Aim

To develop an automated method for segmenting neurons in microscopy images using deep learning techniques to assist in neurobiological research.

Objective

- Implement a segmentation model to identify and delineate neurons in images.
- Utilize the Segment Anything Model (SAM) for generating accurate masks.
- Validate the segmentation results through visual inspection and quantitative metrics.

Summary

This project utilizes the Segment Anything Model (SAM) for neuron segmentation in microscopy images. The process involves loading a pre-trained SAM model, generating segmentation masks for neurons in an image, and visualizing the results. The project is executed on a local machine equipped with an NVIDIA RTX 3060 GPU.

Tools and Libraries Used

- **PyTorch**: For loading the pre-trained SAM model and utilizing GPU acceleration.
- **Torchvision**: For handling image transformations and utilities.
- **NumPy**: For numerical operations and array manipulations.
- **Matplotlib**: For visualizing images and segmentation masks.
- **OpenCV**: For image reading and preprocessing.
- **SAM (Segment Anything Model)**: For automatic mask generation.

Procedure

Code Explanation

1. **Environment Setup:**

CODE:

```
pip install torch==1.7.1+cu110 torchvision==0.8.2+cu110 torchaudio==0.7.2 -f https://download.pytorch.org/whl/torch_stable.html

pip install numpy<2
```

2. **Import Libraries:**

CODE:

```
import torch

import torchvision
import numpy as np
import matplotlib.pyplot as plt
import cv2
import sys
sys.path.append("..")
from segment_anything import sam_model_registry, SamAutomaticMaskGenerator, SamPredictor
```

3. **Check PyTorch and CUDA Setup:**

CODE:

```
print("PyTorch version:", torch.__version__)
print("Torchvision version:", torchvision.__version__)
print("CUDA is available:", torch.cuda.is_available())
```

4. **Load and Preprocess Image:**

CODE:

```
image = cv2.imread('neurons.jpg')  # Try houses.jpg or neurons.jpg
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

plt.figure(figsize=(10,10))
plt.imshow(image)
plt.axis('off')
plt.show()
```

5. **Load SAM Model:**

CODE:

```
sam_checkpoint = "sam_vit_h_4b8939.pth"
model_type = "vit_h"
device = "cuda"

sam = sam_model_registry[model_type](checkpoint=sam_checkpoint)
sam.to(device=device)
```

6. **Generate Masks:**

CODE:

```
mask_generator_ = SamAutomaticMaskGenerator(
    model=sam,
    points_per_side=32,
    pred_iou_thresh=0.9,
    stability_score_thresh=0.96,
    crop_n_layers=1,
    crop_n_points_downscale_factor=2,
    min_mask_region_area=100,  # Requires open-cv to run post-processing
)

masks = mask_generator_.generate(image)
print(len(masks))
```

7. **Display Masks:**

CODE:

```
def show_anns(anns):
    if len(anns) == 0:
        return
    sorted_anns = sorted(anns, key=(lambda x: x['area']), reverse=True)
    ax = plt.gca()
    ax.set_autoscale_on(False)
    polygons = []
    color = []
    for ann in sorted_anns:
        m = ann['segmentation']
        img = np.ones((m.shape[0], m.shape[1], 3))
        color_mask = np.random.random((1, 3)).tolist()[0]
        for i in range(3):
            img[:,:,i] = color_mask[i]
    ax.imshow(np.dstack((img, m*0.35)))

plt.figure(figsize=(10,10))
plt.imshow(image)
show_anns(masks)
plt.axis('off')
plt.show()
```

Highlights

- **Utilization of SAM**: The SAM model is used for automatic and accurate mask generation, which simplifies the segmentation process.
- **GPU Acceleration**: Leveraging the NVIDIA RTX 3060 GPU enhances computational efficiency, allowing for faster processing and model inference.
- **Customizable Parameters**: The mask generator allows customization of parameters like points_per_side, pred_iou_thresh, and min_mask_region_area to optimize segmentation performance.

Conclusion

The neuron segmentation project demonstrates the effectiveness of the Segment Anything Model (SAM) in generating accurate segmentation masks for neurons in microscopy images. By utilizing GPU acceleration, the project achieves efficient processing, making it suitable for large-scale neurobiological studies. The approach can be adapted for various image segmentation tasks, showcasing its versatility and robustness.