

Laboratorio 3: Transformaciones de Imágenes

Marko Sumire Ramos

27 de mayo de 2025

1. Cálculo de Histogramas

Para calcular el histograma de una imagen, se implementó una función que recorre cada píxel de la imagen y cuenta la frecuencia de cada valor de intensidad. Se utilizó un vector de 256 elementos para almacenar las frecuencias de los valores de intensidad de 0 a 255. A continuación, se muestra el código utilizado para calcular el histograma:

```

1  vector<int> getHistogram(const Mat& img) {
2      vector<int> hist(256, 0);
3      for(int i=0; i.rows; i++) {
4          for(int j=0; j.cols; j++) {
5              hist[img.at<uchar>(i, j)]++;
6          }
7      }
8      return hist;
9 }
```

Luego de calcular el histograma, este se almacenó en un archivo csv para su posterior graficación en Python.

```

1 void saveHistogram(vector<int> hist, int n) {
2     string filename = "histogram" + to_string(n) + ".csv";
3     ofstream file(filename);
4     if (file.is_open()) {
5         for (int i = 0; i < 256; i++) {
6             file << i << "," << hist[i] << endl;
7         }
8         file.close();
9     } else {
10        cout << "Unable to open file";
11    }
12 }
```

Una vez calculados y almacenados los histogramas en archivos CSV, utilizando Python, se procedió a graficarlos, y contar cuantas intensidades distintas tienen frecuencias diferentes de cero segun el siguiente código:

```

1 csv_files = glob.glob('*.*.csv')
2 for file in csv_files:
3     df = pd.read_csv(file, header=None)
4     values = df.iloc[:, 0]
5     frequencies = df.iloc[:, 1]
6     non_zero_count = (frequencies != 0).sum()
7     plt.figure(figsize=(10, 6))
8     plt.bar(values, frequencies)
9     plt.title(f'Histogram for {file}')
10    plt.xlabel('Value')
11    plt.ylabel('Frequency')
12    plt.yscale('log')
13    plt.text(
14        0.95, 0.95, f'NonZero entries: {non_zero_count}',
15        transform=plt.gca().transAxes,
16        ha='right', va='top', fontsize=12, bbox=dict(boxstyle='round',
17                                         facecolor='white', alpha=0.8)
18    )
19    plt.show()
```

