International Journal of Aquatic Science
ISSN: 2008-8019
Vol 12, Issue 03, 2021

A Review On Various Lossless Data Compression Technique For Machine Learning And Iot Data

Meenakshi Dhanalakshmi[1],Divya.P[2],D.Palanivel Rajan[3]

[1,2]*Assistant Professor ,Dept of CSE, Bannari Amman Institute of Technology, Sathyamangalam, India*
[3] *Professor, Dept of CSE, CMR Engineering College, Telangana, Hyderabad, Tndia.*

Email: [1]meenakshidhanalakshmi@bitsathy.ac.in,[2]divyap@bitsathy.ac.in, [3]palanivelrajan.d@gmail.com

Abstract:
Compression is the most important technique during the data transmission from one place to another place. Using data compression, the volume of a file can be reduced which will help to decrease the need of new hardware, improve database performance, speed up backups, Provide more secure storage. Compression has two different types which classified as either lossy or lossless. Lossless compression methodology compresses the data to be transferred without any missing in original data. Using this compression the information should not get changed at the place of destination. For example, many sensor parameters can be sensed using sensors placed in various places, which data should be collected and should reach the server without any data loss. In machine learning domain, many data are collected in day by day manner these data should be communicated without any data loss. These kinds of methodology can be used for the secure communication while processing the data. There are many lossless data compression algorithms are available for us to performing the data compression techniques like Huffman coding, Run length Encoding techniques, etc., In this paper we are going to discuss about how data compression techniques will take exciting role in era of rich data used in Machine learning, IoT and so on. We are going to compare algorithms based on energy, performance, encryption and decryption during compression which algorithm will produce better result for these kinds of techniques.

Key words: Huffman coding, Run Length Encoding, Arithmetic Encoding and Dictionary Based Encoding are available

1. INTRODUCTION

In modern days everywhere and everything is data but the secure data communication to the target place is the question mark. For this the data compression methodology is used for secure data communication and also without any reduction in the data. IoT, Machine learning are the domain uses data apparently for various process like communication, analysis, mining, data processing, image processing, evaluation etc., For this protected communication, data compression methodology is used.

Data compression saves storage space, accelerates the transfer of files and reduces storage and bandwidth costs. This technique is extremely useful when transferring the great number of data that are large in size. As data compression is used in an application to transfer data, speed is the key objective. Transmission speed is based on the number of bits sent, the time taken for encode the sending message and decode the receiving message of the original data sent. In a storage application of the data, the degree of compression is the main concern. The compression techniques are categorized as lossy or lossless [1]. Lossless compression as shown in Fig.1 is used to rebuilds to the exact data after the compression of data without any loss. These methods are used in the application like storing health records, text, and photos for legal purposes, computer- executable file.

Fig.1. Lossless Compression

Lossy compression as in Fig.2 is also to rebuilds the data with data loss, and it is considered as irreversible compression. In this technique, the decompression method maybe results in an estimated to rebuild. These techniques are used for the application like multimedia images, video, and audio files to achieve more compact during data compression.

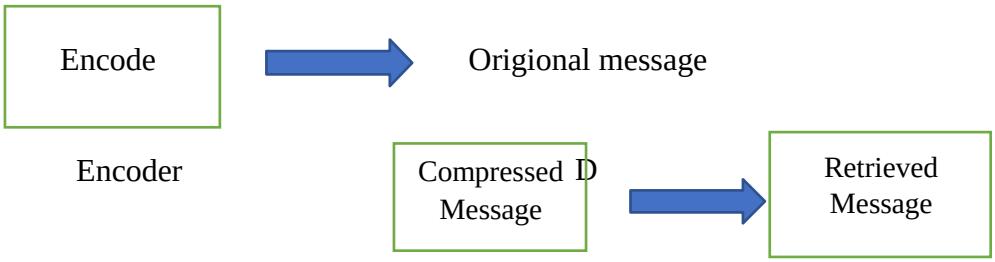

Fig.2. Lossy Compression

Various lossless data compression algorithms are in practice. Some lossless techniques are like Huffman Coding, Run Length Encoding, and Arithmetic Encoding,in this paper, we are going to discuss three compression techniques how they are going to support the data in recent technologies like Machine Learning, IoT.Based on compressing a text data the performance is evaluated and compared.

