FREQUENCY DOMAIN HUFFMAN CODING WITH UNEQUAL PIXEL COSTS FOR IMAGE COMPRESSION AND TRANSMISSION

Zhenhuan Sun, Ziyi Xu

z98sun@uwaterloo.ca - z297xu@uwaterloo.ca
University of Waterloo
Department of Electrical and Computer Engineering
200 University Ave W, Waterloo, ON N2L 3G1

ABSTRACT

A new Huffman coding based lossy image compression method for stable image transmission is introduced in this paper. To ensure effective image transmission in a noise contaminated environment, both high image compression ratio and robustness such as high resistance to error are essential. Our proposed method begins with segregation of magnitude and phase spectra of images, followed by compressing the most informative region on both spectra with our modified Huffman encoding method. Various image transforms can be applied to acquire both spectra. The purpose is to identify and crop the region in which most image energy are concentrated on, whereas different weights are assigned to different pixels in that region according to the importance of each pixel. Huffman coding is adjusted with assigned weights to minimize the cost of encoded image data instead of the length to assure fast and stable transmission of more important pixels. Simulation of this new method has shown that with different weighting functions, more important pixels are encoded with less bits while maintaining good compression ratio, and reconstructed images posses reasonable qualities.

Keywords: Information theory, Huffman coding, Shannon coding, image compression, image transmission

1. INTRODUCTION

In the field of image compression, only two types of methods are concerned, lossless and lossy. Lossless image compression refers to methods which are able to reconstruct compressed images without any theoretical error, but often have unimpressive compression ratios such as entropy coding [1]. On the contrary, lossy image compression methods reconstruct compressed images with errors but usually with very high compression ratio. Our proposed method falls in the category of lossy methods, however an adapted lossless entropy coding method based on Huffman coding is used to encode the most important region of images in our algorithm.

The motivation of our algorithm starts with the awareness of irrationality of assigning each pixel with equivalent importance. The significance of each pixel may not be obvious in spatial domain, however various of image transform methods such Discrete Cosine Transform (DCT) and Discrete Fourier Transform (DFT) can pin down the distribution of image energy in frequency domain, in which case we can easily find out the region of pixels which is more important than other regions. In frequency domain, one can see that pixels are not equally important since most image energy is concentrated on low frequencies, thus during transmission the consequence of losing pixels in the low frequency region is much more severe that losing pixels in high frequency region. On that account, if pixels that are important for the quality of reconstructed image are encoded with a relatively large number of bits, the probability that the codewords for these pixels are contaminated with errors will be larger(?). We intend to find a method which can encode pixels according to their significance while preserving image quality and compression ratio as much as possible.

Huffman coding as a lossless entropy coding method has been attracting researcher's attentions for many years for its simplicity and diversity [2]. With the knowledge of source probability distribution it results in optimal prefix code with minimum average codeword length [3]. However the resulting codewords length of Huffman coding are based on the occurrence of random variable's value in the corresponding alphabet, for example, symbols or pixels that are more likely to occur will be encoded with less bits compare to symbols that are less likely to occur. Despite that the structure of Huffman coding is against our intuition of encoding more important symbols with less bits, we have found a way to modify Huffman coding method in a way such that it will result in optimal prefix code with minimum average cost instead of length.

2. HUFFMAN CODING WITH UNEQUAL PIXEL COSTS

In order to derive formula for this variation of Huffman coding one must first recall how to find the optimal prefix code under certain assumptions. Let $X \in \mathcal{X}$ be a random variable

which draws values from alphabet $\mathcal{X} = \{x_1, x_2, ..., x_m\}$, the probability of X equals to any values in its alphabet is denoted as p_i and the length of each codeword of x_i is denoted as l_i, where $i = \{1, 2, ..., m\}$. Then the question of finding the optimal D-ary code with minimum average length can be formalized as solving minimization problem:

$$\begin{aligned} \min_{l_1,...,l_m} \quad & \sum_{i=1}^{m} p_i l_i \\ \text{s.t.} \quad & \sum_{i=1}^{m} D^{-l_i} \leq 1 \\ & l_i \in \mathbf{Z}^+ \end{aligned} \tag{1}$$

the optimization constraints are coming from Kraft inequality which is the necessary and sufficient condition for a prefix code [3] and length of codewords must be positive integers. This optimization problem can be easily solved if we relax the integer constraint on length to real numbers such that optimal solution will have equality in Kraft inequality [3] since if equality does not hold, decreasing l_i will always improve optimization objective. Once inequality constraint has become equality, Lagrange multiplier can come in handy to solve the relaxed problem. The result of this optimization problem is

$$l_i^* = -\log_D p_i$$

note that this is the solution without integer constraint so in reality the ceiling $\lceil -\log_D p_i \rceil$ is used. To find the optimal code using Huffman coding, we just need to use the source distribution p_i to construct the Huffman tree.

