Tecnológico de Monterrey

Escuela de Ingeniería y Ciencias

Inteligencia Artificial Avanzada para la Ciencia de Datos

Proyecto Final NLP: Herramienta de transcripción y resumen

03/12/2023

Profesor:

Dr. Juan Arturo Nolazco Flores

—-

Alumno:

Pablo Gabriel Galeana Benítez - A01735281 (Equipo 1)

1. Resumen

En el presente documento, se describirá el proceso por el cual se desarrollo una herramienta de transcripción y resumen de audio haciendo uso de modelos de procesamiento del lenguaje natural y de un framework de desarrollo web . De igual forma, se demostrará como funciona la interfaz grafica con el usuario.

1.1. Objetivo

El objetivo principal de este documento es aplicar los conocimientos y métodos aprendidos durante del modulo de Natural Language Processing, mientras se demuestran competencias obtenidas en el mismo que se enfocan principalmente en la correcta implementación de herramientas de NLP a través de APIs y el diseño de una interfaz web con el cual puedan ser utilizadas de forma simple y fácil las antes mencionadas.

1.2. Descripción

El proyecto se desarrollo utilizando el framework de desarrollo web llamado Streamlit en python. Streamlit provee herramientas para la fácil y rápida implementación de modelos de aprendizaje maquina e inteligencia artificial con el fin de agilizar la puesta a prueba de los mismos. Simultáneamente, se utilizaron dos modelos de procesamiento del lenguaje natural, el primero siendo Whisper-GPT de OpenAI el cual es un modelo de reconocimiento de voz y transcripción a texto y el segundo siendo GPT-3.5 un modelo de generación de texto de la misma empresa. Al primero de los modelos se accedió a través de la biblioteca diseñada por lo creadores del modelo y al segundo a través del API key que se puede generar comprando créditos en la pagina correspondiente.

El flujo es simple pero efectivo; primero se requiere que el usuario suba un archivo tipo M4a (audio) y posteriormente este es procesado con el modelo Whisper para su transcripción a texto. Una vez se ha transcrito, el usuario puede decidir si quiere la transcripción o el resumen del mismo. En caso de requerir el resumen, la transcripción es procesada con GPT-3.5 utilizando un prompt base para que genere un resumen en forma de bullet points. Debido a que los objetos de Streamlit tienen una respuesta booleana en cuanto a su uso, si el usuario quiere ahora procesar otro audio, simplemente debe de borrar el que existe actualmente.

La interfaz, además de contar con los objetos definidos por el framework, cuenta con el nombre de la herramienta y una amigable descripción de su uso, así como una advertencia del uso del contenido generado por la herramienta y la confidencialidad de los archivos subidos.

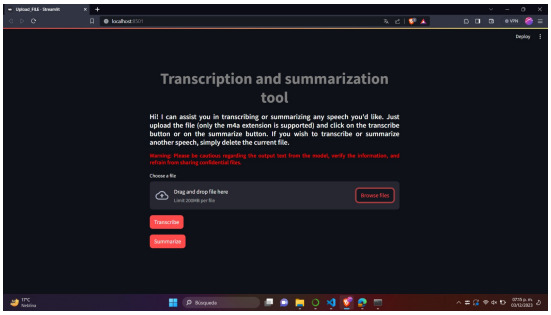
1.3. Comentarios

En general, el proyecto fue bastante agradable. No consideraría que fue demasiado demandante ya que todas las herramientas para su desarrollo fueron provistas durante el curso y además, se

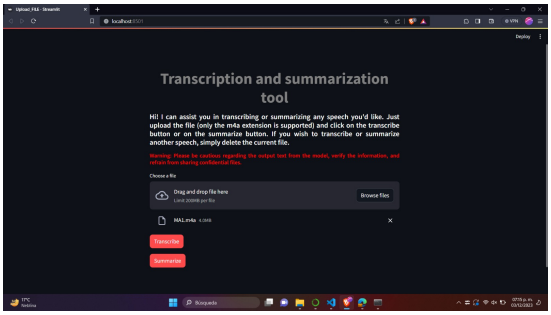
cubrieron los elementos más importantes del mismo de forma muy orgánica.