Huffman Coding

This is the one of the methods of lossless data compression. This technique is used to allot the variable- length codes to the input characters, in which the length is depends on character frequencies. The most frequent character gets the least code and the least frequent character gets the biggest code. Some of the input characters may just need 2 or 3 bits, while other characters which need 7, 10 or 12 bits.
The Huffman coding is classified as Static Huffman code and Dynamic Huffman coding. In this Static coding, the frequencies assigned to the input does not changed during compression, in dynamic huffman source and destination part of the code can be encoded in

real time at the time of compression.
There are number of real-worl applications. ZIP is the most commonly used technique for compression which is based on Huffman Encoding[2]. Brotli Compression[3], the latest of the lossless compression algorithms established last month by Google also uses Huffman Coding. In addition, Let's take one example of how Huffman code functions with fixed duration and variable length
Let us consider the message
x= deer
In the above example we have only three characters d, e, r for encoding. If we do a fixed-length encoding, we need at least two bits for each character [9]. Therefore, we need 8-bits to represent. Suppose same example we are representing in Variable length encoding we decide to use the following bits for variable
d=0, e=11, r=10 0111110
it takes only 6 bits which is less when compared to fixed-length encoding. The reason we choose these values for d, e, r is because of the ambiguity faced by variable length encoding while decoding the text. If you decode 0111110, we get the original message as deer. Suppose we take d=0, e=01, r=00 now we get 0010100 if we decode this there is a possibility of getting dd or r Therefore, proper care should be taken while working with variable length coding[4][11].
Let us consider another example for constructing of Huffman tree as shown in fig 2.1 Huffman algorithm followsoptimal merge pattern that can generate own code[15]. The character value will be added in increasing order.
message = MALAYALAM

Y	M	L	A
1	2	2	4

Fig.3 Huffman Table

Distance calculation = Σfi+di
=1*2+2*2+2*2+4*2 = 18 bits
By using this tree, we can set a code for the character in a message

Table 1.Code Table

Character	**Count**	**Code**	
M	2	01	2*2=4
A	4	11	2*4=8
L	2	10	2*2=4
Y	1	00	1*2=2
4*8=32	9	8	18

Now the size of the message is 18 bitsas shown on Table 1. with this message we need to send the code table or tree for the decoding purpose.
Original message size 9*8-72 bits.

1.1 After encoding

Now Message size is 18 bits (Total character + size of code) 4*8=32 bit + 8 bit = 40 bits

Now encode message size is
message + code table =18+40=58 bits. Decoding will be done by using the code table in Huffman code Algorithm.
Let us compare the various techniques as shown in Table 2 how it's useful for data compression in IoT, Machine Learning.

Table 2. comparison of various implementations using Huffman Coding

Paper Title	**Issues**	**Advantages**
Efficient data Compression for IoT Devices using Huffman Coding Based Technique [2]	The compression process takes considerable energy and resources during network data transmission.	i) Graphs are used for compression for finding the particular patterns and replace the pattern with identifiers which are of variable in length, ii) The space requirements are minimized by compressing the adjacency matrix that is used in the graph.
An energy efficient IoT data compression approach for edge machine learning [7]	i) In cloud it needs high energy consumption for data transmission. ii) Data to be processed and replied should be in a very short time.	i) Edge computing is used to discharge the workload directly from the cloud at the near place of the source location. ii) Prior to the transmission the data must be compressed which consumes of the more energy with the help of IOT devices. iii) Using machine learning techniques the transmitted data are rebuild at the edge node on the cloud. iv) The data collected using Wireless body sensor Networks for the process of compression.
IoT Data Compression: Sensor-agnostic Approach [3]	Bulk data storage in the tiny sensors and data transmission cause more energy consumption.	i) SensCompr methodology is used to extract the embedded useful information properties from the sensor parameter to improve the compression gain while optimizing the loss in information. ii) It is implemented for the independent sensor information.

