International Journal of Recent Research and Review, Vol. XIV, Issue 3, September 2021
ISSN 2277 – 8322

Enhancement of Modern Data Compression Methods Using Standardized Binary Trees

Roman B. Shrestha[1], Seraj Ahmad[2]

[1]*Department of Computer Science, Kathmandu Model Secondary School, Kathmandu, Nepal*

[2]*Head of Department, Department of Computer Science, Kathmandu Model Secondary School, Kathmandu, Nepal*

Email: [1]roman.shrestha312@gmail.com,[2]ahmadseraj.cs@gmail.com

***Abstract*—Despite the wide array of data compression techniques available for technological consumption, there is yet to be an ideal compression technique that meets modern standards for speed and efficiency. Using lossy compression and decompression methods derived from the popular Huffman Code, a variable-length binary algorithm designed for lossless data compression, we propose an enhanced data compression system that relies on standardized codes for messages with more frequently used letters resulting in a shorter code, and less frequently used letters resulting in a slightly longer code. These proposed systemic improvements are designed with the overall aim of reducing code redundancy and accommodating future transmissional demands.**

***Keywords*— Data Compression, Huffman Algorithm, Standardized Trees**

I. INTRODUCTION

Within the various sub processes of signal processing, there are two general types of compression that may be used to reduce the bits of an original file: lossy and lossless compression. Regardless of the compression method that is used, the resulting representation will always be lossy or lossless [1]. In lossy compression, bits are reduced by the removal of superfluous information. While this adeptly lessens the overall data present, some degree of information is discarded from the file. Meanwhile, lossless compression reduces file bits through the use of statistical algorithms that are designed to locate statistically redundant information. Despite the efficiency of this compression methods being measurably less than that of lossy compression, losslessly compressed files maintain the ability to be restored to their unabridged, original form [2]. Such flaws within these available compression methods have become increasingly apparent as technological demands grow, thus prompting the search for improvements upon current processes.

To advance upon modern data compression methods, we must first review the past. The basis of modern data compression can be attributed to the work of David A. Huffman. In 1952, Huffman developed an optimum method of coding an ensemble of messages consisting of a finite number of members. A minimum-redundancy code was constructed in such a way that the average number of coding digits per message was minimized [3]. Huffman code is known to be one of the most efficient algorithms for data compression till the date. However, the repetitive nature of the algorithm's coding calculations severely hinders the processing speed of this compression method [4]. We thereby propose an alternative and optimum method for reducing code redundancy by using generalized alternate ensemble codes for messages.

II. HUFFMAN CODING

A. Huffman Coding Procedure

According to [4], the Huffman coding procedure is based on the following two observations:

- Recurrent symbols will have shorter code words in comparison to a symbol that occurs less frequently.
- The two symbols that occur least frequently will be equal in length.

B. Basis of Huffman Coding Algorithm

The Huffman code is designed by merging the lowest probable symbols. This process is repeated until only two probabilities of two compound symbols remain. As a result, a code tree is generated and Huffman codes are obtained from the labeling of the code tree [4].

III. STANDARDIZED TREE AND TABLE

A generalized tree can save significant amounts of time and effort during the computation process of minimum redundancy codes. The following steps are followed when constructing the tree:

- A set of sample texts is used to calculate the average probable occurrence of letters and symbols.
- Using the Huffman Encoding principle, a binary tree can be created to represent letters and frequently used symbols along with their probabilities of occurrence in message.
- A standardized table is constructed based on the binary tree.

TABLE I

Standardized Table Based on Binary Tree Results

Symbol	Frequency	Probability	Code
space	2541	0.1862	111
e	1486	0.0183	011
t	1021	0.0712	1101
a	889	0.062	1011
o	866	0.0604	1001
n	763	0.0532	0101
i	731	0.051	0100
h	676	0.0471	0011
s	674	0.047	0010
r	624	0.0435	0001
d	485	0.0338	10101
l	439	0.0306	10001
u	285	0.0199	110011
c	269	0.0188	110010
f	262	0.0183	110001
y	253	0.0176	110000
m	252	0.0176	101001
g	240	0.0167	101000
w	219	0.0153	100001
p	173	0.0121	000011
.	173	0.0121	000010
b	144	0.0100	000001
v	105	0.0073	1000001
,	101	0.0070	1000000
k	88	0.0061	0000001
j	33	0.0023	00000001
?	8	0.0006	0000000011
x	8	0.0006	0000000010
q	7	0.0005	0000000001

