*Вайсберг ЕА, ПИ21-1*

**Описание работы программы для транскрибирования аудио с помощью whisper и streamlit**

Для транскрибирования аудио будем использовать предобученную модель whisper, для создания приложения используем streamlit.

Для запуска приложения необходимо написать в терминале **streamlit run streamlit\_whisper.py**

Подключим необходимые библиотеки

*import* whisper

*import* librosa

*import* numpy *as* np

*import* io

*from* st\_audiorec *import* st\_audiorec

*from* bokeh.models.widgets *import* Button

*from* bokeh.models *import* CustomJS

*# import base64*

*# from streamlit.components.v1 import html*

*import* matplotlib.pyplot *as* plt

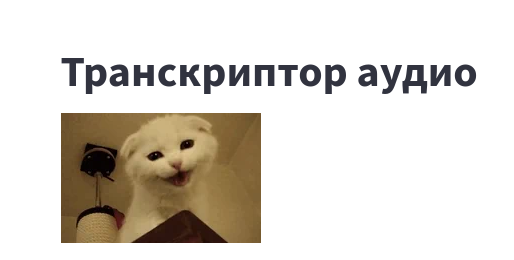
*from* PIL *import* Image

Установим название нашему приложению

st.title("Транскриптор аудио")

Я решила добавить милую гифку с мяукающим котенком, сделаем это с помощью

st.markdown("![Alt Text](https://media4.giphy.com/media/oGP14S1XBi4yk/200w.webp?cid=ecf05e474f01jb4mtj084vbrxms9z82oh00y0qwxpq88s5jd&ep=v1\_gifs\_related&rid=200w.webp&ct=g)")



Загрузим модель и сообщим пользователю, что модель загружена

model = whisper.load\_model("base")

st.success("Модель Whisper загружена")

Добавим кнопку загрузки аудио в нескольких форматах

*# upload audio file with streamlit*

audio\_file = st.file\_uploader("Загрузить файл", type=["wav", "mp3", "m4a"])

Добавим кнопку для записи аудио пользователем

*# online record*

wav\_audio\_data = st\_audiorec()

Также добавим поле для ввода текста и кнопку, чтобы машина прочитала этот текст. Установим событие на кнопку. Текст произносится на русском. При желании можно сменить язык произношения, поменяв u.lang = 'ru'.

text = st.text\_input("Введите текст")

tts\_button = Button(label="Произнести", width=100)

tts\_button.js\_on\_event("button\_click", CustomJS(code=*f*"""

var u = new SpeechSynthesisUtterance();

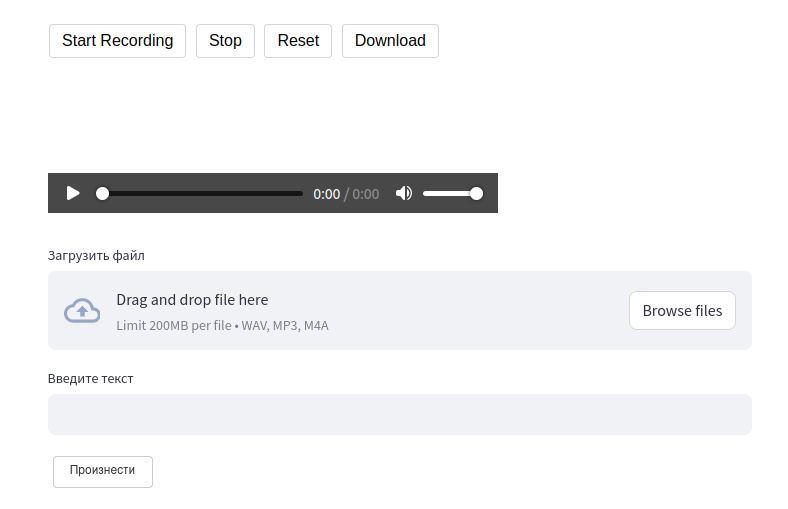
u.text = "{text}";

u.lang = 'ru';

speechSynthesis.speak(u);

"""))

st.bokeh\_chart(tts\_button)



Создадим функцию для графиков, в которую мы будем передавать аудио и sample rate.

