

Maestría en Inteligencia Artificial Aplicada

PROYECTO INTEGRADOR

Avance 5. Modelo final

Equipo 31

JUAN CARLOS VILLAMIL ROJAS A01794003


MATEO CRUZ LANCHERO A01793882

ANDREA MARGARITA OSORIO GONZÁLEZ A01104776

2/6/2024

Avance #5 (Trabajo en progreso, modelo final aún no definido)

```
from google.colab import drive
drive.mount('/content/drive')
```

 Mounted at /content/drive

```
!pip install moviepy transformers
!pip install openai==0.28
!pip install transformers
!pip install bert-extractive-summarizer
!pip install rouge
```



```
installing collected packages: bert-extractive-summarizer
Successfully installed bert-extractive-summarizer-0.10.1
Collecting rouge
  Downloading rouge-1.0.1-py3-none-any.whl (13 kB)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from rouge) (1.16.0)
Installing collected packages: rouge
```

```
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from moviepy.editor import VideoFileClip
from transformers import pipeline, GPT2Tokenizer, GPT2LMHeadModel
from rouge import Rouge
from summarizer import Summarizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.decomposition import TruncatedSVD
from sklearn.metrics.pairwise import cosine_similarity
from moviepy.editor import VideoFileClip
import time
import warnings
warnings.filterwarnings('ignore')

from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
import math

# Configuración de estilo para matplotlib
plt.style.use('ggplot')
```

```

# Directorios de los videos y resúmenes de referencia
video_dir = '/content/drive/My Drive/Videos/'
reference_summary_dir = '/content/drive/My Drive/Reference_Summaries/'
transcript_dir = '/content/drive/My Drive/Transcripts/'

# Obtener listas de archivos de videos, resúmenes de referencia y transcripciones
video_files = [os.path.join(video_dir, f) for f in os.listdir(video_dir) if f.endswith('.mp4')]
reference_summaries = [os.path.join(reference_summary_dir, f) for f in os.listdir(reference_summary_dir) if f.endswith('.txt')]
transcript_files = [os.path.join(transcript_dir, f) for f in os.listdir(transcript_dir) if f.endswith('.txt')]

# Ordenar las listas de archivos por nombre de archivo
video_files.sort()
# Ordenar en su lugar
reference_summaries = sorted(reference_summaries) # Volver a asignar la lista
transcript_files.sort()

# Verificar que el número de videos, resúmenes y transcripciones coincida
assert len(video_files) == len(reference_summaries) == len(transcript_files), "El número de videos, resúmenes de referencia y tr

# Verificar que los nombres de los archivos coincidan en las mismas posiciones
for i in range(len(video_files)):
    assert os.path.splitext(os.path.basename(video_files[i]))[0] == os.path.splitext(os.path.basename(reference_summaries[i]))[0]

# Leer los resúmenes de referencia (corrección: usando reference_summaries)
reference_texts = [open(f).read() for f in reference_summaries]

# Leer las transcripciones
transcripts = [open(f).read() for f in transcript_files]

# Crear un DataFrame para el análisis (ordenar listas antes de crear DataFrame)
data = {
    'Video': [os.path.basename(f) for f in video_files],
    'Transcript': transcripts,
    'Reference Summary': reference_texts
}
df = pd.DataFrame(data)

# Imprimir las primeras 5 filas del DataFrame para verificar
df.head(5)

# Mostrar los nombres de los archivos de los primeros 5 videos
for i in range(5):
    print(f"Video {i+1}: {df['Video'][i]}")

# Mostrar los nombres de los archivos de los primeros 5 videos
#for i in range(5):
#    print(f"Transcript {i+1}: {df['Transcript'][i]}")

# Mostrar los nombres de los archivos de los primeros 5 videos
for i in range(5):
    print(f"Reference Summary {i+1}: {os.path.splitext(os.path.basename(reference_summaries[i]))[0]}")

Video 1: AXEL KAISER y BRUTAL PARTICIPACIÓN en su PRIMER CAPÍTULO.mp4
Video 2: Axel Kaiser y Javier Milei – El renacimiento liberal.mp4
Video 3: Cómo sobrevivir a la destrucción de la industria de la tecnología.mp4
Video 4: Creencias _ Daniel Habif _ TEDxCiudaddePuebla.mp4
Video 5: El nuevo orden mundial de China y la dependencia de Occidente.mp4
Reference Summary 1: AXEL KAISER y BRUTAL PARTICIPACIÓN en su PRIMER CAPÍTULO
Reference Summary 2: Axel Kaiser y Javier Milei – El renacimiento liberal
Reference Summary 3: Cómo sobrevivir a la destrucción de la industria de la tecnología
Reference Summary 4: Creencias _ Daniel Habif _ TEDxCiudaddePuebla
Reference Summary 5: El nuevo orden mundial de China y la dependencia de Occidente

print(os.path.splitext(os.path.basename(video_files[0]))[0])

AXEL KAISER y BRUTAL PARTICIPACIÓN en su PRIMER CAPÍTULO

print(os.path.splitext(os.path.basename(reference_summaries[0]))[0])

AXEL KAISER y BRUTAL PARTICIPACIÓN en su PRIMER CAPÍTULO

print(os.path.splitext(os.path.basename(transcript_files[0]))[0])

AXEL KAISER y BRUTAL PARTICIPACIÓN en su PRIMER CAPÍTULO