1.1. Histogramas de las Imágenes de Prueba

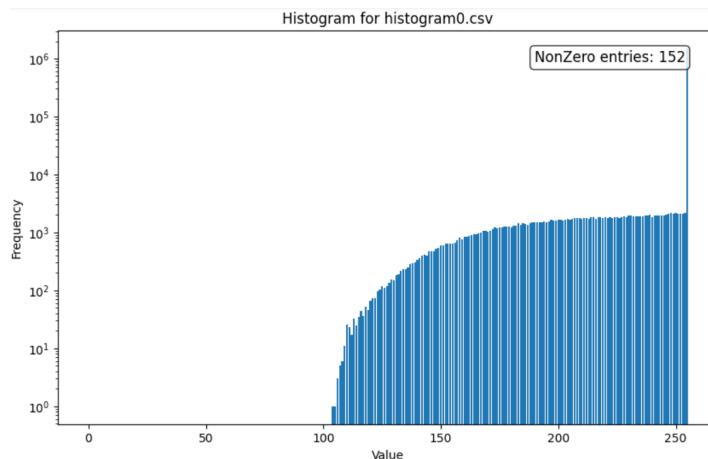


Figura 1: Histograma de la imagen 0

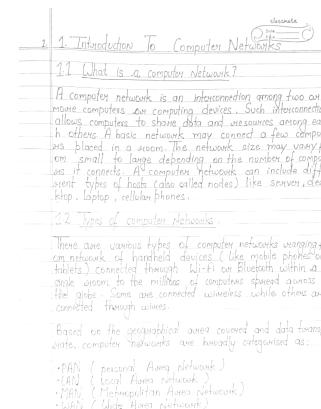


Figura 2: Imagen 0

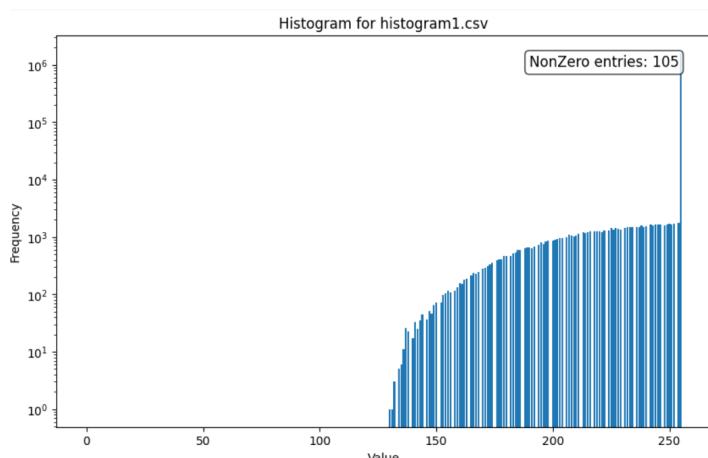


Figura 3: Histograma de la imagen 1

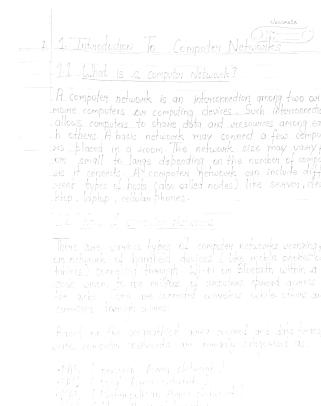


Figura 4: Imagen 1

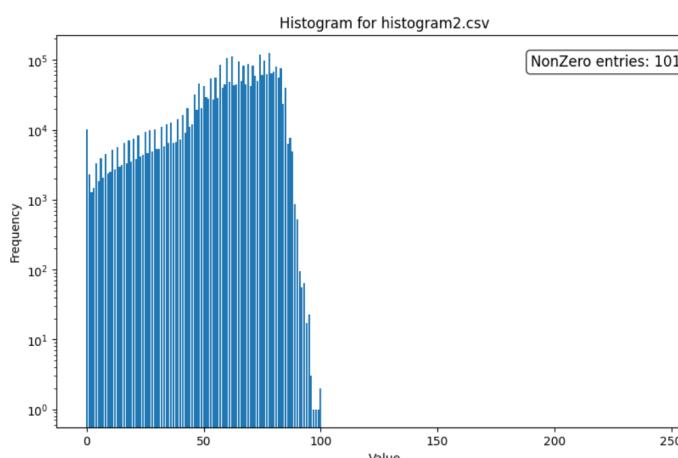


Figura 5: Histograma de la imagen 2

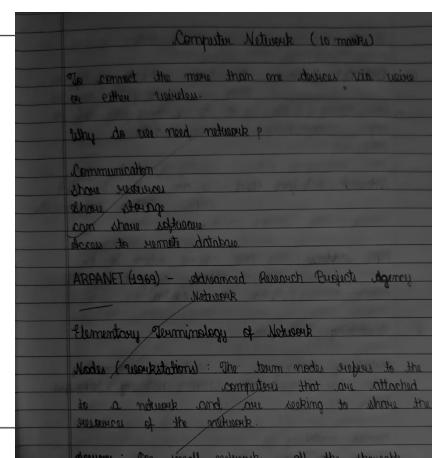


Figura 6: Imagen 2

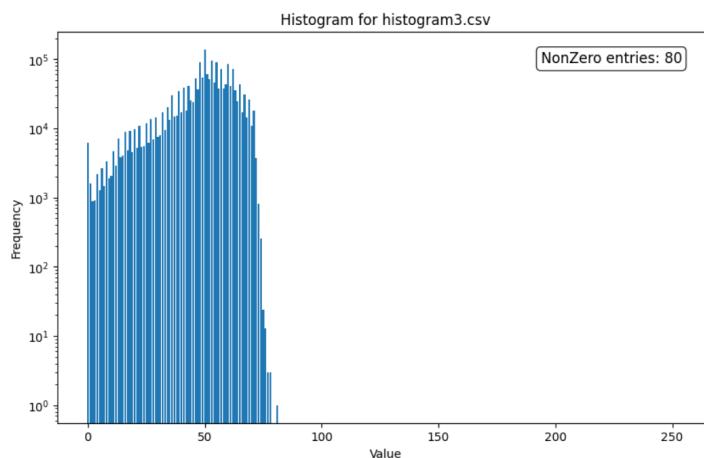


Figura 7: Histograma de la imagen 3

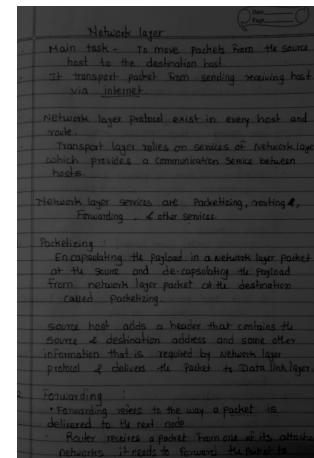


Figura 8: Imagen 3

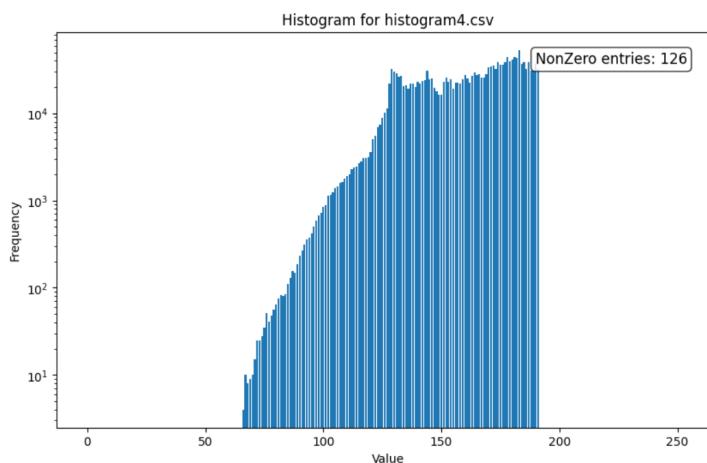


Figura 9: Histograma de la imagen 4

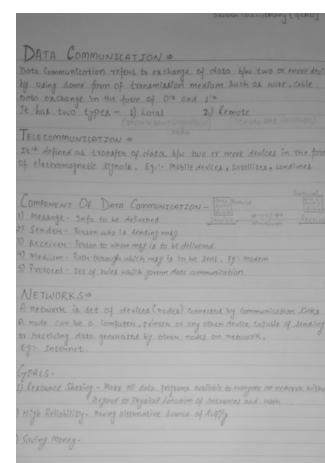


Figura 10: Imagen 4

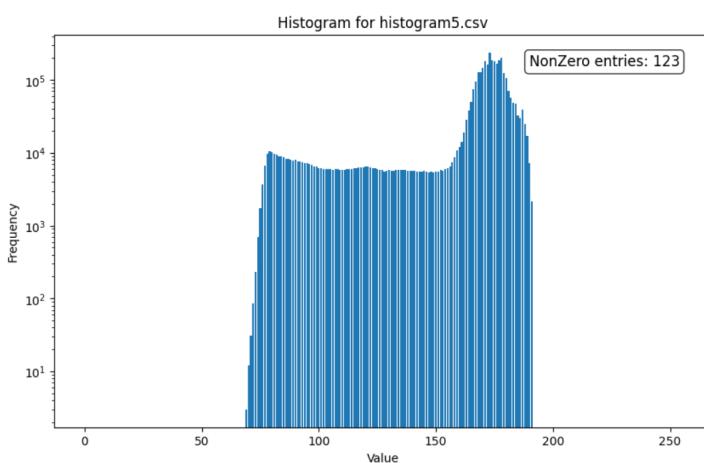


Figura 11: Histograma de la imagen 5

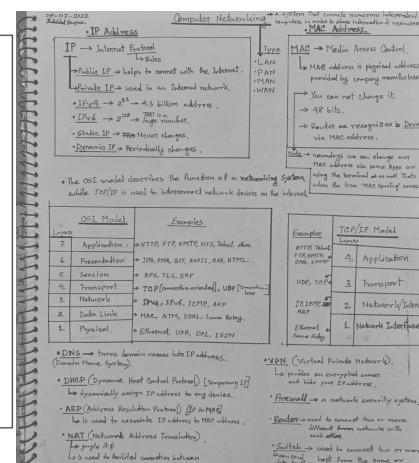


Figura 12: Imagen 5

2. Ecualización de Histograma

Para realizar la ecualización del histograma, se implementó una función que distribuye los valores de intensidad de manera uniforme a lo largo del rango de 0 a 255. Esta función utiliza el histograma

previamente calculado para ajustar los valores de intensidad de la imagen original, al finalizar retorna un vector que mapea cada valor de intensidad original a su nuevo valor ecualizado.

2.1. Ecualización de la imagen 0

La ecualización mapeo intensidades cercanas a 0 a las intensidades distintas de 255, lo que resulta en algo parecido a una binarización con umbral de 255, ademas la cantidad de intensidades usadas en la imagen resultante se reduce de 152 a 27. Esto es debido a la enorme frecuencia de la intensidad 255 relativa a las otras intensidades.

