Informe Técnico: Procesamiento de Texto y Análisis de N-Gramas en Español utilizando Python y NLP.

1. Introducción.

El presente informe describe un programa desarrollado en Python cuyo objetivo es **procesar un corpus textual en español**, aplicar técnicas de limpieza, normalización y lematización, y finalmente **extraer y visualizar la frecuencia de n-gramas** (grupos de palabras consecutivas). Este procesamiento se enmarca dentro del campo del **Procesamiento de Lenguaje Natural (PLN)** y permite extraer patrones léxicos frecuentes en textos de cualquier índole.

El programa hace uso de bibliotecas reconocidas como *NLTK*, *spaCy*, *sklearn* y *matplotlib* para realizar tareas clave del preprocesamiento lingüístico.

2. Objetivo del Programa.

El programa busca lograr los siguientes objetivos:

- Cargar y limpiar un corpus textual en español.
- Tokenizar palabras y oraciones.
- Eliminar palabras vacías (stopwords).
- Lematizar el texto (reducir palabras a su forma base).
- Calcular la frecuencia de bigramas y trigramas.
- Visualizar dichos patrones mediante gráficos de barras.

Todo el procesamiento está orientado a obtener una representación más informativa y estructurada del texto original.

3. Estructura y Componentes del Programa.

A. Carga del corpus.

La función cargar_corpus(ruta) abre un archivo de texto, lo convierte a minúsculas y retorna el contenido como una cadena:

```
def cargar_corpus(ruta):
    with open(ruta, 'r') as corpus:
```

```
        texto = corpus.read().lower()
    return texto
```

B. Tokenización.

Se aplican dos tipos de tokenización:

- **De palabras**: usando nltk.word_tokenize, separa el texto en unidades léxicas (tokens), omitiendo puntuación.
- **De oraciones**: usando nltk.sent_tokenize, divide el texto en frases completas.

C. Eliminación de stopwords.

Con la función quitar_stopwords, se filtran las palabras vacías (como "de", "el", "la") utilizando el conjunto de stopwords en español de NLTK.

D. Lematización.

Se emplea el modelo es_core_news_sm de spaCy para reducir las palabras a su forma canónica (lema). Por ejemplo, "trabajando", "trabaja" y "trabajaron" se transforman en "trabajar".

E. Extracción de N-Gramas.

La función obtener_ngrama_frecuencias utiliza CountVectorizer de sklearn para generar bigramas (n=2) y trigramas (n=3), contabilizando su frecuencia de aparición en el texto ya limpio y lematizado.

Se utiliza el parámetro min_df=1 para asegurar que se incluyan todos los n-gramas que aparezcan al menos una vez.

F**. Visualización gráfica.**

Los n-gramas más frecuentes se representan mediante gráficos de barras con matplotlib, mostrando visualmente los patrones léxicos más repetidos en el corpus:

```
plt.bar(etiquetas, valores, color='mediumseagreen')
```

G. Función integradora: procesar_corpus.

Esta función combina todos los pasos anteriores:

1. Carga el corpus.

2. Tokeniza el texto.
3. Imprime el texto limpio, sin stopwords y luego lematizado.
4. Calcula bigramas y trigramas.
5. Grafica la frecuencia de los n-gramas.

Finalmente, ejecuta el análisis sobre un archivo llamado "CorpusEducacion.txt".

4. Resultados esperados.

Al ejecutarse, el programa muestra en consola:

- Lista de tokens originales.
- Cada oración sin stopwords.
- Cada oración lematizada.
- Dos gráficos: frecuencia de bigramas y trigramas.

Esto permite visualizar qué combinaciones de palabras se repiten con mayor frecuencia, lo cual puede ser útil para:

- Análisis temático.
- Extracción de palabras clave.
- Modelado de lenguaje.
- Diagnóstico de estilo o nivel de un texto.

5. Requisitos del sistema.

Para ejecutar el programa, se requiere:

- **Python 3.7 o superior**
- Instalación previa de las siguientes bibliotecas:
    - nltk (pip install nltk)
    - spaCy (pip install spacy)
    - es_core_news_sm (python -m spacy download es_core_news_sm)
    - scikit-learn (pip install scikit-learn)

- matplotlib (pip install matplotlib)

Además, se necesita un archivo de texto en español como fuente de análisis (en este caso: "CorpusEducacion.txt").

6. Conclusión.

El programa implementado ofrece una herramienta sólida y automatizada para el procesamiento y análisis de textos en español. A través de su pipeline lingüístico, se convierte un texto crudo en una representación lematizada, limpia y analizada por patrones de frecuencia. Gracias a su enfoque modular, puede adaptarse fácilmente a otros idiomas o incluir pasos adicionales como análisis de sentimiento, clasificación de temas o extracción de entidades nombradas.

Este tipo de procesamiento resulta particularmente útil en campos como:

- Educación y evaluación de textos escolares.
- Minería de opiniones y reseñas.
- Análisis de contenido en redes sociales o medios digitales.