```

```
df['Video']
```

```
0 AXEL KAISER y BRUTAL PARTICIPACIÓN en su PRIM...
1 Axel Kaiser y Javier Milei - El renacimiento l...
2 Cómo sobrevivir a la destrucción de la indus...
3 Creencias _ Daniel Habif _ TEDxCiudaddePuebla.mp4
4 El nuevo orden mundial de China y la dependenc...
5 Estados Unidos vs. China_La GUERRA de CHIPS.mp4
6 GPT4o - La Gran Apuesta de OpenAI por la MULTI...
7 Google, Facebook, Amazon - El poder ilimitado ...
8 LO QUE DEBEN SABER HOY LOS AFILIADOS de la EPS...
9 La Entrevista que no quiso hacer Gabi Desangle...
10 La barrera de la que no se habla-ENRIQUE VÁZQ...
11 Musica generado por IA.mp4
12 Persuadir e influir como todo un agente 007_Fe...
13 Pocos Entienden Esto de la Física Moderna....mp4
14 Por los sueños se suspira, por las metas se t...
15 Respuestas con Axel Káiser.mp4
16 Somos lo que Pensamos _ Yirko Sivirich _ TEDxL...
17 Te reto a ser feliz _ Omar Chaparro _ TED.mp4
18 iOS 18, Apple Intelligence y Nueva Siri.mp4
19 sistema_financiero _ Hernan Casciari _ TED.mp4
Name: Video, dtype: object
```

```
df['Reference Summary']
```

```
0 El programa de debate "Strending Topics" prese...
1 Este video de Youtube presenta una conversació...
2 Este video de Platzi aborda la sensación de cr...
3 En esta charla TEDx, Daniel David reflexiona s...
4 Este documental de DW explora la creciente dep...
5 Este video de Youtube explora la creciente ten...
6 Este video de Youtube analiza la nueva tecnolo...
7 Este documental de DW explora el creciente pod...
8 Este video de Youtube trata sobre la decisión ...
9 Este video de Youtube presenta una conversació...
10 En su charla TEDx, Enrique Vázquez, un psicólo...
11 El video explora el impacto de la Inteligencia...
12 El orador del video, Felipe Riano, argumenta q...
13 Este video de Youtube explica la segunda ley d...
14 En esta charla TEDx, Humberto Ramos, un recono...
15 Este video de Youtube presenta una conversació...
16 En esta charla TEDxLima, Yirko Sivirich, un di...
17 En esta charla TEDx, el actor y comediante Oma...
18 Este video de Youtube resume las novedades que...
19 Es una historia contada por Hernán Casciari pa...
Name: Reference Summary, dtype: object
```

```
from google.colab import files
```

```
# Subir el archivo de la clave de API
```

```
#uploaded = files.upload()
```

```
# Leer la clave de API
```

```
#api_key = None
```

```
#for fn in uploaded.keys():
```

```
    #with open(fn) as f:
```

```
        #api_key = f.read().strip()
```

```

#-----> Utilizado para generar las transcripciones, una vez obtenidas no se debe correr de nuevo.
#!pip install pydub
#from pydub import AudioSegment
#import math
#import openai

# Función para dividir el audio en segmentos más pequeños
#def split_audio(audio_path, max_segment_size_bytes=25000000): # 25 MB
#    audio = AudioSegment.from_file(audio_path)
#    duration_ms = len(audio)
#    segments = []
#    segment_length_ms = 0

#    # Calcula la longitud de los segmentos basándose en el tamaño máximo permitido
#    for i in range(duration_ms):
#        if (audio[i:i+1000].frame_rate * audio[i:i+1000].sample_width * audio[i:i+1000].channels * (i+1000) / 8) <= max_segment
#            segment_length_ms += 1000
#        else:
#            break

#    for start in range(0, duration_ms, segment_length_ms):
#        end = min(start + segment_length_ms, duration_ms)
#        segment = audio[start:end]
#        segment_path = f"segment_{start//1000}_{end//1000}.wav"
#        segment.export(segment_path, format="wav")
#        segments.append(segment_path)

#    return segments

# Configurar la clave de API de OpenAI
#openai.api_key = api_key

#def transcribe_audio_openai(audio_path):
#    audio_file = open(audio_path, "rb")
#    transcript = openai.Audio.transcribe("whisper-1", audio_file)
#    return transcript["text"]

#def transcribe_audio_segments(segments):
#    full_transcript = ""
#    for segment in segments:
#        transcript = transcribe_audio_openai(segment)
#        full_transcript += transcript + " "
#    return full_transcript.strip()
#
#def extract_audio_from_video(video_path):
#    video = VideoFileClip(video_path)
#    audio_path = video_path.replace('.mp4', '.wav')
#    video.audio.write_audiofile(audio_path)
#    return audio_path

# Process each video file
#for video_file in video_files:
#    # Extract audio from video
#    audio_path = extract_audio_from_video(video_file)

#    # Split audio into smaller segments
#    segments = split_audio(audio_path)

#    # Transcribe each segment and combine the transcriptions
#    full_transcript = transcribe_audio_segments(segments)

#    # Save the full transcript to a text file
#    transcript_file = video_file.replace('.mp4', '.txt').replace(video_dir, '/content/drive/My Drive/Transcripts/')
#    with open(transcript_file, 'w') as f:
#        f.write(full_transcript)

#    print(f"Transcription for {video_file} saved to {transcript_file}")

```

Preview of dataframe

df.head(5)

	Video	Transcript	Reference Summary	
0	AXEL KAISER y BRUTAL PARTICIPACIÓN en su PRIM...	Pregunta también, sobre todo para Axel, que me...	El programa de debate "Strending Topics" prese...	
1	Axel Kaiser y Javier Milei - El renacimiento l...	Mi nombre es Mara Sedilli, soy Directora de As...	Este video de Youtube presenta una conversació...	
2	Cómo sobrevivir a la destrucción de la indus...	¿Cómo sobrevivir a la destrucción que está ocu...	Este video de Platzi aborda la sensación de	

Próximos pasos: [Generar código con df](#) [Ver gráficos recomendados](#)

```
len(df)
```

```
20
```

```
df.isnull().sum()
```

```
Video          0
Transcript      0
Reference Summary  0
dtype: int64
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20 entries, 0 to 19
Data columns (total 3 columns):
#   Column          Non-Null Count  Dtype
---  ---
0   Video           20 non-null    object
1   Transcript       20 non-null    object
2   Reference Summary 20 non-null    object
dtypes: object(3)
memory usage: 608.0+ bytes
```

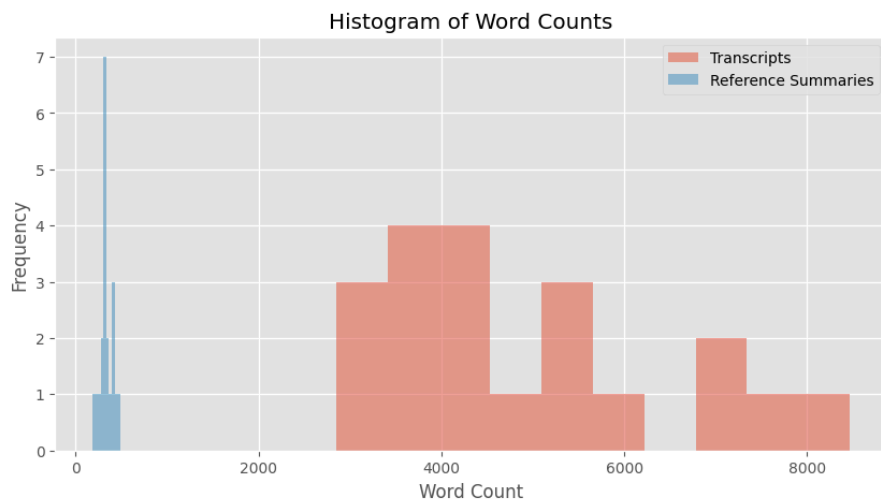
```
# Número de palabras por transcripción y resumen de referencia
df['Transcript Word Count'] = df['Transcript'].apply(lambda x: len(x.split()))
df['Reference Summary Word Count'] = df['Reference Summary'].apply(lambda x: len(x.split()))

# Duración de cada video transcrito en minutos
video_durations = [VideoFileClip(f).duration / 60 for f in video_files] # Duración en minutos
df['Video Duration (minutes)'] = video_durations

# Media del número de palabras
mean_transcript_words = df['Transcript Word Count'].mean()
mean_reference_words = df['Reference Summary Word Count'].mean()
print(f"Mean Transcript Words: {mean_transcript_words}")
print(f"Mean Reference Summary Words: {mean_reference_words}")

plt.figure(figsize=(10, 5))
plt.hist(df['Transcript Word Count'], bins=10, alpha=0.5, label='Transcripts')
plt.hist(df['Reference Summary Word Count'], bins=10, alpha=0.5, label='Reference Summaries')
plt.xlabel('Word Count')
plt.ylabel('Frequency')
plt.title('Histogram of Word Counts')
plt.legend(loc='upper right')
plt.show()
```

Mean Transcript Words: 4869.2
Mean Reference Summary Words: 338.1



```
df['Transcript'][0]
```

'Pregunta también, sobre todo para Axel, que me encanta volver a polimizar con él y para los demás también. ¿Qué piensan ustedes de cómo se podría crear empleo en el corto plazo? Besos y abrazos. Ahí está, Marco Enrique Uminami, ya me lo presentó. ¡Ah, mierda! Le metió cruce, le dejó pregunta instalada. ¡Mierda! Error, error. Junto a nosotros, cada vez que habla Strending Topics, saca noticia en todas partes. Acá tiene los récords de programas junto a nosotros. destacado

```
df['Reference Summary'][0]
```

'El programa de debate "Strending Topics" presenta una discusión acalorada sobre la situación actual de Chile, con un enfoque en la economía, la seguridad y la política.\nPuntos clave:\nEconomía: Axel Kaiser alerta sobre la situación fiscal crítica de Chile, con una deuda pública cercana al 60% del PIB. Critica la falta de crecimiento económico y la incertidumbre jurídica que obstaculizan la inversión. Pronone reformas para crear empleo: reducir el impuesto a las empresas

```
df['Video'][0]
```

