ENTROPY ENCODERS: HUFFMAN CODING AND ARITHMETIC CODING

[1]Ketki R. Jadhav, [2]Jayshree R. Pansare
[1,2] Department of Computer Engineering, M.E.S. College of Engineering, Pune, India

Abstract
Today, there are lot of compression techniques used for reducing memory requirement, bandwidth and time for data. Compression techniques are either lossless or lossy. One of the mostly used technique entropy encoding was first lossless data compression technique. Over time goes on there are lot of changes happens in entropy encoding. Even that this technique is traditional one, but most of researchers still use it today in combination with other techniques to gain expected results for data compression. The paper studies and compares two mostly used entropy encoding algorithms: Huffman encoding and Arithmetic encoding. CR given by the arithmetic coding is better than huffman coding but it is so much complex than huffman coding.
Keywords: Entropy encoding, Arithmetic coding, Huffman coding, Lossless Compression.

I. Introduction

Data compression is useful in reducing size of data without affecting quality of data. Data could be any type of information such as audio, video or text. As these types of contents are growing fast in today's world and too much diversity to handle. Diversified data compression techniques exist today to achieve expected results for data compression. Data compression techniques are either lossy or lossless.

1. Lossless Compression: Everything comes under the word 'lossless' means that decoded information exactly similar to original one. Useful in applications where separation from original content is undesirable. For example, hyperspectral image compression is highly lossless because the type of content that images hold.
2. Lossy Compression: Little bit loss of information is acceptable in some applications when achieving good compression ratio is concern for those applications. The decoded information is not entirely similar to original one. An even lossy compression technique gives better CR; processing speed is slow and doesn't give expected quality. So most of researchers, analyst and data compression technique developers prefer to lossless compression techniques.

Redundancy reduction leads to compression achieved by the reducing two type's redundancies.

1. Spatial Redundancy: Content that duplicated within a structure, such as pixels in a frames and bit pattern in file. Exploiting spatial redundancy is how compression is performed.
2. Temporal redundancy: Temporal redundancy actually found between two adjacent frames. Exploiting temporal redundancy is primary goal of image and video compression techniques.

Benefits of compression

1. Saves the cost by sending less data over the switched telephone network.
2. Reduces noise and saves bit rate.
3. Provide the security against unlawful monitoring.
4. In lot of specified applications certain compression techniques proves to be very useful e.g. Embedded algorithms based compression techniques are useful in compression of HD size videos.

5. It reduces the memory requirement to store large amount of data.

6. Reduces processing time for data.

II. Entropy Encoding

Entropy is a lower bound on the average number of bits needed to represent the symbols. Assigned no of bits to every symbol of the message is either the fixed length code or variable length code. In fixed-length codes all assigned codewords (no of bits) have same size. E.g. A-101, B-100, C-111, D-011

If we saw in above example every symbol consists of fixed length of codeword have length 3. A variable length code assigns different size of codewords to each symbol.

E.g. A-1, B-11, C-100, D-101

If we saw that A has codeword whose length is 1 while B, C, D consists of length of codeword as 2, 3 and 3 respectively. In information theory, entropy encoding is lossless data compression scheme that is independent of the specific characteristics of the medium. The purpose of entropy encoding is create and assign a unique prefix code to each input symbol of message. Then these entropy encoders compress the image by replacing every fixed-length input symbol with the corresponding variable-length prefix free output codeword.

III. Huffman Encoding

Developed by DR. David A. Huffman in 1952, Huffman coding is entropy encoding technique. It is lossless compression technique uses variable-length code table as a main concept. Variable-length code table consist no of codewords for each input symbol according to their frequency of occurrences. Less frequently occurred symbols got compressed by using large sized codewords, while more frequently occurred by using small sized codewords. Huffman coding is prefix-code, which means that no codeword is prefix of any other codeword of symbol. Lot of compression techniques in combination with huffman coding gives better results or either improves performance. There is lot of the areas where huffman code proves to be useful like in information theory and coding signal design for radar and communication and design procedures for asynchronous logical circuits. Huffman coding is similar to Shannon-fano which is first information theory for entropy encoding but instead of using bottom-up approach it uses top-down approach for entropy encoding and proved to be optimal technique.