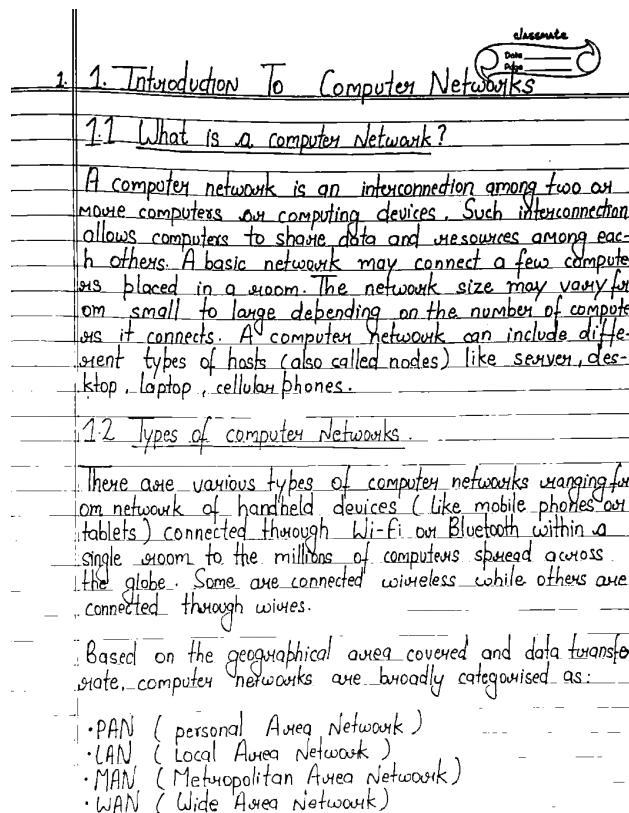


Figura 13: Imagen 0 ecualizada

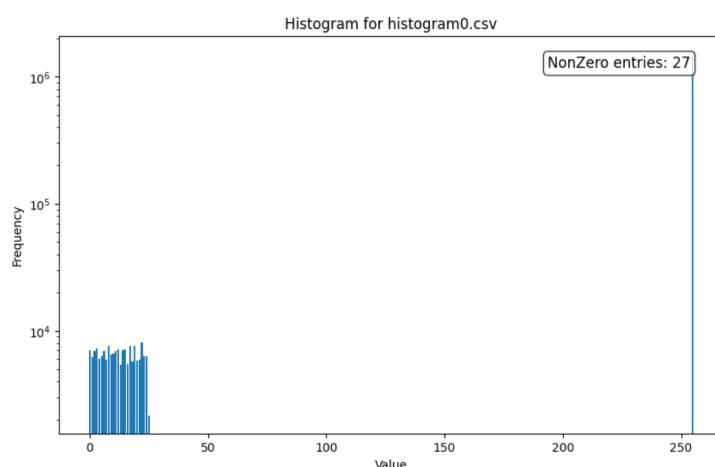


Figura 14: Histograma ecualizado de la imagen 0

2.2. Ecualización de la imagen 1

Un resultado similar al de la imagen 0, se obtuvo al ecualizar la imagen 1, en este caso la cantidad de intensidades usadas en la imagen resultante se reduce de 105 a 13.

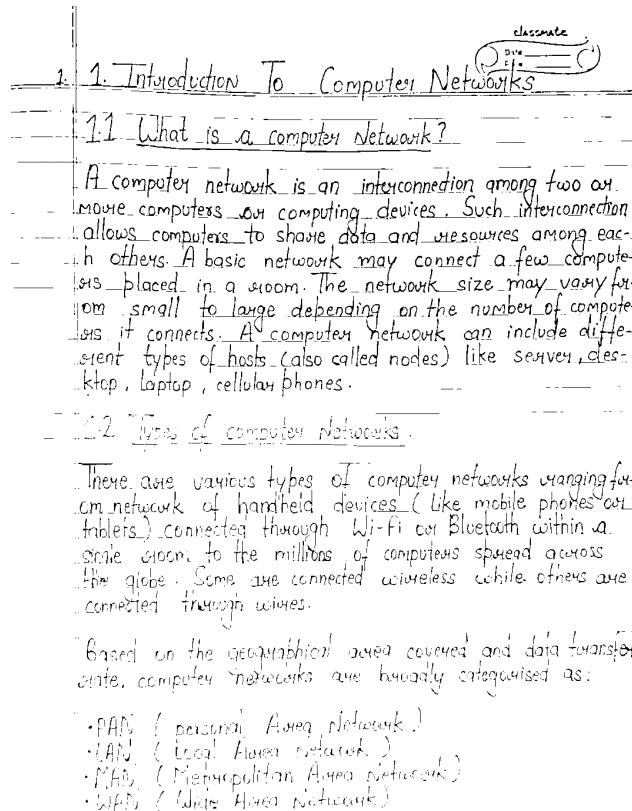


Figura 15: Imagen 1 ecualizada

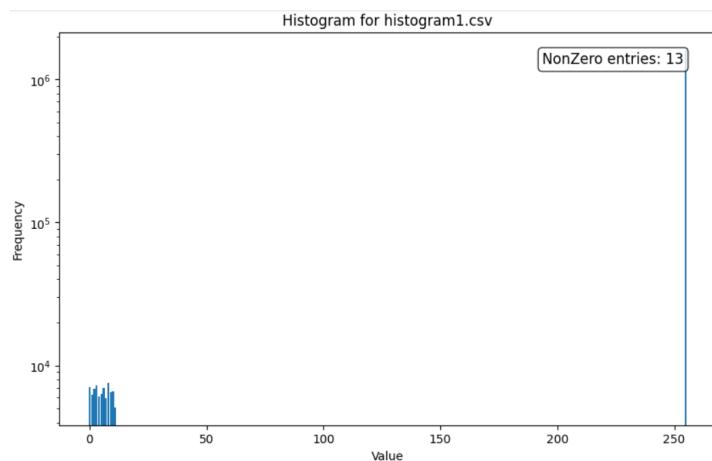


Figura 16: Histograma ecualizado de la imagen 1

2.3. Ecualización de la imagen 2

En la imagen 2, la ecualización hace que las intensidades se distribuyan de manera más uniforme aumentando así los contrastes en general a costa de reducir los contrastes en las intensidades cercanas

a 0 las cuales originalmente tenían una frecuencia alta, lo cual resulta en una especie de difuminación de las letras en las zonas más oscuras. También reduce la cantidad de intensidades usadas en la imagen resultante de 101 a 68.