*def* display\_plots(audio, sr):

st.subheader("Waveform")

plt.figure(figsize=(10, 3))

librosa.display.waveshow(audio, sr=sr)

plt.xlabel("Time (s)")

plt.ylabel("Amplitude")

plt.title("WavePlot")

*# Save it to a temporary buffer.*

buf = io.BytesIO()

plt.savefig(buf, format="png")

*# Use PIL to create an image from the new buffer*

image = Image.open(buf)

*# Display the image in Streamlit*

st.image(image, width=800)

st.subheader("Spectrogram")

*# Use librosa to calculate the spectrogram*

S = librosa.amplitude\_to\_db(np.abs(librosa.stft(audio)), ref=np.max)

plt.figure(figsize=(10, 4))

librosa.display.specshow(S, sr=sr, x\_axis='time', y\_axis='log')

plt.colorbar(format='%+2.0f dB')

plt.title('Spectrogram')

*# Save it to a temporary buffer again*

buf = io.BytesIO()

plt.savefig(buf, format="png")

*# Create an image again*

image = Image.open(buf)

*# Display the image*

st.image(image, width=800)

*# Cleaning up figures to prevent reuse*

plt.close('all')

Наконец добавим кнопку для транскрибирования аудио. В ней, если аудио есть, или в записи пользователя или в загруженных файлах, происходит обработка аудио и ее пропуск через модель. Если аудио нет, то пользователю выводится сообщение о необходимости загрузки файла.

Чтение и преобразование аудио к нужному формату

*# Read the audio file as bytes*

audio\_bytes = audio\_file.read()

*# Use librosa to load the audio file from the byte content*

audio, sr = librosa.load(io.BytesIO(audio\_bytes), sr=None)

*# If model requires single channel (mono) and audio is stereo, take only one channel*

*if* len(audio.shape) > 1 and audio.shape[0] == 2:

audio = librosa.to\_mono(audio)

*# Convert the audio to the required format*

*if* sr != 16000:

audio = librosa.resample(audio, orig\_sr=sr, target\_sr=16000)

audio\_numpy = np.array(audio).astype(np.float32)

Пропускам через модель

transcription = model.transcribe(audio\_numpy)

Сообщаем пользователю язык и транскрипцию

detected\_language = transcription.get('language', 'N/A')

st.sidebar.success(*f*"Язык аудио был определен как: {detected\_language}")

st.sidebar.markdown(transcription["text"])

Затем передаем данные в функцию графика. У нас два условия, т. к. у нас есть несколько вариантов загрузки данных: пользователь сказал что-то в микрофон или пользователь загрузил аудио на сайт. С каждым случаем мы работает отдельно, т. к. данные считанные с этих кнопок имеют разные названия и разные типы выходов.

Код транскрибирования целиком

*if* st.sidebar.button("Траскрибировать аудио"):

*if* audio\_file is not None:

*# Read the audio file as bytes*

audio\_bytes = audio\_file.read()

*# Use librosa to load the audio file from the byte content*

audio, sr = librosa.load(io.BytesIO(audio\_bytes), sr=None)

*# If model requires single channel (mono) and audio is stereo, take only one channel*

*if* len(audio.shape) > 1 and audio.shape[0] == 2:

audio = librosa.to\_mono(audio)

*# Convert the audio to the required format*

*if* sr != 16000:

audio = librosa.resample(audio, orig\_sr=sr, target\_sr=16000)

audio\_numpy = np.array(audio).astype(np.float32)

st.sidebar.success("Транскрибирование аудио...")

transcription = model.transcribe(audio\_numpy)

detected\_language = transcription.get('language', 'N/A')

st.sidebar.success(*f*"Язык аудио был определен как: {detected\_language}")

st.sidebar.markdown(transcription["text"])

display\_plots(audio, sr)

*if* wav\_audio\_data is not None:

*# Use librosa to load the audio file from the byte content*

audio, sr = librosa.load(io.BytesIO(wav\_audio\_data), sr=None)

*# If model requires single channel (mono) and audio is stereo, take only one channel*

*if* len(audio.shape) > 1 and audio.shape[0] == 2:

audio = librosa.to\_mono(audio)

*# Convert the audio to the required format*

*if* sr != 16000:

audio = librosa.resample(audio, orig\_sr=sr, target\_sr=16000)

audio\_numpy = np.array(audio).astype(np.float32)

st.sidebar.success("Транскрибирование аудио...")

