able to complete this entry with a delay time: (re: 256). During this time, an entry (e?: 257) had place on the compression side, again with a character that is unknown to us at the moment.
- And so on: 112 = p, and we now know that the dictionary contains the entry (ep: 257), but also the partially unknown entry (p?: 258).

Let's summarize (here we use a "prev" variable to store the word decoded in the previous step):

Table 6: Word decompression in LZW compression

Step	Code read	Pre	Decoded word	Certain addition to dictionary
0	144		r	
1	101	r	e	re: 256
2	112	e	p	ep: 257
3	101	p	e	pe: 258
4	116	e	t	and: 259
5	105	t	i	ti: 260
6	260	i	ti	it: 261
7	111	ti	o	tio: 262
8	110	o	not	on: 263

5. Experimental Results

The technique of lossless LZW compression uses for reducing the storage space or distributed HADOOP storage complexity in this project. Based on the experiments, the results have been showed that the data storage can save up to 40% space of HADOOP storage using the LZW compression. The experiments are conducted on two types of data, text and images. The text data corresponds to the test results in the form of reports that are stored in the hospital for every patient. The image data corresponds to the scans that the patients take in the form of CT scans, Ultrasounds, xrays etc.

5.1 Lossless compression of Text data

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 17144-17160
Received 25 April 2021; Accepted 08 May 2021.

For the experiments, a medical dataset from UCI Machine learning repository is used. The dataset consists of 13 columns depictingthe parameters and one column corresponding to the labels. The parameters include age, sex, cp, trestbps, chol, fbs, restecg, thalach, exang, oldpeak, slope, ca, thal and target. The total entries in the dataset are 1025. The dataset file heart.csv size is 10.7 KB and the compress version of same file is uploaded to HADOOP. Table 7 shows the Hadoop server parameters running on a local server. The cluster summary is shown in table 8.

Table 7: Hadoop server parameters running on a local server

Started:	Wed Dec 09 15:54:13 IST2020
Version:	2.0.6, Unknown
Compiled:	2017-03-28T17:01Z by user from unknown
Cluster ID:	CID-35ae4611-3ac2-417b-ae57-2a764d68d319
Block Pool ID:	BP-1928167768-172.23.81.17-16075094444656

Heap Memory used 82.25 MB is 58% of Commited Heap Memory 141 MB. Max Hap Memory is 889 MB.

Non Heap Memory used 35.15 MB is 98% of Commited Non Heap Memory 35.84 MB Max Non Heap Memory is -1 B.

Table 8: Cluster Summary

Configured Capacity	:	243.60GB			
DFS Used	:	136 B			
Non DFS Used	:	180.12 GB			
DFS Remaining	:	63.48 GB			
DFS Used%	:	0.00%			
DFS Remanding%	:	26.06%			
Block Pool Used	:	136 B			
Block Pool Used%	:	0.00 %			
DataNodes Usages	:	Min %	Median %	Max %	Stdev %
	:	0.00%	0.00%	0.00%	0.00%
Live Nodes	:	1(Decommissioned: 0)			

Figure 5 shows the screenshot of the files uploaded in the Hadoop server. The original heart.csv file has a size of 10.76KB and the compressed file has a size of 7.32 KB.

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 17144-17160
Received 25 April 2021; Accepted 08 May 2021.