IV. DECODING

After the code has been created, coding and/or decoding is accomplished in a simple look-up table manner [5]. The code itself is an instantaneous, uniquely-decodable block code. It is called a block code because each source symbol is mapped into a fixed sequence of code symbols. The code is considered instantaneous because each codeword in a string of code symbols can be decoded without referencing succeeding symbols. In addition, the code is uniquely decodable because any string of code symbols can be decoded in only one way. Thus, any string of Huffman encoded symbols can be decoded by examining the individual symbols of the string in a left to right manner. For the binary code of table I a left-to-right scan of the encoded string 1110111101 reveals that the first valid codeword is 111, which is the code for symbol a1. The next valid code is 011, which corresponds to symbol a2. Continuing in this manner reveals the completely decoded message, a1,a2,a3. In this manner, the original image or data can be decompressed using Huffman decoding as explained above [4].

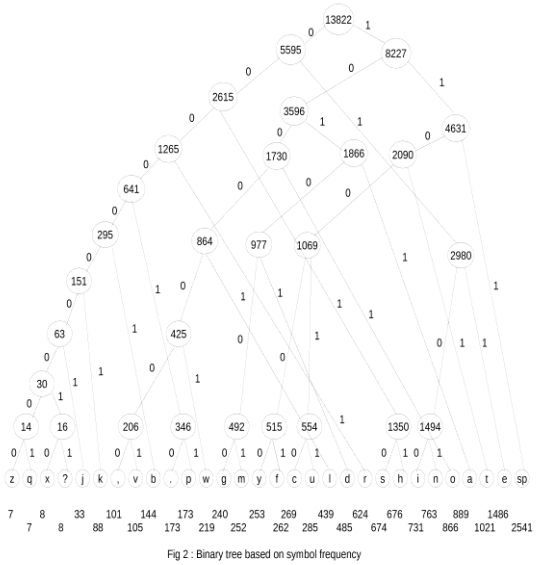

Fig 1: Standardized Binary Tree

V. RESULTS

In this experiment, we used short messages to calculate and compare the compression ratio for both the Huffman coding technique and the proposed generalized alternate ensemble code method. For short messages, the Huffman method produces compressed data that is very similar in size to that of the unabridged data; this leads

us to the logical conclusion that the Huffman coding technique soaks up large amounts of resources to produce very little payoff in terms of compression file size. This was because even if a letter had been introduced in a sentence, it's code had to be included in the reference table so more space was consumed. Meanwhile, in the proposed ensemble code method, a predefined table is available for message codes to be directly assigned, resulting in a more efficient and less-costly way of compression. This proposed compression method managed to achieve up to a 2:1 ratio on short messages. When applied to longer texts, however, both the proposed and Huffman algorithms performed similarly. Despite this, the proposed method still remained slightly faster than the Huffman coding technique due to the presence of a predefined table. This led us to conclude that the proposed algorithm may be best suited for short social media posts or messaging platforms.

VI. LIMITATIONS AND PROSPECTS

Although this new method has a better compression ratio as well as improved time complexity, there are a few limitations that need to be addressed. First, minor losses occur during decompression. These losses include capitalized letters and a few less frequently used symbols. However, the optimum binary tree can be modified in such a way that certain codes trigger the capitalization of messages which can reduce losses to much extent. Finally, the losses can be further reduced by making the binary tree more inclusive but at the expense of the compression ratio.

REFERENCES

[1] Wade, G. (2011). *Signal Coding and Processing*. Cambridge University Press. https://dbpedia.org/page/Data_compression

[2] Mahdi, O.A., Mohammed, M.A., & Mohamed, A.J. (2012). Implementing a novel approach an convert audio compression to text coding via hybrid technique. *International Journal of Computer Science Issues, 9(*6, No. 3*)*, 53–59. http://ijcsi.org/papers/IJCSI-9-6-3-53-59.pdf

[3] Huffman, D. (2001). The number of optimal binary one-ended codes. Research Gate. https://www.researchgate.net/publication/3862148_The_number_of_optimal_binary_one-ended_codes

[4] Pujar, J.H.; Kadlaskar, L.M. (2010). A new lossless method of image compression and decompression using Huffman Coding techniques. *Journal of Theoretical and Applied Information Technology, 15*(1), 18–23, Vol 15 No1.

[5] Aggarwal, M., & Narayan, A. (2000). Efficient Huffman decoding.*Proceedings 2000 International Conference on Image Processing,*1, 936-939.