2.1. Derivation

To modify the existing optimization problem such that we can obtain optimal prefix code with minimum average cost, we first constructed a weight function $w : \mathcal{X} \longrightarrow \mathbf{R}$ which assigns each pixel value $(e.g., x_1, x_2, ..., x_m)$ in the alphabet of X a different weight $(e.g., w_1, w_2, ..., w_m)$ according to their importance. Then we append the weights to objective function so that the minimization goal is weighted average length which is the average cost. The resulting relaxed optimization problem can be formalized in this way:

$$\begin{aligned} \min_{l_1,...,l_m} \quad & \sum_{i=1}^{m} w_i p_i l_i \\ \text{s.t.} \quad & \sum_{i=1}^{m} 2^{-l_i} = 1 \end{aligned} \tag{2}$$

in this optimization problem we chose $D = 2$ for the purpose of finding binary prefix code. Our objective here is to find the weighted distribution on source which results in optimal codeword length with minimum cost. The weighted distribution will then be applied to construct Huffman tree.

Although this optimization problem can be solved by standard Lagrange multiplier method, a different approach from information theory is considered here to show that the probability mass function (PMF) that we constructed is indeed the weighted distribution on source that results in the optimal solution(?).

Let's assume the optimal solution of this optimization problem that satisfy the Kraft equality is denoted as $a_i = 2^{-l_i^*}$, and we construct a probability mass function (PMF) as:

$$q_i = \frac{w_i p_i}{\sum_{i=1}^{m} w_i p_i}$$

given these two notations we can rewrite the objective function in (2) as:

$$\begin{aligned} \sum_{i=1}^{m} w_i p_i l_i &= \sum_{i=1}^{m} w_i p_i \log_2 \frac{1}{a_i} \\ &= \sum_{i=1}^{m} w_i p_i \log_2 (\frac{q_i}{a_i} * \frac{1}{q_i}) \\ &= \sum_{i=1}^{m} w_i p_i \log_2 \frac{q_i}{a_i} - \sum_{i=1}^{m} w_i p_i \log_2 q_i \end{aligned} \tag{3}$$

since q_i is a probability mass function then $\sum_{i=1}^{m} q_i = 1$, and we can multiply $\sum_{i=1}^{m} q_i$ to above equation without changing anything.

$$\begin{aligned} \sum_{i=1}^{m} w_i p_i l_i &= \sum_{i=1}^{m} w_i p_i \log_2 \frac{q_i}{a_i} - \sum_{i=1}^{m} w_i p_i \log_2 q_i \\ &= \sum_{i=1}^{m} w_i p_i \sum_{i=1}^{m} q_i \log_2 \frac{q_i}{a_i} - \sum_{i=1}^{m} w_i p_i \sum_{i=1}^{m} q_i \log_2 q_i \\ &= \sum_{i=1}^{m} w_i p_i D\left(q \| a\right) + \sum_{i=1}^{m} w_i p_i H(q) \end{aligned} \tag{4}$$

where $D\left(q \| a\right)$ is Kullback–Leibler divergence between distributions q and a, it measures the difference between two distributions, and $H(q)$ is the discrete information entropy of distribution q. From definition of entropy we know that $H(q)$ does not depend on l_i, thus to minimize the objective function the best we can do is setting $D\left(q \| a\right)$ to 0, which means q and a are two identical distributions. Therefore, when $a_i = q_i = \frac{w_i p_i}{\sum_{i=1}^{m} w_i p_i}$ the optimal solution for this modified optimization problem is

$$l_i^* = -\log_2 \frac{w_i p_i}{\sum_{i=1}^{m} w_i p_i}$$

This concludes our proof on q_i is the best weighted distribution on source that will result in optimal prefix code with minimum average cost. Thus distribution q_i will be used in our following algorithm to construct Huffman tree and optimal code for every pixels.

2.2. Algorithm Overview

Our algorithm begins by separating original image to its magnitude and phase components using Discrete Fourier Transform (DFT). The magnitude spectrum is used to determine the most informative region in frequency domain since we know where most of image energy is concentrated on, on the other hand, the phase spectrum may not be as useful as magnitude spectrum in terms of selecting important information but it contains phase shift information that are sensitive to human eyes and we need both magnitude and phase components to reconstruct images. Once the important regions are determined, crop operations are performed to these regions in both magnitude and phase spectra, and histogram analysis is conducted to determine the source probability p_i of cropped images. Weight functions are then constructed in a manner that user defined important pixels will receive higher weights than others and we use this weight function to construct weighted source distributions q_i for both spectra. Previously discussed Huffman coding method is applied to cropped image pixels to obtain binary sequence of codewords. The codewords are transmitted and corresponding Huffman decoding will be applied to retrieve the original magnitude and phase spectrum. Finally original image is reconstruction by applying Inverse Discrete Fourier Transform (IDFT) to $|X|e^{j\angle X}$.

