Extracción de bigramas y trigramas

Se debe realizar 1 extracción de bigramas y trigramas y la visualización de los más frecuentes (preferible usar un documento de al menos 1000 palabras de un tema concreto)

```
# Instalacion de paquetes
!pip install nltk matplotlib
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.11/dist-packages (3.9.1)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-packages (3.10.0)
Requirement already satisfied: click in /usr/local/lib/python3.11/dist-packages (from nltk) (8.2.0)
Requirement already satisfied: joblib in /usr/local/lib/python3.11/dist-packages (from nltk) (1.5.0)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.11/dist-packages (from nltk) (2024.11.6)
Requirement already satisfied: tqdm in /usr/local/lib/python3.11/dist-packages (from nltk) (4.67.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.3.2)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (4.58.0)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.4.8)
Requirement already satisfied: numpy>=1.23 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (2.0.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (24.2)
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (11.2.1)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (3.2.3)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.7->matplotlib) (1.17
```

```
# Importando librerias
import nltk
from nltk.util import ngrams
from nltk.tokenize import word_tokenize
from collections import Counter
import matplotlib.pyplot as plt
```

```
# Descargando el mdelo
nltk.download('punkt_tab')
```

```
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt_tab.zip.
```

```
True
```

```
# Texto de un libro
with open("libro.txt", encoding="utf-8") as f:
    texto = f.read()
```

```
# Tokenizando
tokens = word_tokenize(texto.lower())
```

```
# Filtrando palabras
tokens = [word for word in tokens if word.isalpha()]
```

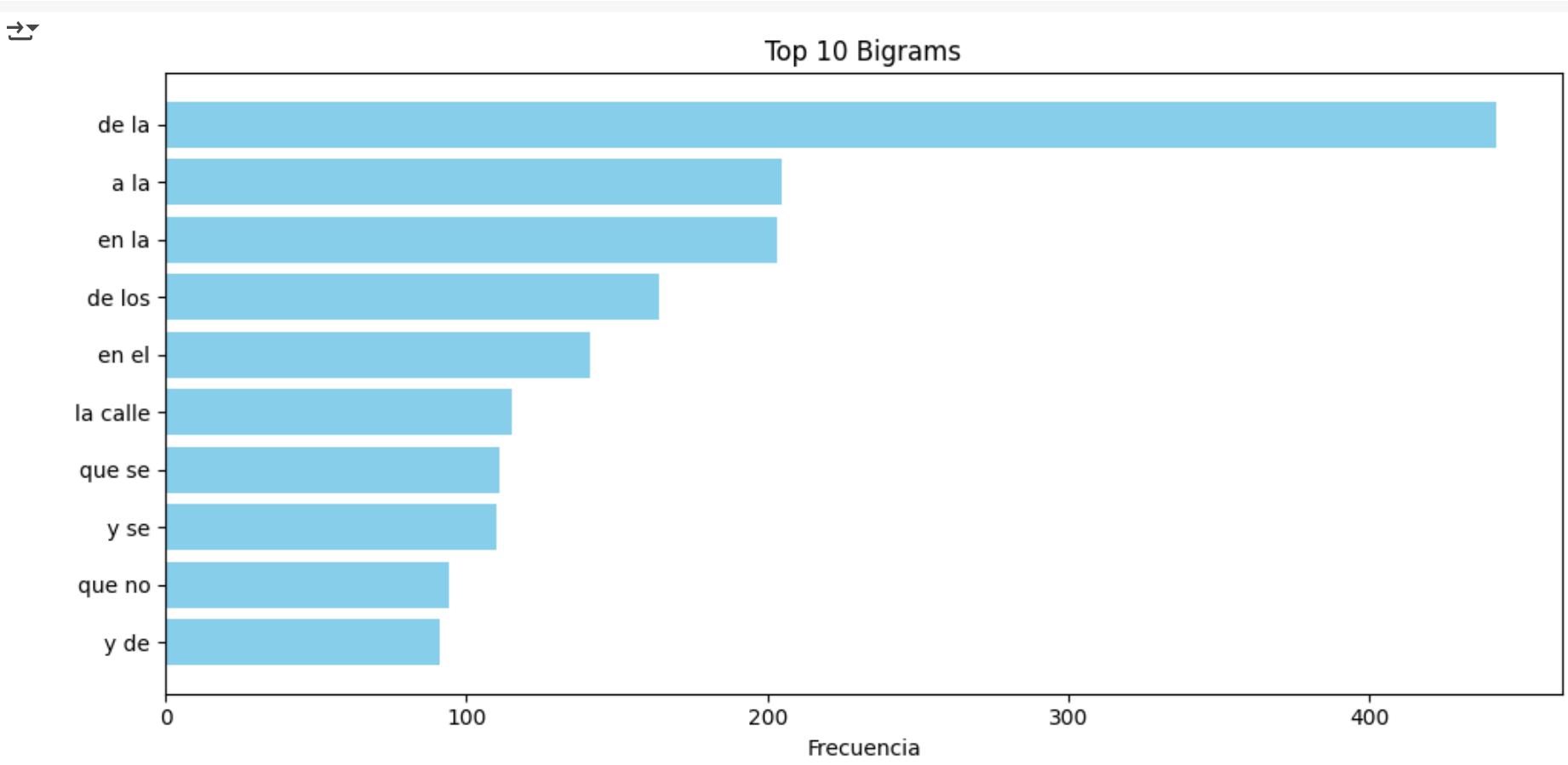
```
# Bigramas y Trigramas
bigrams = list(ngrams(tokens, 2))
trigrams = list(ngrams(tokens, 3))
```