'AXEL KAISER y BRUTAL PARTICIPACIÓN en su PRIMER CAPÍTULO.mp4'

```
# Histogramas del conteo de palabras y duración de videos
```

```
plt.figure(figsize=(15, 5))
```

```
plt.subplot(1, 3, 1)
```

```
plt.hist(df['Transcript Word Count'], bins=10, alpha=0.5, label='Transcripts')
```

```
plt.xlabel('Word Count')
```

```
plt.ylabel('Frequency')
```

```
plt.title('Histogram of Transcript Word Counts')
```

```
plt.legend(loc='upper right')
```

```
plt.subplot(1, 3, 2)
```

```
plt.hist(df['Reference Summary Word Count'], bins=10, alpha=0.5, label='Reference Summaries')
```

```
plt.xlabel('Word Count')
```

```
plt.ylabel('Frequency')
```

```
plt.title('Histogram of Reference Summary Word Counts')
```

```
plt.legend(loc='upper right')
```

```
plt.subplot(1, 3, 3)
```

```
plt.hist(df['Video Duration (minutes)'], bins=10, alpha=0.5, label='Video Duration')
```

```
plt.xlabel('Duration (minutes)')
```

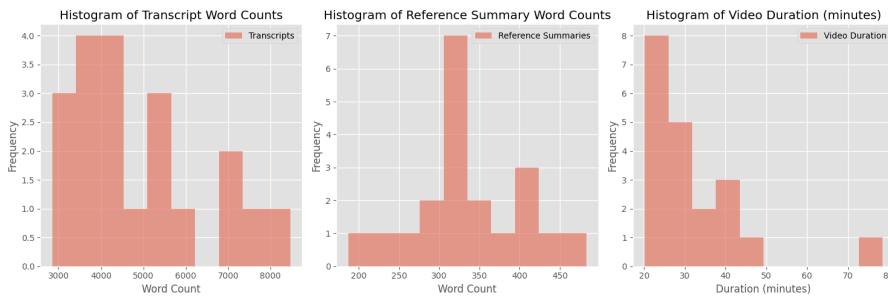
```
plt.ylabel('Frequency')
```

```
plt.title('Histogram of Video Duration (minutes)')
```

```
plt.legend(loc='upper right')
```

```
plt.tight_layout()
```

```
plt.show()
```



```
!python -m spacy download es_core_news_sm
```



```
Collecting es-core-news-sm==3.7.0
  Downloading https://github.com/explosion/spacy-models/releases/download/es\_core\_news\_sm-3.7.0/es\_core\_news\_sm-3.7.0-py3-no
    12.9/12.9 MB 90.8 MB/s eta 0:00:00
Requirement already satisfied: spacy<3.8.0,>=3.7.0 in /usr/local/lib/python3.10/dist-packages (from es-core-news-sm==3.7.0)
Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.11 in /usr/local/lib/python3.10/dist-packages (from spacy<3.8.0,>=3.7.0)
Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.8.0,>=3.7.0)
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.8.0,>=3.7.0)
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in /usr/local/lib/python3.10/dist-packages (from spacy<3.8.0,>=3.7.0)
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in /usr/local/lib/python3.10/dist-packages (from spacy<3.8.0,>=3.7.0)
Requirement already satisfied: thinc<8.3.0,>=8.2.2 in /usr/local/lib/python3.10/dist-packages (from spacy<3.8.0,>=3.7.0)
Requirement already satisfied: wasabi<1.2.0,>=0.9.1 in /usr/local/lib/python3.10/dist-packages (from spacy<3.8.0,>=3.7.0)
Requirement already satisfied: srsly<3.0.0,>=2.4.3 in /usr/local/lib/python3.10/dist-packages (from spacy<3.8.0,>=3.7.0)
Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in /usr/local/lib/python3.10/dist-packages (from spacy<3.8.0,>=3.7.0)
Requirement already satisfied: weasel<0.5.0,>=0.1.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.8.0,>=3.7.0)
Requirement already satisfied: typer<1.0.0,>=0.3.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.8.0,>=3.7.0)
Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.8.0,>=3.7.0)
Requirement already satisfied: requests<3.0.0,>=2.13.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.8.0,>=3.7.0)
Requirement already satisfied: pydantic!=1.8,!1.8.1,<3.0.0,>=1.7.4 in /usr/local/lib/python3.10/dist-packages (from spacy<3.8.0,>=3.7.0)
Requirement already satisfied: Jinja2 in /usr/local/lib/python3.10/dist-packages (from spacy<3.8.0,>=3.7.0)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from spacy<3.8.0,>=3.7.0)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.8.0,>=3.7.0)
Requirement already satisfied: langcodes<4.0.0,>=3.2.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.8.0,>=3.7.0)
Requirement already satisfied: numpy>=1.19.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.8.0,>=3.7.0)
Requirement already satisfied: language-data>=1.2 in /usr/local/lib/python3.10/dist-packages (from spacy<3.8.0,>=3.7.0)
Requirement already satisfied: annotated-types>=0.4.0 in /usr/local/lib/python3.10/dist-packages (from pydantic!=1.8,!1.8.1)
Requirement already satisfied: pydantic-core==2.18.4 in /usr/local/lib/python3.10/dist-packages (from pydantic!=1.8,!1.8.1)
Requirement already satisfied: typing-extensions>=4.6.1 in /usr/local/lib/python3.10/dist-packages (from pydantic!=1.8,!1.8.1)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0,>=2.13.0)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0,>=2.13.0)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0,>=2.13.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0,>=2.13.0)
Requirement already satisfied: blis<0.8.0,>=0.7.8 in /usr/local/lib/python3.10/dist-packages (from thinc<8.3.0,>=8.2.2)
Requirement already satisfied: confection<1.0.0,>=0.0.1 in /usr/local/lib/python3.10/dist-packages (from thinc<8.3.0,>=8.2.2)
Requirement already satisfied: click>=8.0.0 in /usr/local/lib/python3.10/dist-packages (from typer<1.0.0,>=0.3.0)
Requirement already satisfied: shellingham>=1.3.0 in /usr/local/lib/python3.10/dist-packages (from typer<1.0.0,>=0.3.0)
Requirement already satisfied: rich>=10.11.0 in /usr/local/lib/python3.10/dist-packages (from typer<1.0.0,>=0.3.0)
Requirement already satisfied: cloudpathlib<1.0.0,>=0.7.0 in /usr/local/lib/python3.10/dist-packages (from weasel<0.5.0,>=0.1.0)
Requirement already satisfied: smart-open<8.0.0,>=5.2.1 in /usr/local/lib/python3.10/dist-packages (from weasel<0.5.0,>=0.1.0)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from Jinja2)
Requirement already satisfied: marisa-trie>=0.7.7 in /usr/local/lib/python3.10/dist-packages (from language-data>=1.2)
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.10/dist-packages (from rich>=10.11.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.10/dist-packages (from rich>=10.11.0)
Requirement already satisfied: wrapt in /usr/local/lib/python3.10/dist-packages (from smart-open<8.0.0,>=5.2.1)
Requirement already satisfied: mdurl>=0.1 in /usr/local/lib/python3.10/dist-packages (from markdown-it-py>=2.2.0)
Installing collected packages: es-core-news-sm
Successfully installed es-core-news-sm-3.7.0
✓ Download and installation successful
You can now load the package via spacy.load('es_core_news_sm')
▲ Restart to reload dependencies
If you are in a Jupyter or Colab notebook, you may need to restart Python in order to load all the package's dependencies. You can do this by selecting the 'Restart kernel' or 'Restart runtime' option.
```



```

# Funciones para el análisis de datos y preprocesamiento de texto
import spacy
import nltk
from nltk.corpus import stopwords
import re

nltk.download('punkt')
nltk.download('stopwords')
nlp = spacy.load("es_core_news_sm") # Cargar el modelo de SpaCy para español

# Función para limpiar el texto
def clean_text(text):
    text = text.lower()
    # Remover texto entre corchetes
    text = re.sub(r'\[.*?\]', '', text)
    # Remover palabras con números (opcional, puedes mantenerlas)
    # text = re.sub(r'\w*\d\w*', '', text)
    # Remover caracteres especiales
    text = re.sub(r'[%s]' % re.escape(r'!"#$%&'()*+,-./:;<=>@[\\]^_`{|}~'), '', text)
    # Remover espacios adicionales
    text = re.sub(r'\s+', ' ', text).strip()
    return text

# Función para preprocesar el texto
def preprocess_text(text):
    stop_words = set(stopwords.words('spanish'))
    stop_words.update(['uno', "dos", "tres", "cuatro", "cinco", "seis", "siete", "ocho", "nueve", "diez",
                      "si", "voy", "subtítulos", "realizados", "amaraorg", "ahí", "di", "va", "van", "etcétera", "ver", "gracias"])

    # Detectar plurales y evitar lematización
    words = text.split()
    for i, word in enumerate(words):
        # Reglas más completas para detectar plurales
        if word.endswith(("s", "es", "as", "os", "ces", "zes", "ies")) or \
            (word.endswith("a") and word[:-1].endswith(("s", "es", "as", "os", "ces", "zes", "ies"))):
            words[i] = word # Evita lematizar la palabra

    text = ' '.join(words) # Reconstruir la frase

    text = clean_text(text) # Limpiar el texto después de detectar plurales
    doc = nlp(text)

    # Mantener los números como tokens (opcional)
    words = [token.text for token in doc if (token.is_alpha or token.like_num) and token.text.lower() not in stop_words]

    # Omitir palabras que empiezan con "www"
    words = [word for word in words if not word.startswith("www")]

    return ' '.join(words) # Devuelve la frase preprocesada y None

# Aplicar la limpieza y el preprocesamiento a las transcripciones y los resúmenes de referencia
df['Cleaned Transcript'] = df['Transcript'].apply(preprocess_text)
df['Cleaned Reference Summary'] = df['Reference Summary'].apply(preprocess_text)

print("Transcripciones y Resúmenes de Referencia preprocesados:")
print(df[['Video', 'Cleaned Transcript', 'Cleaned Reference Summary']])

# Número de palabras por transcripción y resumen de referencia
df['Cleaned Transcript']
df['Cleaned Transcript Word Count'] = df['Cleaned Transcript'].apply(lambda x: len(x.split()))
df['Cleaned Reference Summary Word Count'] = df['Cleaned Reference Summary'].apply(lambda x: len(x.split()))

# Media del número de palabras
mean_cleaned_transcript_words = df['Cleaned Transcript Word Count'].mean()
mean_cleaned_reference_words = df['Cleaned Reference Summary Word Count'].mean()
print(f"Mean Cleaned Transcript Words: {mean_cleaned_transcript_words}")
print(f"Mean Cleaned Reference Summary Words: {mean_cleaned_reference_words}")

# Histogramas del conteo de palabras
plt.figure(figsize=(10, 5))
plt.hist(df['Cleaned Transcript Word Count'], bins=10, alpha=0.5, label='Cleaned Transcript')
plt.hist(df['Cleaned Reference Summary Word Count'], bins=10, alpha=0.5, label='Cleaned Reference Summaries')
plt.xlabel('Cleaned Word Count')
plt.ylabel('Frequency')
plt.title('Histogram of Cleaned Word Counts')