2.3. Encoding Algorithm

1. Separate image to its magnitude and phase components.
2. Apply contrast enhancement to magnitude spectrum to determine the informative region.
3. Crop magnitude and phase spectra.
4. Find source distribution on both spectra.
5. Choose weight functions and construct weighted distributions for both spectra.
6. Apply Huffman coding to cropped images using weighted distributions.
7. Transmit encoded data.

2.4. Decoding Algorithm

1. Apply Huffman coding to received data.
2. Apply counter contrast enhancement to magnitude spectrum.
3. Combine magnitude and phase spectra with magnitude and phase formula.
4. Reconstruct image by applying IDFT to the result of 3.

2.5. Choice of Weight Function

Our purposed method aims to encode pixels that have more significance with fewer bits and pixels that are less important with more bits. However the definition of important pixels varies from application to application, different applications may define a same pixel with completely different importance. We listed a few weight functions that we think might be useful during image compression and transmission.

In contrast enhanced magnitude spectrum, pixels located at low frequencies with intensity value close to 255 are considered as important pixels since they usually contain most of image energy.

- Square weight function: It takes a threshold value and assign all intensity values greater than that threshold a fixed weight while intensity values smaller than the threshold are weighted 0.
- Gaussian weight function: Instead of assigning fixed weight to intensity values greater than threshold, a 1D Gaussian distribution with mean 255 and a user defined variance is used to assign weight in increasing order.

Although standard Huffman encoding method has already encoded pixels with high probability of occurrence with fewer bits, our proposed method supports users to encode those pixel with even fewer bits. The only thing needs to be changed is weight function.

- Distributive weight function: Assigning higher weight to pixel values that are more likely to occur according to image source probability distribution.

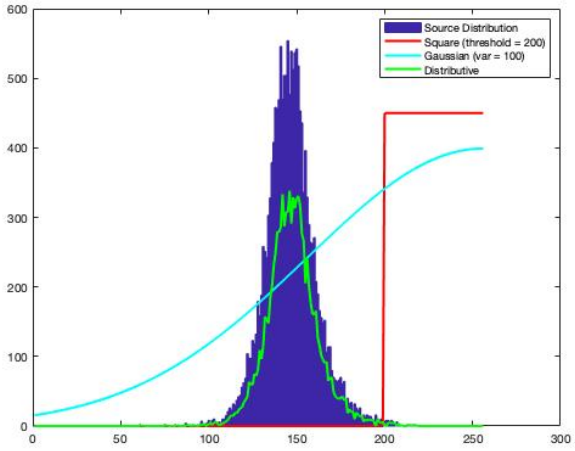

Fig. 1. Different weight functions comparison under same source distribution

3. SIMULATION RESULTS

The objectives of this simulation are to find out if our modified Huffman coding algorithm can encode pixels according to their importance and evaluate the performance of proposed algorithm.

Table 1. Average Codeword Length for Important Pixels Only(All Pixels) Using Distributive Weight Function

Importance	Modified Huffman	Huffman
Probability $\geq 1\%$	5.6000(53.1406)	5.5714(47.3828)
Probability $\geq 1.5\%$	5.2414(53.1406)	5.3448(47.3828)
Probability $\geq 2\%$	4.8636(53.1406)	5.1364(47.3828)
Probability $\geq 3\%$	4.3333(53.1406)	5(47.3828)

Table 2. Average Codeword Length for Important Pixels Only(All Pixels) Using Gaussian Weight Function

Importance	Modified Huffman	Huffman
Intensity ≥ 200	31.5439(47.3672)	31.7544(47.3828)
Intensity ≥ 210	32.9574(47.3672)	33.0851(47.3828)
Intensity ≥ 220	32.9459(47.3672)	32.9730(47.3828)
Intensity ≥ 230	27.9259(47.3672)	27.9630(47.3828)

Table 3. Average Codeword Length for Important Pixels Only(All Pixels) Using Square Weight Function