transcription = model.transcribe(audio\_numpy)

detected\_language = transcription.get('language', 'N/A')

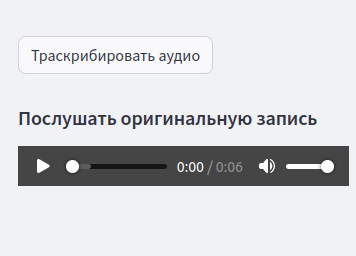
st.sidebar.success(*f*"Язык аудио был определен как: {detected\_language}")

st.sidebar.markdown(transcription["text"])

display\_plots(audio, sr)

*else*:

st.sidebar.error("Пожалуйста, загрузите аудио файл")



Ну и, конечно, добавим пользователю возможность прослушать свою аудиозапись после загрузки

*if* audio\_file is not None:

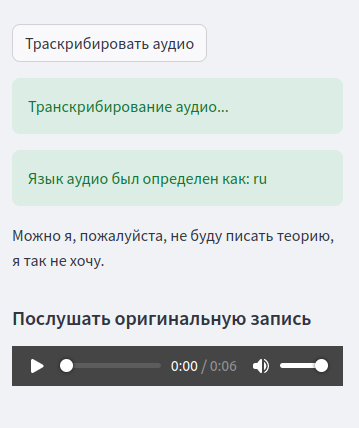
st.sidebar.header("Послушать оригинальную запись")

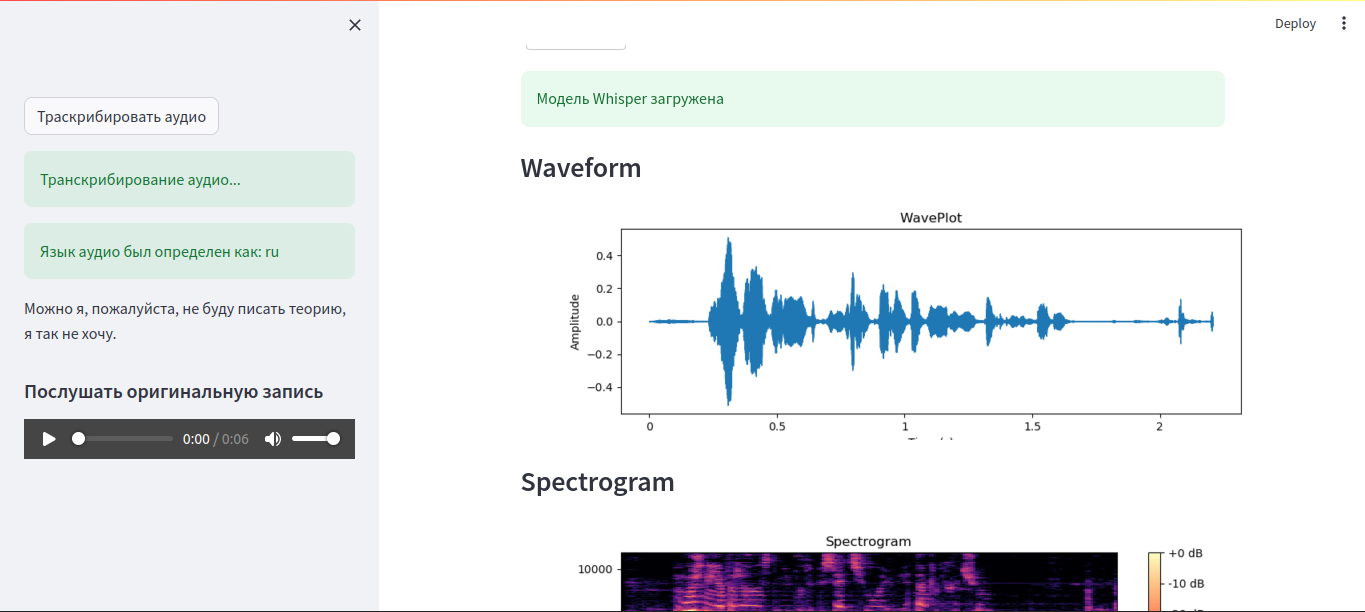
st.sidebar.audio(audio\_file)

*if* wav\_audio\_data is not None:

st.sidebar.header("Послушать оригинальную запись")

st.sidebar.audio(wav\_audio\_data)





Ура, конец!