Vectorización y compresión de imágenes. Descomposición RGB

Juan Sebastán Peñaloza Quintana
JsebastianDS@proton.me
Github: JsebastianUVPRQ

14 Marzo 2025

1 Conversión a Escala de Grises

La imagen original en color (RGB) se convierte a escala de grises mediante la combinación lineal:

$$I_{\text{gray}} = 0.2989 \cdot R + 0.5870 \cdot G + 0.1140 \cdot B \tag{1}$$

Aquí se combinan los tres canales en una única matriz que codifica la intensidad luminosa. La normalización de los valores al rango [0,1] es esencial para garantizar consistencia numérica, especialmente cuando la imagen original utiliza valores enteros en el rango [0,255]. La operación de conversión preserva la estructura espacial de la imagen mientras reduce su dimensionalidad, facilitando procesamientos posteriores.

Fundamento

- Los coeficientes ponderados reflejan la sensibilidad espectral del ojo humano (mayor peso en el canal verde)
- Normalización de valores al rango $[0, 1]$ cuando la imagen está en formato entero (8 bits por canal):

$$I_{\text{norm}} = \frac{I_{\text{gray}}}{255} \tag{2}$$

2 Generación de la Miniatura por Promediado de Bloques

Procedimiento

1. **Recorte de la imagen**: Ajuste de dimensiones para divisibilidad por el tamaño de bloque $b \times b$:

$$H_{\text{new}} = H - (H \mod b) \tag{3}$$
$$W_{\text{new}} = W - (W \mod b) \tag{4}$$

2. **División en bloques**: Reorganización matricial:

$$\text{blocks} = \texttt{reshape}\left(I_{\text{recortada}}, \left(\frac{H_{\text{new}}}{b}, b, \frac{W_{\text{new}}}{b}, b\right)\right) \tag{5}$$

Transposición para agrupación espacial:

$$\text{blocks} = \texttt{transpose}(0, 2, 1, 3) \tag{6}$$

3. **Promediado espacial**: Cálculo por bloque:

$$\text{thumbnail}[i,j] = \frac{1}{b^2} \sum_{x=0}^{b-1} \sum_{y=0}^{b-1} \text{blocks}[i,j,x,y] \tag{7}$$

Listing 1: Operaciones completas

```python
# Cargar la imagen
image = plt.imread('input.png')
# Convertir a escala de grises con operaciones de álgebra lineal
if image.ndim == 3:
    # Normalizar valores si la imagen está en formato entero (0-255)
    if np.issubdtype(image.dtype, np.integer):
        red = image[:, :, 0].astype(float) / 255.0
        green = image[:, :, 1].astype(float) / 255.0
        blue = image[:, :, 2].astype(float) / 255.0
    else:
        red, green, blue = image[:, :, 0], image[:, :, 1], image[:, :, 2]

    gray_image = 0.2989 * red + 0.5870 * green + 0.1140 * blue
else:
    gray_image = image.astype(float)
    if np.issubdtype(image.dtype, np.integer):
        gray_image = gray_image / 255.0
# ----------------
# Definir el tamaño del bloque para la miniatura
block_size = 2
# Ajustar dimensiones para que sean divisibles por block_size
H, W = gray_image.shape
H_new = H - H % block_size
W_new = W - W % block_size
cropped_image = gray_image[:H_new, :W_new]
# Dividir en bloques y calcular promedios
blocks = cropped_image.reshape(H_new // block_size, block_size, W_new // block_size, block_size)
blocks = blocks.transpose(0, 2, 1, 3) # Reorganizar para agrupar bloques
thumbnail = blocks.mean(axis=(2, 3)) # Promedio sobre los bloques
# ----------------
# Vectorizar la miniatura
thumbnail_vector = thumbnail.flatten()
# ----------------
# Visualización
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
plt.title('Imagen en Escala de Grises')
plt.imshow(gray_image, cmap='gray', vmin=0, vmax=1)
plt.subplot(1, 2, 2)
plt.title(f'Miniatura ({thumbnail.shape[0]}x{thumbnail.shape[1]})')
plt.imshow(thumbnail, cmap='gray', vmin=0, vmax=1)
plt.show()
# ----------------
print("Dimensiones de la miniatura:", thumbnail.shape)
print("Tamaño del vector:", thumbnail_vector.shape)
```

Listing 1: Operaciones completas

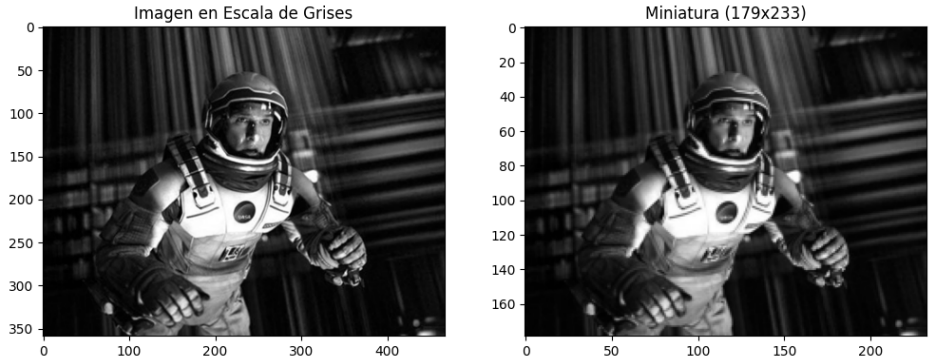

Figure 1: Se perdió información, siempre se pierde información.

3 Vectorización de la Miniatura

La vectorización transforma la matriz 2D de la miniatura en un vector 1D mediante la operación flatten(). Este proceso conserva toda la información de la miniatura pero la estructura en una única dimensión:

$$\mathbf{v} = \texttt{flatten}(\text{thumbnail}) \in \mathbb{R}^N \tag{8}$$

Para el caso de la celda ejecutada, se obtuvo:

- Dimensiones de la miniatura: $(179, 233)$
- Tamaño del vector: $(41707,)$

Propiedades del Vector Resultante

- Dimensionalidad:

$$N = \frac{H_{\text{new}} \cdot W_{\text{new}}}{b^2} \tag{9}$$

- Cada componente v_i representa el valor promedio de intensidad en el i-ésimo bloque
- Preserva la estructura espacial mediante ordenamiento por filas