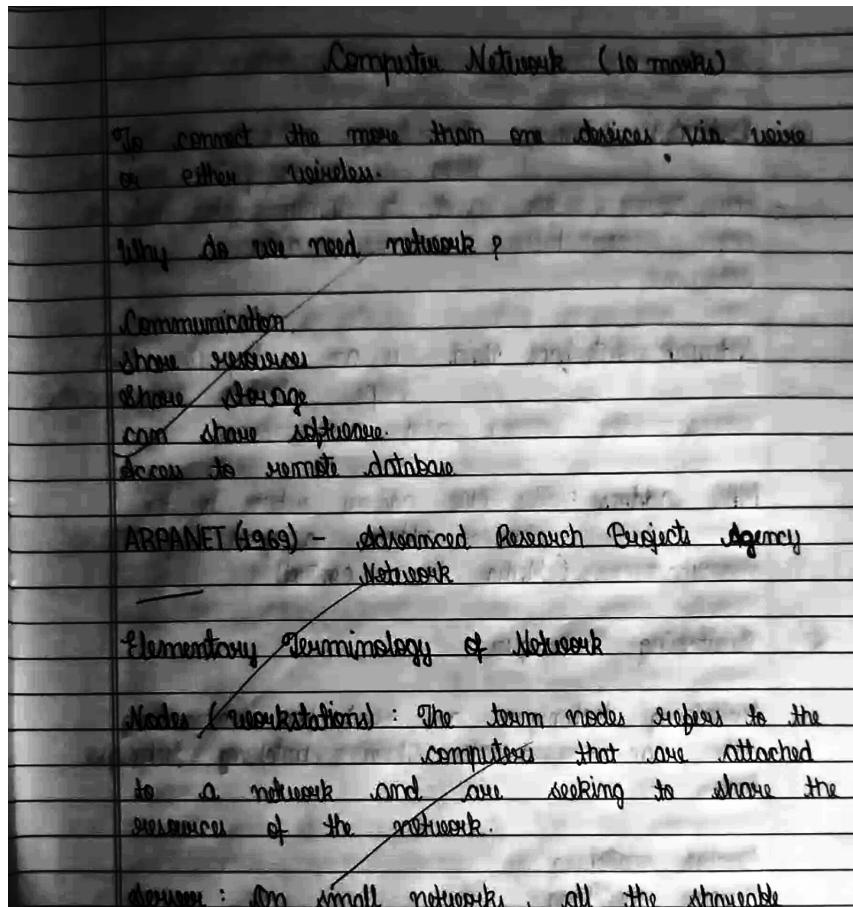


Figura 17: Imagen 2 ecualizada

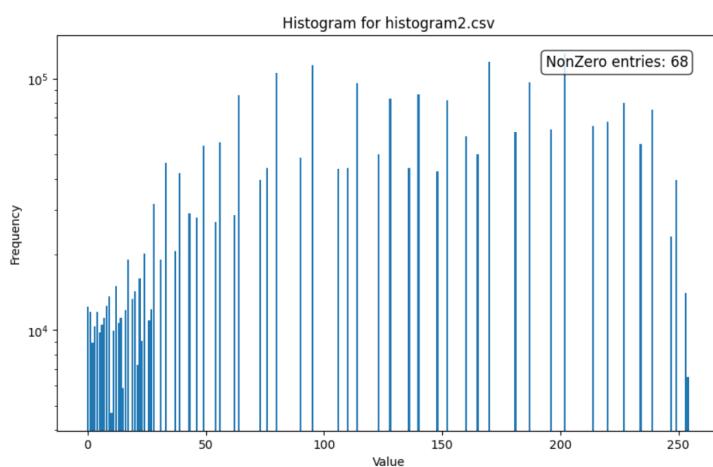


Figura 18: Histograma ecualizado de la imagen 2

2.4. Ecualización de la imagen 3

Un resultado similar al de la imagen 2, se obtuvo al ecualizar la imagen 3, aunque en este caso el efecto de difuminación es menos notorio. La cantidad de intensidades usadas en la imagen resultante se reduce de 80 a 63.

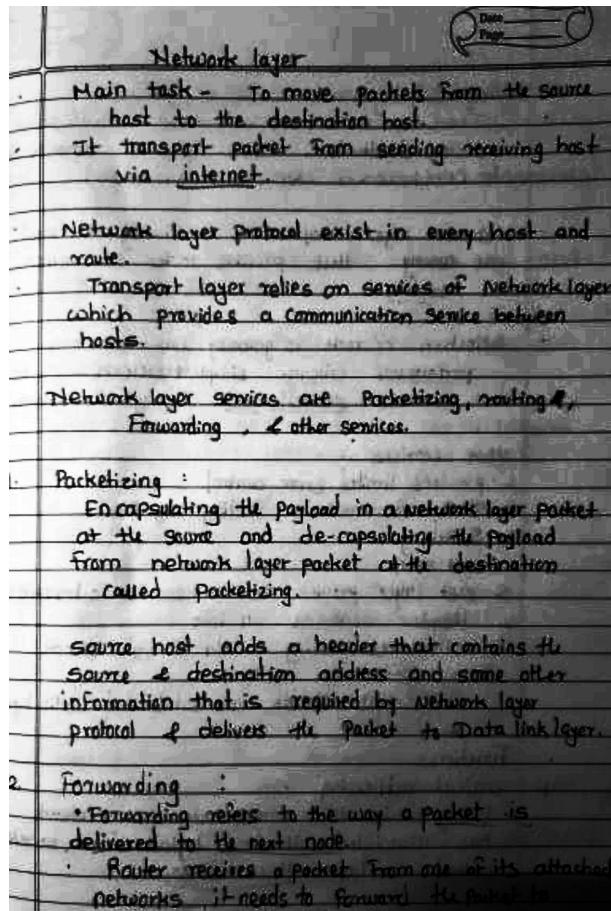


Figura 19: Imagen 3 ecualizada

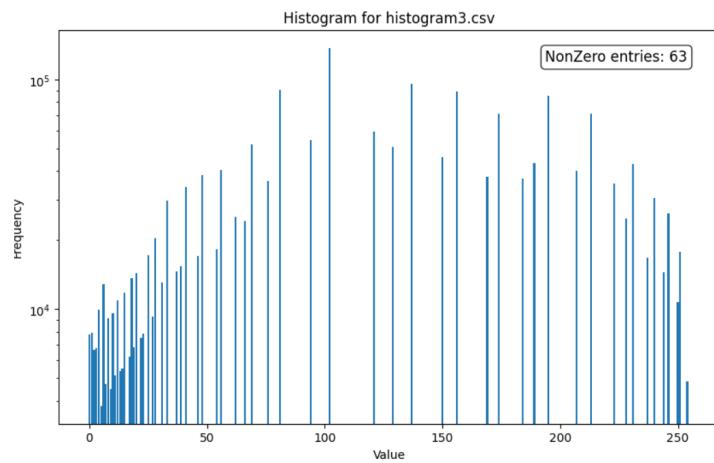


Figura 20: Histograma ecualizado de la imagen 3

2.5. Ecualización de la imagen 4

En la imagen 4, la ecualización mejora el contraste de las letras y reduce el efecto de difuminación en las zonas claras. La cantidad de intensidades usadas en la imagen resultante se reduce de 126 a 76.

