Experiment 2 Report- Lossless Compression

R Basheer Ahammad - 22EE65R19

March 18, 2023

1 Dataset - 1

1. Huffman coding is the method of compressing data. It provides a code by analyzing the frequencies of symbols in a message or text. Symbols that have high frequency will be encoded as a shorter-bit string while symbols with low frequency will be encoded as longer-bit strings

2. Here, in the dataset 1 with a text lengths of N =[50,100,500,1000,5000], we generated 10 textfiles with each N and the name given as file-nn-ii.txt, where n is the text size and i is the text file number.

3. The time taken by each text file for listing of sizes in list N are listed below:

   - For N=50 the encoding times taken for each text file are
   - 0.0015,0.0029,0.0027, 0.0,0.0020,0.0,0.0055,0.0,0.0089, 0.0
   - For N=50 the decoding times taken for each encoded file are
   - 0.0025, 0.0, 0.0, 0.0, 0.0016, 0.0, 0.0, 0.0, 0.0017, 0.0
   - For N=100 the encoding times taken for each text file are
   - 0.0, 0.0, 0.0028, 0.0, 0.0, 0.0, 0.0075, 0.0055, 0.0068, 0.0
   - For N=100 the decoding times taken for each encoded file are
   - 0.0, 0.0080, 0.0, 0.0, 0.0086, 0.0, 0.0, 0.0, 0.0, 0.0

4. From the above observations, Huffman encoding compression takes more time than decompression because the encoding process involves building a frequency table for the input data, constructing a Huffman tree based on the frequency table, and then encoding the input data using the generated Huffman codes. This process can be time consuming mainly for large input data sets.

5. Huffman decoding involves simply traversing the Huffman tree based on the encoded input data and decoding each symbol as we reach its corresponding leaf node. This process is generally faster than encoding, as it involves only simple tree traversal and decoding operations.

