An Introduction of Image Lossless Compression Approches

Shengzhong Zhang[a,*], Lei Yu[b], Yinqian Cheng[c]

Information Network Center, China University of Geosciences (Beijing), Beijing, China
[a]zhsz@cugb.edu.cn, [b]yul@cugb.edu.cn, [c]chengyq@cugb.edu.cn
[*]Corresponding author

Abstract: *The purpose of image compression is to reduce the number of bits required to represent data by removing data redundancy. Due to the large amount of image data, it is very difficult to store, transmit, and process, so the compression of image data becomes very important. This article introduced several lossless compression approaches such as Huffman, Fano, Run Length, Arithmetic, and LZW (Lempel–Ziv–Welch) Coding. The approach which can achieve compact code is regarded as closing to the best solution.*

Keywords: *Lossless Compression, Entropy, Compact Code, Run Length Coding, Arithmetic Coding*

1. Introduction

The purpose of image compression is to maintain the continuity of original sampling, so that a high bit rate image can become a compressed image with high fidelity [1]. High fidelity can be evaluated by quantitative criteria such as mean variance, visual quality evaluation as subjective criteria or application-specific statistical criteria, so that communication and storage of the image can obtain the best possible quality. Lossless compression utilizes compression technologies that do not lose information, allowing compressed data to recover back to the original image without error, but the compression ratio is not high, usually 2:1 to 3:1. One of the basic approaches is to try to change the probability distribution of the source to make it as non-uniform as possible, and then use the best coding method to make the average code length approximate to the entropy of the source. Common lossless compression methods include Huffman, run length, arithmetic, Rice algorithm, etc. [2]. The minimum amount of lossless compression data is limited by its information entropy, and the efficiency is limited. Information entropy coding is based on the principle of information entropy. Those with high probability of occurrence are represented by short codewords, while those with low probability are represented by long codewords. The most common ones are Huffman, run length and arithmetic coding.

2. Entropy

Assuming there are M random variables α_1, α_2, •••, α_M with probabilities P (α_1), P (α_2), ••••, P (α_M), then the information content of each random variable α_K is defined as:

$$I(\alpha_K) = -\log_a P_K \tag{1}$$

Then Entropy is defined as:

$$H = \sum_{k=1}^{M} P_K I(\alpha_K) = -P_K \log_a P_K \tag{2}$$

The physical meaning of entropy is a measure of the degree of uncertainty in a set of random variables, and the concept of entropy provides a criterion to measure the performance of bit-specific codes.

If the input set of the decoder is W_1, •••, W_M with probability P_1, •••, P_M, and design a codeword C_1, •••, C_M with a word length of β_1, •••, β_M is designed, then the average number of bits of the encoder is:

$$\mathrm{R} = \sum_{K=1}^{M} \beta_K P_K \tag{3}$$

When R is close to H, the encoder is close to the best, and the encoding is called compact code [3], that is, there is a distortion-free coding, which makes the average code length R approximate the information entropy H. Typical coding methods include Huffman, Faon and Shannon encoding methods, among which Huffman method is the best. The basic principle of entropy coding is that the average number of input pixels is minimum, that is, short code words are used for codes with high probability.

3. Lossless Compresssion Approaches

Lossless compression is to use the minimum bit to represent a given image, which is generally divided into two steps:

(1) De-correlation, statistical redundancy in the image is removed and residual image is obtained, which can be regarded as a step of first-order entropy reduction.

(2) Encoding, using the encoder to turn the residual image into the output bit stream [4] [5].

3.1. Huffman Coding

In 1952, Huffman proposed the method of generating compact codes. The steps are as follows [3]:

(1) Arrange several information source signals from large to small according to the probability distribution P(xi).

(2) Combine the two information source signals with the lowest probability into one to form a new probability set, so as to obtain a new information source containing only (n-1) symbols, and then rearrange according to (1), repeat until last two probabilities remain.

(3) Assign code words. From the last step backwards, for the last two probabilities, one is assigned "0", the other is assigned "1", and so on in reverse to the beginning.

The average code length R approaches the entropy H. Therefore, Huffman code is compact code and an optimal encoding.

For Huffman coding, the following problems should be noted:

(1) The codeword constructed by Huffman coding is not unique. Reason as follows:

(a) The choice of upper and lower branch assignment is arbitrary.

(b) When the probabilities of two symbols are equal, the one who comes first is also random, so the codeword is not unique, but for the same source, the average code length remains unchanged and the coding efficiency is the same.

(2) Huffman encoding is a variable length encoding, which is inconvenient for hardware implementation and difficult to identify errors.

(3) Transmission time is required to transmit Huffman encoding table.

3.2. Fano Coding

Feno encoding is a coding algorithm published by Shannon and Fano at the same time. The specific method [3] is:

(1) The set of known input probabilities is arranged in order of probability from largest to smallest, and

(2) It is separated from the middle place into two parts, “I” and “II”, making the two probabilities are close to each other. Then “I” is assigned a codeword (such as “1”), “II” is assigned a codeword (such as “0”), and repeated the method to obtain the final code.

3.3. Run Length Coding (RLC)

The main method of RLC is to replace a continuous string of the same value with a numeric value and length. In image coding, adjacent pixels with the same length along the scanning direction of the image are a group, and their continuation length is called run. The end position of the run is determined by the relative distance from the end of the previous run, that is, the number of the same pixels. In this way, the gray run can be used to represent the image data. For example, if there is a string of M pixels with the same grayscale N along the horizontal direction, then passes only two values (N, M) it could replace M grayscale values N of M pixels. Image scanning direction can be horizontal, vertical, or zigzag.