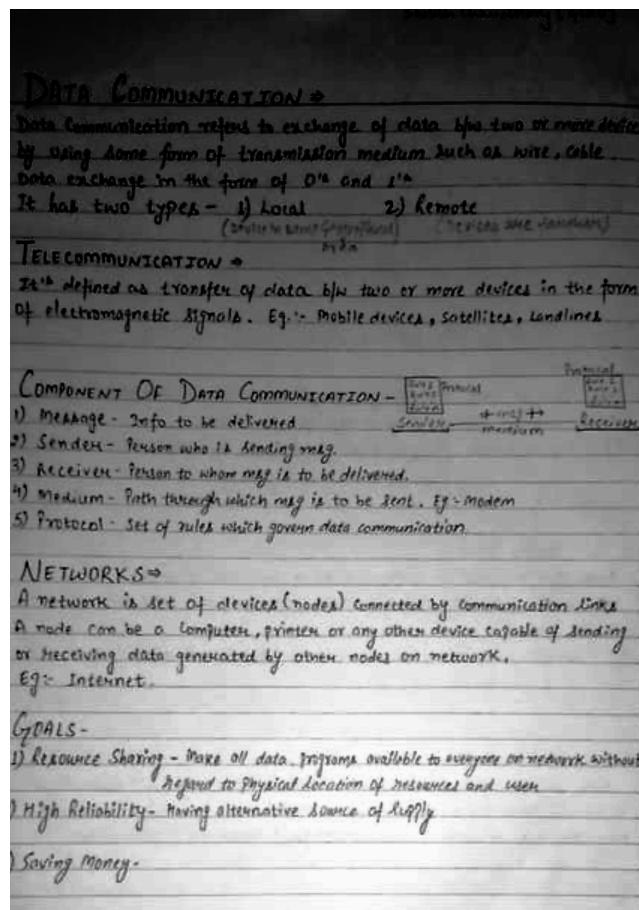


Figura 21: Imagen 4 ecualizada

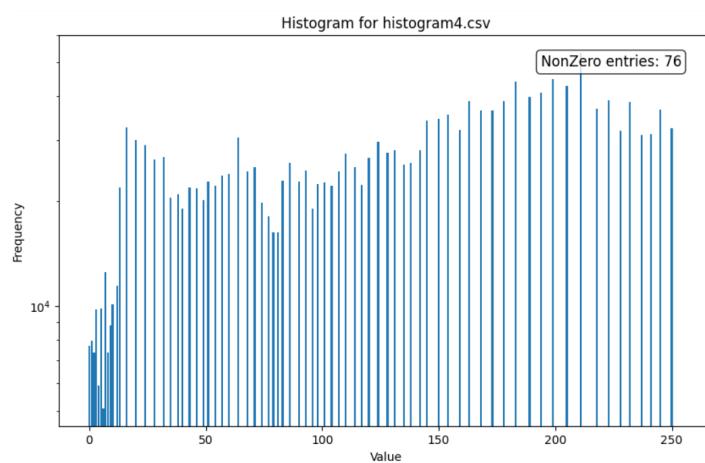


Figura 22: Histograma ecualizado de la imagen 4

2.6. Ecualización de la imagen 5

A pesar de que la imagen 5 originalmente era la mas legible de todas, la ecualización hace que el papel que originalmente parecía uniforme adquiera manchas oscuras y claras, esto puede ser debido a la reducción de la cantidad de intensidades usadas en la imagen resultante, de 123 a 72, y a la tendencia a una distribución uniforme de las intensidades que tiene la ecualización, lo que hace que las zonas oscuras y claras se mezclen.

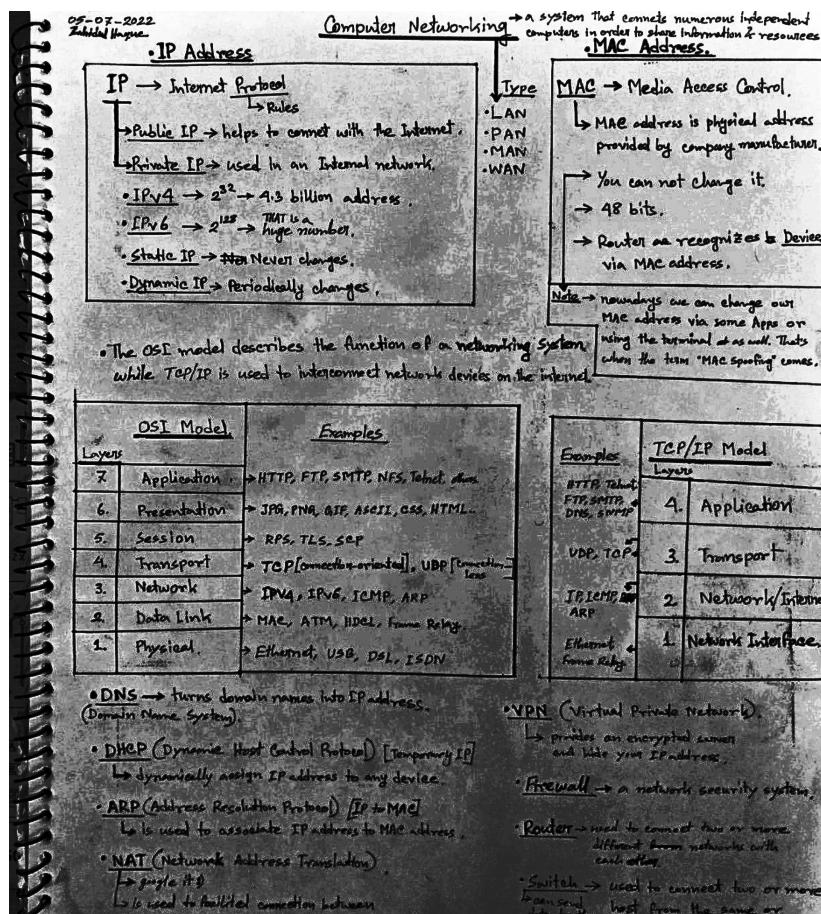


Figura 23: Imagen 5 ecualizada

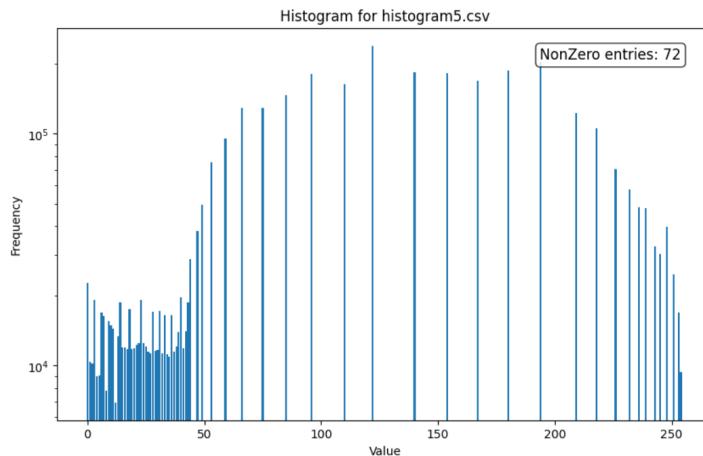


Figura 24: Histograma ecualizado de la imagen 5

3. Binarización de Imágenes

Se realizó el proceso de binarización de las imágenes procesadas con la ecualización de histograma. Se estableció los umbrales de binarización $thresholds = \{255, 255, 125, 125, 125, 120\}$, para las imágenes desde la 0 a la 5 respectivamente. Los resultados de la binarización se muestran a continuación:

1. Introduction To Computer Networks

1.1 What is a computer Network?

A computer network is an interconnection among two or more computers or computing devices. Such interconnection allows computers to share data and resources among each others. A basic network may connect a few computers placed in a room. The network size may vary from small to large depending on the number of computers it connects. A computer network can include different types of hosts (also called nodes) like server, desktop, laptop, cellular phones.

1.2 Types of computer Networks.

There are various types of computer networks ranging from network of handheld devices (like mobile phones or tablets) connected through Wi-Fi or Bluetooth within a single room to the millions of computers spread across the globe. Some are connected wireless while others are connected through wires.

Based on the geographical area covered and data transfer rate, computer networks are broadly categorised as:

- PAN (personal Area Network)
- LAN (Local Area Network)
- MAN (Metropolitan Area Network)
- WAN (Wide Area Network)

Figura 25: Imagen 0 binarizada

1. Introduction To Computer Networks

1.1 What is a computer Network?

A computer network is an interconnection among two or more computers or computing devices. Such interconnection allows computers to share data and resources among each others. A basic network may connect a few computers placed in a room. The network size may vary from small to large depending on the number of computers it connects. A computer network can include different types of hosts (also called nodes) like server, desktop, laptop, cellular phones.

1.2 Types of computer Networks

There are various types of computer networks ranging from network of handheld devices (like mobile phones or tablets) connected through Wi-fi or Bluetooth within a single room to the millions of computers spread across the globe. Some are connected wireless while others are connected through wires.