Huffman Tree Construction

Suppose if there are no of symbols given with their frequency of occurrences huffman tree generation process goes through the following stages

1. Start
2. According to frequency occurrences of symbols algorithm put them in descending order.

3. Merges two smallest probability symbols results in high probability symbols.

4. According to top to bottom manner it assigns zero and ones to every branch of tree.

5. It merges lowest probability symbols until it doesn't get one single node which is root of tree and having highest probability value.

6. After root node is found it reads transition bits on branches from top to bottom to generate the codewords.

7. End

E.g. Assume there are 5 different symbols {A, B, C, D, E} with frequency of occurrences {100, 10, 9, 7, 5}. If we see that frequency occurrence of symbol A is 100 which is highest than all other symbols. Huffman tree and their final codes are shown figure 1 and table 1

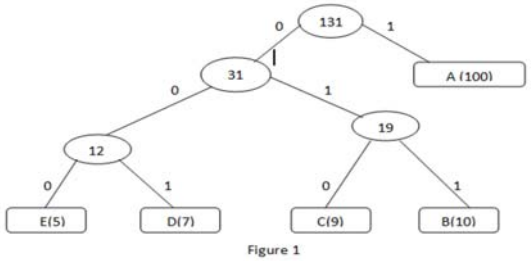

Figure 1

Table 1 represent frequency occurrences of symbols and codeword obtained using huffman tree that are sent to decoder

Symbols	Codewords	Frequency
A	1	100
B	011	10
C	010	9
D	001	7
E	000	5

Table 1

IV. Arithmetic Coding

Instead of working on each symbol, arithmetic coding works on entire message or on whole input string by assigning sequence of bits. It considers whole symbols or message as one unit. It doesn't use concept of variable-length code table for compression of message or string of symbols like huffman coding. Bits used for compression of symbols depend on probability of corresponding symbol. Low arithmetic coding is assign interval to each symbol starts with interval from 0 to 1 and find out subintervals according probability of corresponding symbols. One subinterval results as interval for next symbol.

Process of arithmetic coding

1. First, algorithm assigns each symbol their frequency of occurrences that resides between intervals 0 to 1.
2. And then checks first symbol of input string or message and range of occurrence of that symbol.
3. Range of first input symbol is then arithmetic code takes as an interval for next input symbol.
4. For second input symbol it finds probability of occurrence for each symbol within input interval using following formulae.

   4.1 It first finds out the range difference d using upper bound value and lower bound value within input interval.

   d= upper bound-lower bound.

Where upper bound is highest range value and lower bound is lowest range value of first input symbol respectively.

   4.2 And then for finding range for each symbol within input interval, algorithm uses following formula.

   Range of symbol= lower limit: lower limit +d(Probability of symbol)

   4.3 It repeats these process until it doesn't found frequency of occurrences for each symbol within input interval.

5. Then it takes next symbols of input string or message.
6. Each time interval for next input symbol is equal to range of previously obtained input symbol.
7. It repeats process for each upcoming symbol similar to stage 4.
8. Repeats process until it doesn't reach end of file.

E.g. If we consider example given in table 2 arithmetic coding works as follows.

Symbol	Probability	Range
A	0.1	(0, 0.1)
B	o.2	(0.1, 0.3)
C	0.4	(0.3, 0.6)
D	0.3	(0.6, 0.9)
E	0.1	(0.9, 1.0)

Table 2

We suppose that the input message consist of following symbols BAADA and it starts from left to right. Figure 2 explains the graphical representation of the arithmetic coding of this message from left to right. As can be seen, the first probability range is 0.1 to 0.3 because the first symbol is B.

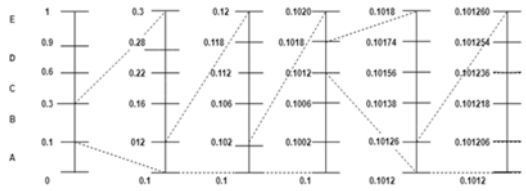

Figure 2

The encoded interval for mentioned example is [0.1012, 0.101260]. A sequence of bits is assigned to a number that located in this range.

V. Advantages of Arithmetic coding

1. Gives better compression efficiency.
2. In combination with adaptive model, gives better encoding if alphabet or its properties are changing over the file.

