Katherine Michelle Ortiz Burgos
Código: 506221041
Juan Esteban Acosta Rodriguez
Código: 506221068
Profesor: Oscar Alexander Mendez Aguirre
Fundación Universitaria Konrad Lorenz

Prototipo Análisis De Textos

I. Resumen

Este prototipo consiste en la creación de un Servicio de Análisis de Sentimientos para Textos Largos tiene como objetivo desarrollar un sistema que evalúe sentimientos en textos de más de 1000 tokens, con una interfaz gráfica y la capacidad de analizar comentarios en redes sociales. El prototipo se divide en la implementación de modelos de procesamiento de lenguaje natural (NLP) para el análisis de sentimientos. Las herramientas y tecnologías utilizadas incluyen Python con FastAPI o Django para el backend, HTML, CSS y JavaScript para el frontend, y Transformers de Hugging Face para el NLP. El prototipo se estructura en cinco sesiones que abarcan desde la introducción al NLP hasta los ajustes finales y la implementación de filtros de publicación.

II. Introducción

El prototipo de análisis de sentimientos para textos largos que hemos desarrollado integra de manera eficiente el frontend y el backend para ofrecer una experiencia completa a través de una interfaz web. El frontend de la aplicación se encarga de la parte visible para el usuario en su navegador web. Esto incluye la interfaz gráfica donde los usuarios pueden ingresar textos largos (de más de 1000 tokens) y visualizar los resultados del análisis de sentimientos. Además, ofrece controles interactivos, como botones para enviar los textos para análisis y opciones para ver los resultados detallados de los sentimientos detectados.

Por otro lado, el backend de la aplicación maneja la lógica detrás del análisis de sentimientos. Utiliza modelos avanzados de procesamiento de lenguaje natural (NLP) proporcionados por los Transformers de Hugging Face para analizar el sentimiento en textos largos. El backend también se encarga de procesar los datos recibidos y enviar los resultados al frontend para su visualización.

Esta colaboración entre el frontend y el backend permite que la aplicación funcione de manera fluida y eficiente. Mientras que el frontend proporciona una interfaz amigable para que los usuarios interactúen con el sistema, el backend realiza el procesamiento intensivo de datos y el análisis de sentimientos, garantizando una experiencia de usuario óptima y resultados precisos.

Figura 1. Google Colab.

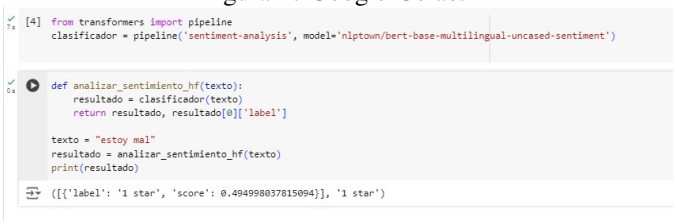

Figura 2. Google Colab.

III. Diseño y desarrollo de la API backend

El análisis de sentimientos se ha convertido en una herramienta crucial para entender las opiniones y emociones expresadas en el texto. Nuestra aplicación aprovecha las capacidades avanzadas del procesamiento de lenguaje natural (NLP) para proporcionar un análisis preciso de los sentimientos en textos ingresados por el usuario y comentarios en redes sociales. El backend de esta aplicación, desarrollado con FastAPI, se encarga de manejar las solicitudes, realizar el análisis de sentimientos utilizando modelos de Hugging Face y NLTK, y extraer datos de Twitter para su análisis.

1. Configuración Del Entorno: El backend de nuestra aplicación está construido con FastAPI, un framework de Python reconocido por su alto rendimiento y eficiencia en la creación de APIs web. FastAPI facilita la creación de endpoints que pueden manejar grandes volúmenes de solicitudes de manera concurrente, lo que es fundamental para el análisis en tiempo real.
2. Procesamiento De Lenguaje Natural : Para realizar el análisis de sentimientos, utilizamos el modelo nlptown/bert-base-multilingual-uncased-sentiment de Hugging Face. Este modelo está entrenado para clasificar sentimientos en múltiples idiomas, permitiendo una amplia versatilidad en el análisis. Además, empleamos NLTK (Natural Language Toolkit) para la tokenización de oraciones, lo que nos permite dividir textos largos en oraciones individuales para un análisis más preciso.