Run Length Encoding:
Run-Length encoding (RLE) is a lossless compression methodology in which the sequence of data elements can be used consecutively and can be stored as a single data value. This methodology is applied in text, animations, and drawings.

1.2 *Example :1*
Before Encoding Total = 14 bits
After Applying RLE (Run Length Encoding)

X	0	3	Y	0	3	Z	0	4	C	0	4

X	X	X	Y	Y	Y	Z	Z	Z	Z	C	C	C	C

Total =12 bits

A, R, N, D are data value, A=03 is count of data value

The encoding of run length is particularly applied when there is the need for compressing images and changing images. It lies behind many of the techniques used to extract information from images. There may be potential drawbacks to this method:[8]

- This compression is only efficient with the files which contain lots of repetitive data.
- Computer generated images are also not more suitable for the RLE encoding.
- The size of the consecutive characters increased from 3 to 4 characters, which may disturb the efficiency of the compression of certain data.

Let us compare the various techniquesas shown in Table 3.1 how it's useful for data compression in IoT, Machine Learning

Table 3. comparison of various implementations using RLE

Paper Title	Issues	Advantages
Integration of IoT Streaming Data with Efficient Indexing and Storage Optimization [14]	Due to distribution of these IoT devices, integration and management of the large amount of data is the new challenge.	i) Rectified by developing the indexing technique with the run-length encoding using time-series data compression technique. **ii)** It takes the time stamp from the compressed data while the process of decompression**.**
Differential Run-Length Encryption in Sensor Networks[6]	Energy is the main problem in the sensor networks while communicating the data.	i) This issue is rectified by introducing Differential run-length encoding with the 3 layers of approach. ii) This is used to divide the values into subgroups and apply the compression technique and then convert this into binary form.
Performance Analysis of Data	Main challenge for maintaining	For this data compression is used by identify and
Compression using Lossless Run Length Encoding[5]	the data is to transform, store and retrieve in a secure way.	eliminate the statistical and redundant input data

LZW Compression Technique:

Using a table-based lookup algorithm this technique transforms a file into a lesser file [16]. LZW compression is used to compress the text files, often.

Each input string of bits of the specified length takes a specific message the pattern itself generates an input to a table as a smaller number[17]. As the input is read, every pattern read before the shorter code is replaced effectively compresses the total input to something smaller. The LZW algorithm does include the codes lookup table. Using this algorithm it

processes the encoded input, the decoding programme that uncompressed the file will create the table itself.

1.3 Example: ababbabcababba

i) Construct Small Table with the characters available in message as shown in Table 4

Table 4. Small Table

Index	Entry
1	a
2	b
3	c

ii) Output Sequence Table:as shown in Table 5

Table 5. Output Table

Encoded Output	Index	Entry
	1	a
	2	b
	3	c
1	4	ab
2	5	ba
4	6	abb
5	7	bab
2	8	bc
3	9	ca
4	10	aba
6	11	abba
1	-	-

Encoded Sequence 124523461
It is easy to implement, and has the potential for provide high throughput for hardware deployment. It is the commonly used Unix file compression utility algorithm and also GIF image format. Due to its simplicity and flexibility, LZW is the leading approach for general purpose data compression. Let us compare the various techniques as shown in Table 4.3 how it's useful for data compression in IoT, Machine Learning

