Tamil to English –bot:  
  
from transformers import MarianMTModel, MarianTokenizer

import gradio as gr

# Load the tokenizer and model

model\_name = "Helsinki-NLP/opus-mt-mul-en"

tokenizer = MarianTokenizer.from\_pretrained(model\_name)

model = MarianMTModel.from\_pretrained(model\_name)

def translate\_text(prompt, src\_lang):

    if src\_lang == "ta":  # Checking if the source language is Tamil

        # Encode the input and generate translation

        inputs = tokenizer(prompt, return\_tensors="pt", padding=True)

        translated = model.generate(\*\*inputs)

        translated\_text = tokenizer.decode(translated[0], skip\_special\_tokens=True)

        return translated\_text

    return "Currently supports Tamil ('ta') language only."

# Gradio interface

demo = gr.Interface(

    fn=translate\_text,

    inputs=[gr.Textbox(lines=2, label="Text to Translate"),

            gr.Dropdown(choices=['ta'], label="Source Language")],  # Tamil as source language

    outputs=gr.Textbox(label="Translated Text"),

)

demo.launch()

Tamil to English to image:  
  
from transformers import MarianMTModel, MarianTokenizer

from diffusers import StableDiffusionPipeline

import gradio as gr

import torch

# Load the translation model and tokenizer

translation\_model\_name = "Helsinki-NLP/opus-mt-mul-en"  # Tamil to English

translation\_tokenizer = MarianTokenizer.from\_pretrained(translation\_model\_name)

translation\_model = MarianMTModel.from\_pretrained(translation\_model\_name)

# Load the Stable Diffusion model for image generation

image\_model\_id = "CompVis/stable-diffusion-v1-4"

image\_pipe = StableDiffusionPipeline.from\_pretrained(image\_model\_id, torch\_dtype=torch.float16)

image\_pipe = image\_pipe.to("cuda")  # Use GPU for inference

def translate\_and\_generate\_image(prompt):

    # Translate Tamil to English

    inputs = translation\_tokenizer(prompt, return\_tensors="pt", padding=True)

    translated\_tokens = translation\_model.generate(\*\*inputs)

    translated\_text = translation\_tokenizer.decode(translated\_tokens[0], skip\_special\_tokens=True)

    # Generate an image based on the translated text

    image = image\_pipe(translated\_text).images[0]

    return image, translated\_text

# Gradio interface

demo = gr.Interface(

    fn=translate\_and\_generate\_image,

    inputs=gr.Textbox(lines=2, label="Enter Tamil Text to Translate"),

    outputs=[gr.Image(label="Generated Image"), gr.Textbox(label="Translated Text")],

    title="Tamil to English Translation and Image Generation",

    description="Enter Tamil text to translate it to English and generate an image based on the translated text."

)

# Launch the app

demo.launch()

FULL CODE:

from transformers import MarianMTModel, MarianTokenizer, GPTNeoForCausalLM, GPT2Tokenizer

from diffusers import StableDiffusionPipeline

import gradio as gr

import torch

# Load the translation model and tokenizer

translation\_model\_name = "Helsinki-NLP/opus-mt-mul-en"  # Tamil to English

translation\_tokenizer = MarianTokenizer.from\_pretrained(translation\_model\_name)

translation\_model = MarianMTModel.from\_pretrained(translation\_model\_name)

# Load the Stable Diffusion model for image generation

image\_model\_id = "CompVis/stable-diffusion-v1-4"

image\_pipe = StableDiffusionPipeline.from\_pretrained(image\_model\_id, torch\_dtype=torch.float16)

image\_pipe = image\_pipe.to("cuda")  # Use GPU for inference

# Load GPT-Neo model and tokenizer for content generation

gpt\_neo\_model\_name = "EleutherAI/gpt-neo-1.3B"  # You can choose a larger model if needed

gpt\_neo\_tokenizer = GPT2Tokenizer.from\_pretrained(gpt\_neo\_model\_name)

gpt\_neo\_model = GPTNeoForCausalLM.from\_pretrained(gpt\_neo\_model\_name)

def translate\_and\_generate\_image(prompt):

    # Translate Tamil to English

    inputs = translation\_tokenizer(prompt, return\_tensors="pt", padding=True)

    translated\_tokens = translation\_model.generate(\*\*inputs)

    translated\_text = translation\_tokenizer.decode(translated\_tokens[0], skip\_special\_tokens=True)

    # Generate creative text using GPT-Neo

    gpt\_input\_ids = gpt\_neo\_tokenizer.encode(translated\_text, return\_tensors="pt")

    gpt\_output = gpt\_neo\_model.generate(gpt\_input\_ids, max\_length=100)

    creative\_text = gpt\_neo\_tokenizer.decode(gpt\_output[0], skip\_special\_tokens=True)

    # Generate an image based on the creative text

    image = image\_pipe(creative\_text).images[0]

    return image, translated\_text, creative\_text

# Gradio interface

demo = gr.Interface(

    fn=translate\_and\_generate\_image,

    inputs=gr.Textbox(lines=2, label="Enter Tamil Text to Translate"),

    outputs=[gr.Image(label="Generated Image"),

             gr.Textbox(label="Translated Text"),

             gr.Textbox(label="Creative Text from GPT-Neo")],

    title="Tamil to English Translation, Creative Text Generation, and Image Generation",

    description="Enter Tamil text to translate it to English, generate creative text based on the translation, and create an image based on the creative text."

)

# Launch the app

demo.launch()