Importance	Modified Huffman	Huffman
Intensity ≥ 200	22.9298(121.7031)	31.7544(47.3828)
Intensity ≥ 210	24.1915(121.7031)	33.0851(47.3828)
Intensity ≥ 220	24(121.7031)	32.9730(47.3828)
Intensity ≥ 230	19(121.7031)	27.9630(47.3828)

As you can see from table 1, 2, and 3, our modified Huffman coding method has in fact reduced the number of bits required to encode pixels that are defined to be important. Note that the average codeword length in table 1 is much smaller than the length in table 2 and 3, which is expected since they have different definitions on pixels that are important. Standard Huffman coding encode symbols that have high probability of occurrence with less bits, and modified Huffman coding with distributive weight function improved upon standard huffman coding by assigning even more weight on those pixels, thus these pixels end up with having codeword with smaller length. On the other hand, pixels with intensity value greater than 200 are less likely to occur, thus they require more bits to encode. Regardless of the definitions of important pixels, our modified Huffman coding algorithm has managed to encode important pixels with less bits than standard Huffman coding. Different weight functions reduced the average codeword length for important pixels by different amount, square weight function has the most significant bits reduction on important pixels however the average codeword length for all pixels is increased a lot in order to compensate for the bits reduction. On the flip side Gaussian weight function may not reduced the codeword length for important pixels by a huge amount, but it also has managed to reduce the codeword length for all pixels by a very small amount. We can conclude that modified Huffman coding algorithm indeed achieve the effect of encoding important pixels with less bits, but sometimes average codeword length for all pixels are increased to compensate for that effect.

The result of compression using proposed algorithm has shown in figure 2. We have tested the performance of using different weight function on Lenna image's magnitude spectrum, and we have found that Gaussian weight function results in the best compression ratio and overall quality. We used uniform weight function on phase spectrum since all pixel values are seem uniformly distribution in the area of interest, so it makes sense to assign them with equal weights.

Table 4. Quality Metrics for Reconstructed Image

MSE	SNR	Compression Ratio
73.7925	48.1172	9.2177:1

(a) Original Image (b) Reconstructed Image

Fig. 2. Original and Reconstructed Image Comparison

Table 4 and figure 2 have shown that reconstructed image have great compression ratio while persevering reasonable image quality.

4. NOVELTY

Our proposed algorithm was based on the intuition that pixels in images should not be treated equally. There are many image transform methods have suggested that most of image energy was concentrated on a very small region in the frequency domain, it is only natural to take extra care to those pixels fall in important regions during image compression or transmission. Due to this reason, Huffman coding is modified in a way such that pixels are treated unequally during encoding process. Given a user defined pixel importance, more important pixels or pixels that have more significant impact on reconstructing original image are encoded with less bits, whereas,

less important pixels are encoded with more bits. There are several benefits of doing this, the first benefit is that such encoding manner would give important pixels more resistance to error during transmission. For a noisy discrete memoryless channel, errors during transmission are almost inevitable, however, less bits for important pixels can certainly reduce the chance of errors occur to those pixels. Codewords with more bits are more likely to contaminate with errors during transmission since they will need to spend more time on the channel and they will have more opportunities to deal with errors because of their length. In opposition, codewords with less bits are less likely to catch an error and even if they did their resistance to error is much higher than those with more bits because code with less bits has narrower value range. For example, a 2 bits binary code has value range between 0 and 3 thus even if there is an error during transmission, the codebook can very likely recover from the error by finding its closest value, which is impossible for codewords with long length. In this way important pixels are more protected during transmission, and less important pixels are more exposed to error, which is acceptable as they don't contribute too much to image quality. In addition to error resistance, compression ratio can be improved significantly if there is a way to extract only important pixels from images. Although, in our proposed method a simple crop operation is used to extract important pixels, but there are still many insignificant pixels in that cropped image which are encoded with many bits. If we can extract only the those important pixels, for example, the cross area with bright pixels, then we might be able to further improve the compression ratio and image quality.

5. CONCLUSION

In this paper we have introduced a new image compression method which encodes pixels according to their significance. Simulation results have shown that proposed method indeed encodes important pixels with less bits than before, however the bits reduction on important pixels by different weight functions has in various degree influence on average codeword length for all pixels. Despite the fact that proposed method may comes with side effect, nevertheless, it still has space for improvement and great potential.

6. REFERENCES

[1] David JC MacKay, David JC Mac Kay, et al., *Information theory, inference and learning algorithms*, Cambridge university press, 2003.

[2] David A Huffman, "A method for the construction of minimum-redundancy codes," *Proceedings of the IRE*, vol. 40, no. 9, pp. 1098–1101, 1952.

[3] MTCAJ Thomas and A Thomas Joy, *Elements of information theory*, Wiley-Interscience, 2006.