```

```
plt.legend(loc='upper right')  
plt.show()
```

```

1 Axel Kaiser y Javier Neco - El Rendimiento de...
2 Cómo sobrevivir a la destrucción de la indus...
3 Creencias _ Daniel Habif _ TEDxCiudaddePuebla.mp4
4 El nuevo orden mundial de China y la dependenc...
5 Estados Unidos vs. China_La GUERRA de CHIPS.mp4
6 GPT4o - La Gran Apuesta de OpenAI por la MULTI...
7 Google, Facebook, Amazon - El poder ilimitado ...
8 LO QUE DEBEN SABER HOY LOS AFILIADOS de la EPS...
9 La Entrevista que no quiso hacer Gabi Desangle...
10 La barrera de la que no se habla-ENRIQUE VÁZQ...
11 Musica generado por IA.mp4
12 Persuadir e influir como todo un agente 007_Fe...
13 Pocos Entienden Esto de la Física Moderna...mp4
14 Por los sueños se suspira, por las metas se t...
15 Respuestas con Axel Kaiser.mp4
16 Somos lo que Pensamos _ Yirko Sivirich _ TEDxL...
17 Te reto a ser feliz _ Omar Chaparro _ TED.mp4
18 iOS 18, Apple Intelligence y Nueva Siri.mp4
19 sistema_financiero _ Hernan Casciari _ TED.mp4

```

Cleaned Transcript \

```

0 pregunta axel encanta volver polimizar demás p...
1 nombre mara sedilli directora asuntos públicos...
2 cómo sobrevivir destrucción ocurriendo industr...
3 aquí presentes permitirme formar parte pedazo ...
4 día advierten comienza noticia alarmante china...
5 fábricas empresa tsmc ciudad tainan taiwán tsm...
6 hace par semanas openai hizo gusta evento relá...
7 silicon valley california aquí apple google co...
8 doctora ayano buenos días buenos días nistor j...
9 vivo conversando momento mujeres borde agustín...
10 nervios perdón tardé llegar 1987 mujer sala em...
11 hit musical escrito inteligencia artificial so...
12 bueno vamos comenzar preguntas ustedes primera...
13 hoy trata conceptos importantes embargo menos ...
14 hola humberto ramos mexicano sagitario impensi...
15 hola iván duque bienvenidos respuestas espacio...
16 girko bienvenido tal tremendo dejo bueno chico...
17 hola buenas tardes tijuana escuchan gusto priv...
18 hola bienvenidos wwdc apple park empezar event...
19 hija 13 años 2012 colapso financiero cayeron t...

```

Cleaned Reference Summary

```

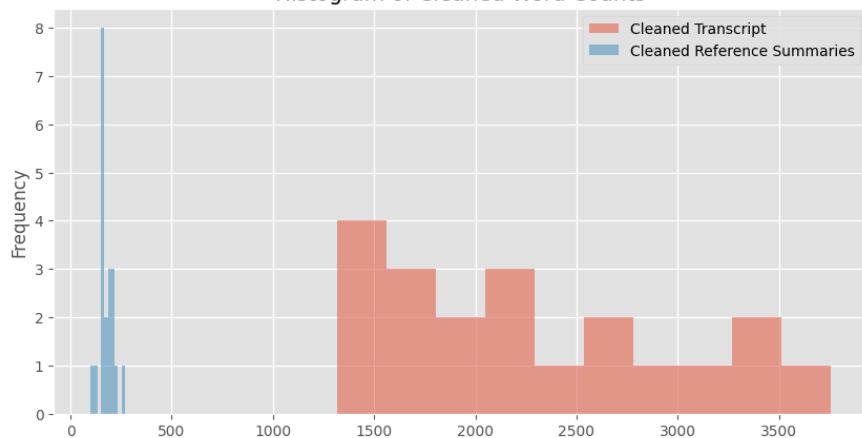
0 programa debate strending topics discusión aca...
1 economista escritor libertario áxel kaiser dip...
2 platzi aborda sensación crisis industria tecno...
3 daniel david reflexiona vida experiencias dest...
4 documental dw creciente dependencia alemania h...
5 creciente tensión geopolítica unidos china con...
6 analiza nueva tecnología multimodal openai imp...
7 documental dw creciente poder grandes empresas...
8 trata decisión eps sura solicitar desmonte pro...
9 periodista néstor escritor conferencista agust...
10 enrique vázquez psicólogo activista parálisis ...
11 impacto inteligencia artificial ia industria m...
12 orador felipe riano argumenta realidad constru...
13 explica segunda ley termodinámica impacto univ...
14 humberto ramos reconocido dibujante cómics mex...
15 iván duque ex presidente colombia áxel kaiser ...
16 tedxlima yirko sivirich diseñador moda peruano...
17 actor comediante omar chaparro comparte herra...
18 resume novedades apple anunció evento wwdc 202...
19 historia contada hernán cascari explicar cris...

