# Informação e Codificação

First project report

Bruno Gomes, Diogo Silva

Universidade de Aveiro

# Informação e Codificação

DEPARTAMENTO DE ELETRÓNICA
TELECOMUNICAÇÕES E INFORMÁTICA

First project report

Bruno Gomes, Diogo Silva
(103320) brunofgomes@ua.pt, (104341) diogobranco.as@ua.pt

20/10/2024

**Abstract**

This project is mainly composed of three parts, the first one allowed us to understand how to manipulate text data programmatically. The second part focused on how to read and manipulate audio data. And finally, the third part predominantly focuses on reading and manipulating image files.

# Contents


**1 Introduction: 1**

**2 Methodology: 2**
2.1 Text data manipulation . . . . . 2
2.2 Audio data manipulation . . . . . 4
2.3 Image Data Processing . . . . . 6

**3 Results 7**
3.1 Text data . . . . . 7
3.2 Audio data . . . . . 15
3.2.1 Quantization Impact . . . . . 15
3.2.2 Histogram Analysis . . . . . 18
3.3 Image Data . . . . . 19

**4 Performance analysis 25**
4.1 Text data . . . . . 25
4.1.1 Processing time: . . . . . 25
4.1.2 Trade-offs: . . . . . 25
4.2 Audio data . . . . . 25
4.2.1 Processing time: . . . . . 25
4.2.2 Trade-offs: . . . . . 25
4.3 Image data . . . . . 26
4.3.1 Processing time: . . . . . 26
4.3.2 Trade-offs: . . . . . 26

**5 Discussion 27**
5.1 Text Data . . . . . 27
5.1.1 Outcomes: . . . . . 27
5.1.2 Strengths and Limitations: . . . . . 27
5.1.3 Possible Improvements . . . . . 27
5.2 Audio data . . . . . 27
5.2.1 Outcomes: . . . . . 27
5.2.2 Strengths and Limitations: . . . . . 28
5.3 Image data . . . . . 29

5.3.1 Outcomes: . . . . . 29
5.3.2 Strengths and Limitations: . . . . . 29
5.3.3 Possible improvements: . . . . . 29

**6 Conclusion** **30**
6.1 Text Data . . . . . 30
6.1.1 Key Takeaways: . . . . . 30
6.1.2 What was learned: . . . . . 30
6.2 Audio Data . . . . . 30
6.2.1 Key Takeaways: . . . . . 30
6.2.2 What was learned: . . . . . 30
6.3 Image Data . . . . . 31
6.3.1 Key Takeaways: . . . . . 31
6.3.2 What was learned: . . . . . 31

**7 Appendices** **32**

# Chapter 1

# Introduction:

The purpose of this project is to explore and implement methods for processing, manipulating and visualizing data. As such it is divided into three main parts:

1. Text data manipulation : In this part we implement techniques to read, transform and analyze text data. The tasks include file reading, creation of histograms of characters, word frequency calculations and visualizations of these statistics.

2. Audio data manipulation: This section relates to the manipulation of audio files, focusing on loading and visualizing audio samples. The key parts include waveform plotting, amplitude histograms, quantization, and quality analysis using metrics like Mean Squared Error (MSE) and Signal-to-noise ratio (SNR). These analyses will provide insight into the effects of audio processing, size and quality reduction.

3. Image Data Processing: In this final part we work with image files, using the OpenCV library for loading, displaying and processing images. The main focus of this part is centered around grayscale conversion and RGB channels, histogram calculation, image filtering with the use of Gaussian Blur and quantization using metrics like MSE and Peak Signal-to-Noise Ratio (PSNR).

The final results of this project should give us an insight on how the transformation of data impacts quality and size by the use of compression techniques.

# Chapter 2

# Methodology:

## 2.1 Text data manipulation

For this task, we aimed to develop a program written in *C++* which allowed us to extract several features about text data, such as character frequency and word frequency. In this section it will be explained what we did in order to manipulate and process this type of data effectively.

1. **Key Transformations and Algorithms Applied**

   In this project, several key transformations and algorithms were applied to process the textual data effectively. These transformations were necessary to ensure the accuracy and consistency of the word and character occurrences.

   - **Lowercase Conversion:**
     - **Description:** All words and characters we converted to lowercase;
     - **Purpose:** We decided to apply this method in order to avoid counting the same word or character in different cases, for instance, "Data" and "data" or "A" and "a", would be counted as different words or characters if they didn't follow the same principle of being written in lowercase. In natural language capitalization does not make a difference in the semantic meaning of a word, so this step was crucial in unifying data and prevent redundancy.
   - **Punctuation Removal:**
     - **Description:** All punctuation marks were stripped from words and characters;
     - **Purpose:** Removing the punctuation marks made it easier for us to "tokenize" each word individually, because sometimes there were cases where we would get words with commas attached to

them, this would create situations where we got a "word," and a "word" which would be counted as two different occurrences. For character analysis, punctuation was irrelevant, so we also removed it.

- **Whitespace Trimming:**
  - **Description:** Leading and trailing whitespaces around words were removed;
  - **Purpose:** Sometimes, extra spaces might appear before or after words and characters due to formatting issues. These spaces can also interfere with accurate counting making "cat" and "cat " count as different words. The removal of whitespaces proved to be an essential component to count words and characters accurately.
- **Character Conversion:**
  - **Description:** In the character frequency analysis, uppercase letters were converted to lowercase and accented characters were preserved.
  - **Purpose:** For character frequency, the same letter in upper and lowercase should not be treated differently, because we would have another redundancy problem that would also affect word occurrences, for example, "A" and "a" would need to be regarded as the same letter. Additionally, preserving accented letters was proven to be essential for languages like Portuguese, because accents give a different semantic meaning to characters.
- **XML Tag Filtering:**
  - **Description:** Any lines containing XML tags were ignored.
  - **Purpose:** Most datasets that our algorithm processed contained metadata or structural information enclosed in tags. Since, these tags are not part of the natural text, we decided that it was best to ignore them to focus on the actual content of the text.

2. **Tools and Libraries Used**

   Several tools and libraries were used in this process, providing essential functionality for reading, processing and analyzing the text data:

   - **C++ Standard Libraries:**
     - **<fstream>:** Used to read input text files (via wifstream) and to write the results (word and character frequencies) to output CSV files. The wifstream was particularly important because it allows handling of wide-character input (a wide character is a datatype that has a size greater than 8 bits), ensuring that accented characters and other non-ASCII characters in the text were read correctly.

  - **<map>:** The map data structure was the most suitable to our needs in this particular case, because each unique word or character was stored as a key, with its associated frequency as the value. This allowed for efficient lookup and updating of occurrences of each key.
  - **<locale> and <codecvt>:** These libraries allowed us to do a proper handling of *UTF-8* encoded text. These were critical for ensuring that accented characters and other special characters were processed correctly. By setting the locale, and using it with data streams that allowed for wide string and wide characters, made our program able to handle multilingual text files seamlessly.
- **Data Output Tools:**
  - **Comma-separated values (CSV) Files:** The results of the analysis were saved to CSV files. This format was chosen for its simplicity and compatibility with *Python* which was also an essential tool to aid us in the visualization of the obtained results.
  - **Matplotlib:** To views the results obtained, we opted to create a script in *Python* which used the library *Maplotlib*, to create a histogram and a bar chart, to view character and word frequencies respectively

## 2.2 Audio data manipulation

For the audio data task, we applied various algorithms to to analyze and process audio files. The main ones were the visualization of the audio waveform, the histogram of amplitude values, Quantization of audio and the MSE and SNR.

