PorousMediaGAN

Implementation and data repository for

Reconstruction of three-dimensional porous media using generative adversarial neural networks

Authors

Lukas Mosser Twitter
Olivier Dubrule
Martin J. Blunt
Department of Earth Science and Engineering, Imperial College London

Results

Cross-sectional views of the three trained models

- Beadpack Sample
  Beadpack Comparison
- Berea Sample
  Berea Comparison
- Ketton Sample
  Ketton Comparison

Methodology

Process Overview

Instructions

Pre-requisites

- To run any of the `jupyter` notebooks follow instructions here or install via pip.

```
pip install jupyter
```

- In addition we make heavy use of `pandas`, `numpy`, `scipy` and `numba`
- We recommend the use of anaconda
- For numba instructions, you can find a tutorial and installation guideline here.
- For the torch version of the code training and generating code please follow the instructions here
- In addition you will need to have installed torch packages `hdf5` and `dpnn`

```
luarocks install hdf5
luarocks install dpnn
```

- For the pytorch version you will need to have installed `h5py` and `tifffile`

```
pip install h5py
pip install tifffile
```

- Clone this repo

```
git clone https://github.com/LukasMosser/PorousMediaGAN
cd PorousMediaGAN
```

Pre-trained model (Pytorch version only)

We have included a pre-trained model used for the Berea sandstone example in the paper in the repository.

- From the pytorch folder run `generate.py` as follows

```
python generator.py --seed 42 --imageSize 64 --ngf 32 --ndf 16 --nz 512 --netG [path to generator
```

Use the modifier `--imsize` to generate the size of the output images.
`--imsize 1` corresponds to the training image size
Replace `[path to generator checkpoint].pth` with the path to the provided checkpoint e.g.
`checkpoints\berea\berea_generator_epoch_24.pth`
Generating realizations was tested on GPU and CPU and is very fast even for large reconstructions.

Training

We highly recommend a modern Nvidia GPU to perform training.
All models were trained on `Nvidia K40` GPUs.
Training on a single GPU takes approximately 24 hours.
To create the training image dataset from the full CT image perform the following steps:

- Unzipping of the CT image

```
cd ./data/berea/original/raw
#unzip using your preferred unzipper
unzip berea.zip
```

- Use `create_training_images.py` to create the subvolume training images. Here an example use:

```
python create_training_images.py --image berea.tif --name berea --edgelength 64 --stride 32 --targ
```

This will create the sub-volume training images as an hdf5 format which can then be used for training.

- Train the GAN
  Use `main.py` to train the GAN network. Example usage:

```
python main.py --dataset 3D --dataroot [path to training images] --imageSize 64 --batchSize 128 --
```

Additional Training Data

High-resolution CT scan data of porous media has been made publicly available via the Department of Earth Science and Engineering, Imperial College London and can be found here

Data Analysis

We use a number of jupyter notebooks to analyse samples during and after training.

- Use `code\notebooks\Sample Postprocessing.ipynb` to postprocess sampled images

    - Converts image from hdf5 to tiff file format
    - Computes porosity
- Use `code\notebooks\covariance\Compute Covariance.ipynb` to compute covariances
    - To plot results use `Covariance Analysis.ipynb` and `Covariance Graphs.ipynb` as an example on how to analyse the samples.

Image Morphological parameters

We have used the image analysis software Fiji to analyse generated samples using MorpholibJ.
The images can be loaded as tiff files and analysed using `MorpholibJ\Analyze\Analyze Particles 3D`.

Results

We additionally provide the results used to create our publication in `analysis`.

- Covariance S2(r)
- Image Morphology
- Permeability Results
  The Jupyter notebooks included in this repository were used to generate the graphs of the publication.

Citation

If you use our code for your own research, we would be grateful if you cite our publication ArXiv


```
@article{pmgan2017,
	title={Reconstruction of three-dimensional porous media using generative adversarial neural networks},
	author={Mosser, Lukas and Dubrule, Olivier and Blunt, Martin J.},
	journal={arXiv preprint arXiv:1704.03225},
	year={2017}
}
```

Acknowledgement

The code used for our research is based on DCGAN
for the torch version and the pytorch example on how to implement a GAN.
Our dataloader has been modified from DCGAN.

O. Dubrule thanks Total for seconding him as a Visiting Professor at Imperial College.