Prototype Development for Image Captioning Using the BLIP Model and Gradio Framework

AIM:

To design and deploy a prototype application for image captioning by utilizing the BLIP image-captioning model and integrating it with the Gradio UI framework for user interaction and evaluation.

PROBLEM STATEMENT:

Image captioning is the task of generating a textual description for a given image. This is crucial for applications like accessibility tools for visually impaired users, content generation, and image indexing. Traditional systems often rely on predefined labels or are limited in context understanding. By leveraging the BLIP model—a state-of-the-art vision-language pretraining model—this project aims to create an intuitive and efficient application for real-time image captioning, accessible via the Gradio interface.

DESIGN STEPS:

STEP 1:

Model Preparation

Use a pre-trained BLIP image-captioning model available from Hugging Face Transformers or similar libraries. Ensure the model supports inference on diverse image types and contexts.

STEP 2:

Framework

Use Gradio to create a UI with the following components: Input: File upload for images. Output: Textbox showing the generated caption.

STEP 3:

Workflow

Load the BLIP model and tokenizer. Accept an image as input via Gradio's file upload. Preprocess the image for the BLIP model. Generate a caption using the BLIP model's inference pipeline. Display the caption on the Gradio interface.

STEP 4:

Testing and Deployment

Test the application with various image types to ensure the captions are meaningful and diverse. Deploy the application on a public URL using Gradio’s hosting features or external platforms like Hugging Face Spaces.

PROGRAM:

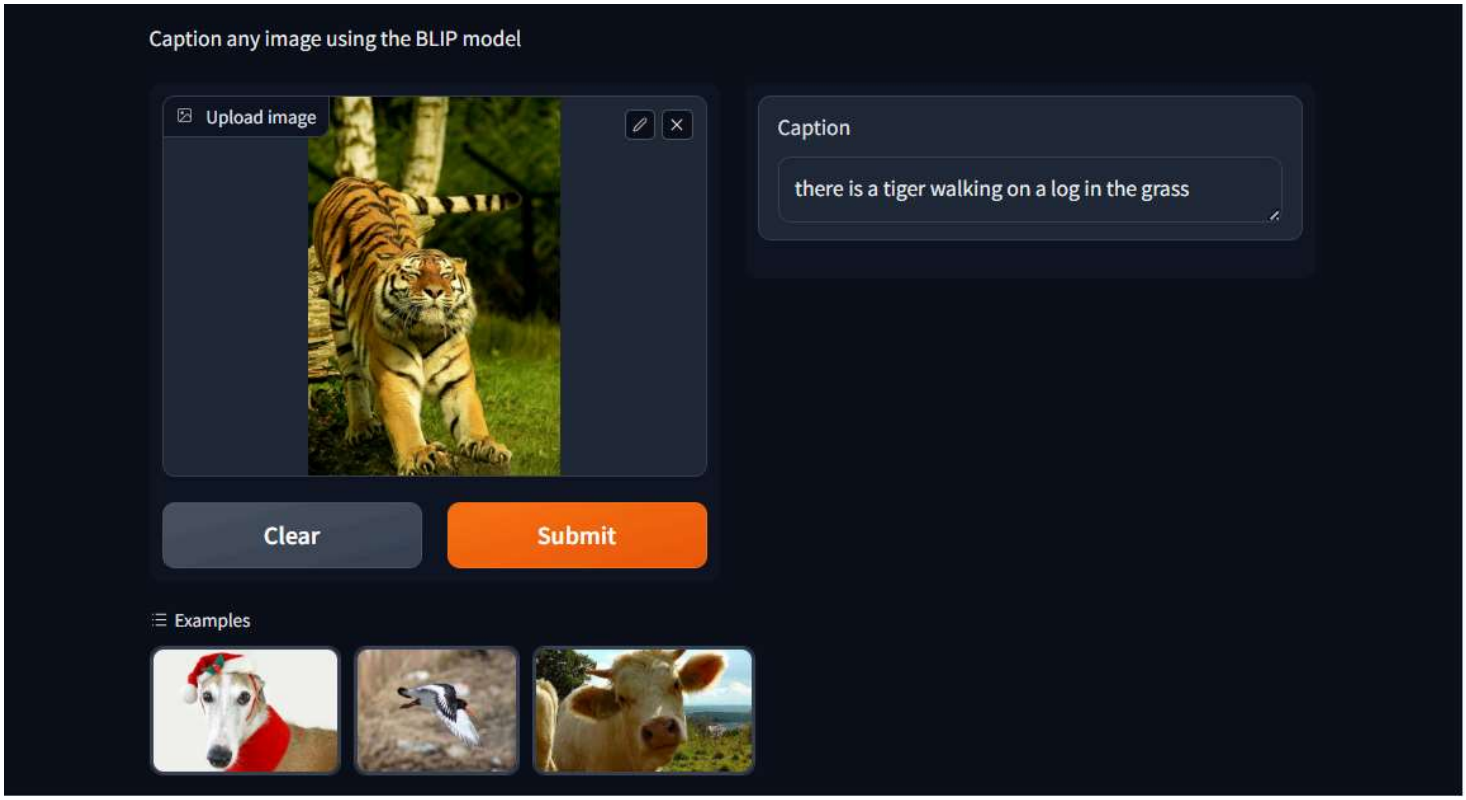
```
import os
import io
import IPython.display
from PIL import Image
import base64
from dotenv import load_dotenv, find_dotenv
_ = load_dotenv(find_dotenv()) # read local .env file
hf_api_key = os.environ['HF_API_KEY']

# Helper functions
import requests, json

#Image-to-text endpoint
def get_completion(inputs, parameters=None, ENDPOINT_URL=os.environ['HF_API_ITT_BASE']):
```

```
    headers = {
      "Authorization": f"Bearer {hf_api_key}",
      "Content-Type": "application/json"
    }
    data = { "inputs": inputs }
    if parameters is not None:
        data.update({"parameters": parameters})
    response = requests.request("POST",
                                ENDPOINT_URL,
                                headers=headers,
                                data=json.dumps(data))
    return json.loads(response.content.decode("utf-8"))

image_url = "https://free-images.com/sm/9596/dog_animal_greyhound_983023.jpg"
display(IPython.display.Image(url=image_url))
get_completion(image_url)

import gradio as gr

def image_to_base64_str(pil_image):
    byte_arr = io.BytesIO()
    pil_image.save(byte_arr, format='PNG')
    byte_arr = byte_arr.getvalue()
    return str(base64.b64encode(byte_arr).decode('utf-8'))

def captioner(image):
    base64_image = image_to_base64_str(image)
    result = get_completion(base64_image)
    return result[0]['generated_text']

gr.close_all()
demo = gr.Interface(fn=captioner,
                    inputs=[gr.Image(label="Upload image", type="pil")],
                    outputs=[gr.Textbox(label="Caption")],
                    title="Image Captioning with BLIP",
                    description="Caption any image using the BLIP model",
                    allow_flagging="never",
                    examples=["christmas_dog.jpeg", "bird_flight.jpeg", "cow.jpeg"])

demo.launch(share=True, server_port=int(os.environ['PORT1']))
```

OUTPUT:

Example Input and Output

Input:



Output:

there is a tiger walking on a log in the grass

RESULT:

The application successfully generates high-quality images based on user-provided text prompts. The Stable Diffusion model ensures visually appealing results, and the Gradio interface makes it accessible and interactive.