```

Mean Cleaned Transcript Words: 2272.75

Mean Cleaned Reference Summary Words: 177.9

Histogram of Cleaned Word Counts



```
df['Cleaned Transcript'][0]
```

```
'pregunta axel encanta volver polimizar demás piensan ustedes cómo podría crear
empleo corto plazo besos abrazos marco enrique uminami presentó ah mierda metió
cruce dejó pregunta instalada mierda error error junto cada vez habla strending
topics saca noticia todas partes acá récords programas junto destacado abogado
economista axel kayser cómo axel buenas noches bienvenido bien contento vuelta
acá gonzalo invitación bueno marco metió economía exactamente quería hablar pro
```

```
df['Cleaned Reference Summary'][0]
```

```
'programa debate strending topics discusión acalorada situación actual chile en
foque economía seguridad política puntos clave economía axel kaiser alerta situ
ación fiscal crítica chile deuda pública cercana 60 pib critica falta crecimien
to económico incertidumbre jurídica obstaculizan inversión propone reformas cre
ar empleo reducir impuesto empresas 20 reintegrar sistema iva 100 derogar ley p
ersecución empresarial seguridad nancho iavier acusa gobierno tener agenda perm
```

```
cnt=0
for i in df['Cleaned Reference Summary']:
    if(len(i.split())<=225):
        cnt=cnt+1
print(cnt/len(df['Cleaned Reference Summary']))
```

```
0.9
```

```
cnt=0
for i in df['Cleaned Transcript']:
    if(len(i.split())<=3350):
        cnt=cnt+1
print(cnt/len(df['Cleaned Transcript']))
```

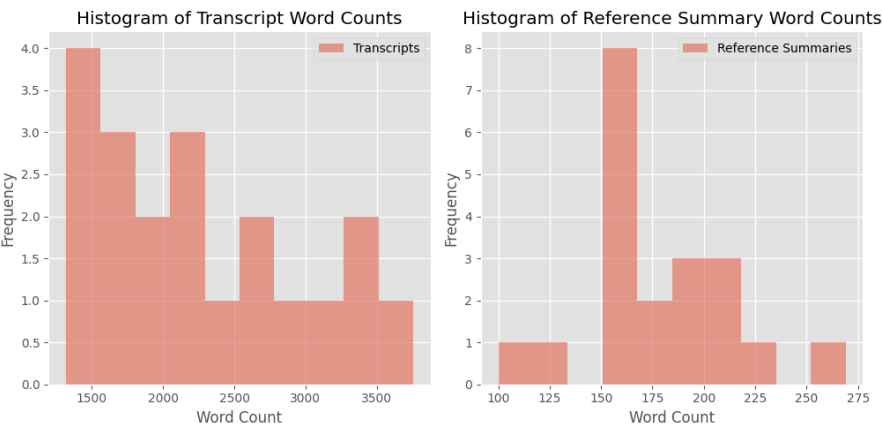
```
0.9
```

```
# Histogramas del conteo de palabras y duración de videos
plt.figure(figsize=(15, 5))
```

```
plt.subplot(1, 3, 1)
plt.hist(df['Cleaned Transcript Word Count'], bins=10, alpha=0.5, label='Transcripts')
plt.xlabel('Word Count')
plt.ylabel('Frequency')
plt.title('Histogram of Transcript Word Counts')
plt.legend(loc='upper right')
```

```
plt.subplot(1, 3, 2)
plt.hist(df['Cleaned Reference Summary Word Count'], bins=10, alpha=0.5, label='Reference Summaries')
plt.xlabel('Word Count')
plt.ylabel('Frequency')
plt.title('Histogram of Reference Summary Word Counts')
plt.legend(loc='upper right')
```

```
plt.tight_layout()
plt.show()
```



```
df_limpio = pd.DataFrame({
    'Video': df['Video'],
    'Transcript': df['Cleaned Transcript'],
    'Reference Summary': df['Cleaned Reference Summary']
})

df_limpio.to_csv('df_limpio.csv', index=False)

df_limpio.head(5)
```



	Video	Transcript	Reference Summary
0	AXEL KAISER y BRUTAL PARTICIPACIÓN en su PRIM...	pregunta axel encanta volver polimizar demás p...	programa debate strending topics discusión aca...
1	Axel Kaiser y Javier Milei - El renacimiento l...	nombre mara sedilli directora asuntos públicos...	economista escritor libertario áxel kaiser dip...
2	Cómo sobrevivir a la destrucción de	cómo sobrevivir destrucción ocurriendo	platzi aborda sensación

Próximos pasos: [Generar código con df_limpio](#) [Ver gráficos recomendados](#)

Analyzing n-grams - Transcript

```
X = df['Cleaned Transcript']
y = df.drop('Cleaned Transcript', axis=1)

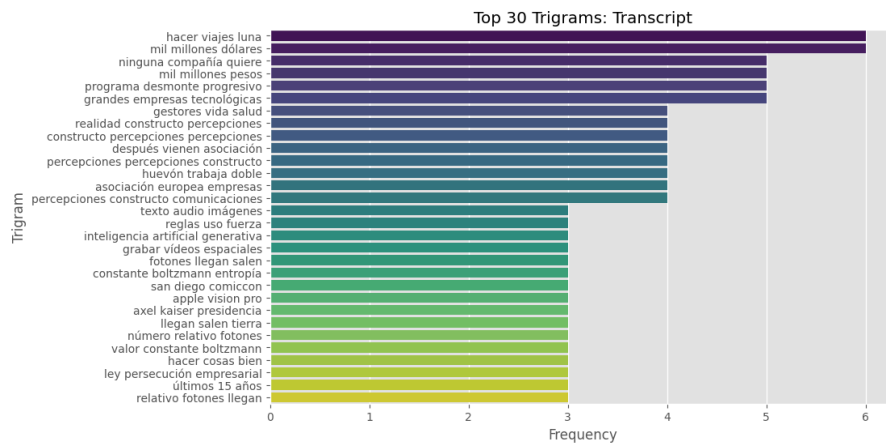
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(ngram_range = (3,3))
X_count = cv.fit_transform(X)
X_count = pd.DataFrame.sparse.from_spmatrix(X_count)
X_count.columns = sorted(cv.vocabulary_)
X_count.set_index(y.index, inplace=True)

all_tri_labels = X_count.sum().sort_values(ascending = False)[0:30]
```

```
# Create a bar plot
plt.figure(figsize=(10, 6))
sns.barplot(x=all_tri_labels.values, y=all_tri_labels.index, palette='viridis')

# Set plot labels and title
plt.xlabel('Frequency')
plt.ylabel('Trigram')
plt.title('Top 30 Trigrams: Transcript')

# Display the plot
plt.show()
```