1. Waveform Visualization
   - Purpose: Used to see the original audio file waveform and used to compare it with the quantized version.
   - Method: We plotted the waveforms of the left and right channels of the audio files and in the case of quantization compare it before and after it happens.
   - Libraries: we used Python's Pandas library to plot the waveforms and SFML library for audio processing.
2. Histogram Analysis
   - Purpose: To analyze the frequency and distribution of amplitude values in the audio samples.
   - Method: We created histograms for left and right channels as well as the MID and SIDE channels, to visualize the distribution of amplitude values.

- Libraries: Python's Panda's library for visualization

3. Quantization

   - Purpose: Quantization is used to reduce the number of bits used to represent each audio sample, reducing the file size and impacting audio quality.
   - Method: We implemented uniform quantization, mapping the sample values to a smaller set of discrete values. The number of quantization levels used is determined by the number of bits used.
   - Libraries: We used SFML to retrieve the audio files channel amplitudes and Python's Panda's for visualization.

4. Comparison of audio with MSE and SNR

   - Purpose: Both of these metrics are used to access the quality of the quantization
   - Method: MSE was used to calculate the average difference between two corresponding sample points between audio samples. SNR was used to quantify the amount of noise introduced in the processed signal compared to the original.

## 2.3 Image Data Processing

1. Grayscale Conversion
   - Purpose: Convert color images to grayscale
   - Method: Use OpenCV's built in function's
   - Libraries: OpenCV
2. Histogram Calculation
   - Purpose: Calculate the histogram of grayscale images.
   - Method: Using OpenCV's calcHist function and a python function for visualization.
   - Libraries: OpenCV, Pandas.
3. Gaussian Blur:
   - Purpose: Reduce image noise and detail
   - Method: Use OpenCV's GaussianBlur function
   - Libraries: OpenCV
4. Image Comparison:
   - Purpose: Compare two images using MSE and PSNR as standard metrics
   - Method: using the respective algorithms.
5. Quantization
   - Purpose: Reduce the number of distinct pixel values, achieving image compression.
   - Method: Custom function to map pixel values to quantized levels

# Chapter 3

# Results

## 3.1 Text data

We experimented mainly with 2 large datasets (PT and EN datasets), 2 small datasets (Moby Dick and Dom Casmurro datasets), and a very short text that contained Russian characters. Below, the results for each one will be shown.