Table 6. Comparison of various implementations using LZW

Paper Title	Issues	Advantages
Edge computing by using LZW Algorithm P [10]	As this information rises day by day, it becomes difficult to effectively and efficiently store and distribute the data in less time.	Here it plays a vital role as a portal where we use the LZW algorithm to compress all-time data. Upon completion of the process in the gateway block, the compressed data is sent to the data store where the data is

		permanently stored like a database
Lossless Data Compression Algorithm to Save Energy in Wireless Sensor Network [13]	By transferring the information, the battery life of node is decreasing.	By developing an algorithm as a table-based lookup structure/ In this a character string is a sequence of two characters or more, provides the unique token.to it By using its key, the next character should be identified the string which is replaced by itsKey.
MultiPLZW: A novel multiple patterns matching search in LZW-compressed	The protection and privacy need to be provided for the encrypted or compressed data without sacrificing performance.	Alookup table, a mapping table and a generalised suffix tree context tree will be developed, in terms of time complexity, the algorithm proposed is superior, and spatial complexity
Data [12]		maintains the same order as the best of the current algorithms.

Comparison between Three Compression Algorithms

In this section we are going to compare performance of three different algorithms, by using a different set of characters as an input to determine the performance [4]. To analyse lossless algorithm efficiency. We could consider some metrics of the output as follows.

i) **Compression Ratio** is the ratio between the size of the compressed file and the size of the base file.
Compress Ratio = size after compression / size before compression

ii) **Saving Percentage** calculates the shrinkage of the source file as a percentage.
Saving Percentage = size before compression - size after compression / size before compression *100
Let us consider 4 different types inputs for evaluate the effectiveness of these algorithms and compare their results.

Table 6.Results of Huffman Code

Input file size (bytes)	**Output file size (bytes)**	**Huffman Code (Compression Ratio)**	**Huffman Code (Saving Percentage)**
104	45	43.27	60.7
2168	1131	52.17	47.8
5568	3046	54.17	45.29
192	102	58.13	46.8
677	133	48.9	80.3

Table 7.Results of RLE

Input file size (bytes)	**Output file size (bytes)**	**RLE (Compression Ratio)**	**RLE (Saving Percentage)**
162	122	75.3	24.69
418	232	55.5	44.4
1110	571	51.4	48.5
32	16	50	50
52	29	55.7	44

Table 8. Results of LZW

Input file size (bytes)	**Output file size (bytes)**	**LZW (Compression Ratio)**	**LZW (Saving Percentage)**
600	536	89	10.6
6440	4640	72	27.9
208	178	85.5	14.4
272	244	89.7	10.29
2168	1728	79	20.2

The above Table 6,7,8 shows the compression resuls and saving percentage of Huffman code, RLE, LZW algorithm with different input files.The below Fig.4,5,6 shows graph represntation of comaprision forHuffman code, RLE, LZW algorithm.