Analyzing n-grams - Reference Summary

```
X = df['Cleaned Reference Summary']
y = df.drop('Cleaned Reference Summary', axis=1)

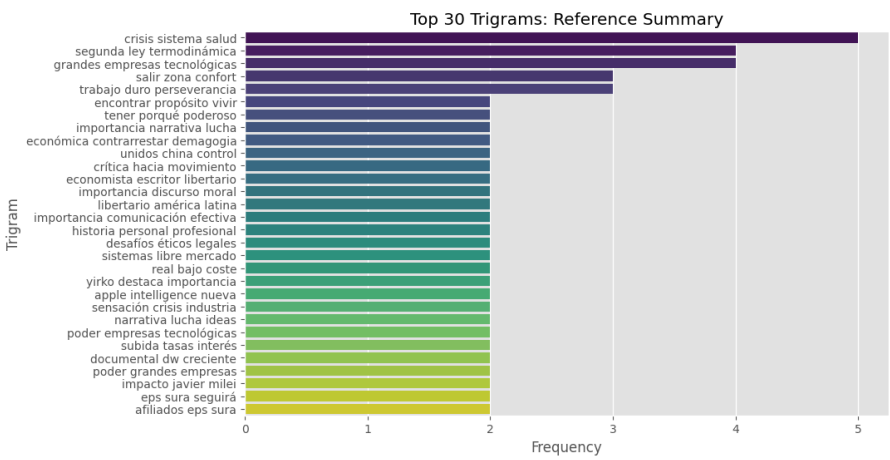
cv = CountVectorizer(ngram_range = (3,3))
X_count = cv.fit_transform(X)
X_count = pd.DataFrame.sparse.from_spmatrix(X_count)
X_count.columns = sorted(cv.vocabulary_)
X_count.set_index(y.index, inplace=True)

all_tri_labels = X_count.sum().sort_values(ascending = False)[0:30]

# Create a bar plot
plt.figure(figsize=(10, 6))
sns.barplot(x=all_tri_labels.values, y=all_tri_labels.index, palette='viridis')

# Set plot labels and title
plt.xlabel('Frequency')
plt.ylabel('Trigram')
plt.title('Top 30 Trigrams: Reference Summary')

# Display the plot
plt.show()
```



```
# --- Modelos de Resumen ---
```

```
# Función para dividir el texto en partes más pequeñas
```

```
def split_text(text, max_length=512):
    sentences = text.split('.')
    chunks = []
    current_chunk = []
    current_length = 0

    for sentence in sentences:
        sentence_length = len(sentence.split())
        if current_length + sentence_length > max_length:
            # Solo agrega el chunk si tiene contenido
            if current_chunk:
                chunks.append(' '.join(current_chunk) + '.')
                current_chunk = [sentence]
                current_length = sentence_length
            else:
                current_chunk.append(sentence)
                current_length += sentence_length

    # Solo agrega el último chunk si tiene contenido
    if current_chunk:
        chunks.append(' '.join(current_chunk) + '.')

    return chunks
```

```
# Función para resumir cada parte del texto
```

```
def summarize_text_chunks(chunks, summarizer):
    summaries = [] # Declarar summaries como una lista vacía
    total_time = 0

    for chunk in chunks:
        input_length = len(chunk.split())
        max_length = min(150, input_length) # Ajustar max_length a la longitud del fragmento
        min_length = max(25, max_length // 2) # Ajustar min_length basado en max_length

        if input_length > 5:
            start_time = time.time()
            try:
                # Call directly if it's BART or Pegasus
                if summarizer.__name__ in ['summarize_with_bart', 'summarize_with_pegasus', 'T5-Base', 'TextRank']:
                    summary, time_taken = summarizer(' '.join(chunks))
                else: # Other models that handle max_length and min_length
                    summary = summarizer(chunk, max_length=max_length, min_length=min_length, do_sample=False)
                # Verificar si el modelo devuelve una lista con el resumen
                if isinstance(summary, list) and len(summary) > 0 and 'summary_text' in summary[0]:
                    summaries.append(summary[0]['summary_text'])
                else:
                    print(f"El modelo no devolvió un resumen válido: {summary}")
                    summaries.append(chunk) # Añadir el fragmento original si hay un error
            except Exception as e:
                print(f"Error al resumir el fragmento: {e}")
                summaries.append(chunk) # Añadir el fragmento original si hay un error
            end_time = time.time()
            elapsed_time = end_time - start_time
            total_time += elapsed_time
        else:
            summaries.append(chunk) # Si el fragmento es muy corto, usarlo como está
    return ' '.join(summaries), total_time
```



```
# --- Modelos de Resumen ---
```

```
from transformers import PegasusTokenizer, PegasusForConditionalGeneration
```

```
pegasus_tokenizer = PegasusTokenizer.from_pretrained("google/pegasus-xsum")
```

```
pegasus_model = PegasusForConditionalGeneration.from_pretrained("google/pegasus-xsum")
```

```
def summarize_with_pegasus(text):
    start_time = time.time()
    inputs = pegasus_tokenizer(text, return_tensors="pt", max_length=1024, truncation=True)
    summary_ids = pegasus_model.generate(inputs['input_ids'], max_length=150, min_length=30, num_beams=4, early_stopping=True)
    summary = pegasus_tokenizer.decode(summary_ids[0], skip_special_tokens=True)
    end_time = time.time()
    elapsed_time = end_time - start_time
    return summary, elapsed_time
```

```
tokenizer_config.json: 100% 87.0/87.0 [00:00<00:00, 7.87kB/s]
spiece.model: 100% 1.91M/1.91M [00:00<00:00, 4.43MB/s]
special_tokens_map.json: 100% 65.0/65.0 [00:00<00:00, 5.70kB/s]
tokenizer.json: 100% 3.52M/3.52M [00:00<00:00, 23.1MB/s]
config.json: 100% 1.39k/1.39k [00:00<00:00, 127kB/s]
pytorch_model.bin: 100% 2.28G/2.28G [00:08<00:00, 151MB/s]
Some weights of PegasusForConditionalGeneration were not initialized from the mo
You should probably TRAIN this model on a down-stream task to be able to use it
generation_config.json: 100% 259/259 [00:00<00:00, 21.0kB/s]
```

```
from transformers import BartTokenizer, BartForConditionalGeneration
```

```
bart_tokenizer = BartTokenizer.from_pretrained("facebook/bart-large-cnn")
```

```
bart_model = BartForConditionalGeneration.from_pretrained("facebook/bart-large-cnn")
```

```
def summarize_with_bart(text):
    start_time = time.time()
    inputs = bart_tokenizer(text, return_tensors="pt", max_length=1024, truncation=True)
    summary_ids = bart_model.generate(inputs['input_ids'], max_length=150, min_length=30, num_beams=4, early_stopping=True)
    summary = bart_tokenizer.decode(summary_ids[0], skip_special_tokens=True)
    end_time = time.time()
    elapsed_time = end_time - start_time
    return summary, elapsed_time
```

```
vocab.json: 100% 899k/899k [00:00<00:00, 1.38MB/s]
merges.txt: 100% 456k/456k [00:00<00:00, 30.5MB/s]
tokenizer.json: 100% 1.36M/1.36M [00:00<00:00, 3.16MB/s]
config.json: 100% 1.58k/1.58k [00:00<00:00, 96.5kB/s]
model.safetensors: 100% 1.63G/1.63G [00:04<00:00, 412MB/s]
generation_config.json: 100% 363/363 [00:00<00:00, 31.8kB/s]
```

```
# Función para generar resumen con LSA con medición de tiempo
```

```
def summarize_with_lsa(text):
    start_time = time.time()
    sentences = text.split('. ')
    vectorizer = TfidfVectorizer()
    X = vectorizer.fit_transform(sentences)
    lsa = TruncatedSVD(n_components=1, n_iter=100)
    lsa.fit(X)
    scores = lsa.components_[0]

    ranked_indices = np.argsort(scores, axis=0)[: -1]
    ranked_sentences = [sentences[i] for i in ranked_indices if i < len(sentences)]

    summary = '. '.join(ranked_sentences[:5])
    end_time = time.time()
    elapsed_time = end_time - start_time
    return summary, elapsed_time
```

```

# Función para generar resumen con TF-IDF con medición de tiempo
from nltk.stem.snowball import SnowballStemmer
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
import math

def summarize_with_tfidf(text):
    start_time = time.time()

    sentences = sent_tokenize(text)

    # 1. Preprocesamiento del texto
    clean_words = text_preprocessing(sentences)

    # 2. Creación de la matriz TF
    tf_matrix = create_tf_matrix(sentences)

    # 3. Creación de la matriz IDF
    idf_matrix = create_idf_matrix(sentences)

    # 4. Calculo de la matriz TF-IDF
    tf_idf_matrix = create_tf_idf_matrix(tf_matrix, idf_matrix)

    # 5. Calculo de la puntuación de las frases
    sentence_value = create_sentence_score_table(tf_idf_matrix)

    # 6. Determinación del umbral
    threshold = find_average_score(sentence_value)

    # 7. Generación del resumen
    summary = generate_summary(sentences, sentence_value, threshold)

    end_time = time.time()
    elapsed_time = end_time - start_time

    return summary, elapsed_time

def text_preprocessing(sentences):
    """
    Preprocesamiento del texto para eliminar palabras innecesarias.
    """
    stop_words = set(stopwords.words('spanish')) # Ajustar para español
    ps = SnowballStemmer("spanish") # Define el stemmer aquí
    clean_words = []
    for sent in sentences:
        words = word_tokenize(sent)
        words = [ps.stem(word.lower()) for word in words if word.isalnum()] # Stemming en español
        clean_words += [word for word in words if word not in stop_words]
    return clean_words

def create_tf_matrix(sentences):
    """
    Crea una matriz de frecuencia de términos (TF).
     $TF(t) = (\text{Número de veces que el término } t \text{ aparece en un documento}) / (\text{Número total de términos en el documento})$ 
    """
    tf_matrix = {}
    for sentence in sentences:
        tf_table = {}
        words_count = len(word_tokenize(sentence))
        words = word_tokenize(sentence)
        word_freq = {}
        for word in words:
            word_freq[word] = (word_freq[word] + 1) if word in word_freq else 1
        for word, count in word_freq.items():
            tf_table[word] = count / words_count
        tf_matrix[sentence[:15]] = tf_table
    return tf_matrix

def create_idf_matrix(sentences):
    """
    Crea una matriz de frecuencia de documentos inversa (IDF).
     $IDF(t) = \log_e(\text{Número total de documentos} / \text{Número de documentos con el término } t \text{ en ellos})$ 
    """
    idf_matrix = {}
    documents_count = len(sentences)
    sentence_word_table = {}
    for sentence in sentences:

