Tecnológico de Monterrey

Speech2Summarization

María Fernanda Torres Alcubilla A01285041

23 de noviembre del 2023

Natural Language Processing

Profesor. Juan Arturo Nolazco Flores

Se realizó una interfaz para la carga de un archivo de audio .m4a, con el objetivo de transcribir este a texto y con esto, realizar un resumen en forma de puntos importantes. Para esto, se hizo uso de las siguientes librerías, junto con su utilidad:

- Streamlit. para realizar la interfaz conectar el modelo con el usuario.
- Whisper. para la transcripción del audio, con el modelo 'base'.
- Openai. a partir del texto generado con la librería anterior, se genera un resumen con el prompt: *"You are an office administer, summarize the text in key points"*

De manera general, el proceso que se realiza dentro de la interfaz es la siguiente:

1. Se sube un archivo .m4a para empezar el modelo, en el caso que no haya algún archivo se despliega un error.
2. Se da la opción de reproducir el audio para verificar el archivo o resolver dudas específicas.
3. Se empieza el proceso de transcripción y se notifica su finalización.
4. Al finalizar el paso anterior, se inicia el proceso de resumen.
5. Se despliegan en toggles la transcripción o resumen (según la selección)

A continuación se exponen imágenes del funcionamiento de la interfaz, así como el código utilizado:

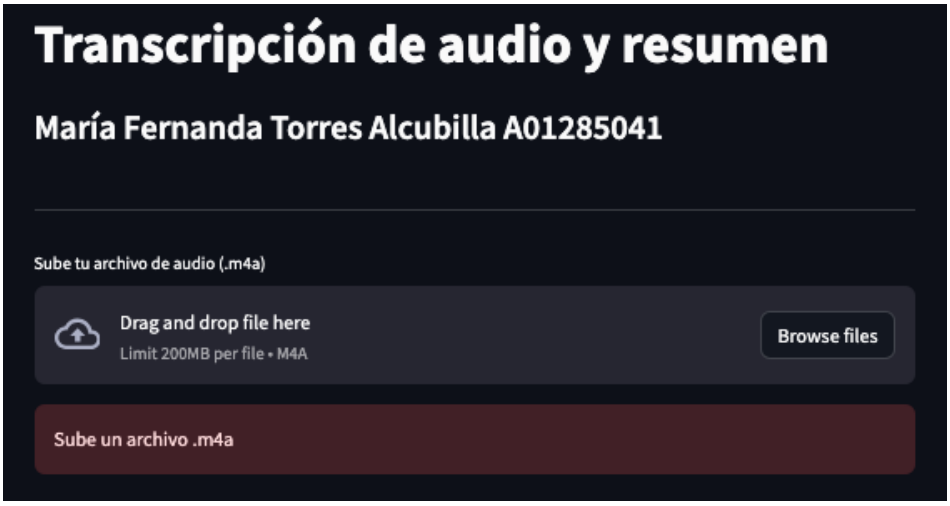

Imagen 1. Interfaz inicial sin carga de archivo (paso 1)