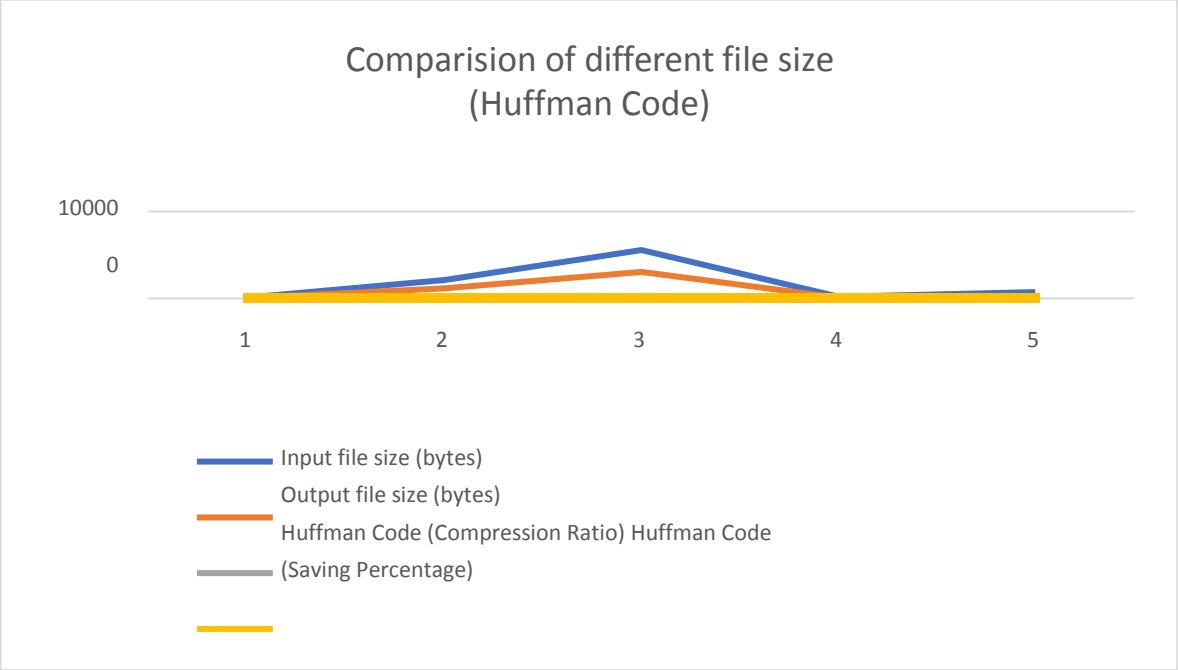

Fig.4 Comparison chart for Huffman code

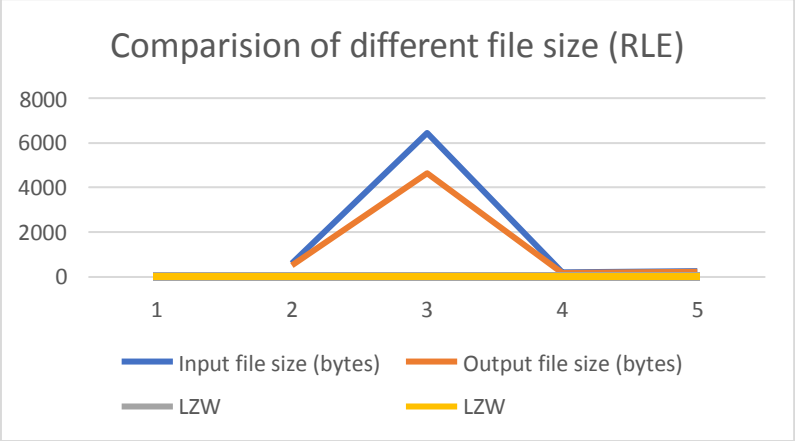

Fig.5. Comparison chart for RLE

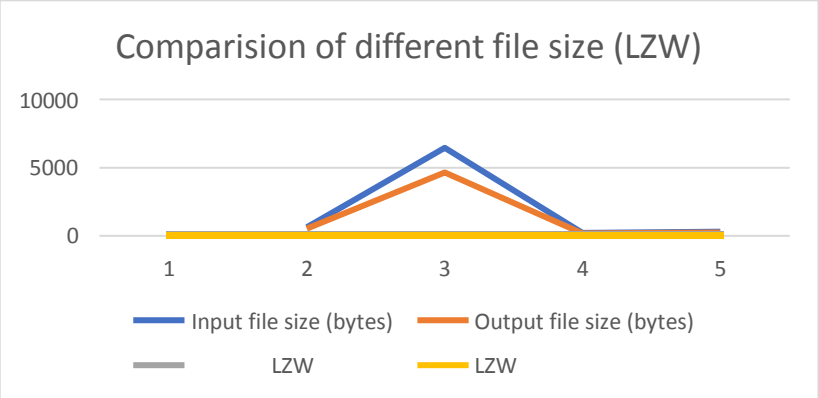

Fig.6. Comparison chart for LZW

As per comparison of the different compression Algorithm the Huffman code produces better result when we use different size of inputs, but in RLE produces better result when the input size of the file is high and LZW also shows some variations in results of different input size.