Based on the geographical area covered and data transfer rate, computer networks are broadly categorised as:

- PAN (personal Area Network)
- LAN (Local Area Network)
- MAN (Metropolitan Area Network)
- WLAN (Wide Area Network)

Figura 26: Imagen 1 binarizada

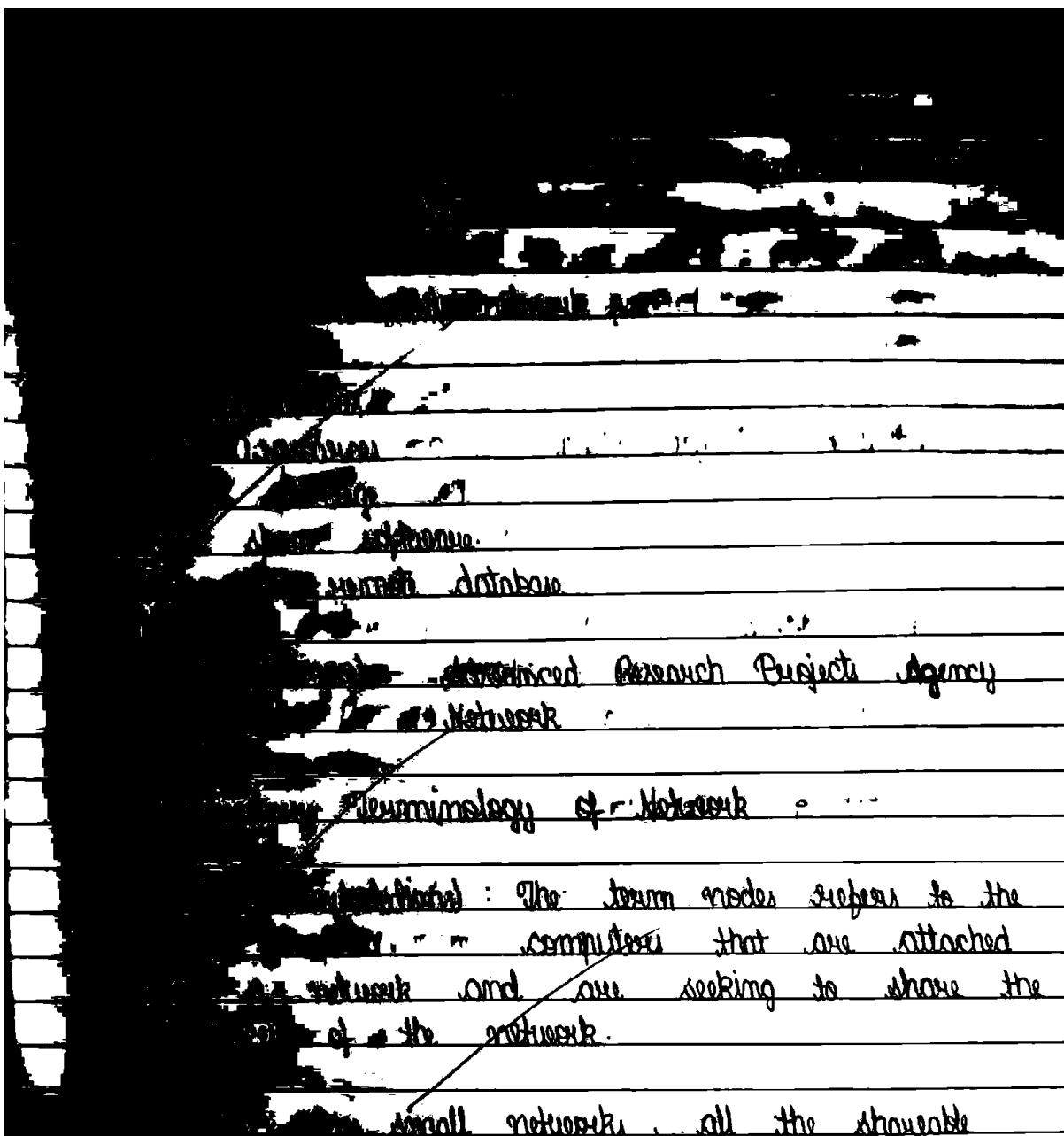


Figura 27: Imagen 2 binarizada

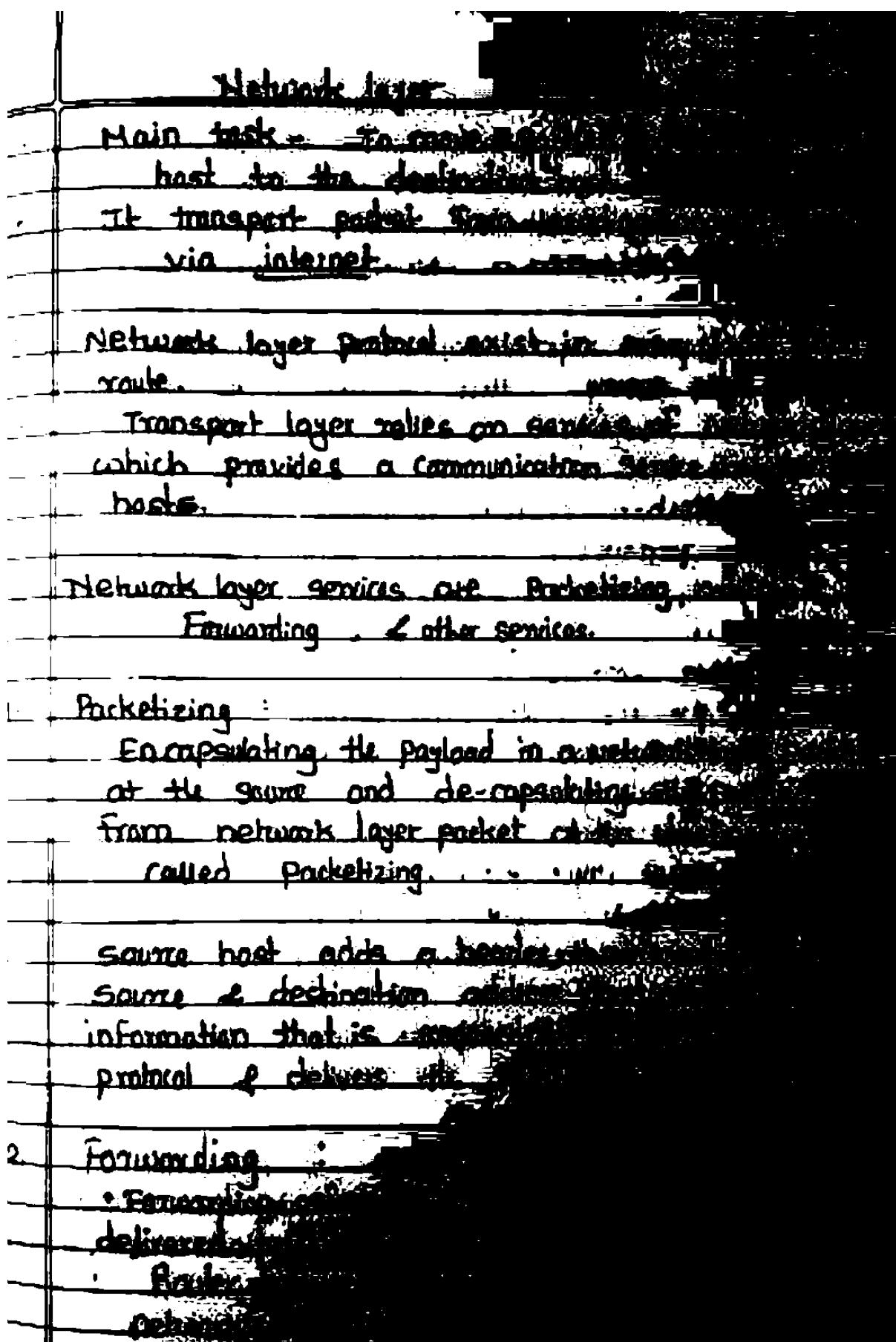
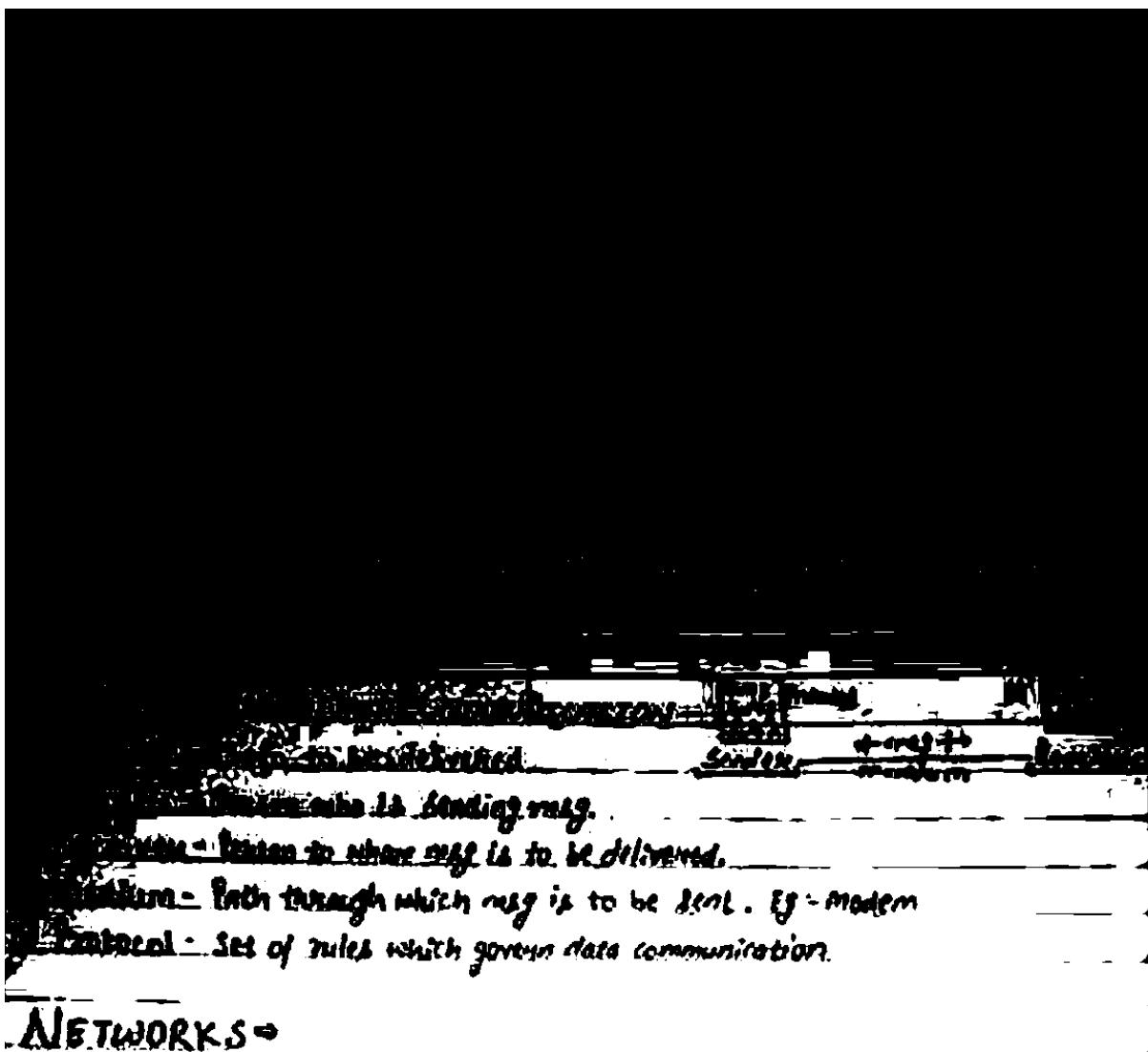


Figura 28: Imagen 3 binarizada



NETWORKS -

A network is set of devices (nodes) connected by communication links.
A node can be a computer, printer or any other device capable of sending or receiving data generated by other nodes on network.
Eg:- Internet

GPA LS -

- 1) Resource Sharing - Share all data in form available to everyone on network without having to physical location of resources and user.
- 2) High flexibility - Many alternative lines of supply
- 3) Saving money -