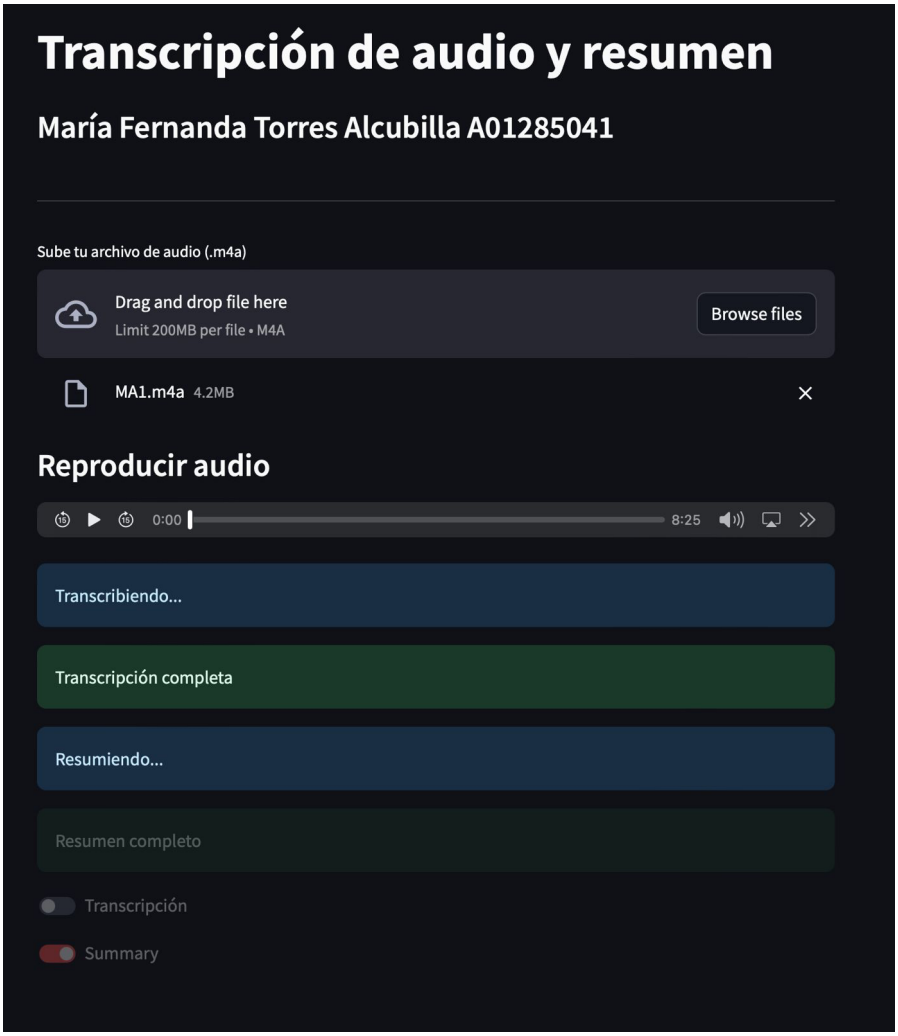

Imagen 2. Interfaz con archivo, opción de reproducción, transcripción realizada y en proceso de resumen (hasta paso 4)

