Building data compression application using Huffman coding

Abstract

As civilization evolves, the data is in exponential growth. The ubiquity of Internet-based applications, coupled with emerging technologies like the Internet of Things (IoT), artificial intelligence, cloud computing, and machine learning, has led to an unprecedented surge in the volume of data transmission and storage. Managing this influx of data has become a paramount challenge. Scientists and engineers are working closely to develop new and improve existing technologies used to compress the data to increase the speed of transmission, reduce the cost of storage, and ultimately reduce the total cost of the maintenance of the data. Although there are numerous ways of compressing the data, this research paper focuses on the pivotal role of Huffman coding in addressing the challenges posed by rapidly increasing data.

Introduction:

Establishing the significance of data compression in the contemporary digital landscape. Highlighting the surge in data volumes propelled by IoT, artificial intelligence, and machine learning applications.

Background:

Even though many believe that data compression is a new branch in the field of technology, the reality is that the first signs of data compression are coming from 1838 when the Morse code was invented. As the earliest example of data compression, morse code used shorter codewords for letters such as "e" and "t" which are very common in the English language. More than 100 years later, the modern era of data compression began with a significant development of the information theory. In 1977, Abraham Lempel and Jacob Ziv proposed the basic concept of pointer-based encoding. By the mid-1980s, after work by Terry Welch, the LZW algorithm quickly became the preferred method for most general-purpose compression systems. It found use in various programs like PKZIP and hardware devices like modems. As digital images became more widespread in the late 1980s, standards for compressing them started emerging. In the early 1990s, lossy compression methods also gained widespread use. Current image compression standards include FAX CCITT 3 (using run-length encoding with codewords

determined by Huffman coding based on a specific distribution of run lengths), GIF (LZW), JPEG (lossy discrete cosine transform followed by Huffman or arithmetic coding), BMP (using run-length encoding, etc.), and TIFF (FAX, JPEG, GIF, etc.). Typical compression ratios achieved for text are approximately 3:1, for line diagrams and text images around 3:1, and for photographic images around 2:1 for lossless compression and 20:1 for lossy compression. [3 REFERENCE TODO]

This paper focuses on Huffman Coding which originally started in 1951. That year, David A. Huffman and his classmates at MIT studying information theory had to choose between writing a term paper or taking a final exam. Robert M. Fano was the professor who assigned them a term paper about finding the best and most efficient way to create a binary code. Huffman was stuck for quite some time, unable to prove that any of the given codes were the best choice. He almost gave up, and considered studying for the final exam instead. Suddenly, he had a breakthrough idea. Instead of using the traditional method, Huffman thought of organizing the binary code based on how often each symbol appeared (this will be discussed more in Fundamentals of Huffman Coding and Entropy). He quickly proved that this approach, using a frequency-sorted binary tree, was the most efficient way. In achieving this, Huffman surpassed Fano, who, along with Claude Shannon, had developed a similar code. Huffman's bottom-up method of building the tree guaranteed optimality, a better approach than the top-down method used in Shannon–Fano coding. [4]

Reference for this [3] - Stephen Wolfram, A New Kind of Science (Wolfram Media, 2002), page 1069.

Tracing the historical evolution of data compression techniques.
Introducing Huffman coding as a prominent solution in the realm of lossless compression.
Exploring the theoretical foundation of Huffman coding in relation to information entropy.

Fundamentals of Huffman Coding and Entropy:

Detailing the core principles and mechanisms behind Huffman coding.
Establishing the connection between Huffman coding and information entropy.
Exploring how Huffman coding achieves entropy-based compression by assigning shorter codes to more frequent symbols.

Applications of Huffman Coding:

Analyzing real-world scenarios where Huffman coding finds practical utility.

Showcasing its effectiveness in diverse domains such as image and text compression.

Performance Evaluation:

COMPARE How it works with compression and without compression on 100k requests.
Presenting empirical studies and performance comparisons with other compression algorithms.
Discussing the trade-offs and limitations associated with Huffman coding.

Future Directions:

MAYBE MENTION GRPC?

Identifying potential advancements and avenues for further research in Huffman coding.
Considering its adaptability to emerging technologies and evolving data landscapes.

Conclusion:

Summarizing the key findings and contributions of the research.
Emphasizing the ongoing relevance and future prospects of Huffman coding in the dynamic field of data compression.

Sources

1. [General about data compression](#)
2. [Data compression use cases](#) (Here also mentioned Huffman coding, nice for transition)
3. Stephen Wolfram, A New Kind of Science (Wolfram Media, 2002), page 1069.
4. Huffman, Ken (1991). "Profile: David A. Huffman: Encoding the "Neatness" of Ones and Zeroes". Scientific American: 54–58.