2. Demostración

A continuación se muestra como funciona la herramienta. Primero, como se ve la interfaz recién ha sido abierta y una vez que se ha cargado el archivo.



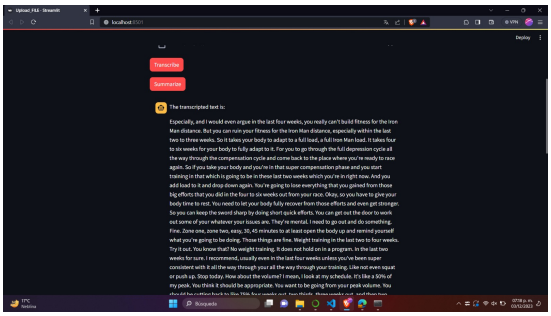
(a) Interfaz base



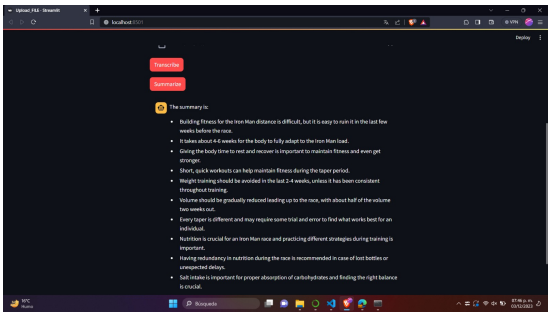
(b) Archivo cargado

Figura 1: Principio de la interfaz

Posteriormente, una vez que se ha seleccionado la opcion de resumir o transcribir.



(a) Transcripción.



(b) Resumen.

Figura 2: Transcripción y resumen

3. Código

Listing 1: Codigo desarrollado para la aplicación.

```
1 import streamlit as st
2 import openai
```

```
import whisper

#Definimos la API key de OpenAI y el modelo de Whisper (GPT-3)
openai.api_key = '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'
model = whisper.load_model("base")

#Funciones para transcribir el audio usando el modelo
def transcribe_audio(model, file_path):
    transcript = model.transcribe(file_path)
    return transcript['text']

#Funcion para resumir el texto a partir un base prompt y el input del usuario
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


#Titulo de la app
st.markdown("<h1 style='text-align: center; color: grey;'>Transcription and summarization tool</h1>", unsafe_allow_html=True)

#Descripcion de la app
st.markdown("<h5 style='text-align: justify; color: white;'>Hi! I can assist you in transcribing or summarizing any speech you'd like. Just upload the file (only the m4a extension is supported) and click  on the transcribe button or on the summarize button. If you wish to transcribe or summarize another speech, simply delete the current file.</h5>", unsafe_allow_html=True)

#Advertencia
st.markdown("<h6 style='text-align: justify; color: red;'>Warning: Please be cautious regarding the output text from the model, verify the information, and refrain from sharing confidential files.</h6>", unsafe_allow_html=True)

#Definimos el boton para subir el archivo
uploaded_file = st.file_uploader("Choose a file")

#Definimos el boton para resumir el archivo
button = st.button("Transcribe", type="primary")

button_2 = st.button("Summarize", type="primary")

#Si el usuario sube un archivo y presiona el boton, se ejecuta el codigo
```

```
if uploaded_file is not None and button:

    #Guardamos el archivo en el directorio
    with open("uploaded_file_name.m4a", "wb") as f:
        f.write(uploaded_file.read())

    #Definimos el path del archivo
    path =  'uploaded_file_name.m4a'


    #Transcribimos el audio y resumimos el texto
    transcription = transcribe_audio(model, path)
    summary = CustomChatGPT(transcription)

    #Mostramos el resumen
    with st.chat_message("assistant"):
        st.write('The transcripted text is: \n', '\n', transcription)



if uploaded_file is not None and button_2:
    #Guardamos el archivo en el directorio
    with open("uploaded_file_name.m4a", "wb") as f:
        f.write(uploaded_file.read())

    #Definimos el path del archivo
    path =  'uploaded_file_name.m4a'


    #Transcribimos el audio y resumimos el texto
    transcription = transcribe_audio(model, path)
    summary = CustomChatGPT(transcription)

    #Mostramos el resumen
    with st.chat_message("assistant"):
        st.write('The summary is: \n', '\n', summary)
```