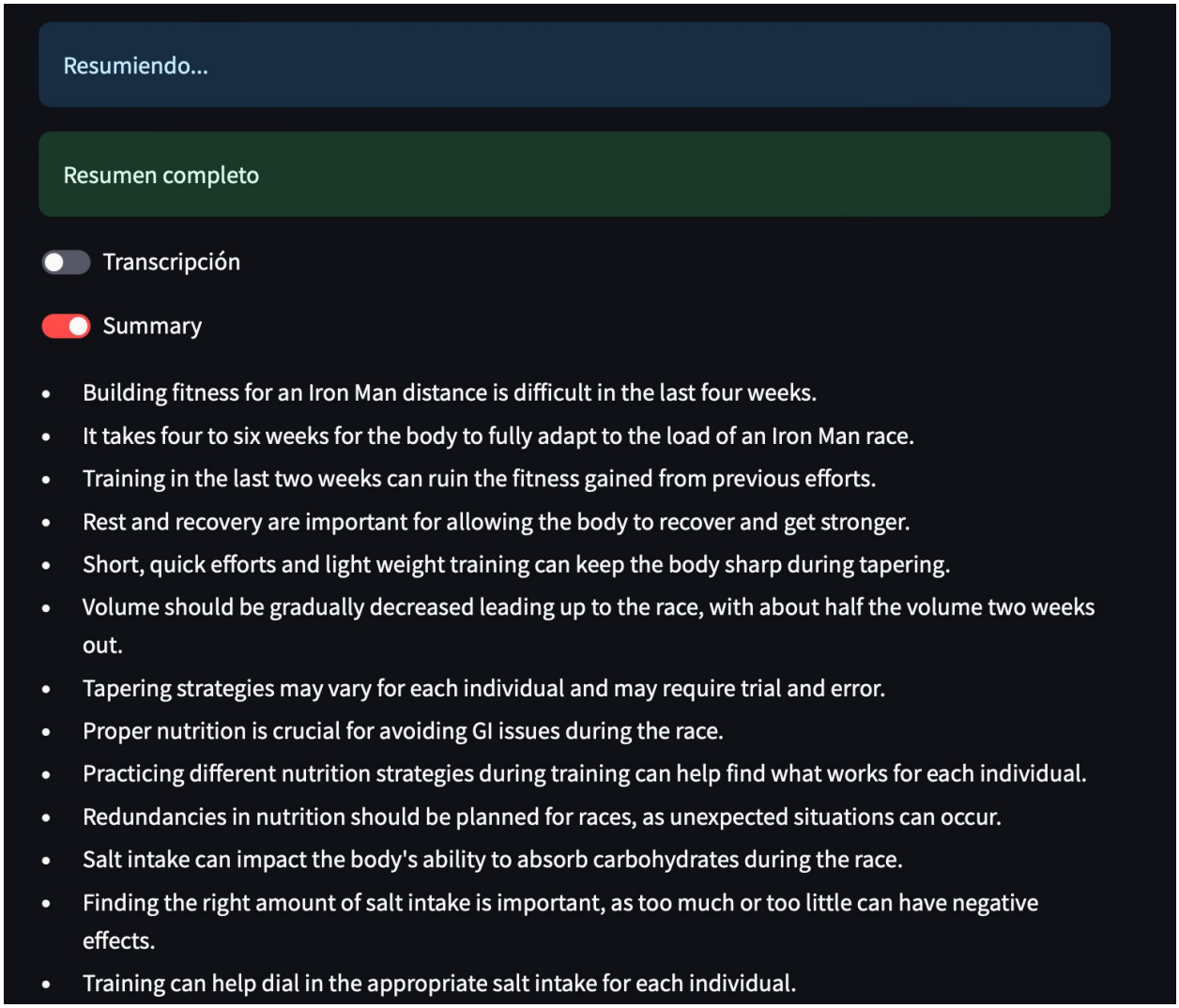

Imagen 3. Ejemplo de toggle de resumen del audio (paso 5)

```python
import streamlit as st

import openai
import whisper

st.title("Transcripción de audio y resumen")
st.subheader("María Fernanda Torres Alcubilla A01285041")
st.divider()

uploaded_file = st.file_uploader("Sube tu archivo de audio (.m4a)", type="m4a")

openai.api_key = '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'
model = whisper.load_model("base")


def transcribe_audio(model, file_path):
    transcript = model.transcribe(file_path)
    return transcript['text']


def CustomChatGPT(user_input):
    messages = [{"role": "system",
                 "content": "You are an office administer, summarize the text in key points"}]
    messages.append({"role": "user", "content": user_input})
    response = openai.ChatCompletion.create(
        model="gpt-3.5-turbo",
        messages=messages
    )
    ChatGPT_reply = response["choices"][0]["message"]["content"]
    return ChatGPT_reply


if uploaded_file is not None:
    st.subheader('Reproducir audio')
    st.audio(uploaded_file)

    st.info('Transcribiendo...')
    transcription = transcribe_audio(model, uploaded_file.name)
    st.success('Transcripción completa')

    st.info('Resumiendo...')
    summary = CustomChatGPT(transcription)
    st.success('Resumen completo')

    transc = st.toggle('Transcripción')
    if transc:
        st.markdown(transcription)

    summ = st.toggle('Summary')
    if summ:
        st.markdown(summary)

else:
    st.error('Sube un archivo .m4a')
```

Imagen 4. Código de la interfaz