

Code List

Classification.py	Code of classification task replacing all train & valid images of pitting with GAN generated images
DCGAN.py	code of Deep Convolutional GAN
evaluation_main.py	Main code of the evaluation GUI
evaluation_ui.ui	interface of evaluation GUI generated by Qt Designer
evaluation_ui.py	Code of Interface of evaluation GUI transformed from evaluation_ui.ui
LSGAN.py	code of Least Square GAN
Metric_FID.py	Code of calculation of FID Score
total_ui.ui	interface of main GUI generated by Qt Designer
total_ui.py	Code of Interface of main GUI transformed from total_ui.ui
tsne.py	Code of t-SNE dimension reduction
WGAN_div.py	code of Wasserstein GAN divergence

Instruction of using the GUI as the supplementation of running codes of the Master Thesis

Two GUIs are designed. The first one can be called by running 'total_ui_main.py'. Interface is showed below:

Interface for GAN and its related classification

Input parameters for GAN training

Generator
gen5

Discriminator
dis1

batch size_GAN
50

training loops/epochs
10001

optimizer D adam
0.00020
0.50

optimizer G adam
0.00020
0.50

Training round D in one loop/epoch
1

Training round G in one loop/epoch
1

Training images channels
1

latent vector size for generator input
100

real label
0.90

fake label
0.10

special for DCGAN
original: real=1, fake=0, when smoothed, this can be tuned

saving interval
1000

display row/column
6

training image dataset (load npy file)
select
pitting_tv.npy

result saving folder name (incl. display images, generated images, model)
test1

Train with loop-wise
Train with epoch-wise

Train with loop-wise
Train with epoch-wise

Train with loop-wise
Train with epoch-wise

select GAN model to generate synthetic images
dcgan_result/test1/
generator_model_10000_1_100.h5

generated images amount, best 700
700

optimizer adam
0.00020
0.50

training epochs
200

batch_size_classification
32

result saving folder name (incl. loss and acc values, confusion metrics, metrics table)
d1

train and test

Loss function curve
select
Training, validation, test loss
Training, validation, test loss
Training, validation, test loss

Accuracy curve
select
Training, validation, test accuracy
Training, validation, test accuracy
Training, validation, test accuracy

Confusion Matrix
select
Confusion Matrix
Confusion Matrix
Confusion Matrix

Metrics Table, 0-pitting, 1-intact
select
Metrics Table, 0-pitting, 1-intact
Metrics Table, 0-pitting, 1-intact

Image generation result display
select images


Please follow the order below to input parameters in the blank and click the button:

1. Generator: select the variant of the generator of GAN (gen5 is the best)
2. Discriminator: select the variant of the discriminator of GAN (dis1,2,3 are for DCGAN, dis4,5,6 are for WGAN-div and LSGAN)
3. Batch size_GAN: setting the batch size of GAN training
4. Training loops/epochs: setting the number of loops/epochs of GAN training
5. optimizer D adam: adam optimizer for the discriminator, above is the learning rate of adam optimizer, below is the beta1 of adam optimizer
6. optimizer G adam: adam optimizer for the generator, above is the learning rate of adam optimizer, below is the beta1 of adam optimizer
7. Training round D in one loop/epoch: number of training time of the discriminator per loop/epoch
8. Training round G in one loop/epoch: number of training time of the generator per loop/epoch
9. Training images channels: image channel. 1 for gray images, 3 for RGB images
10. latent vector size for generator input: the input vector size of the generator
11. saving interval: it is the number 'n' which the training results including display images and generator model etc. will be saved every time when training n loops/epochs finished.
12. Display row/column: the row and column of generated stitched images for display after each saving interval. For example: the above display images in the

GUI screenshot is 6x6 stitched images. To generate such display images, you should give the number 5 in the blank.

13. Training image dataset (load npy file): you can click the 'select' button near it to select the a npy file of an original image dataset (for example: the train set of grayscale pitting images) in the npy folder.