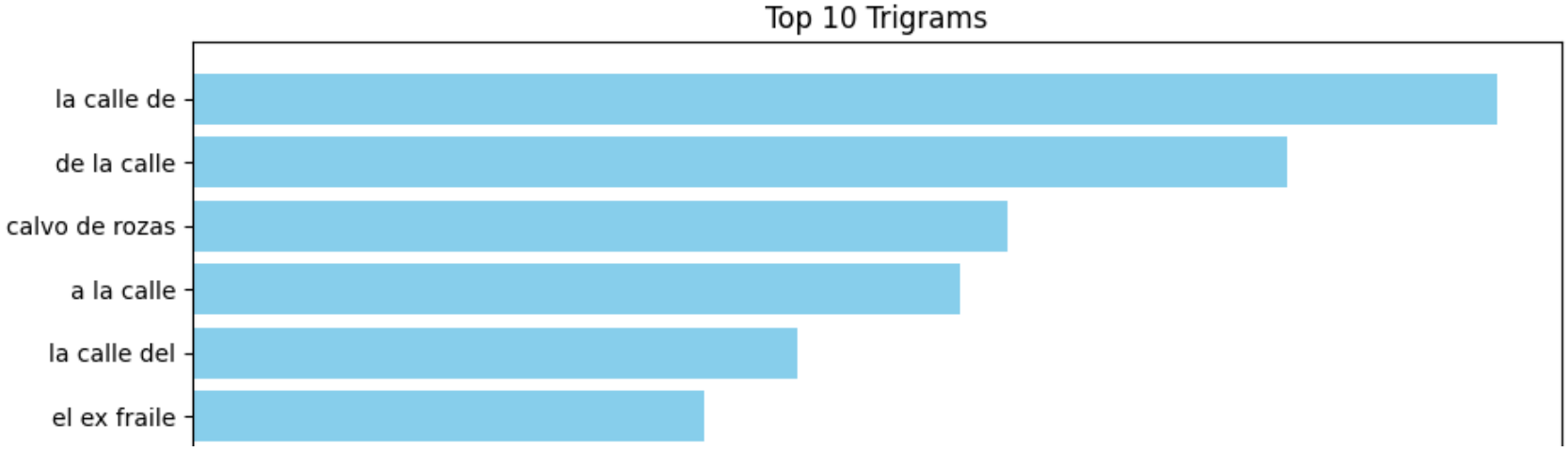
```
# Contando los comunes
top_bigrams = Counter(bigrams).most_common(10)
top_trigrams = Counter(trigrams).most_common(10)
```

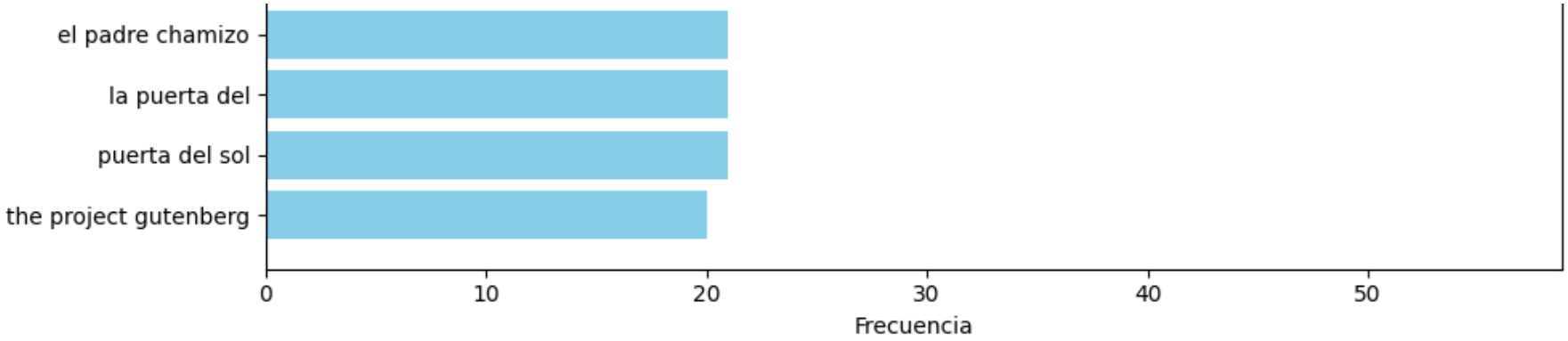
```
# Visualizacion
def plot_ngrams(ngram_freqs, title):
    frases = [' '.join(ng) for ng, _ in ngram_freqs]
    freqs = [freq for _, freq in ngram_freqs]

    plt.figure(figsize=(10, 5))
    plt.barh(frases, freqs, color='skyblue')
    plt.xlabel("Frecuencia")
    plt.title(title)
    plt.gca().invert_yaxis()
    plt.tight_layout()
    plt.show()
```

```
# Graficando
plot_ngrams(top_bigrams, "Top 10 Bigrams")
plot_ngrams(top_trigrams, "Top 10 Trigrams")
```


Top 10 Bigrams
de la
a la
en la
de los
en el
la calle
que se
y se
que no
y de
0
100
200
300
400
Frecuencia


Top 10 Trigrams
la calle de
de la calle
calvo de rozas
a la calle
la calle del
el ex fraile



Conclusion

Al trabajar con un texto largo sobre historia, se logró extraer combinaciones frecuentes de dos y tres palabras, conocidas como bigramas y trigramas. Primero se limpió el texto, dejando solo palabras reales y en minúsculas, para luego formar esas secuencias. Después se contaron cuántas veces aparecían y se generaron gráficas para mostrar cuáles fueron las más comunes.

Este tipo de análisis ayuda bastante a entender los patrones de lenguaje que más se repiten en un tema específico. También permite detectar estructuras comunes y posibles frases clave dentro del texto. La visualización hace más fácil notar rápidamente las combinaciones importantes sin tener que leer todo palabra por palabra.