3. Integración Con Redes Sociales: La aplicación también se integra con Twitter mediante la biblioteca Tweepy. Esto permite al backend extraer tweets y sus comentarios en tiempo real para analizar los sentimientos expresados en ellos. Esta funcionalidad es especialmente útil para monitorear el sentimiento público sobre eventos o temas específicos.
4. Archivo Principal : El archivo main.py contiene la configuración y la lógica principal del backend. Esto incluye la configuración de FastAPI, las funciones de análisis de sentimientos, la integración con Twitter, y la configuración de rutas para servir la interfaz gráfica y los archivos estáticos. Se importan las clases y funciones necesarias de FastAPI para configurar la aplicación. En nuestro backend podemos encontrar algunos pasos que llevamos a cabo:
   - Se crea una instancia de FastAPI() llamada `app`.
   - Se monta el directorio `static` para servir archivos estáticos (como CSS, JavaScript, imágenes) en la ruta `/static`.
   - Se configuran las plantillas Jinja2 almacenadas en el directorio `templates`.
   - Se configura el logging para registrar eventos y errores.
   - Se define una ruta en la raíz del servidor (`/`) que devuelve una respuesta HTML utilizando una plantilla llamada `index.html`.
   - Se define una ruta `/sentimientos/{texto}` que recibe texto como parámetro y realiza el análisis de sentimientos sobre él.
   - El texto se divide en oraciones utilizando la función `sent_tokenize` de NLTK.
   - Se utilizan las funciones `analizar_sentimientos_oraciones` y `combinar_resultados` para realizar el análisis de sentimientos y obtener un promedio de los resultados.
   - Se mapean los valores del promedio de sentimientos a etiquetas más comprensibles y se devuelve el resultado.
   - Este archivo configura el servidor, define las rutas para la API y se encarga de manejar las solicitudes entrantes. La función `read_root` sirve la interfaz web principal, mientras que `analizar_sentimiento_hf` realiza el análisis de sentimientos sobre el texto proporcionado.

IV. Creación de la interfaz gráfica y conexión con el backend

La creación de una interfaz gráfica intuitiva y su conexión efectiva con el backend son elementos cruciales para garantizar una experiencia de usuario fluida en nuestra aplicación de análisis de sentimientos. En esta sección, abordaremos cómo diseñamos la interfaz gráfica utilizando HTML, CSS y JavaScript, así como cómo establecemos la conexión con el backend utilizando FastAPI y JavaScript para realizar solicitudes asíncronas.

Estructura HTML

1. Se define un documento HTML básico con un título y un enlace al archivo CSS.
2. Dentro del cuerpo del documento, se crea un contenedor (div.container) que alberga el título, un área de texto para ingresar el texto a analizar, un botón para iniciar el análisis y un elemento ¡p¿ para mostrar los resultados.
3. Se incluye un script al final del cuerpo del documento que carga el archivo JavaScript para manejar la lógica de la aplicación.

