Audio2Art: Project Design - Part 2

# 1. File Structure

- app.py: Main application interface using Streamlit

- ImageModel.py: Contains model logic (promptgen, text2image)

- requirements.txt: Lists necessary Python packages

# 2. promptgen Function

- Uses transformers.Wav2Vec2Tokenizer and Wav2Vec2ForCTC

- Transcribes the uploaded .wav audio file into text prompt

* Uses librosa for audio waveform extraction and PyTorch for inference

Code:

from transformers import Wav2Vec2Tokenizer, Wav2Vec2ForCTC

import librosa as lb

import torch

def promptgen(file):

tokenizer = Wav2Vec2Tokenizer.from\_pretrained('facebook/wav2vec2-base-960h')

model = Wav2Vec2ForCTC.from\_pretrained('facebook/wav2vec2-base-960h')

waveform, rate = lb.load(file, sr=16000)

input\_values = tokenizer(waveform, return\_tensors='pt').input\_values

logits = model(input\_values).logits

predicted\_ids = torch.argmax(logits, dim=-1)

transcription = tokenizer.batch\_decode(predicted\_ids)

return transcription[0]

# 3. text2image Function

- Accepts a string prompt and model identifier

- Uses StableDiffusionPipeline from diffusers

* Generates an image using the specified model and returns it along with generation time

Code:

from diffusers import StableDiffusionPipeline

import torch

import time

def text2image(prompt, repo\_id):

seed = 2024

generator = torch.manual\_seed(seed)

if torch.cuda.is\_available():

pipe = StableDiffusionPipeline.from\_pretrained(repo\_id, torch\_dtype=torch.float16)

pipe = pipe.to("cuda")

else:

pipe = StableDiffusionPipeline.from\_pretrained(repo\_id, torch\_dtype=torch.float32)

start = time.time()

image = pipe(prompt, guidance\_scale=7.5).images[0]

end = time.time()

return image, start, end

# 4. Streamlit Integration

- Handles file upload, model selection, and form submission

- Calls promptgen and text2image

- Displays image and provides download button

* Includes sidebar instructions for usability

Code:

import streamlit as st

from ImageModel import promptgen, text2image

from io import BytesIO

def app():

st.title("Audio2Art: Transforming Audio Prompts into Visual Creations")

upload\_file = st.file\_uploader("Choose your .wav audio file", type=["wav"])

option = st.selectbox("Choose a model", ["CompVis/stable-diffusion-v1-4"])

with st.form("my\_form"):

submit = st.form\_submit\_button(label="Submit Audio File!")

if submit:

with st.spinner("Generating Image … It may take some time."):

prompt = promptgen(upload\_file)

im, start, end = text2image(prompt, option)

st.image(im)

buf = BytesIO()

im.save(buf, format="PNG")

st.download\_button("Download Image", buf.getvalue(), file\_name="output.png")

# 5. Deployment Plan

- Hosted and tested on Google Colab using LocalTunnel for public access

- Model inference accelerated using GPU environment

- Suitable for migration to Streamlit Cloud or Heroku in future