1. **Europarl PT dataset:**

   - **Word Frequencies:**

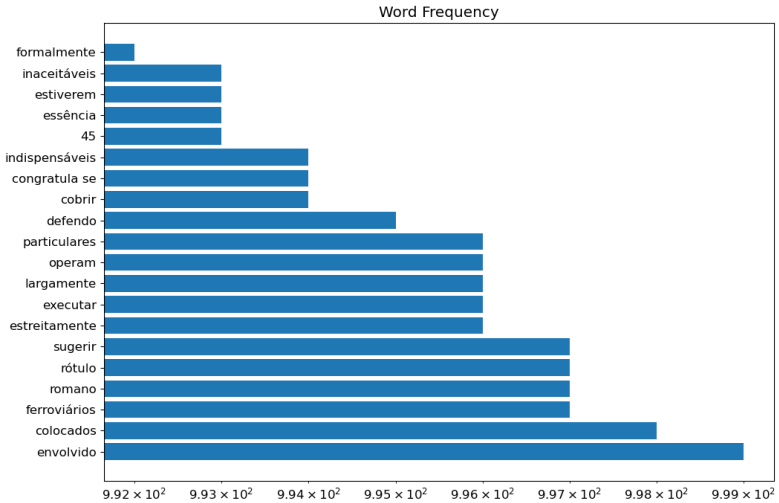


Figure 3.1: Word frequencies for PT dataset

- **Character frequencies:**

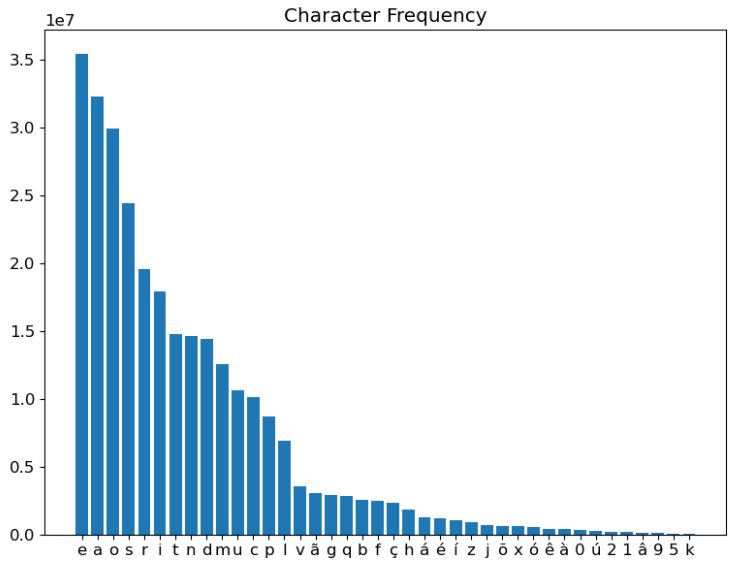


Figure 3.2: Character frequencies for PT dataset

2. **Europarl EN dataset:**

   - **Word Frequencies:**

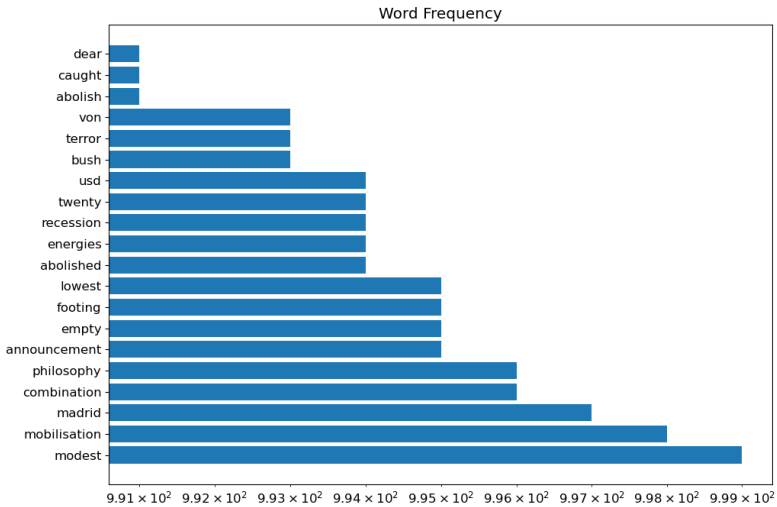


Figure 3.3: Word frequencies for EN dataset

- **Character frequencies:**

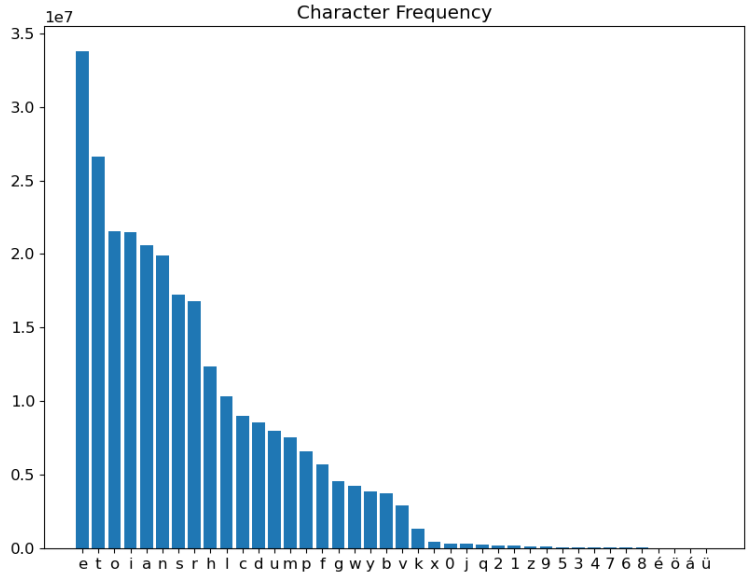


Figure 3.4: Character frequencies for EN dataset

3. **Moby Dick dataset:**

- **Word Frequencies:**

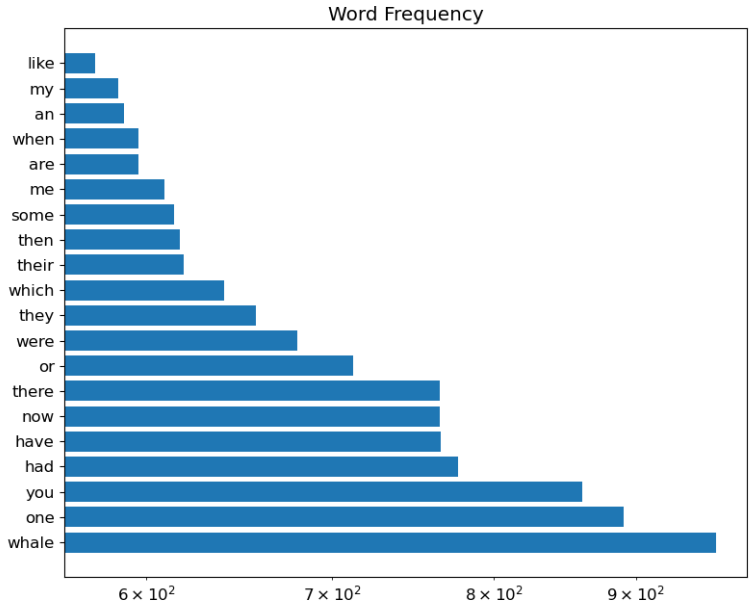


Figure 3.5: Word frequencies for Moby Dick dataset

- **Character frequencies:**

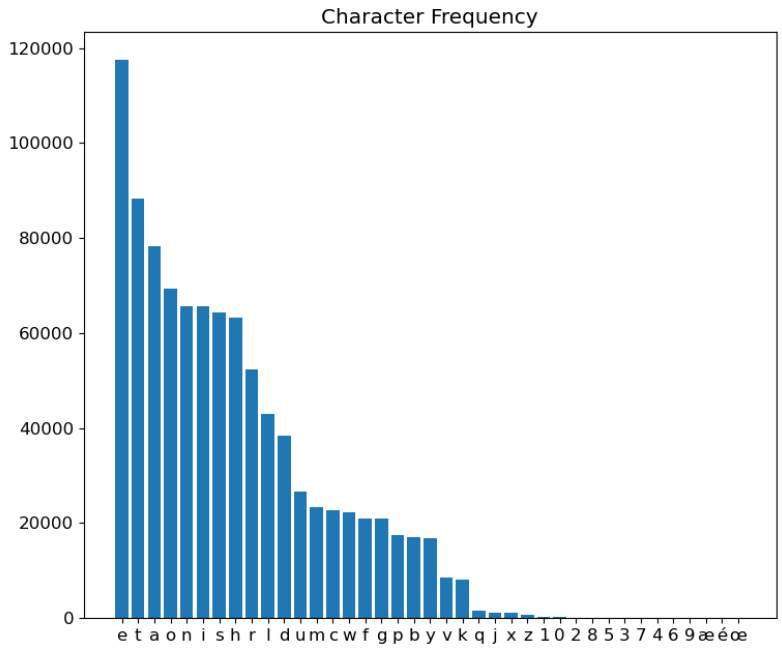


Figure 3.6: Character frequencies for Moby Dick dataset

4. **Dom Casmurro dataset:**

   - **Word Frequencies:**

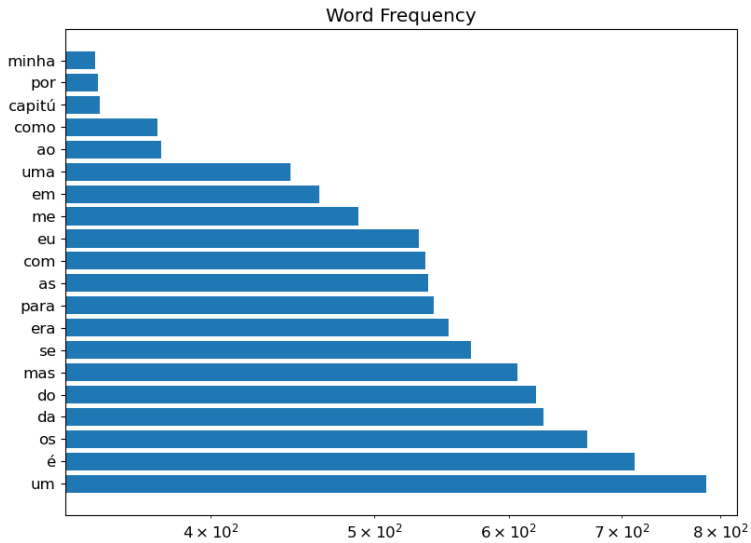


Figure 3.7: Word frequencies for Dom Casmurro dataset

- **Character frequencies:**

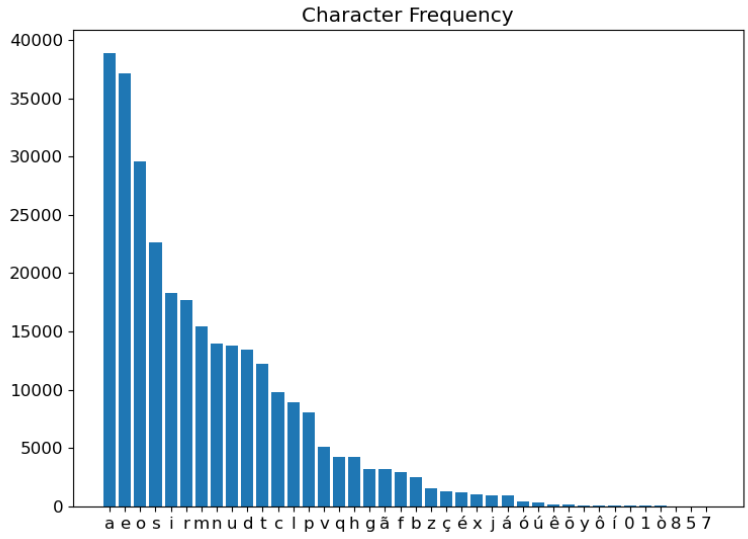


Figure 3.8: Character frequencies for Dom Casmurro dataset

5. **Small Russian dataset:**

```
на : 2
настроение : 1
научные : 1
наш : 1
нашу : 1
новые : 1
обучение : 1
отличный : 1
планирую : 1
поддерживать : 1
полезно : 1
помогают : 1
популярны : 1
популярным : 1
посетить : 1
привет : 1
природных : 1
проблем : 1
прошлом : 1
путешествия : 1
```

Figure 3.9: Russian words found by our algorithm

To sum up, the results show the key differences between the English language and the Portuguese language, these are:

1. **Common Characters:**