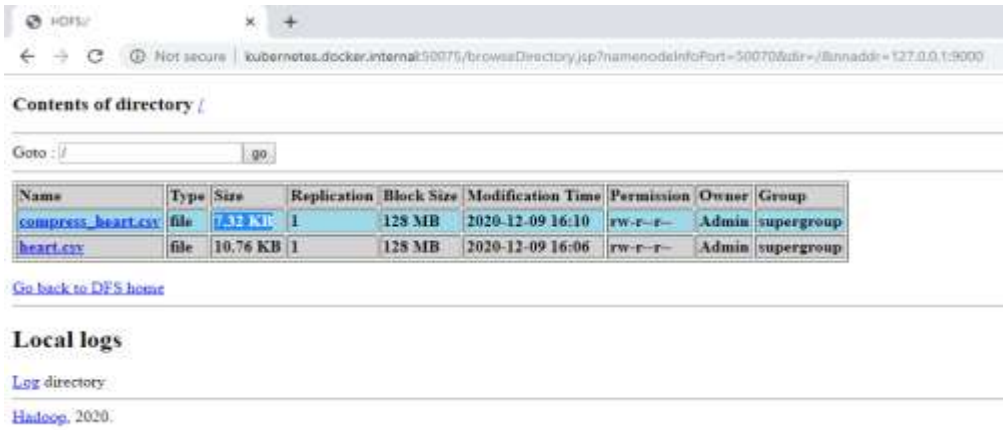


Figure 5: sizes of original data and compressed data

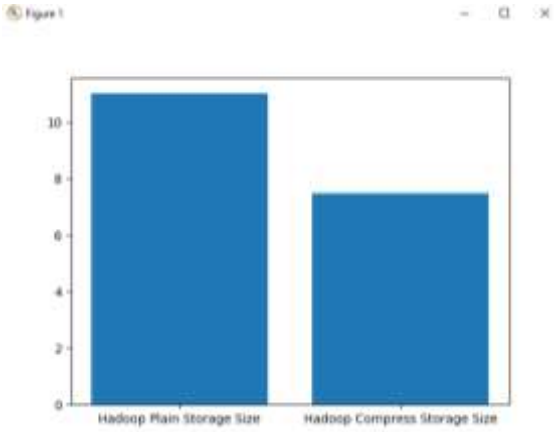


Figure 6: Comparison of original and compressed data

The technique name represents with x-graph and storage size indicates by y-axis in above graph. The compression storage space of nearly 40% can save with the proposed method.

5.2 Lossless compression of Image data

The image data consists of medical scans like CT scans, X ray scans etc. The proposed algorithm in implemented on several types of medical images. The pixel values from the image are extracted and LZW compression algorithm is implemented on it. The following example depicts the implementation of LZW compression on a chest X-ray.

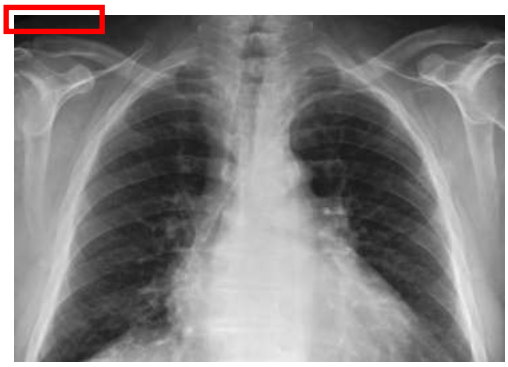


Figure 7: Chest X ray image

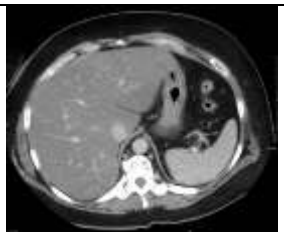
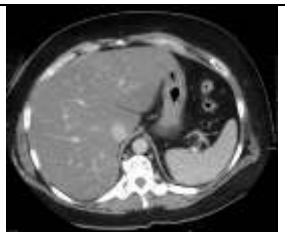
Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 17144-17160
Received 25 April 2021; Accepted 08 May 2021.

The pixel values of the highlighted part in the figure 7 are show in table 9. The first row shows the input image pixel values. The second row shows the encoded values by LZW algorithm. The number of input pixel values in 11 and the number of output encoded code words is 7. As the number 11 is repeated multiple times, the algorithm replaced the repeated pixel values with encoded values thereby reducing the size of the data and compressing it.

Table 9: Sample pixels input data and encoded data

Input Data	12	11	9	10	11	11	11	11	11	11	11
Encoded Data	13	12	10	11	12	261	262				

Table 10: Lossless compression analysis on medical scans

Input Medical Image	**Image Dimensions**	**Original Size**	**Compressed Size**	**Decompressed Image**
CT Scans				
	401*480	1,92,480	1,23,252	
MRI Scan				
	500*534	2,67,000	1,41,150	
X-ray				
	352*498	1,75,296	1,23,980	
Ultrasound Scanning				
	735*976	7,17,360	4,07,459	
Fluoroscopy				

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 17144-17160
Received 25 April 2021; Accepted 08 May 2021.

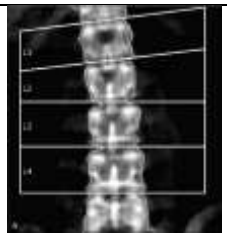
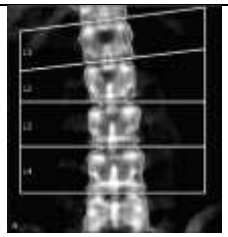
	494*673	3,32,462	2,21,590	
Bone densitometry				
	717*677	4,85,409	1,45,062	

Table 10 results of applying lossless compression on medical scans. The table shows the input image, the dimension of the image, original size, compressed size and the decompressed image. LZW compression being lossless, restores the image completely after decompression.

