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

Prototipo Detector De Emociones

I. Resumen

Este prototipo consiste en la detección de emociones en imágenes mediante la creación de un microservicio API con FastAPI y una aplicación web. El microservicio, implementado en FastAPI, recibe imágenes y utiliza un modelo de inteligencia artificial entrenado para predecir emociones, devolviendo los resultados de manera eficiente. La aplicación web, diseñada con HTML, CSS y JavaScript para la interfaz y jQuery para la interacción, permite a los usuarios cargar imágenes y visualizar las emociones detectadas.

II. Introducción

En este prototipo, la finalidad es proporcionar a los usuarios una herramienta fácil de usar que permita analizar las emociones presentes en imágenes cargadas, utilizando para ello un modelo de inteligencia artificial previamente entrenado. Este prototipo se divide en dos componentes principales: el backend, que gestiona la lógica del procesamiento de imágenes y predicción de emociones, y el frontend, que ofrece una interfaz intuitiva y atractiva para la interacción del usuario. La combinación de estos dos componentes proporciona una experiencia de usuario fluida y eficiente, facilitando la carga de imágenes y la visualización de resultados en tiempo real.

En el lado del servidor, se emplea FastAPI, un marco web rápido para Python, para manejar las solicitudes HTTP. Utilizando TensorFlow y TensorFlow Hub, se carga un modelo de aprendizaje profundo previamente entrenado, capaz de reconocer emociones en imágenes. Cuando se recibe una imagen a través de una solicitud POST, el backend procesa la imagen y devuelve la emoción detectada como respuesta.

Por otro lado, en el lado del cliente, el frontend proporciona una interfaz intuitiva para que los usuarios carguen imágenes. Esto se logra mediante HTML y CSS para estructurar y diseñar la página web, y JavaScript con jQuery para manejar la interacción del usuario. Los usuarios pueden seleccionar una imagen, enviarla al servidor y recibir la emoción detectada en respuesta.

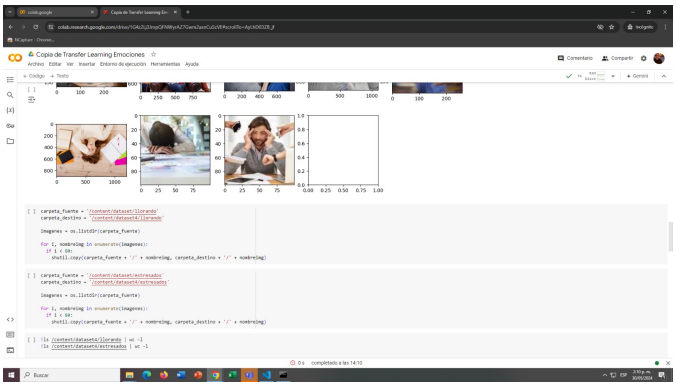

Figura 1. Colab.

III. Diseño y desarrollo de la API backend

El desarrollo de una API Backend son fundamentales para la creación de aplicaciones escalables. La API Backend sirve como un puente entre el frontend de la aplicación, la parte con la que interactúan los usuarios, y el backend, la parte encargada de procesar los datos y la lógica de la aplicación. En este contexto, el backend desempeña un papel crucial al proporcionar los servicios y la funcionalidad necesarios para que la aplicación funcione de manera eficiente y segura.

Estructura main.py

1. Se importan los módulos y clases necesarios de FastAPI y otras bibliotecas como TensorFlow y PIL.
2. ISe configuran las rutas de los archivos estáticos y las plantillas para servir archivos HTML y recursos estáticos como imágenes y estilos.
3. Se configura el middleware CORS para permitir el acceso a la API desde un origen específico (en este caso, la aplicación frontend que se ejecuta en http://127.0.0.1:5501).
4. Se define una ruta para manejar las solicitudes GET en la raíz de la aplicación.
5. Cuando se recibe una solicitud GET en esta ruta, se devuelve la plantilla HTML index.html.
6. Se define una ruta para manejar las solicitudes GET en la raíz de la aplicación.
7. Cuando se recibe una solicitud GET en esta ruta, se devuelve la plantilla HTML index.html.

Estructura script.js

1. Se ejecuta cuando el DOM (Document Object Model) está completamente cargado y listo para ser manipulado. Es el punto de entrada principal para el código JavaScript en la página.

2. Se agrega un evento de cambio al elemento de entrada de tipo archivo (<input type="file" >) con el ID image. Cuando el usuario selecciona una imagen, este evento se activa y ejecuta la función asociada.
3. Se hace uso de FileReader para leer los datos de la imagen seleccionada por el usuario. Cuando se completa la lectura de la imagen, la función onload se activa y ejecuta el código dentro de ella. En este caso, se muestra la imagen seleccionada al usuario.
4. Hay que agregar un evento de envío al formulario con el ID uploadForm. Cuando el usuario envía el formulario (por ejemplo, al hacer clic en el botón de enviar), este evento se activa y ejecuta la función asociada.
5. La imagen se envía como datos del formulario (FormData). Se configura contentType y processData como false para asegurar que la imagen se envíe correctamente como un archivo binario.
6. La función success maneja la respuesta exitosa del servidor, y la función error maneja cualquier error que pueda ocurrir durante la solicitud AJAX.

