Improved Lossless Compression with Adaptive Prediction

Rahul Gupta (202211069)
Ajaydeep Rawat (202211001)
Under the Guidance of: Dr. Jignesh Patel

1 Core Idea

This project enhances a baseline lossless compression algorithm. The original method uses simple **inter-column differencing** (comparing a pixel only to its left neighbor) and a static **Canonical Huffman code table** to compress the difference.

The **improvement** was to replace this simple predictor with a more powerful, adaptive predictor: the **Median Edge Detection (MED)** predictor, which is used in the JPEG-LS standard.

2 The Improvement: Adaptive MED Predictor

The simple predictor ($D = p(i,j) - p(i,j-1)$) is not very efficient on images with vertical edges or complex textures.

The **MED predictor** is smarter. To predict the current pixel $p(i,j)$, it looks at its three neighbors:

- A: The pixel to the left.
- B: The pixel directly above.
- C: The pixel to the upper-left (diagonal).

It then uses these three values to make an intelligent guess (p_x) that "detects" and follows edges:

$$p_x = \begin{cases} \min(A,B) & \text{if } C \geq \max(A,B) \text{ (detects vertical edge)} \\ \max(A,B) & \text{if } C \leq \min(A,B) \text{ (detects horizontal edge)} \\ A + B - C & \text{otherwise (smooth area)} \end{cases}$$

This prediction is much more accurate, so the *error* (or "residual") $R = p(i,j) - p_x$ is much smaller. Compressing these smaller, more-centered residuals leads to a better compression ratio.

3 Updated Methodology

The algorithm's 3-stage process was modified to support this new predictor:

1. **Offline Training:** The static Huffman code table is now built from the frequencies of **MED residuals** calculated from the training images.
2. **Encoding:**
   - The algorithm calculates the MED residuals for the input image.
   - It encodes these residuals using the pre-built Huffman table.
   - **Crucially, it adds a 1-byte "predictor flag"** to the start of the compressed file. This flag tells the decompressor that the file was compressed using MED.
3. **Decoding:**

- The decompressor first reads the 1-byte predictor flag.
- Seeing that MED was used, it decodes the residuals from the bitstream.
- It then perfectly reconstructs the image by applying the MED formula in reverse: $p(i, j) = p_x + R$.

4 Results and Analysis

The performance of the baseline algorithm (Inter-column differencing) and the improved algorithm (Median Edge Detection) were directly compared by compressing the same validation image, `FLIR_08863.tiff`. The training phase for both methods used the same 53 training images.

4.1 Performance Comparison

The key metrics from the validation test are summarized in Table 1.

Table 1: Compression Performance Comparison for `FLIR_08863.tiff`

Metric	Base Paper (Inter-column)	Improved Paper (MED)
Phase 1: Training		
Analyzed Images	53	53
Generated Code Table Entries	303	303
Phase 2: Validation		
Processing File	FLIR_08863.tiff	FLIR_08863.tiff
Original Size	316,186 bytes	316,186 bytes
Compressed Size	**215,379 bytes**	**205,238 bytes**
Compression Ratio	**1.47:1**	**1.54:1**
MSE	**0.000000 (LOSSLESS)**	**0.000000 (LOSSLESS)**

4.2 Analysis and Conclusion

The results in Table 1 clearly show that the **improved paper (MED) is superior**.

- **Higher Compression:** The improved MED-based method achieved a **1.54:1** compression ratio, which is measurably better than the baseline's **1.47:1** ratio. This resulted in a compressed file that is over 10,000 bytes smaller (205,238 vs. 215,379 bytes) for the same input image.
- **Why it's better:** This improvement is because the MED predictor is "smarter" than simple inter-column differencing. It adapts to local image features like edges, resulting in more accurate predictions. More accurate predictions create smaller, more-centered residuals (errors), which have lower entropy and can be compressed more efficiently by the Huffman encoder.
- **Perfectly Lossless:** Both methods remain **fully lossless**, achieving a Mean Squared Error (MSE) of **0.0**.

In conclusion, by integrating a more intelligent, adaptive predictor, the algorithm's compression efficiency was significantly improved while retaining the high speed and perfect reconstruction of the original method.

Code Availability

Both the base paper implementation and the improved version are available at:

`https://github.com/ADVAYA1/Image-Compression`