Local Frequency Table for Context-based Adaptive Arithmetic Coding in Text Compression

Yi-Lin Tsai and Jian-Jiun Ding, *Senior Member, IEEE*
National Taiwan University, Taipei, Taiwan
jjding@ntu.edu.tw

***Abstract*-- Adaptive arithmetic coding is widely adopted in text compression. However, to achieve a high coding gain, an advanced way for context assignment is required. In this paper, the local frequency table technique is adopted to improve the coding efficiency of context-based adaptive arithmetic coding in text compression. With it, the probability distribution of the characters to be encoded can be estimated more accurately and a higher compression ratio can be achieved. Experimental results show that the proposed coding algorithm with a local frequency table outperforms other methods and achieves better compression performance for text compression.**

I. Introduction

Text compression is to reduce the number of bits to represent the text. Since the characters in the real world is rarely uniformly distributed, we can compress a text by removing statistical redundancy. Since any loss in the text is not tolerated, all text compression algorithms are lossless.

For a lossless compression algorithm, entropy coding plays a critical role. There are several well-known entropy coding methods, including Huffman coding [1], static arithmetic coding (SAC) [2], adaptive arithmetic coding (AAC) [3], and context-modeling based adaptive arithmetic coding (CAAC) [4, 5]. The Huffman code encodes each input independently and the SAC requires extra bits to encode the probability of each characters, their coding efficiencies may not be as high as that of AAC. CAAC is a further improvement of AAC. It determines the context from the surrounding data in the causal part and uses different frequency table for different contexts to achieve an even higher coding efficiency.

In this work, we apply the technique of the local frequency table [6, 7] to further improve the coding efficiency of AAC or CAAC. Moreover, a more advanced way to assign the context for text encoding is also proposed. The local frequency table is constructed based on the global frequency table. However, if it is estimated that the probability of a certain symbol to be encoded is different from that derived from the global frequency table, the local frequency table adjusted from the global frequency table is used to encode the symbol. Experiments show that, with the local frequency table and the proposed context assignment method, an even higher compression ratio for text compression can be achieved.

II. Proposed Algorithm

For the original CAAC algorithm, initially, we set the interval variables as $L = 0$ and $U = 1$ and initialize the frequency table as $T_C[a_k] = 1$ for all C and k where c means the context and $a_1, a_2, \ldots, a_K$ are the possible values of the input symbol. When encoding the i^{th} input (denoted by x_i), we first estimate the context c from the causal surrounding inputs. If $x_i = a_j$, then, L and U are adjusted by

$$L_{next} = L + (U - L)P(X \le a_{j-1}), \quad (1)$$

$$U_{next} = L + (U - L)P(X \le a_j). \quad (2)$$

where $$P(X \le a_j) = \sum_{k=1}^{j} T_C[a_k] \,/\, \sum_{k=1}^{n} T_C[a_k]. \quad (3)$$

Then, the frequency table T_C is adjusted by

$$T_C[a_i] = T_C[a_i] + 1. \quad (4)$$

The above process is repeated until all the inputs have been addressed. Finally, if

$$B \cdot 2^{-N} \le L < U \le (B+1) \cdot 2^{-N} \quad (5)$$

for some integers B and N, then the input sequence is encoded by B in terms of N bits.

In the proposed algorithm, we apply two techniques to improve the coding efficiency of CAAC for text encoding. If X_{i-1} is a consonant, we set the context to c_1. If X_{i-1} is a vowel, then the context is set to c_2. If $i = 1$ or X_{i-1} is a space and X_{i-2} is a period, a question mark, or an exclamation mark, then the context is c_3. If X_{i-1} is a space and X_{i-2} is neither a period, a question mark, nor an exclamation mark, then the context is c_4. If X_{i-1} is a number, then the context is c_5. In other cases, the context is set to c_6. The idea is based on the fact that the probability distribution of the current character is highly dependent on the previous character. To elaborate, for c_1 and c_2, if the preceding character is a consonant, then the input character is more likely to be a lowercase vowel. If the previous one is a vowel, then the input character is more likely to be a lowercase consonant. For c_3, if the last character is a space and the last two character is a period, a question mark, or an exclamation mark, then the input character is more likely to be an uppercase letter. For c_5, if the preceding character is a number, then the input character is more likely to be a number.

Moreover, we also apply the frequency table to improve the coding efficiency of CAAC. Although six contexts have been adopted, they are still not specific enough to cover all the cases where the probability distribution of the characters is changed. For instance, when X_{i-1} is "q", the probability of X_i being "u" should be much higher than other characters. Nevertheless, CAAC predicts the probability of X_i by T_1, which is the frequency table for the case where X_{i-1} is a consonant, instead of specifying for the case where X_{i-1} is "q". Hence, the probability of "u" obtained from T_1 is much less than the real probability. Accordingly, using only CAAC still cannot predict the probability of the input character accurately.

Increasing the number of contexts cannot solve the problem. It costs lots of memory. Moreover, the probability distribution estimated by the frequency table well approximates the true probability distribution only if the amount of data is enough. If there are too many contexts, the amount of data for each context is dropped, which leads to a poor estimated probability.