2. CONCLUSION

In this paper we have presented the different technique for the lossless compression of the text data. Different strategies were explored along with their algorithms and disadvantages. It is shown that no algorithm provides promising results which can be used to compress the data in realistic applications. Therefore, in the future, a lossless text compression algorithm needs to be developed that can better compress text data and can also be used in various functional applications where text data compression is required.

3. REFERENCES

[1]. Amarjit Kaur, Navdeep Singh and SethiHarinderpal Singh, "A Review on Data Compression Techniques", International Journal of Advanced Research in Computer Science and Software Engineering Volume5, issue 1, January
- 2015, pp. 769-773.
[2]. Amlan Chatterjee, RushabhJitendrakumar Shah,Khondker S. Hasan," Efficient Data Compression for IoT Devices usingHuffman Coding Based Techniques",2018 IEEE International Conference on Big Data (Big Data).
[3]. Arijit Ukil, Soma Bandyopadhyayand Arpan Pal,"IoT Data Compression: Sensor-agnostic Approach", 2015 Data Compression Conference.
[4]. Amandeep Singh Sidhu et al.:" Research Paper on Text Data CompressionAlgorithm using Hybrid Approach",International Journal of Computer Science and Mobile Computing, Vol. 3, Issue. 12, December 2014, pg.01 – 10.
[5]. Chetan R. Dudhagaraiet al.:" Performance Analysis of Data Compression
[6]. usingLossless Run Length Encoding",Oriental Journal of Computer Science and Technology, Vol. 10, No. (3) 2017, Pg. 703-707.
[7]. ChiratheepChianphatthanakitet al.:" Differential Run-Length Encryption in Sensor Networks", Sensors, July 2019.

[8]. Joseph Azar, Abdallah Makhoul, Mahmoud Barhamgi, Raphael Couturier, "An energy efficient IoTdata compression approach for edge machine learning". Future Generation Computer Systems, Elsevier,2019, 96, pp.168 - 175.
[9]. KussayNugamesh Mutteret al.:" Automatic Fingerprint Identification Using Gray Hopfield Neural NetworkImproved by Run-Length EncodingFifth International Conference on Computer Graphics, Imaging and Visualization, IEEE 2008.
[10]. LakshmiNarasimha, Devulapalli Venkata, "Application ofHuffman DataCompression Algorithm in HashingComputation", Western Kentucky University, Spring 2018.
[11]. K. Mohana Ravali Chowdary et al.:" Edge computing by using LZW Algorithm", International Journal of Advanced Research, Idea and Innovations in Technology, Volume 5, Issue 1,2019.
[12]. MontherAldwairi et al.:" MultiPLZW: A novel multiple pattern matching search in LZW Compressed Data. Computer Communications .2019.
[13]. S.R. Kodituwakku et. al.," Indian Journal of Computer Science and Engineering", Vol 1, issue 4, pp 416-425.
[14]. Tuong Ly Le, et al.:" Lossless Data Compression Algorithm to Save Energy in Wireless
[15]. SensorNetwork", 4th International Conference on Green Technology and Sustainable Development(GTSD),2018.
[16]. Q.-T. Doan et al.:" Integration of IoT Streaming Data with Efficient Indexing and Storage Optimization", IEEE Access, March 2020.
[17]. Yaqiong Liu, Yuzhuo Wen, Dingrong Yuan and Yuwei Cuan," A Huffman Tree-BasedAlgorithm for Clustering Documents", Springer International Publishing Switzerland 2014,pp. 630–640.
[18]. Classification and prediction of social attributes By K-Nearest Neighbor Algorithm with Socially-aware wireless networking-A study To cite this article: Sujatha Krishanmoorthy et al 2020 IOP Conf. Ser.: Mater. Sci. Eng. 937 01205
[19]. Amin Salih Mohammed, Saravana Balaji B, Saleem Basha M S, Asha P N, Venkatachalam K(2020),FCO — Fuzzy constraints applied Cluster Optimization technique for Wireless AdHoc Networks,Computer Communications, Volume 154,Pages 501-508