Taller de Tokenización de Texto

Nombre: *Isaac Reyes*

Desarrollo del Código

En este taller, se desarrolló un código en Python, empleando la biblioteca NLTK, con el objetivo de tokenizar un texto específico, fragmentándolo inicialmente en sentencias y luego en palabras. Este proceso se ejecutó a través de las siguientes etapas:

1. Se importaron las funcionalidades necesarias de NLTK.
2. Se preparó una función para leer y tokenizar el texto de un archivo, abriendo el archivo para leer el texto, reemplazando saltos de línea por espacios para mantener la continuidad del texto.
3. Posteriormente, se dividió el texto en sentencias y cada sentencia en palabras, utilizando las funciones **sent_tokenize** y **word_tokenize**.
4. Finalmente, se mostraron los resultados, incluyendo el total de sentencias, las palabras tokenizadas por sentencia, y el total de palabras en cada sentencia.

python

```
import nltk

from nltk.tokenize import sent_tokenize, word_tokenize


# Downloading necessary NLTK data

nltk.download('punkt')


# Defining the function to tokenize text from a file

def tokenize_text(file_path):

  try:

    # Open and read the file

    with open(file_path, 'r', encoding='utf-8') as file:

      text = file.read().replace('\n', ' ')
```

```
        # Tokenizing the text into sentences
        sentence_tokens = sent_tokenize(text)

        # Tokenizing each sentence into words
        word_tokens = [word_tokenize(sentence) for sentence in sentence_tokens]

        # Displaying the results
        print(f"Total sentences: {len(sentence_tokens)}")
        for i, sentence in enumerate(sentence_tokens):
            print(f"\n[Sentence {i}] -> {sentence}")
            print(f"Word Tokens: {word_tokens[i]}")
            print(f"Total words in sentence {i}: {len(word_tokens[i])}")

    except FileNotFoundError:
        print(f"File not found: {file_path}")

# The path to the text file (to be replaced with the actual path of the text file)
file_path = 'texto.txt'
tokenize_text(file_path)
```

Material Utilizado para Demostración

El texto seleccionado para la demostración fue un extracto de "Lorem Ipsum", resaltando su uso tradicional en la industria de la impresión y su evolución hacia los medios digitales, ilustrando así la continuidad de su relevancia desde el siglo XVI hasta la actualidad.

Funcionamiento Detallado del Programa

El funcionamiento del programa se detalla en una secuencia ordenada que comienza con la lectura del texto desde un archivo. A continuación, el texto se divide en sentencias, y estas a su vez en palabras, con el fin de obtener unidades más pequeñas de análisis. Esta estructura permite una presentación detallada de los resultados, que incluye la cantidad de sentencias, la visualización de las palabras tokenizadas por sentencia y el recuento de palabras por sentencia.

Conclusiones y Reflexiones

- Se concluyó que la tokenización es un paso esencial en el preprocesamiento de datos para el procesamiento de lenguaje natural (NLP), facilitando un análisis más granular y profundo del texto.
- La habilidad para descomponer el texto en sentencias y palabras subraya la importancia de la flexibilidad en el procesamiento de texto, permitiendo la aplicación de técnicas específicas de NLP a diferentes niveles de detalle.

Recomendaciones Prácticas

- Se subraya la importancia de la limpieza previa de los datos antes de la tokenización, recomendando acciones como la eliminación de caracteres especiales, la corrección ortográfica y la homogeneización del texto a minúsculas para asegurar la consistencia.
- Para textos con estructuras más complejas o para el trabajo en diversos idiomas, se recomienda explorar herramientas de tokenización más avanzadas, como las ofrecidas por la biblioteca SpaCy o modelos de transformers como BERT, buscando mejorar la precisión en la tokenización y adaptabilidad a diferentes contextos lingüísticos.