3. Encoding procedure can be naturally and simply extended to encompass even an infinite alphabet. Huffman type codes require special considerations for each such alphabet.

VI. Advantages of huffman on arithmetic coding

1. Arithmetic algorithm runs to much slower than huffman algorithm, which can be critical in some applications.
2. Arithmetic coding is more complex than huffman coding and more difficult to understand for system user.
3. Main purpose of arithmetic code obtains compression efficiency which is hard to get in most realistic situations. As well as for text based applications, the saving is typically very small.
4. Lot of applications uses inaccurate probabilities, where saving may actually be negative.

VII. Conclusion

We studied two entropy encoding algorithms: Huffman coding and Arithmetic coding. Even though entropy encoding techniques traditional one, but still lot of researchers uses it for data compression. Entropy encoding is lossless compression technique. From process of execution of algorithms and from given examples we can understand that Arithmetic coding is better one than huffman coding and gives better compression ratio. But it has slowest processing speed than huffman coding and more complex to understand for user. But both are useful in specific applications. Diversified applications according to their type of compression either use huffman coding or arithmetic coding.

Acknowledgment

It gives me great pleasure and satisfaction in presenting seminar on "Entropy Encoders: Huffman coding and Arithmetic coding," I would like to thank my project guide, Prof. J. R. Pansare mam for her guidance and support. Without her constant support, guidance and assistance this seminar report would not have been completed. Without their coordination, guidance and reviewing this task could not be completed alone. I would like to thank all those, who have directly or indirectly helped me for the completion of the work during this project.

REFERENCES

1. Ahsan Habib and Mohammad Shahidur Rahman, "Balancing decoding speed and memory usage for Huffman codes using quaternary tree," pp. 1-15, 2017.
2. Asadollah Shahbahrami, Ramin Bahrampour, Mobin Sabbaghi Rostami, Mostafa Ayoubi Mobarhan "Evaluation of Huffman and Arithmetic Algorithms for Multimedia Compression Standards".
3. Gaurav Vijayvargiya, Dr. Sanjay Silakari and Dr.Rajeev Pandey, "A Survey: Various Techniques of Image Compression," (IJCSIS) International Journal of Computer Science and Information Security, Vol. 11, No. 10, October 2013.
4. Nehal Markandeya, Prof.Dr.Sonali Patil, "Image Compression Using Huffman Coding," International Journal Of Engineering And Computer Science, Vol. 6, Issue 1, pp. 19999-20002, Jan. 2017.
5. Rachit Patel, Virendra Kumar, Vaibhav Tyagi and Vishal Asthana, "A Fast and Improved Image Compression Technique Using Huffman Coding," IEEE WiSPNET, pp. 2283-2286. March 2016.
6. Mamta Sharma, "Compression Using Huffman Coding," (IJCSNS) International Journal of Computer Science and Network Security, VOL.10 No.5, pp. 133-141, May 2010.
7. Sanjay Kumar Gupta, "An Algorithm For Image Compression Using Huffman Coding Technique," (IJARSE) International Journal of Advance Research in Science and Engineering, Vol. 5, No. 7, pp. 69-75, July 2016.
8. A. Bookstein and S. T. Klein, "Is Huffman coding dead?" Computing, vol. 50, no. 4, pp. 279–296, 1993.

9. Ms. Pallavi M. Sune Prof. Vijaya K. Shandilya., "Image Compression Techniques based On Wavelet and Huffman Coding," International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 3, Issue 4, pp. 524-528, April 2013.

10. Paul G. Howard and Jeffrey Scott Vitter, "Analysis Of Arithmetic Coding For Data Compression," Information Processing & Management, Vol. 28, No. 6. pp. 749-763, 1992.

11. Ezhilarasu P. , Krishnaraj N. , Dhiyanesh B., "Arithmetic Coding for Lossless Data Compression- A Review," International Journal of Computer Science Trends and Technology (IJCST), Vol. 3, Issue 3, pp. 89-94, May-June 2015.

12. Mridul Kumar Mathur, Seema Loonker, Dr. Dheeraj Saxena, "Lossless Huffman Coding Technique For Image Compression And Reconstruction Using Binary Trees," IJCTA, Vol. 3, No. 1, pp. 76-79, Jan-Feb 2012.