   - In both graphs the most common characters are **e,a,o** which are central to both languages, however the order differs slightly:
     - **English dataset:** Characters like **e,t,o,i** are among the top four.
     - **Portuguese dataset:** Characters like **e,a,o,s,r** are more dominant reflecting different language structures.

2. **Special Characters and accents**

   - One of the most noticeable differences in the Portuguese dataset is the presence of accented characters, which are very frequent in the language.
     - **Portuguese:** Characters like ã, ç, á appear frequently due to the use of accents in words like "não", "ação", "fácil", and so on.
     - **English:** Accented characters are absent, as the English language usually does not use accents.

3. **Less Frequent Letters:**

   - **English dataset:** Letters such as **q,z,j** are less frequent, which is typical for English.
   - **Portuguese dataset:** Similarly, characters like **k,w,y** are rare in Portuguese.

4. **Numeric Characters:**

   - Both datasets show the presence of numeric characters, but their frequency is relatively low compared to letters.

Finally, we decided to include a short russian dataset, in order to show that our algorithm can in fact handle these kind of characters, however we could not visualize them through graphs like we showed before for other datasets.

## 3.2 Audio data

### 3.2.1 Quantization Impact

We experimented with different quantization levels (4,6,8 and 10 bits) to observe the impact on audio quality. The results were assessed using MSE and SNR as well as graphic visualization