Figura 29: Imagen 4 binarizada

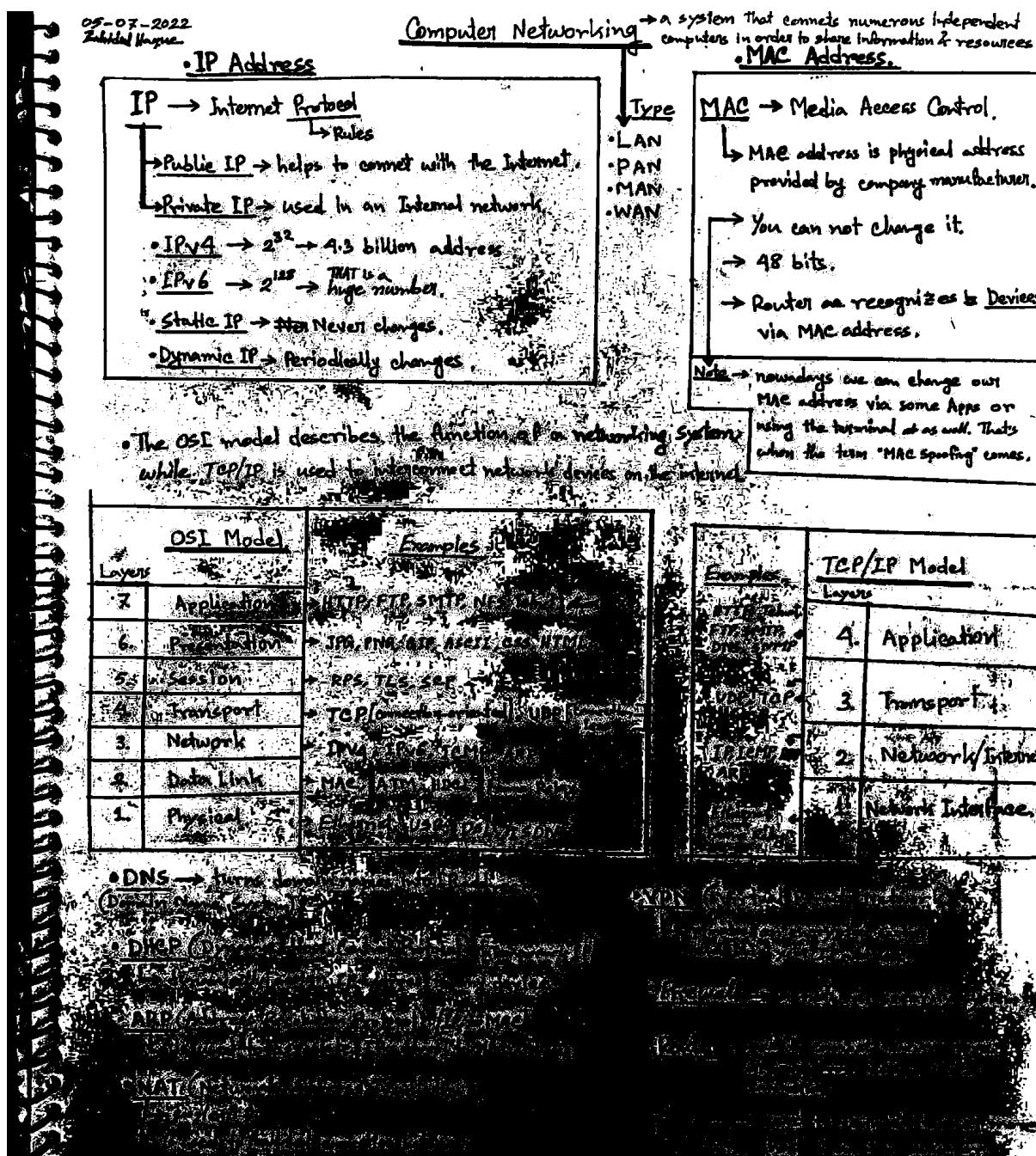


Figura 30: Imagen 5 binarizada

4. Dificultades Encontradas

- Visualización de histogramas:** Al graficar los histogramas en Python, los histogramas de las imágenes 0 y 1 mostraban una gran cantidad de frecuencias en la intensidad 255, lo que dificultaba la visualización de las otras intensidades. Se resolvió utilizando una escala logarítmica para el eje y, lo que permitió observar mejor las frecuencias de las intensidades más bajas.
- Binarización:** Elegir los umbrales adecuados para cada imagen requirió pruebas manuales, aunque un umbral común de 128 para todas las imágenes daba resultados muy similares a los presentados en la guía, sin embargo, se optó por usar umbrales distintos para cada imagen para obtener resultados más identicos a los de la guía.
- Documentación en LATEX:** Fue tedioso cargar las imágenes y los histogramas en el documento

de L^AT_EX, especialmente al tener que ajustar el tamaño de las imágenes y asegurarse de que se vean bien en el documento final.

5. Código Implementado

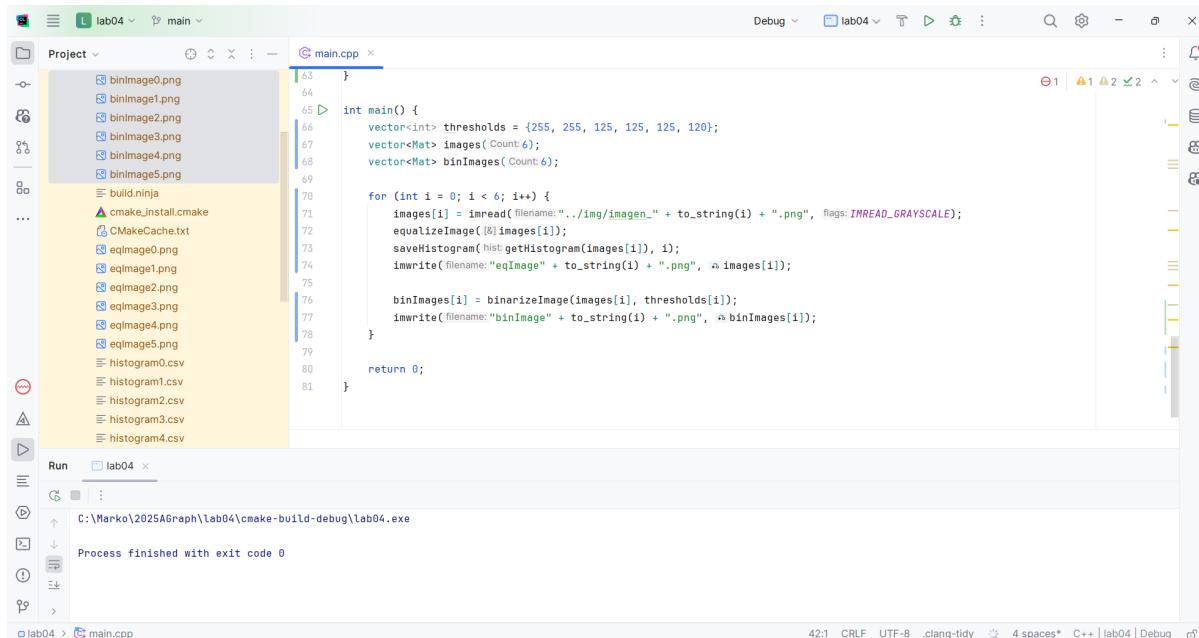


Figura 31: Ejecución del programa

```

1 #include <opencv2/opencv.hpp>
2 #include <iostream>
3 #include <fstream>
4
5 using namespace std;
6 using namespace cv;
7
8 vector<int> getHistogram(const Mat& img) {
9     vector<int> hist(256, 0);
10    for(int i=0; i.rows; i++) {
11        for(int j=0; j.cols; j++) {
12            hist[img.at<uchar>(i, j)]++;
13        }
14    }
15    return hist;
16}
17
18 void saveHistogram(vector<int> hist, int n) {
19     string filename = "histogram" + to_string(n) + ".csv";
20     ofstream file(filename);
21     if (file.is_open()) {
22         for (int i = 0; i < 256; i++) {
23             file << i << "," << hist[i] << endl;
24         }
25         file.close();
26     } else {
27         cout << "Unable to open file";
28     }
29}
  
```

```

31 vector<int> equalizeHistogram(vector<int> hist, int nPixels) {
32     vector<int> f(256);
33     f[0] = 0;
34     int cumulativeAbsFreq = hist[0];
35     for (int i = 1; i < 255; i++) {
36         f[i] = (cumulativeAbsFreq * 255) / nPixels;
37         cumulativeAbsFreq += hist[i];
38     }
39     f[255] = 255;
40     return f;
41 }
42
43 void equalizeImage(Mat& img) {
44     vector<int> hist = getHistogram(img);
45     int nPixels = img.rows * img.cols;
46     vector<int> f = equalizeHistogram(hist, nPixels);
47
48     for(int i = 0; i < img.rows; i++) {
49         for(int j = 0; j < img.cols; j++) {
50             img.at<uchar>(i, j) = f[img.at<uchar>(i, j)];
51         }
52     }
53 }
54
55 Mat binarizeImage(const Mat& img, int threshold) {
56     Mat binaryImg = Mat::zeros(img.size(), CV_8UC1);
57     for(int i = 0; i < img.rows; i++) {
58         for(int j = 0; j < img.cols; j++) {
59             binaryImg.at<uchar>(i, j) = (img.at<uchar>(i, j) < threshold) ? 0 :
60                                         255;
61         }
62     }
63     return binaryImg;
64 }
65
66 int main() {
67     vector<int> thresholds = {255, 255, 125, 125, 125, 120};
68     vector<Mat> images(6);
69     vector<Mat> binImages(6);
70
71     for (int i = 0; i < 6; i++) {
72         images[i] = imread("../img/imagen_" + to_string(i) + ".png",
73                           IMREAD_GRAYSCALE);
74         equalizeImage(images[i]);
75         saveHistogram(getHistogram(images[i]), i);
76         imwrite("eqImage" + to_string(i) + ".png", images[i]);
77
78         binImages[i] = binarizeImage(images[i], thresholds[i]);
79         imwrite("binImage" + to_string(i) + ".png", binImages[i]);
80     }
81
82     return 0;
83 }
```