```

```

    words = word_tokenize(sentence)
    sentence_word_table[sentence[:15]] = words
word_in_docs = {}
for sent, words in sentence_word_table.items():
    for word in words:
        word_in_docs[word] = (word_in_docs[word] + 1) if word in word_in_docs else 1
for sent, words in sentence_word_table.items():
    idf_table = {}
    for word in words:
        idf_table[word] = math.log10(documents_count / float(word_in_docs[word]))
    idf_matrix[sent] = idf_table
return idf_matrix

def create_tf_idf_matrix(tf_matrix, idf_matrix):
    """
    Crea una matriz TF-IDF que es la multiplicación de tf * idf para cada palabra individual.
    """
    tf_idf_matrix = {}
    for (sent1, f_table1), (sent2, f_table2) in zip(tf_matrix.items(), idf_matrix.items()):
        tf_idf_table = {}
        for (word1, value1), (word2, value2) in zip(f_table1.items(), f_table2.items()):
            tf_idf_table[word1] = float(value1 * value2)
        tf_idf_matrix[sent1] = tf_idf_table
    return tf_idf_matrix

def create_sentence_score_table(tf_idf_matrix):
    """
    Determina la puntuación promedio de las palabras de la frase con su valor TF-IDF.
    """
    sentence_value = {}
    for sent, f_table in tf_idf_matrix.items():
        total_score_per_sentence = 0
        count_words_in_sentence = len(f_table)
        if count_words_in_sentence != 0:
            for word, score in f_table.items():
                total_score_per_sentence += score
            sentence_value[sent] = total_score_per_sentence / count_words_in_sentence
        else:
            sentence_value[sent] = 0
    return sentence_value

def find_average_score(sentence_value):
    """
    Calcula el valor promedio de una frase en la tabla de puntuación de frases.
    """
    sum = 0
    for val in sentence_value:
        sum += sentence_value[val]
    average = sum / len(sentence_value)
    return average

def generate_summary(sentences, sentence_value, threshold):
    """
    Genera un resumen extrayendo las frases con puntuaciones mayores que el valor umbral.
    """
    sentence_count = 0
    summary = ''
    for sentence in sentences:
        if sentence[:15] in sentence_value and sentence_value[sentence[:15]] >= threshold:
            summary += sentence + " "
            sentence_count += 1
    return summary

df['Cleaned Transcript'][0]

→ 'pregunta axel encanta volver polimizar demás piensan ustedes cómo podría crear
empleo corto plazo besos abrazos marco enrique uminami presentó ah mierda metió
cruce dejó pregunta instalada mierda error error junto cada vez habla strending
topics saca noticia todas partes acá récords programas junto destacado abogado
economista axel kayser cómo axel buenas noches bienvenido bien contento vuelta
acá onzalo invitación bueno marco metió economía exactamente quería hablar nro

```

--- Evaluación de Modelos ---

```

modelos = {
    'T5-Base': pipeline("summarization", model="t5-base"),

```

```

'LSA': summarize_with_lsa,
'TF-IDF': summarize_with_tfidf,
'Textrank': pipeline("summarization", model="sshleifer/distilbart-cnn-12-6"),
'Pegasus': summarize_with_pegasus
}

# Función para calcular y comparar ROUGE
def evaluar_modelos(df):
    resultados = []
    rouge = Rouge()

    # Ajustar el límite de recursión
    import sys
    sys.setrecursionlimit(2000)

    # Asegurarse de que los textos no sean demasiado largos
    max_evaluation_length = 512

    # Validar el orden de los datos
    #assert df['Video'].tolist() == sorted(df['Video'].tolist()), "El orden de los videos en el DataFrame no es correcto."
    #assert df['Transcript'].tolist() == sorted(df['Transcript'].tolist()), "El orden de las transcripciones en el DataFrame no e
    #assert df['Reference Summary'].tolist() == sorted(df['Reference Summary'].tolist()), "El orden de los resúmenes de referenci

    # Función para evaluar ROUGE de manera segura
    def safe_rouge_score(summary, reference, rouge):
        if not summary.strip():
            return {'rouge-1': {'f': 0.0}, 'rouge-2': {'f': 0.0}, 'rouge-l': {'f': 0.0}}
        return rouge.get_scores(summary[:max_evaluation_length], reference[:max_evaluation_length])[0]

    for i in range(len(df)):
        # Obtener la transcripcion y resumen de referencia
        transcript = df['Transcript'][i]
        reference_summary = df['Reference Summary'][i]

        # Dividir el texto en fragmentos
        chunks = split_text(transcript)

        # Evaluar cada modelo
        for model_name, model_func in modelos.items():
            print(f"Ejecutando modelo: {model_name}") # Imprime el modelo actual
            print(f"Procesando video: {df['Video'][i]}") # Imprime el video actual

            # Si es TF-IDF, utilizar la transcripcion limpia
            if model_name == 'TF-IDF':
                cleaned_transcript = df['Cleaned Transcript'][i]
                cleaned_reference_summary = df['Cleaned Reference Summary'][i]
                summary, time_taken = model_func(cleaned_transcript)
                scores = safe_rouge_score(summary, cleaned_reference_summary, rouge)
            # Para los demás modelos, utilizar la transcripcion original
            else:
                if model_name in ['DistilBART-CNN', 'T5-Base', 'Textrank', 'BART', 'Pegasus']:
                    try:
                        summary, time_taken = summarize_text_chunks(chunks.copy(), model_func)
                    except Exception as e:
                        print(f"Error al resumir el fragmento: {e}")
                        summary = "" # Asignar un resumen vacío si hay error
                        time_taken = 0 # Asignar tiempo 0 si hay error
                else:
                    summary, time_taken = model_func(transcript)
                scores = safe_rouge_score(summary, reference_summary, rouge)

            # Agregar los resultados a la lista
            resultados.append({
                'Video': df['Video'][i],
                'Modelo': model_name,
                'rouge1': scores['rouge-1']['f'],
                'rouge2': scores['rouge-2']['f'],
                'rougeL': scores['rouge-l']['f'],
                'Tiempo': time_taken,
                'Resumen Generado': summary, # Agregar el resumen generado
                'Transcripción': transcript, # Agregar la transcripcion
                'Resumen Referente': reference_summary # Agregar el resumen referente
            })

    # Mostrar información del primer video y preguntar si continuar
    if i == 0 and model_name == 'Pegasus':
        print(f"Transcripción: {transcript}")
        print(f"Resumen Referente: {reference_summary}")