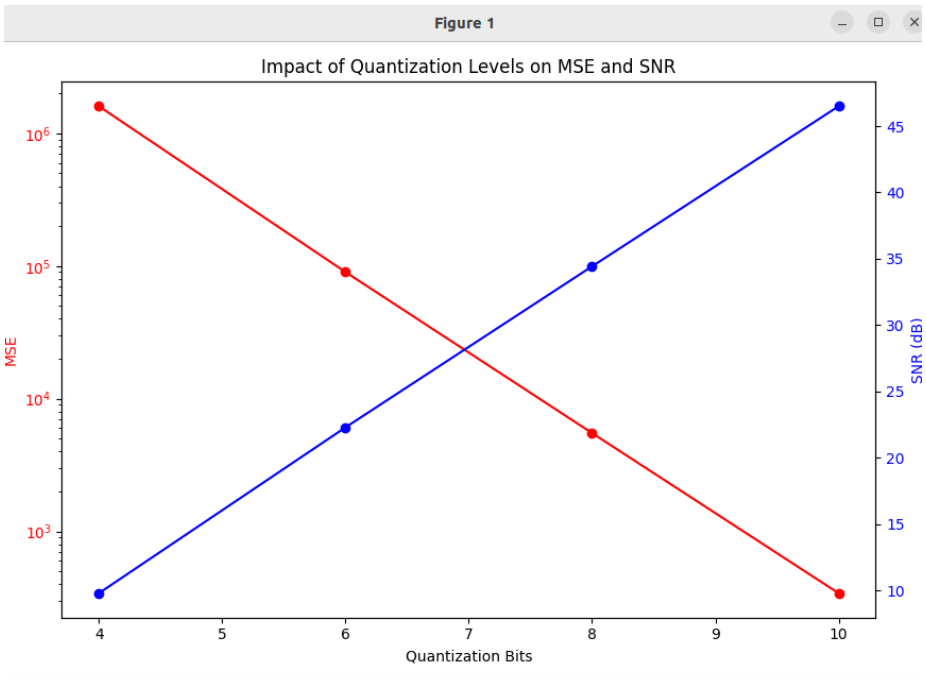


Figure 3.10: MSE and SNR results in sample01.wav

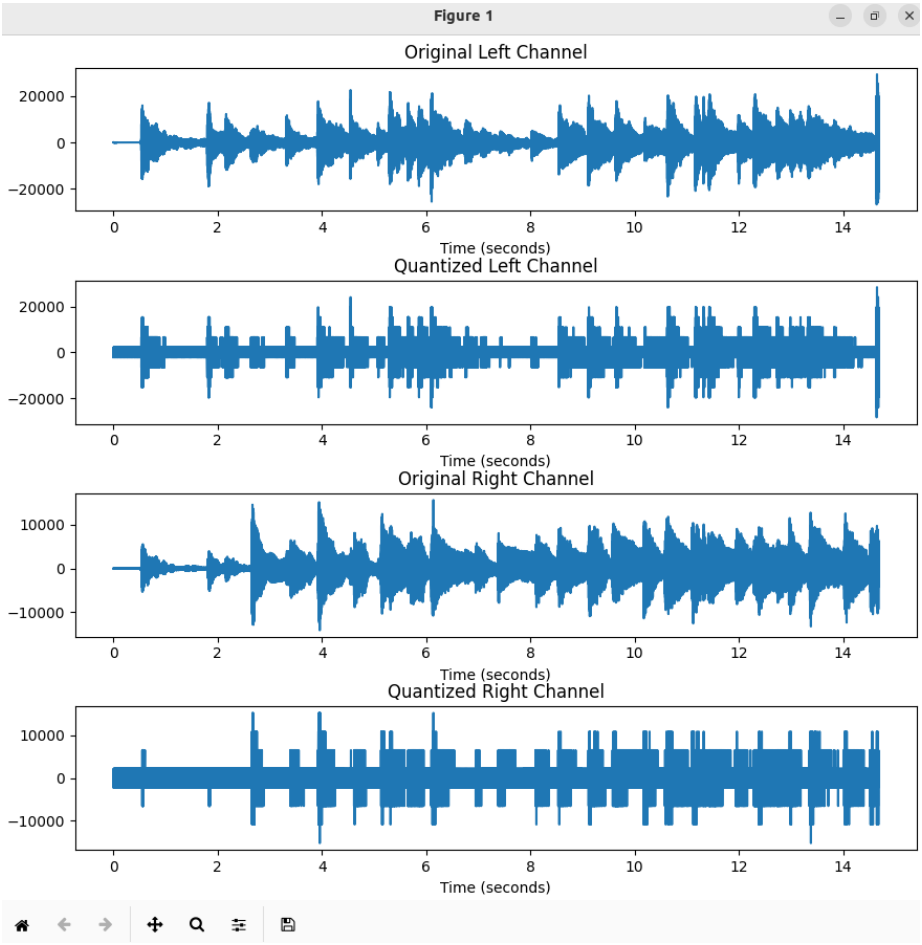


Figure 3.11: sample02.wav with 4 bit quantization

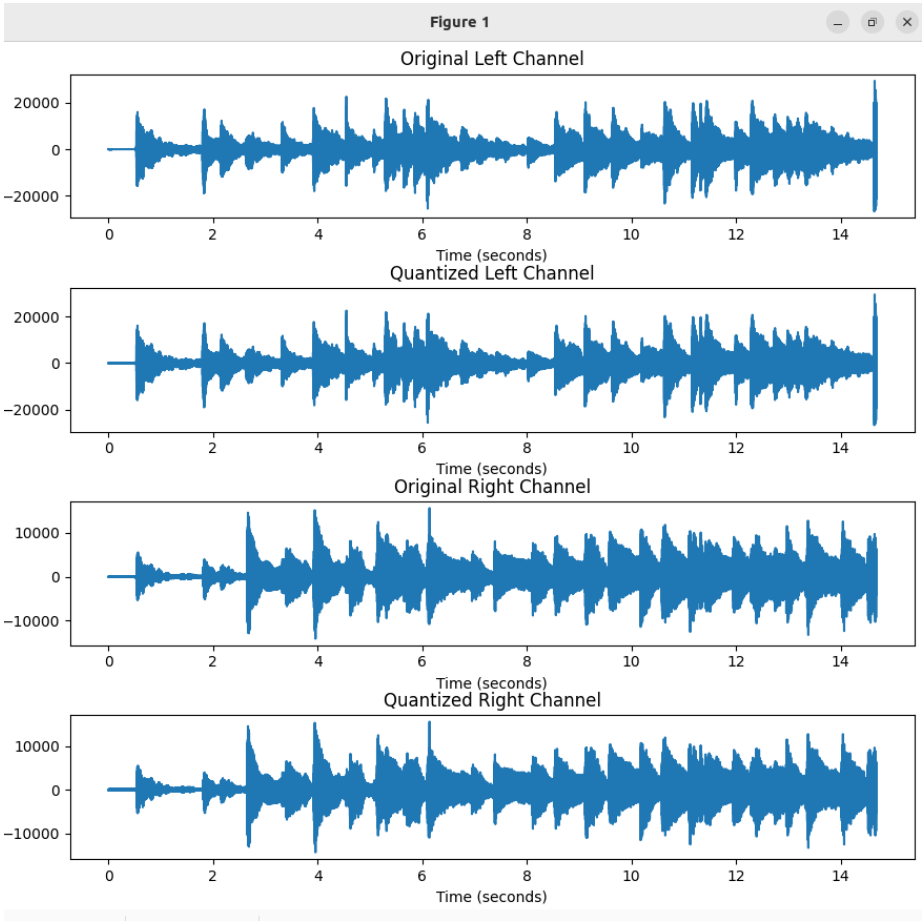


Figure 3.12: sample02.wav with 8 bit quantization

### 3.2.2 Histogram Analysis

Histograms were created for the left and right channels, as well as the MID and SIDE channels. These histograms provided us insight into the distribution of amplitude values across the entire audio file.