Run-length coding is suitable for binary image coding. It is an efficient coding method for the same gray level or continuous image. For multi-gray level images, run length coding is often mixed with other coding methods. For example, JPEQ, H.261 and MPEG use a hybrid encoding method of run length coding and Huffman encoding to compress image data.

3.4. Arithmetic Coding

The Huffman method can achieve the best encoding effect for source data stream. But because the smallest unit of storage and processing in computer is "bit", the actual compression effect is far from the limit of theoretical compression ratio. To address this issue, arithmetic coding has been proposed. It was first introduced by J. Rissanen in 1976 in the form of "last in, first out" coding.

In 1979, G. G. Langdon systematized it, eliminated multiplication and simplified processing. In 1981, it was extended to binary image coding: for binary stationary Markov information source, the efficiency was higher than 95%. Theoretically, arithmetic coding gives the best compression. The arithmetic encoder encodes the data in a minimum number of bytes if a correct probability distribution model of the compressed data events can be provided.

However, to achieve optimal compression, exact real arithmetic must be used (at least when the probabilities are rational, use exact rational arithmetic). In fact, arithmetic coding uses fixed-point numbers to approximate exact arithmetic, which can obtain a quasi-optimal compression.

Arithmetic coding’s principle is: it represents the encoded information as an interval between the real number 0 and 1. The longer the information, the smaller the interval between encoded representations, indicating the more binary bits it contains. The successive symbols in the information source are reduced in intervals based on a certain generation probability. The basic steps are as follows:

(1) Initiate current interval [0,1];

(2) For each symbol, perform two steps:

(a) Divide the initial interval into smaller intervals, and the sub-interval of each symbol is proportional to its probability of becoming the next symbol;

(b) The symbol actually appears in the next digit corresponding to the subinterval as the current interval;

(3) Output enough bits to distinguish the final determined interval from other possible results.

Decoding steps [6]:

(1) Determine the first character of the current code based on its range and output it.

(2) Eliminate the impact of decoded characters on subsequent operations, subtract the lower limit of the probability value of the translated character, and then divide it by the width of the probability range of the translated characters, which is the code of the corresponding subsequent string.

(3) If the code string processing is completed, decoding ends.

However, to achieve optimal compression, precise real number arithmetic must be used (at least when the probability is rational, precise rational arithmetic must be used). In fact, arithmetic encoding uses fixed point numbers to approximate precise arithmetic, resulting in quasi optimal compression.

Arithmetic encoding features [7]:

(1) The adaptive mode of arithmetic coding gets unique advantages, while other entropy coding methods do not have this characteristic. This mode does not require a pre-defined probability model, and is particularly suitable for situations where probability statistics are not performed.

(2) Arithmetic coding is recommended when source symbol probabilities are relatively close, because Huffman coding efficiency is not high at this time.

(3) The implementation of arithmetic encoding is more complex than the Huffman encoding method, especially in hardware implementation.

3.5. LZW (Lempel–Ziv–Welch) Coding

LZW is a universal lossless data compression algorithm created by Abraham Lempel, Jacob Ziv, and Terry Welch. It was published by Welch in 1984 as an improved implementation of the LZ78 algorithm published by Lempel and Ziv in 1978 [8]. LZW encoding is based on a dictionary encoding method, which generates a string table and corresponding code during compression encoding. Before the start of LZW compression encoding, there are only a single character and a string table corresponding to the encoding in the dictionary. After the compression encoding start, the string is read and corresponding to the string in the table, and the encoding is output until the corresponding string cannot be found. The character string that can be found and cannot be found in the table are included for corresponding encoding, then the string table will gradually expand, and the number of occurrences of strings that need to be compressed in the table will also increase. Finally remove duplicate information from the image data.

The advantages of LZW encoding are strong logic, low cost, high efficiency, simple to implement and the ability to achieve good compression results, with high throughput in hardware implementations [9].

4. Summary

Image compression will reduce transmission and storage cost significantly, while lossless compression can maximize the preservation of original image information. In this article, some common image lossless compression approaches are presented, such as Huffman, Fano, Run Length, arithmetic and LZW coding. Within them, the approach which can reach compact code is nearly the best solution.

Acknowledgements

This research is supported by China Earthquake Science Experiment project, China Earthquake Administration (2019CSES0113).

References

[1] Robert M. Gray, et. al, Image Compressing and Tree-structured Vector Quantization, Image and Text Compression, 1992, Vol.176: 3-24
[2] Xiaoguang Jia, Lei Wang, Latest Development of Aerospace Remote Sensing Image Compression Technology, Chinese Journal of Graphics and Images, 1997, 2(10): 697-700
[3] Xianglin Li, Digital Image Processing, Graduate School of China University of Science and Technology, 1997.9
[4] Jinfei Wang, Kaizhong Zhang, et. al, Spectral and Spatial Decorrelation of Landsat-TM Data for Lossless Compression , IEEE Trans. on Geoscience and Remote Sensing, 1995, 33(5): 1277-1285
[5] Nazir D. Memon et. al, Lossless Compression of multispectral Image data, IEEE Trans. on Geoscience and Remote Sensing, 1994, 32(2): 282-289
[6] Mei Yuan, Wen Yuan, Data Compression Technology and Its Applications, Electronic Industry Press, 1995.12
[7] Wen Gao, Multimedia Data Compression Technology, Electronic Industry Press, 1994.4
[8] Fouzia I Khandwani, P E Ajmire, A Survey of Lossless Image Compression Techniques, International Journal of Electrical Electronics & Computer Science Engineering, 2018, 5(1): 39-42
[9] Yun Gan, Tao Ran, Research on Image Compression Coding Technology and Lossless Compression Strategies, Computer Knowledge and Technology, 2020, 16(22): 212-213