The proposed local frequency table method is described as follows. We denote the frequency tables corresponding to the six contexts as $\{T_1, T_2, \ldots, T_6\}$ and denote the local frequency table as T_L. Assume that we are encoding the i^{th} character X_i. Let E be an event that occurred in the causal part (e.g. X_{i-1} is the character "q"), and b be the character whose probability should be adjusted (e.g. b is the character "u"). Then we set

$$T_L[b] = kT_C[b], \qquad T_L[a] = T_C[a] \quad \text{if } a \neq b, \tag{6}$$

$$k = \frac{P_2}{P_1}, \tag{7}$$

where P_1 is the probability estimated from the original frequency table is and we predict that the real conditional probability of b given the event E is $P_2 = P(X = b|E, C)$. We use the local frequency table to encode the input data. However, this local frequency table is only applied to encode the current symbol and we still use the same method as that of CAAC to adjust the original frequency table (treated as the global frequency table). Therefore, the global frequency table is the same as that of CAAC.

The local frequency table does not cost an extra memory. Meanwhile, a local frequency table considers more contexts than using only CAAC, which makes the probability obtained from a local frequency table closer to the true probability. Hence, we can further enhance the compression rate without increasing the memory requirement.

In our algorithm, to well adopt the technique of the local frequency table, first, we counted the frequency of the characters following the first-order and second-order contexts from training texts. The order of the context means the number of characters to determine the context. Then, the conditions that the frequency table should be applied are as follows. The first condition is that the character is the most frequent one in the context. The second one is that the ratio of the character frequency in the context to the total frequency of the context reaches the threshold.

For example, for the case where "q" is the first-order context, the mechanism of the local frequency table can be applied to the character "u" because "u" is the most frequent following character of "q" and the ratio of the frequency of "qu" to the frequency of "q" is more than the threshold. We trained the model with 12 training texts and chose parameters to minimize the code length. In sum, there are 213 cases where the local frequency table is applied, including 32 first-order context cases and 181 second-order contexts cases. For each case, we determine the parameters k from (7) for each case.

III. Simulations

In Table I, we selected six books to evaluate the compression performance of each method for text encoding.

TABLE I
Performance Comparison for Text Encoding (in Terms of Bits per Character)

Text Name	Huffman	SAC	AAC	CAAC	**Proposed**
Thank you for arguing	4.72	4.68	4.64	4.14	**3.87**
The 7 Habits of Highly Effective People	4.55	4.52	4.48	4.02	**3.79**
What Money Can't Buy	4.58	4.56	4.52	4.03	**3.79**
Normal People	4.54	4.52	4.48	3.96	**3.68**
Wealth, Poverty, and Politics	4.52	4.50	4.47	3.98	**3.74**
Where the Crawdads Sing	4.58	4.54	4.51	3.99	**3.72**
Average	4.58	4.55	4.52	4.02	**3.76**

All of these books contain 200,000 characters and their ASCII values range from 0 to 127. The results show that the proposed method much outperforms other methods. Compared to AAC, the improve rates of the proposed algorithm ranges from 15.4% to 17.9%. Compared to CAAC, which applies the six contexts in Section II but does not apply the local frequency table, the improve rate ranges from 5.72% to 7.07%.

IV. Conclusion

In this work, two techniques are applied to improve the coding efficiency of text compression. First, we classify the causal part into six contexts and use six frequency tables to estimate the probability distribution under these contexts. Moreover, to well address the specific conditions without increasing the number of contexts, the local frequency table is applied. It can handle more specific contexts and bridge the gap between the probability predicted by the global frequency table and the real probability. With these techniques, the coding efficiency of text compression can be much improved.

References

[1] A. Moffat, "Huffman coding," *ACM Computing Surveys*, vol. 52, no. 4, pp. 1-35, 2019.

[2] S. Shanmugasundaram and R. Lourdusamy, "A comparative study of text compression algorithms," *Int. J. Wisdom Based Computing*, vol. 1, no. 3, pp. 68-76, Dec. 2011.

[3] A. Masmoudi, W. Puech, and M. S. Bouhlel. "Efficient adaptive arithmetic coding based on updated probability distribution for lossless image compression," *J. Electronic Imaging*, vol 19, no. 2, article 023014, Apr. 2010.

[4] J. Wang, X. Ji, S. Zhao, X. Xie, and J. Kuang. "Context-based adaptive arithmetic coding in time and frequency domain for the lossless compression of audio coding parameters at variable rate," *EURASIP J. Audio, Speech, and Music Processing*, vol. 2013, pp. 1-13, May 2013.

[5] S. Kim and N. I. Cho, "Hierarchical prediction and context adaptive coding for lossless color image compression," *IEEE Trans. Image Processing,* vol. 23, no. 1, pp. 445-449, Jan. 2014.

[6] J. J. Ding, I. H. Wang, and H. Y. Chen, "Improved efficiency on adaptive arithmetic coding for data compression using range-adjusting scheme, increasingly adjusting step, and mutual-learning scheme," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 28, issue 12, pp. 3412-3423, Dec. 2018.

[7] J. T. Wu and J. J. Ding, "Improved angle freeman chain code using improved adaptive arithmetic coding," *IEEE Asia Pacific Conf. Circuits and Systems*, Ha Long Bay, Vietnam, pp. 181-184, Dec. 2020.