14. result saving folder name: input the name of the folder which the result of one experience of GAN training is saved. (for example: test1, test2...)

15. to train with DCGAN, the block area of 'special for DCGAN should be filled. For 'real label', please input the value of smoothed label of real image data in the discriminator (for example: 0.9) and please input the value of 'fake label', which is the smoothed label of the generator in the discriminator (for example: 0.1)

16. Three different GAN variant can be chosen, each one with two training methods: loop-wise and epoch-wise. Click the corresponding button to train the GAN

17. Image generation result display: click 'select image' to select the images saved in the folder which is named in 14.

The following part is designed for doing the classification task which all train & valid set of pitting images are replace by GAN generated images. Please follow the order to input and click in the bottom part of the GUI.

18. select GAN model to generate synthetic images: select the generator model from saved results in 14

19. fill the 'generated images amount', which is the number of generated images. Recommend setting as 700 to be equal as the number of train + valid set of intact surface images.

20. setting adam optimizer for classification like in step 6.

21. setting training epochs of the classification and the batch size.

22. make sure that the latent vector size window in step 10 has correctly being filled.

23. give the folder name of one classification experiment, in which all results are saved. For example: 'cl1'. The folder will be in the main folder 'classification_result'.

24. to visualize the classification result in GUI, click 'select' near the 'Loss function curve' and select the loss curve graph following the path: 'classification_result/cl1/loss_tv.jpg'. cl1 is the example given in step 23.

25. the same manipulation as step 23, click 'select' near accuracy curve to load the graph 'acc_tv.jpg'; click 'select' near Confusion Matrix' to load 'confusion_matrix.jpg'; click 'select' near 'Metrics Table' to select 'report.jpg'.

The second one can be called by running 'total_ui_main.py'. Interface is showed below:

Evaluation

choose original dataset

npz/pitting_tv.npz

choose generator model

drgan_result/pt/
generator_model_30000_z_100.h5

latent variable dimension according to the trained model
should be same as being tuned in GUI "total_ui"

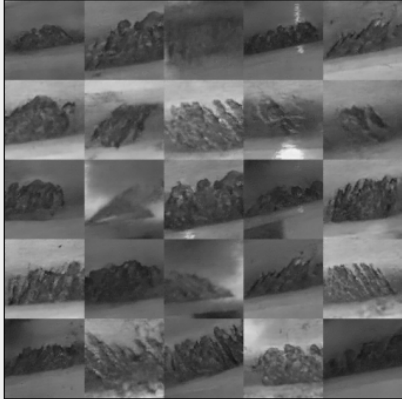
100

rows&columns of generated display images

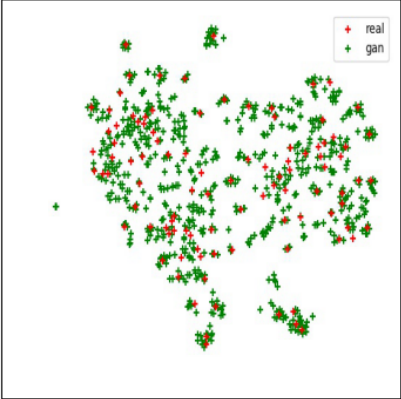
5

random generation

generated display images



t-SNE 2D display



FID SCORE

calculate

107.88918

generated images amount

700

plot_tsne

- 1.click 'choose original dataset' to select real image dataset.
- 2.click 'choose generator model' to select the generator model which generates GAN images and make comparison with real image dataset.
3. fill in the 'latent variable dimension' on the right. You can get the number from the name of the generator name. For example: 'generator_model_30000_z_100.h5' shows that the latent vector size is 100.
4. select the number of rows and columns of generated display images. Images will show in the left side.
5. click 'calculate' to get FID score of the generated image dataset.
6. fill in the number of generated images amount on the bottom right and click the button 'plot_tsne' to generate 2D t-SNE graph.