Estilos CSS

1. Estilos para el cuerpo del documento (body):El cuerpo del documento se estiliza para establecer una apariencia atractiva para toda la página web. Utilizando propiedades como font-family, display, justify-content y align-items, se logra centrar vertical y horizontalmente el contenido, proporcionando una disposición ordenada y equilibrada. Además, la declaración background se utiliza para incrustar una imagen de fondo, agregando profundidad visual y estilo al conjunto de la página.
2. Estilos para el contenedor principal (div.container):El contenedor principal de la página, definido por la clase .container, recibe estilos específicos para resaltar su importancia y estructura. Los estilos incluyen propiedades como text-align, que centraliza horizontalmente el contenido de texto dentro del contenedor, y background-color, que establece un color de fondo distintivo. Además, se aplican relleno, bordes redondeados y una sombra sutil para mejorar la estética y la legibilidad del contenido.
3. Estilos para el área de texto (textarea):El área de texto, representada por el elemento textarea, se estiliza para proporcionar una experiencia de entrada de texto cómoda y estéticamente agradable. Se establece un ancho y una altura adecuados para que el área de texto se ajuste bien dentro del contenedor principal, y se aplican bordes y relleno para definir claramente su límite y mejorar su aspecto visual.
4. Estilos para el botón (button):El botón de acción, definido por el elemento button, recibe estilos que lo hacen destacar visualmente y fácil de usar. Se establece un color de fondo llamativo y un color de texto contrastante para mejorar la visibilidad. Además, se aplica relleno para que el botón tenga un tamaño adecuado y sea fácil de hacer clic.
5. El área donde se muestran los resultados del análisis de sentimientos se estiliza para que sea claramente distinguible y legible. Se agrega espacio en la parte superior para separar visualmente los resultados del área de entrada de texto, y se establece un tamaño de fuente adecuado para mejorar la legibilidad. Estos estilos garantizan que los resultados sean fácilmente accesibles y comprensibles para los usuarios, proporcionando una experiencia de usuario satisfactoria.

Script.js

1. Se define una función analizarSentimientos() que se activa al hacer clic en el botón de análisis.
2. Se obtiene el texto ingresado por el usuario del elemento de área de texto.
3. Se valida si se ingresó un texto antes de realizar la solicitud al backend.
4. Se realiza una solicitud asíncrona (fetch()) al endpoint de análisis de sentimientos del backend utilizando el texto ingresado.
5. Se muestra el resultado devuelto por el backend en el elemento ¡p¿ designado para mostrar los resultados.
6. Se manejan los errores que puedan ocurrir durante el proceso de análisis de sentimientos.

Figura 3. Prototipo Análisis De Texto.

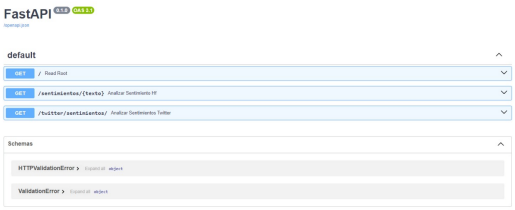

Figura 4. FastApi.

V. Conclusiones

1. El prototipo demuestra la sinergia entre diversas disciplinas tecnológicas, desde el procesamiento de lenguaje natural (NLP) hasta el desarrollo web y la integración de APIs de redes sociales. Esta combinación permite la creación de una aplicación completa y versátil que aprovecha lo mejor de cada campo para ofrecer una solución integral a la detección de sentimientos en textos largos.
2. Tanto en el diseño de la interfaz gráfica como en la implementación del backend, se evidencia un enfoque centrado en el usuario. La interfaz se presenta de manera intuitiva y amigable, facilitando la interacción del usuario con la aplicación. Además, la integración con redes sociales como Twitter muestra una comprensión profunda de las necesidades del usuario y la importancia de analizar comentarios en tiempo real para una retroalimentación efectiva.
3. En el prototipo se hace uso de tecnologías de vanguardia como FastAPI, Hugging Face Transformers y Tweepy, lo que demuestra un compromiso con la excelencia técnica y la capacidad de adaptarse a las últimas tendencias en desarrollo de software. Esta elección de herramientas no solo garantiza un rendimiento óptimo y escalable, sino que también permite futuras expansiones y mejoras en función de las necesidades cambiantes del proyecto y del usuario.

VI. Referencias

1. Google Colab. (s/f). Google.com. Recuperado el 26 de mayo de 2024, de https://colab.research.google.com/drive/1MIoprqtyD8XjFyaa_rjHZZ8xW6RRN4yp?hl=es