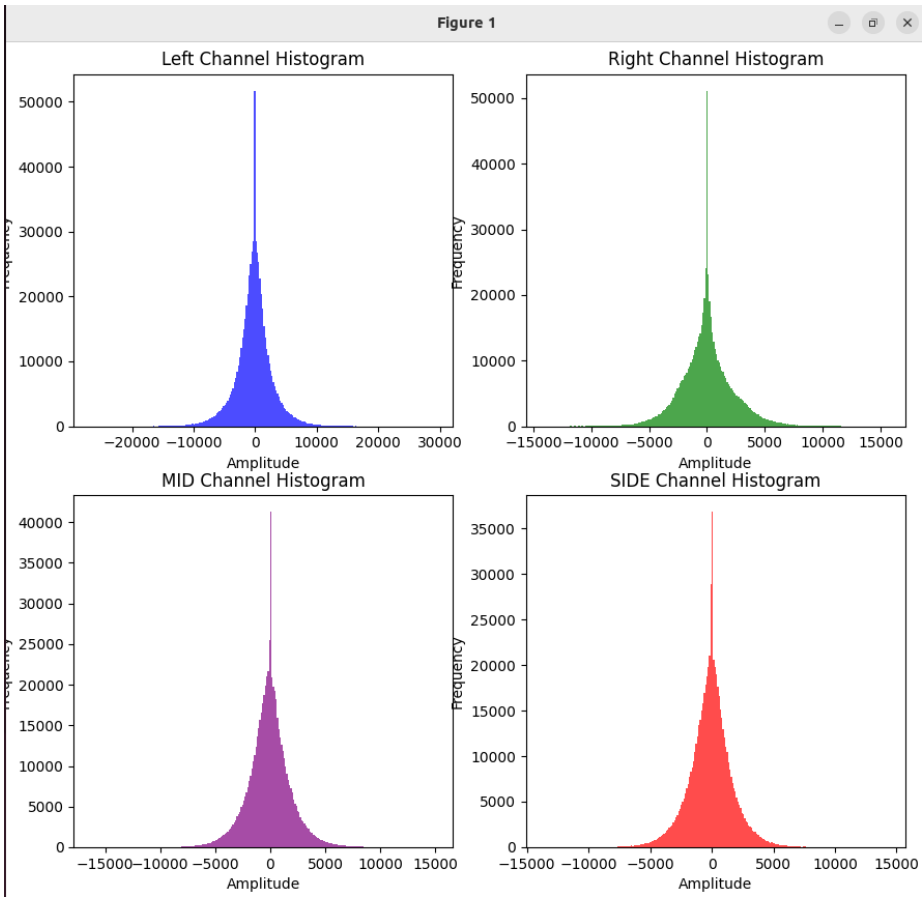


Figure 3.13: sample02.wav histogram

## 3.3 Image Data

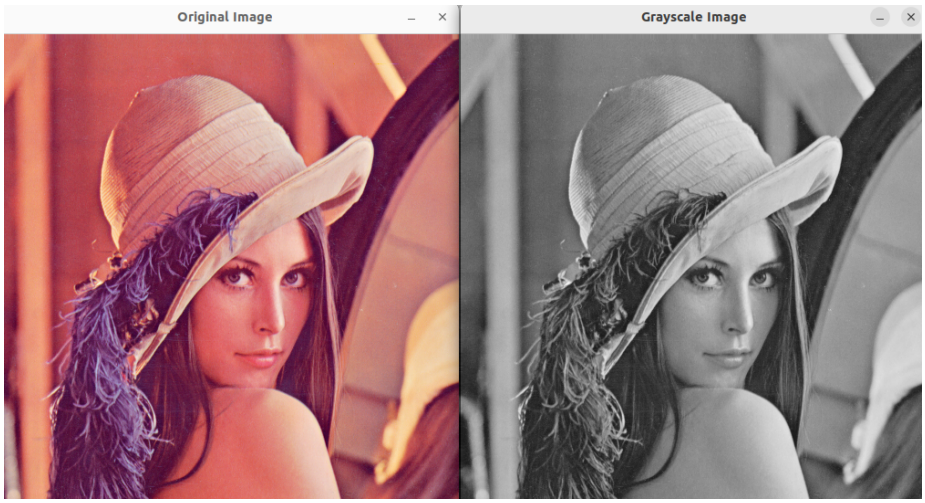


Figure 3.14: Original and grayscale comparison

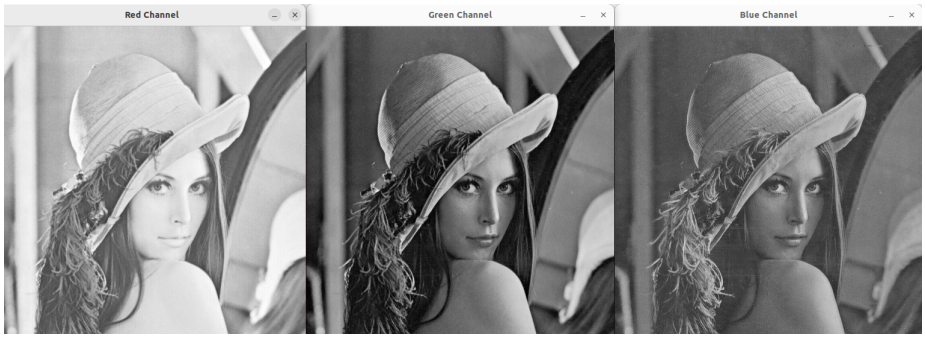


Figure 3.15: RGB channels comparison

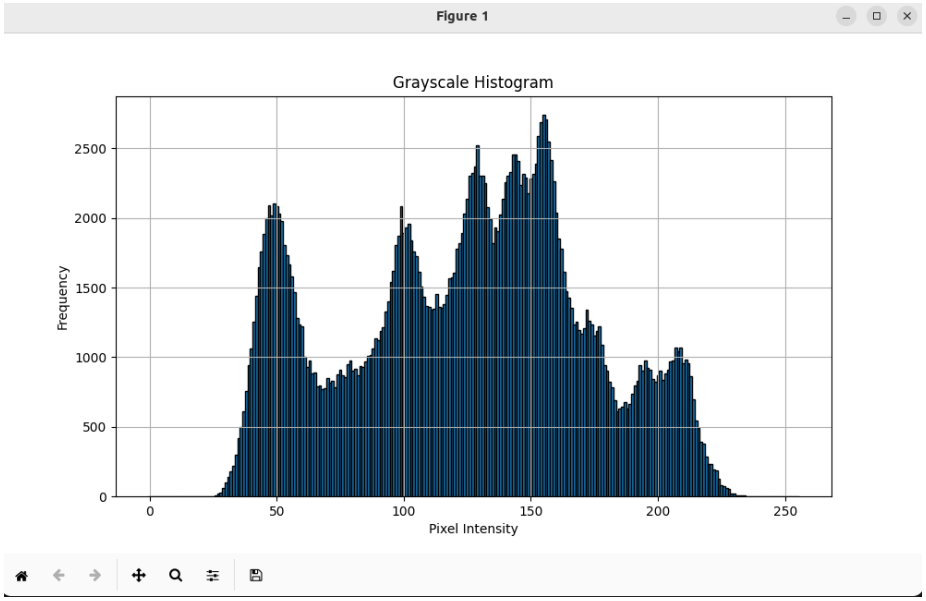


Figure 3.16: Lena Image Histogram

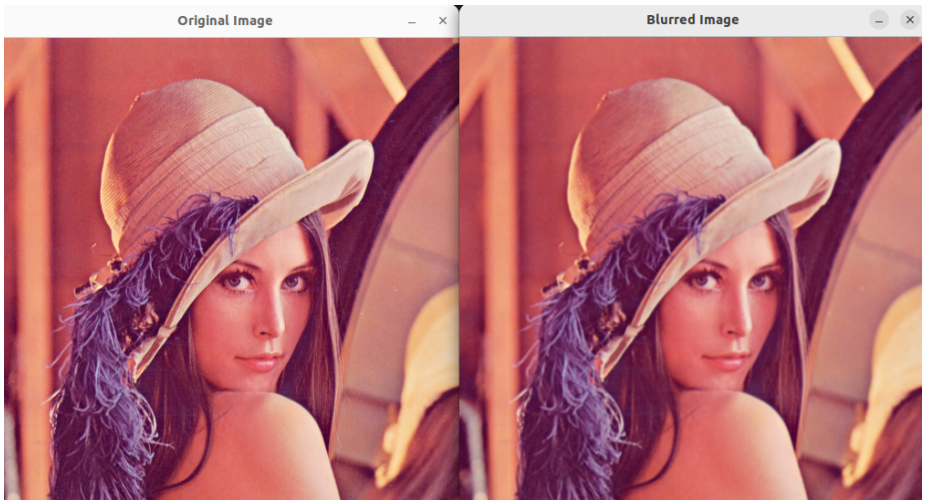


Figure 3.17: Lena Gaussian Blue with a kernel size of 3

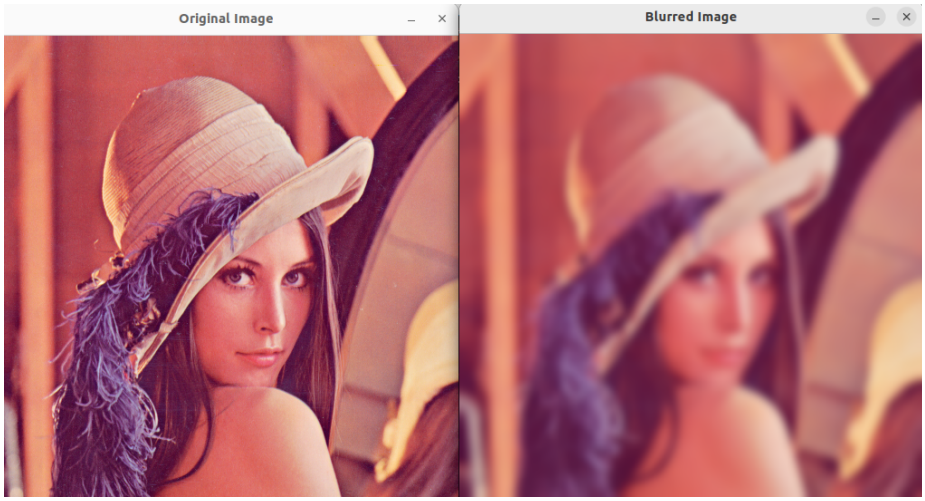


Figure 3.18: Lena Gaussian Blue with a kernel size of 31

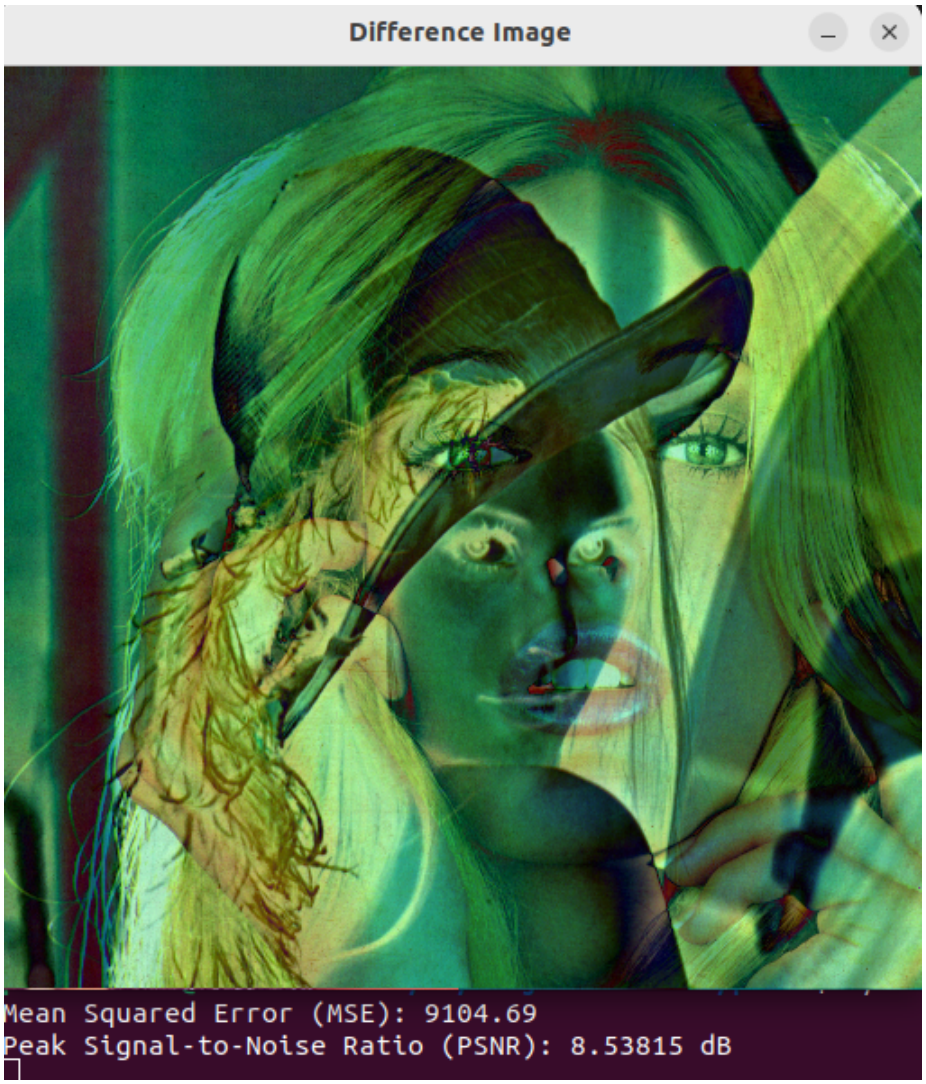


Figure 3.19: Comparison betweend Lena and girl pictures

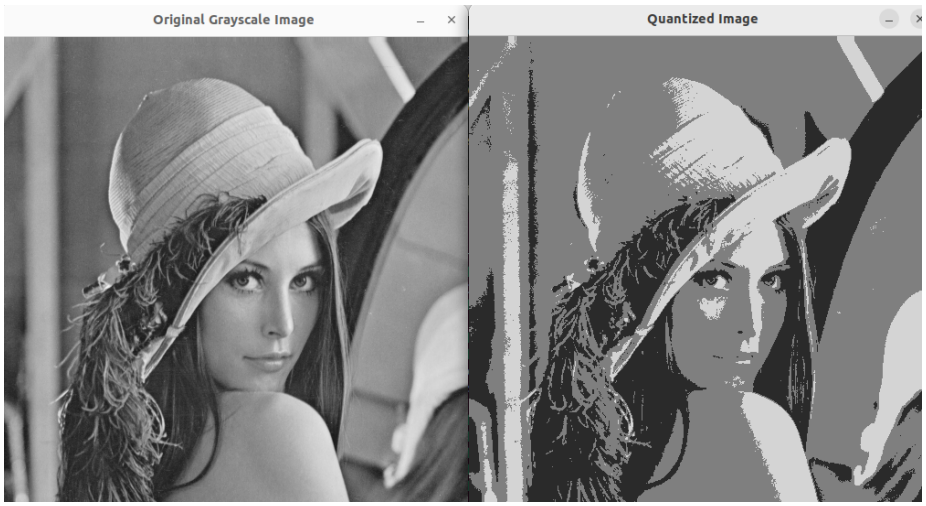


Figure 3.20: Image with 3 bit quantization

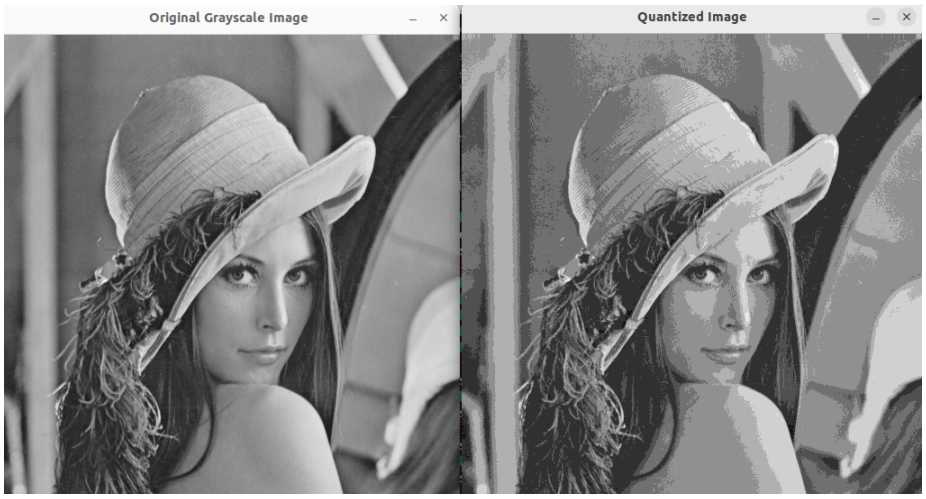


Figure 3.21: Image with 8 bit quantization

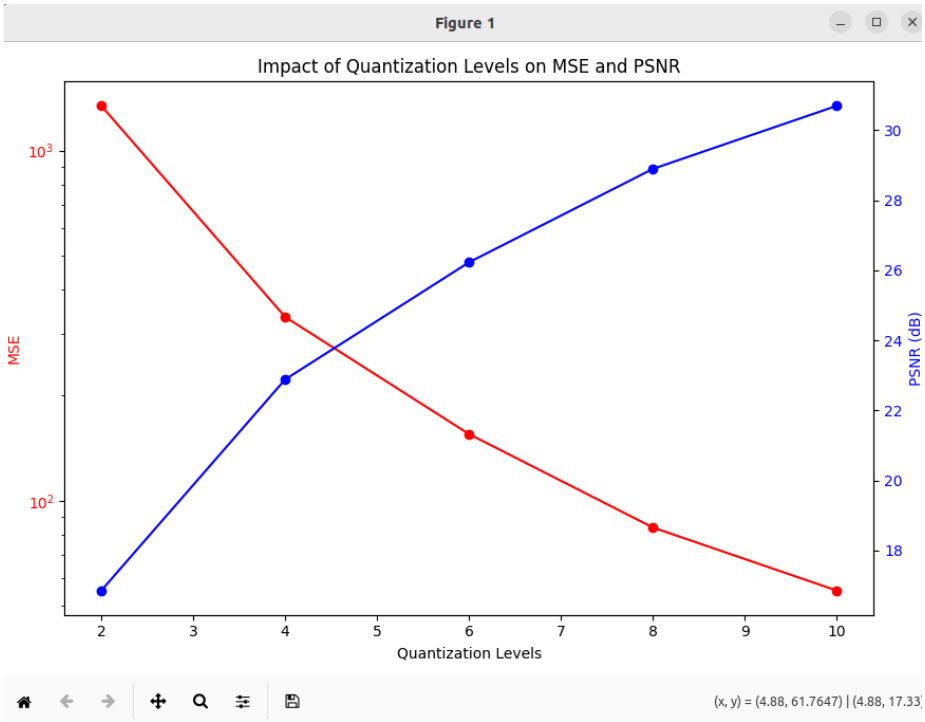


Figure 3.22: Lena image MSE and PSNR comparison through Quantization levels

# Chapter 4

# Performance analysis

## 4.1 Text data

### 4.1.1 Processing time:

The time taken to analyze the datasets varied greatly, for both the PT and EN Europarl datasets the time was approximately 240 seconds, for the Moby Dick dataset the time was 35 ms, for the Dom Casmurro dataset it took 16 ms, the russian text took 6 ms to be processed.

### 4.1.2 Trade-offs:

This variations in time are mainly due to the content of the files, larger files with more text take more time to be processed, additionally the biggest datasets also had to go through more processing methods in order to remove Extensible Markup Language (XML) tags.

## 4.2 Audio data

### 4.2.1 Processing time:

The time taken for different steps, such as quantization and visualization. While on the whole the general waiting time was low (4-8 seconds), there were insignificant but noticeable differences. Higher quantization levels resulted in longer processing times due to the increased number of calculations

### 4.2.2 Trade-offs:

There is a trade-off between processing time, compression efficiency and quality. Higher compression ratios (fewer bits) resulted in lower SNR values, indicating a loss of quality. Somthing that was visible by the sample02 quantization waveform 3.11