6. Therefore, the time taken for the compression and decompression depends on size of the input data.

7. The mean square error obtained for each text file is zero which is observed in the code

8. The Notch box plots for Compression factor vs N, Encoding time vs N and Decoding time vs N are shown below

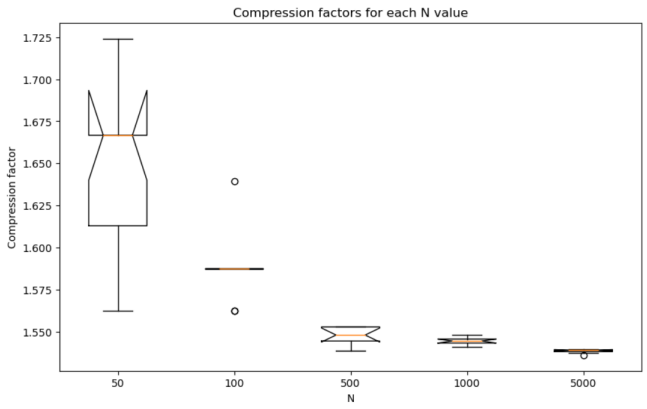

Figure 1: Notch box plot of Compression factor vs N

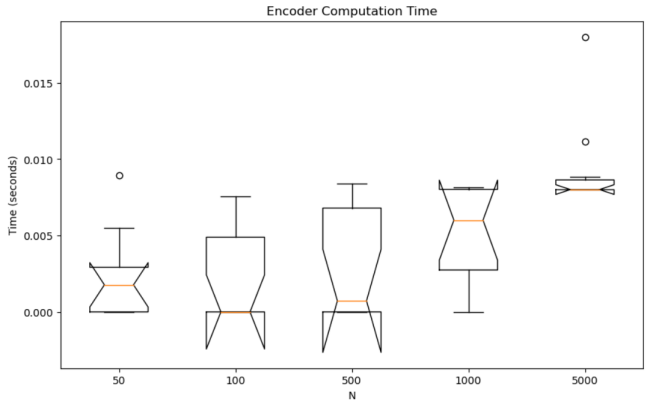

Figure 2: Notch box plot for Encoder computation time vs N

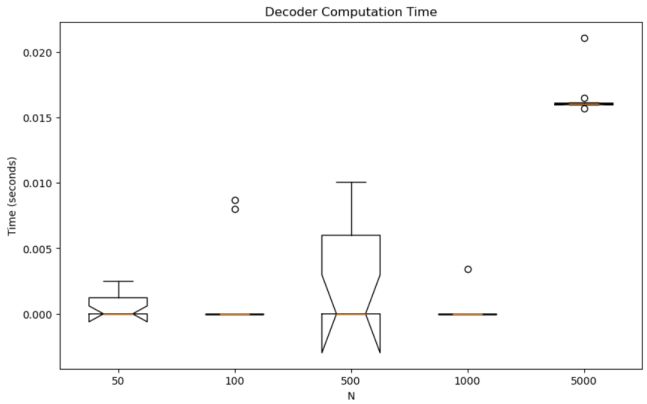

Figure 3: Notch box plot for Decoder computation time vs N

2 Dataset - 2

1. Huffman coding for image data set is done as follows
    - The image is first divided into small blocks of pixels, typically 8x8 or 16x16 pixels
    - Each block is then transformed into a frequency distribution of the color intensities of the pixels.
    - The Huffman algorithm is applied to each frequency distribution to generate a unique variable-length code for each color intensity value.
    - The codes are then used to represent the color intensities of the pixels in the image
    - The resulting compressed image file contains the codes for each block, along with information about the codebook used to decode them.
2. The encoding times for compressed image in olivetti dataset are 0.0082, 0.0152, 0.0104, 0.0098, 0.0101, 0.0146, 0.0100, 0.0133, 0.0085, 0.0099.
3. The decoding times for compressed image in olivetti dataset are 0.0333, 0.0261, 0.0273, 0.0301, 0.0331,0.0252, 0.0314, 0.0250, 0.0241, 0.0281.
4. From the above observations, Huffman encoding compression takes more time than decompression because the encoding process involves building a frequency table for the input data, constructing a Huffman tree based on the frequency table, and then encoding the input data using the generated Huffman codes. This process can be time consuming mainly for large input data sets.
5. If the image contains a lot of random noise or has a wide range of color intensities, then Huffman encoding may not be as effective.
6. The Compression factor Variation in shown in figure 4
7. The scatter plot for compression factor on x-axis and encoder time on y-axis over all images are shown in below figure 5
8. The scatter plot for compression factor on x-axis and decoder time on y-axis over all images are shown in below figure 6

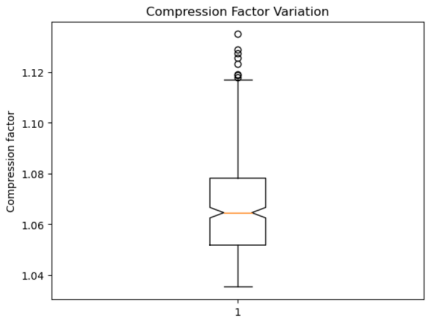

Figure 4: Notch box plot of Compression factor Variation

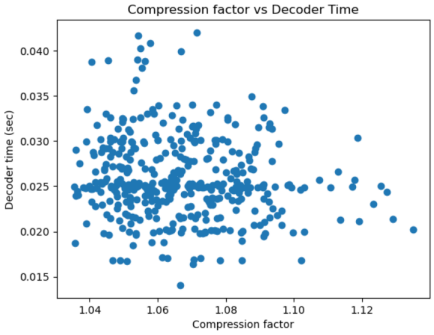

Figure 5: Scatter plot for Encoder computation time vs Compression time

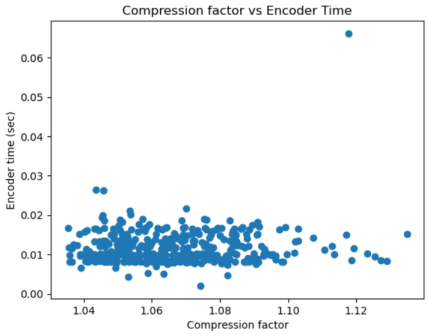

Figure 6: Scatter plot for Decoder computation time vs Compression time

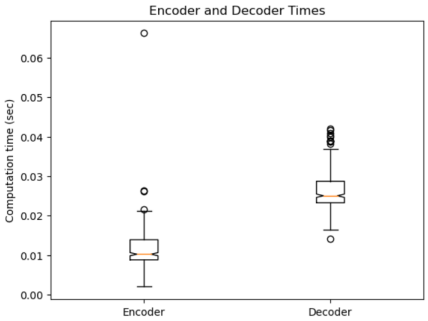

Figure 7: Notch box plot for Encoder time and Decoder time vs Compression time