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

El código del frontend es la parte de una aplicación web que los usuarios interactúan directamente en sus navegadores. Está compuesto por HTML, CSS y JavaScript, que trabajan juntos para crear la interfaz de usuario y brindar una experiencia interactiva y atractiva.

Estructura index.html

El archivo index.html sirve como el punto de entrada de la aplicación web, presentando la estructura y el contenido que los usuarios interactúan en sus navegadores. Desde el título de la página hasta los elementos de formulario y áreas de visualización de resultados, cada componente juega un papel crucial en la experiencia del usuario. En él, se enlazan hojas de estilo CSS para definir el aspecto visual, incluyendo la popular biblioteca Bootstrap para un diseño responsivo. Los elementos de formulario permiten a los usuarios seleccionar y enviar imágenes al backend para su procesamiento. El uso de scripts JavaScript, incluyendo la biblioteca jQuery y código personalizado en script.js, facilita la interactividad y la comunicación con el servidor, completando la experiencia del usuario en la aplicación web..

Estilos CSS

El archivo styles.css contiene reglas de estilo que definen la apariencia visual de la página web. En él, se establecen propiedades como el tamaño, el color y el espaciado de los elementos HTML para crear un diseño coherente y atractivo. Por ejemplo, las reglas aplicadas al contenedor principal container limitan su ancho y lo centran en la página, mientras que las reglas para el botón de envío btn-primary definen su apariencia y comportamiento al interactuar con él. Además, se proporcionan estilos específicos para el formulario, como el espaciado entre los elementos del formulario, y para el contenedor de resultados, asegurando una presentación visualmente agradable y funcional para los usuarios. Estos estilos son esenciales para mejorar la usabilidad y la estética de la página web, proporcionando una experiencia de usuario mejorada y coherente.

En el prototipo proporcionado, la conexión entre el backend y el frontend se establece a través de solicitudes HTTP. Cuando un usuario interactúa con la interfaz de usuario en el frontend, como seleccionar una imagen y enviarla al servidor, se activa un evento en JavaScript que desencadena una solicitud AJAX al backend. Esta solicitud se procesa en el servidor, donde se lleva a cabo la lógica de negocio, como la detección de emociones en la imagen. Una vez completado el procesamiento, el backend devuelve una respuesta al frontend, que puede contener resultados o mensajes de estado. Esta respuesta se maneja en el frontend mediante una función JavaScript, que actualiza dinámicamente la interfaz de usuario para mostrar los resultados al usuario.

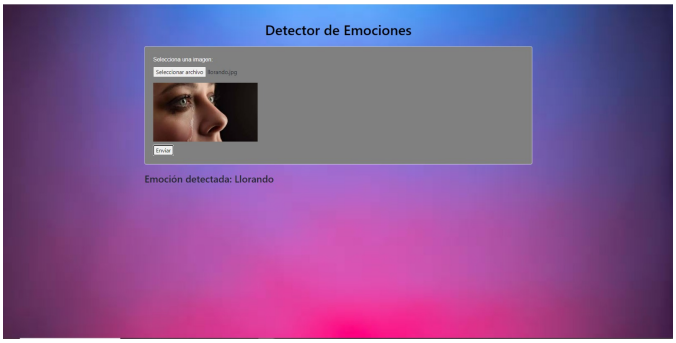

Figura 2. Prototipo Análisis De Imagénes.

V. Conclusiones

1. El uso de AJAX en JavaScript permite una experiencia de usuario más interactiva al permitir que el frontend envíe y reciba datos del backend sin necesidad de recargar la página completa. Esto mejora la experiencia del usuario al proporcionar respuestas en tiempo real y una sensación de fluidez en la aplicación.
2. El prototipo demuestra un potencial significativo en la detección de emociones a través del análisis de imágenes. Al integrar tecnologías de inteligencia artificial o machine learning en el backend, la aplicación puede analizar imágenes enviadas por los usuarios y detectar emociones como tristeza, enojo. Esta capacidad puede ser útil en una variedad de aplicaciones, desde análisis de sentimientos en redes sociales hasta sistemas de atención al cliente basados en emociones.
3. La conexión entre el frontend y el backend permite una experiencia del usuario centrada en los resultados. Después de enviar una imagen al backend para su procesamiento, los usuarios reciben una respuesta en tiempo real que muestra los resultados del análisis de emociones. Esta experiencia directa y receptiva puede

aumentar la satisfacción del usuario al proporcionar una retroalimentación inmediata sobre la imagen enviada.

VI. Referencias

1. Google colab. (s/f). Google.com. Recuperado el 30 de mayo de 2024, de`https://colab.research.google.com/drive/1G4z2Lj3JmpQFNWyrAZ7Gwm2asnCuScVE`