Instructions

Requirements

In this project, we use PyTorch as our deep learning Python library. In addition to the default `comp0197-cw1-pt` conda environment, please also install the following libraries:

- `matplotlib`
- `einops`

with pip or conda, or use:

```
pip install -r requirements.txt
```

Self-Supervised Pre-Training with SimMIM on ImageNet-1K

1. Navigate to the project's `/data/` folder and download ImageNet-1K by either running these commands below in a bash command line, or manually using the links to these 3 files (devkit, validation, train):

   ```
   cd data
   wget https://image-net.org/data/ILSVRC/2012/ILSVRC2012_devkit_t12.tar.gz --no-check-certificate
   wget https://image-net.org/data/ILSVRC/2012/ILSVRC2012_img_val.tar --no-check-certificate
   wget https://image-net.org/data/ILSVRC/2012/ILSVRC2012_img_train.tar --no-check-certificate
   ```

   Note that this may take upwards of **4 hours**, depending on your connection. You may choose to download the validation set and the devkit files only and train on this smaller subset, in which case use the `--val_set` flag in the next step.
2. Run the following to start pre-training (with the default settings used in the report, on a smaller subset of data):

   ```
   python main_pretrain.py \
       --config vit_4M_pretrain \
       --train_size 10000 \
   ```

   This will first extract the compressed ImageNet files, then start printing training statistics and save the model weights as a `.pth` file every epoch. Use the flag `--run_plots` to save reconstructions during training, and the `--val_set` flag to use the smaller (validation) set only, for quicker testing.

Fine-Tuning on Oxford-IIIT Pets

1. With the pre-trained encoder weights in the `/weights/` folder, run this command for fine-tuning, which will download the Oxford-IIIT Pets dataset and start training initialised with the weights given:

   ```
   python main_finetune.py \
       --config vit_4M_finetune \
       --train_size 6000 \
       --test_size 1000 \
       --weights weights/encoder_vit_4M_pretrain_200K.pth
   ```

   Loss is printed every epoch while test set pixel accuracy and mean IoU is calculated after training is complete. Segmentation predictions will be saved under `/figures/`.
2. To run a baseline model with **no pre-training**, omit the `--weights` argument, i.e. use the following command:

   ```
   python main_finetune.py \
       --config vit_4M_finetune \
   ```

```
    --train_size 6000 \
    --test_size 1000 \
```

Intermediate-Task Fine-Tuning on Intel Image Classification Dataset

1. First, the Intel Image Classification dataset needs to be downloaded. From the project's root directory, run:

```
cd data
wget https://huggingface.co/datasets/miladfa7/Intel-Image-Classification/resolve/main/archive.zip?
download=true
```

2. With the pre-trained encoder weights in the `/weights/` folder, run the following command to perform intermediate fine-tuning on this dataset, followed by segmentation fine-tuning on Oxford-IIIT Pets:

```
    python main_finetune.py \
    --config vit_4M_finetune \
    --train_size 6000 \
    --test_size 1000 \
    --weights weights/encoder_vit_4M_pretrain_200K.pth \
    --int_finetune
```

Evaluation

To plot reconstructions from pre-trained models on ImageNet validation set (download above):

```
python evaluation.py \
    --config vit_4M_pretrain \
    --weights weights/mim_vit_4M_pretrain_200K.pth \
```

To evaluate a fine-tuned segmentation model on the Oxford-IIIt Pets test set, use a command like the following, replacing the weights with those saved after fine-tuning (see above):

```
python evaluation.py \
    --config vit_4M_finetune \
    --weights weights/vit_4M_finetune_data_250.pth \
    --test_size 1000 \
    --train_size 250
```