Image Generation with Stable Diffusion

Installation and Imports:

Installs necessary libraries: diffusers and transformers using pip. Imports required modules from pathlib, tqdm, torch, pandas, numpy, diffusers, transformers, matplotlib, and cv2.

Configuration:

Defines a CFG class to store configuration settings: device: Sets the device to "cuda" for GPU acceleration. seed: Sets the random seed for reproducibility. image_gen_steps: Specifies the number of inference steps for image generation. image_gen_model_id: Specifies the Stable Diffusion model ID. image_gen_size: Sets the desired image size. image_gen_guidance_scale: Controls the influence of the prompt on the generated image.

Model Loading:

Loads the Stable Diffusion pipeline from the specified model ID using your Hugging Face token. Moves the model to the specified device (GPU). **Image Generation Function:**

Defines a function generate_image that takes a prompt and the model as input: Generates an image using the provided prompt and configuration settings. Resizes the image to the desired size. Returns the generated image. Image Generation and Saving:

Generates two images using the prompt "raining in village and children are playing in rain. Create nostalgic view with full functional hands". Saves the generated images as "image0.png" and "image1.png". Displaying Images:

Imports matplotlib.pyplot and matplotlib.image for image display. Displays the saved images using plt.imshow and plt.show.

```
!pip install --upgrade diffusers transformers -q
```

```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 44.4/44.4 kB 2.2 MB/s eta 0:00:00
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.2/3.2 MB 48.8 MB/s eta 0:00:00
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 9.7/9.7 MB 112.9 MB/s eta 0:00:00
```

```
from pathlib import Path
import tqdm
import torch
import pandas as pd
import numpy as np
from diffusers import StableDiffusionPipeline
from transformers import pipeline, set_seed
import matplotlib.pyplot as plt
```

```
import matplotlib.pyplot as plt
import cv2
```

```
The cache for model files in Transformers v4.22.0 has been updated. Migrating your old cache. This is a one-time only operation. You can interru
0/0 [00:00<?, ?it/s]
```

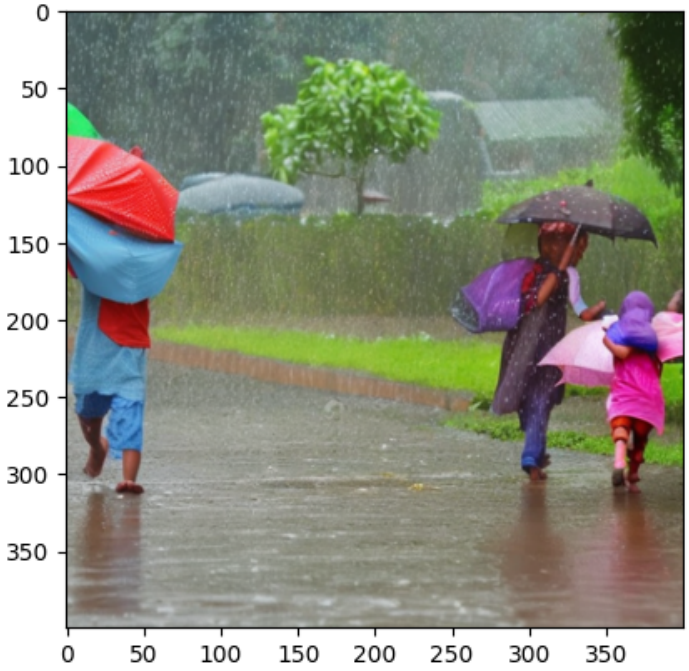
```
class CFG:
    device = "cuda"
    seed = 42
    generator = torch.Generator(device).manual_seed(seed)
    image_gen_steps = 35
    image_gen_model_id = "stabilityai/stable-diffusion-2"
    image_gen_size = (400,400)
    image_gen_guidance_scale = 9
    prompt_gen_model_id = "gpt2"
    prompt_dataset_size = 6
    prompt_max_length = 12


image_gen_model = StableDiffusionPipeline.from_pretrained(
    CFG.image_gen_model_id, torch_dtype=torch.float16,
    revision="fp16", use_auth_token='<Hugging Face Token>', guidance_scale=9
)
image_gen_model = image_gen_model.to(CFG.device)
```

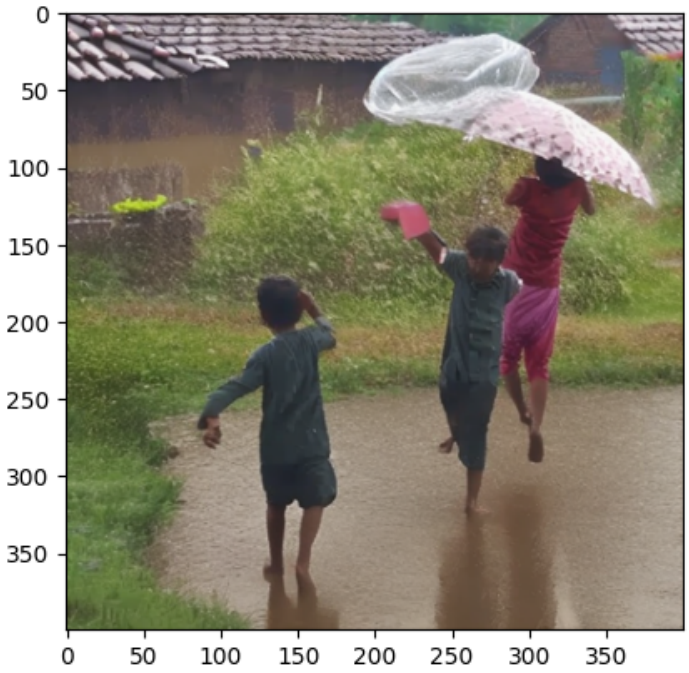
```
Show hidden output
```

```
def generate_image(prompt, model):
    image = model(
        prompt, num_inference_steps=CFG.image_gen_steps,
        generator=CFG.generator,
        guidance_scale=CFG.image_gen_guidance_scale
    ).images[0]

    image = image.resize(CFG.image_gen_size)
    return image


for i in range(2):
  image = generate_image("raining in village and children are playing in rain. Create nostalgic view with full functional hands", image_gen_model)
  image.save(f"image{i}.png")
```

```
100%  35/35 [00:12<00:00,  2.96it/s]
100%  35/35 [00:11<00:00,  2.96it/s]
```

```
# prompt: show png images

import matplotlib.pyplot as plt
import matplotlib.image as mpimg

for i in range(2):
  img = mpimg.imread(f"image{i}.png")
  imgplot = plt.imshow(img)
  plt.show()
```


0
50
100
150
200
250
300
350
0 50 100 150 200 250 300 350


0
50
100
150
200
250
300
350
0 50 100 150 200 250 300 350

```
generate_image("IT Studennts", image_gen_model)



# Just an Additional to convert image to text

!pip install transformers
from transformers import pipeline

# Use a better image-to-text model from Hugging Face
image_to_text = pipeline("image-to-text", model="Salesforce/blip-image-captioning-base")


def generate_story_from_image(image_path):
    # Process the image and generate text
    text = image_to_text(image_path)[0]["generated_text"]
    return text

# Example usage (assuming you have images saved as "candle_light_brunch_0.png", etc.)
for i in range(5):
    image_path = f"image{i}.png"
    story = generate_story_from_image(image_path)
    print(f"Story for image {i}:\n{story}\n")

# # Process the IT Students image
# story = generate_story_from_image("IT Students.png")
# print(f"Story for IT Students image:\n{story}\n")
```