```

```
print(f'Resumen Referente: {reference_summary}')
print(f'Resumen Generado: {summary}')
print(f'Puntajes ROUGE: {scores}')
continuar = input("¿Desea continuar con la evaluación? (s/n): ")
if continuar.lower() != 's':
    # Crear un DataFrame con los resultados parciales
    resultados_df = pd.DataFrame(resultados)

    # Guardar resultados_df en un archivo .csv
    resultados_df.to_csv('resultados_resumen_parcial.csv', index=False)

    return # Devuelve los resultados hasta ahora

# Crear un DataFrame con los resultados
resultados_df = pd.DataFrame(resultados)

# Calcular el promedio de ROUGE por modelo
promedio_rouge = resultados_df.groupby('Modelo')[['rouge1', 'rouge2', 'rougeL']].mean()

# Mostrar el promedio de ROUGE por modelo
print("Promedio de ROUGE por modelo:")
print(promedio_rouge)

# Guardar resultados_df en un archivo .csv
resultados_df.to_csv('resultados_resumen.csv', index=False)
return resultados_df

# Llamar a la función evaluar_modelos y mostrar los resultados
resultados_df = evaluar_modelos(df)
print(resultados_df)
```



config.json: 100%	1.21k/1.21k [00:00<00:00, 115kB/s]
model.safetensors: 100%	892M/892M [00:01<00:00, 404MB/s]
generation_config.json: 100%	147/147 [00:00<00:00, 11.9kB/s]
spiece.model: 100%	792k/792k [00:00<00:00, 28.1MB/s]
tokenizer.json: 100%	1.39M/1.39M [00:00<00:00, 6.45MB/s]
config.json: 100%	1.80k/1.80k [00:00<00:00, 145kB/s]
pytorch_model.bin: 100%	1.22G/1.22G [00:03<00:00, 372MB/s]
tokenizer_config.json: 100%	26.0/26.0 [00:00<00:00, 2.08kB/s]
vocab.json: 100%	899k/899k [00:00<00:00, 30.3MB/s]
merges.txt: 100%	456k/456k [00:00<00:00, 30.6MB/s]

Ejecutando modelo: T5-Base

Procesando video: AXEL KAISER y BRUTAL PARTICIPACIÓN en su PRIMER CAPÍTULO.mp4

[illegible]

Ejecutando modelo: LSA

Procesando video: AXEL KAISER y BRUTAL PARTICIPACIÓN en su PRIMER CAPÍTULO.mp4

Ejecutando modelo: TF-IDF

Procesando video: AXEL KAISER y BRUTAL PARTICIPACIÓN en su PRIMER CAPÍTULO.mp4

Ejecutando modelo: Textrank

Procesando video: AXEL KAISER y BRUTAL PARTICIPACIÓN en su PRIMER CAPÍTULO.mp4

[illegible]

Ejecutando modelo: Pegasus

Procesando video: AXEL KAISER y BRUTAL PARTICIPACIÓN en su PRIMER CAPÍTULO.mp4

[illegible]

Resumen Referente: El programa de debate "Strending Topics" presenta una discusión acalorada sobre la situación actual de Ch

Economía: Axel Kaiser alerta sobre la situación fiscal crítica de Chile, con una deuda pública cercana al 60% del PIB. Crítica Seguridad: Pancho Javier acusa al gobierno de tener una agenda que permite la delincuencia, señalando la liberación de preso Política: El debate se centra en las acciones del gobierno y su supuesta falta de agenda. Axel Kaiser critica la gestión del Giorgio Jackson: Se discute la demanda de Giorgio Jackson contra 23 diputados de la UDI por difamación. Axel Kaiser lo califica Guachipato: Se menciona la crisis en Guachipato y el posible impacto en la economía, con 30.000 personas que podrían perder

El debate refleja la polarización política en Chile, con diferentes visiones sobre la gestión del gobierno, la economía, la
 Resumen Generado: Pregunta también, sobre todo para Axel, que me encanta volver a polimizar con él y para los demás también.
 Puntajes ROUGE: {'rouge-1': {'r': 0.1774193548387097, 'p': 0.14666666666666667, 'f': 0.16058393665086063}, 'rouge-2': {'r':

¿Desea continuar con la evaluación? (s/n): s

Ejecutando modelo: T5-Base

Procesando video: Axel Kaiser y Javier Milei - El renacimiento liberal.mp4

```
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
```

Ejecutando modelo: LSA

Procesando video: Axel Kaiser y Javier Milei - El renacimiento liberal.mp4

Ejecutando modelo: TF-IDF

Procesando video: Axel Kaiser y Javier Milei - El renacimiento liberal.mp4

Ejecutando modelo: Textrank

Procesando video: Axel Kaiser y Javier Milei - El renacimiento liberal.mp4

```
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
```

Ejecutando modelo: Pegasus

Procesando video: Axel Kaiser y Javier Milei - El renacimiento liberal.mp4

```
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
```

Ejecutando modelo: T5-Base

Procesando video: Cómo sobrevivir a la destrucción de la industria de la tecnología.mp4

```
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
```

https://colab.research.google.com/drive/1pgVtvzQaloCLu4HD7rQTMsyO9QgFry6p#scrollTo=6Z3mX0_-U56K&printMode=true


```
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Ejecutando modelo: Pegasus
Procesando video: GPT4o – La Gran Apuesta de OpenAI por la MULTIMODALIDAD.mp4
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Ejecutando modelo: T5-Base
Procesando video: Google, Facebook, Amazon – El poder ilimitado de los consorcios digitales _ DW Documental.mp4
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Ejecutando modelo: LSA
Procesando video: Google, Facebook, Amazon – El poder ilimitado de los consorcios digitales _ DW Documental.mp4
Ejecutando modelo: TF-IDF
Procesando video: Google, Facebook, Amazon – El poder ilimitado de los consorcios digitales _ DW Documental.mp4
Ejecutando modelo: Textrank
Procesando video: Google, Facebook, Amazon – El poder ilimitado de los consorcios digitales _ DW Documental.mp4
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Ejecutando modelo: Pegasus
Procesando video: Google, Facebook, Amazon – El poder ilimitado de los consorcios digitales _ DW Documental.mp4
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Ejecutando modelo: T5-Base
Procesando video: LO QUE DEBEN SABER HOY LOS AFILIADOS de la EPS SURA.mp4
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Ejecutando modelo: LSA
Procesando video: LO QUE DEBEN SABER HOY LOS AFILIADOS de la EPS SURA.mp4
Ejecutando modelo: TF-IDF
Procesando video: LO QUE DEBEN SABER HOY LOS AFILIADOS de la EPS SURA.mp4
Ejecutando modelo: Textrank
Procesando video: LO QUE DEBEN SABER HOY LOS AFILIADOS de la EPS SURA.mp4
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
```

https://colab.research.google.com/drive/1pgVtvzQaloCLu4HD7rQTMsyO9QgFry6p#scrollTo=6Z3mX0_-U56K&printMode=true

[illegible]

```
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Ejecutando modelo: T5-Base
Procesando video: Pocos Entienden Esto de la Física Moderna....mp4
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Ejecutando modelo: LSA
Procesando video: Pocos Entienden Esto de la Física Moderna....mp4
Ejecutando modelo: TF-IDF
Procesando video: Pocos Entienden Esto de la Física Moderna....mp4
Ejecutando modelo: Textrank
Procesando video: Pocos Entienden Esto de la Física Moderna....mp4
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
Ejecutando modelo: Pegasus
Procesando video: Pocos Entienden Esto de la Física Moderna....mp4
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Error al resumir el fragmento: index out of range in self
Ejecutando modelo: T5-Base
Procesando video: Por los sueños se suspira, por las metas se trabaja-Humberto Ramos _ TED.mp4
Error al resumir el fragmento: 'SummarizationPipeline' object has no attribute '__name__'
```