## 4.3 Image data

### 4.3.1 Processing time:

The time taken was also measured here. Most operations took less than 1 second to be done so I could not notice any perceptible changes.

### 4.3.2 Trade-offs:

Higher compression ratios in Quantization resulted in lower PSNR values and higher MSE indicating a loss in quality. Higher kernel sizes resulted in a higher "Blur" filter on the image, reducing it's quality.

# Chapter 5

# Discussion

## 5.1 Text Data

### 5.1.1 Outcomes:

Text data processing allowed us to to see the main contrasts between different datasets, however, the more text we have to process the more delay will be introduced by the algorithm. Datasets with less punctuation, XML tags and less uppercase letters will take smaller times to be processed, since the algorithm does not have to process those details.

### 5.1.2 Strengths and Limitations:

Our program allowed us to verify character and word frequencies, however, we do not have a way to check if every single character was counted or encoded correctly, in addition we can not display some characters on our graphs because the font we are using which covers a good amount of characters does not have the ability to cover every single character existent, sometimes errors may occur.

### 5.1.3 Possible Improvements

Adding parallelization in order to make the program run faster for the bigger datasets and also implement more encoding types to be able to represent even more characters and words.

## 5.2 Audio data

### 5.2.1 Outcomes:

The quantization process demonstrated the impact of reducing the number of bits on audio quality. Lower bit levels introduced noticeable MSE while higher bit introduced a higheer SNR and preserved the audio quality.

### 5.2.2 Strengths and Limitations:

The methods implemented provided a clear understanding of the effects of quantization but there was way for the program to return the quantized results back into a wav file, for further demonstration on said effects.

## 5.3 Image data

### 5.3.1 Outcomes:

- The methods implemented were effective in performing the desired image manipulations. Quantization successfully reduced the number of distinct pixel values and Gaussian blur reduced noise