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

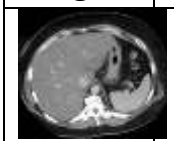
Where I_1 and I_2 are the two images of size MxN.

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

MAX indicates the maximum pixel value in the image.

The Peak Signal to Noise Ratio (PNSR) values of all the decompressed images is infinity and the Mean Square Error (MSE) is 0. The compressed files can be easily stored in the database with much lesser size compared to the original data.

Table 11: Comparative analysis

Input Medical Image	**Original Size**	**Run-length encoding**		**Huffman coding**		**LZW compression**	
		Compressed Size	**Compression ratio**	**Compressed Size**	**Compression ratio**	**Compressed Size**	**Compression ratio**
	1,92,480	3,18,176	0.604948	1,61,931	1.188654	1,23,252	1.561679
	2,67,000	3,72,582	0.716621	2,20,671	1.209946	1,41,150	1.891605

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 17144-17160
Received 25 April 2021; Accepted 08 May 2021.

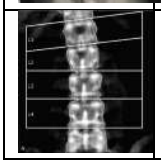
	1,75,296	3,01,008	0.582363	1,68,930	1.037684	1,23,980	1.413905
	7,17,360	10,25,902	0.699248	5,66,810	1.265609	4,07,459	1.76057
	3,32,462	4,43,734	0.749237	2,90,145	1.145848	2,21,590	1.500347
	4,85,409	3,11,242	1.559587	2,95,003	1.559587	1,45,062	3.346217

Table 11 shows the comparison of proposed LZW compression with RLE and Huffman coding.

$$Compression\ ratio\ (CR) = \frac{Size_{uncompressed}}{Size_{compressed}}$$

Greater the compression ratio, better the compression. The compression ratio indicates that the proposed method performs better when compared to the other techniques.

6. Conclusion

Compression of medical data before storage is very important to save storage space. The stored data can be accessed online with very less bandwidth. The proposed method would be very usefull in developing telemedicine applications. LZW compression used in the proposed method compressed the medical data better than run length coding and huffman coding. The experimental results show that the proposed method compressed outperformed the existing compression methods.

References:

[1]. Fong, B., A.C.M. Fong, and C. K. Li, telemedicine technologies: Information technologies in medicine and telehealth. 2011: John Wiley & Sons.
[2]. Algaet, M.A., et., Telemedicine and its application in telemedicine management. 2014, outskirts Press, Inc.
[3]. Orth M, Averina M, Chatzipanagiotou S, Faure G, Haushofer A, Kusec V, et al. Opinion: redefining the role of the physician in laboratory medicine in the context of emerging technologies, personalised medicine and patient autonomy (‘4P medicine’). J Clin Pathol 2017. [Epub ahead of print].
[4]. Orth M. Direct-to-consumer testing: the business with lifestyle tests. Point Care 2017; 16:124–7.
[5]. Orth M, Luppa PB. Direct-to-consumer-testing: Fluch oder Segen für die Patienten? Dtsch Arztebl Int 2015; 112: A-174.
[6]. Ramos E, Weissman SM. The dawn of consumer-directed testing. Am J Med Genet C Semin Med Genet 2018; 178:89–97.
[7]. Annes JP, Giovanni MA, Murray MF. Risks of presymptomatic direct-to-consumer genetic testing. N Engl J Med 2010;363: 1100–1.
[8]. Myers JE, El-Sadr Davis OY, Weinstein ER, Remch M, Edelstein A, Khawja A, et al. Availability, accessibility, and price of rapid HIV self-tests, New York City Pharmacies, Summer 2013. AIDS Behav 2017; 21:515–24.

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 17144-17160
Received 25 April 2021; Accepted 08 May 2021.

[9]. Fiala C, Diamandis EP. The meteoric rise and dramatic fall of Theranos: lessons learned for the diagnostic industry. Clin Chem Lab Med 2018. [Epub ahead of print].
[10]. Tobon DP, Jayaraman S, Falk TH. Spectro-temporal electrocardiogram analysis for noise-robust heart rate and heart rate variability measurement. IEEE J Transl Eng Health Med 2017; 5:1900611.
[11]. Reichel J. Oversight of EU medical data transfers – an administrative law perspective on cross-border biomedical research administration. Health Technol 2017; 7:389–400.