### 5.3.2 Strengths and Limitations:

- Strengths: The methods were straightforward to implement due to the already existing functions and documentation of OpenCV.
- Limitations: Some methods such as quantization resulted in a great loss of image quality

### 5.3.3 Possible improvements:

- There might be better algorithms that might achieve the same results while minimizing the quality loss.

# Chapter 6

# Conclusion

## 6.1 Text Data

### 6.1.1 Key Takeaways:

Text data processing and manipulation depends on several factors such as which type of characters are we counting, which type of encoding do they use, what type of processing should we do in order to unify data and make it easier to present with the main objective of gathering insightful information.

### 6.1.2 What was learned:

Learned how to manipulate strings and characters using the wide character and wide string dataypes from *C++*, learned useful data processing techniques, learned how to visualize dataset information using plots that were shown by a python script every time our data text processing algorithm executed.

## 6.2 Audio Data

### 6.2.1 Key Takeaways:

The project highlighted the importance of quantization in audio compression and its impact on quality. The use of MSE and SNR metrics provided a quantitative assessment of the quality loss.

### 6.2.2 What was learned:

Gained experience with SFML and Pandas library.

## 6.3 Image Data

### 6.3.1 Key Takeaways:

Learned various image processing techniques, including grayscale conversion, histogram calculation, Gaussian blur, image comparison and quantization.

Understood the trade-offs between processing time, compression efficiency and quality loss

### 6.3.2 What was learned:

Gained experience with OpenCV and Pandas.

Learned how to quantify image quality through the use of MSE and PSNR.

# Chapter 7

# Appendices

https://github.com/BrunoGomes22/Projeto-01---IC

# Work Contributions

Both authors contributed equally towards the realization of this project, asking for 50% each.

# Acrónimos

**MSE** Mean Squared Error

**SNR** Signal-to-noise ratio

**PSNR** Peak Signal-to-Noise Ratio

**CSV** Comma-separated values

**XML** Extensible Markup Language