MLDS HW3-1

TAs
b05902013@csie.ntu.edu.tw
b05902127@csie.ntu.edu.tw

Outline

- Timeline
- Task Descriptions
- Model & Training tips
- Submission & Rules
- Q&A

Timeline

Three Parts in HW3

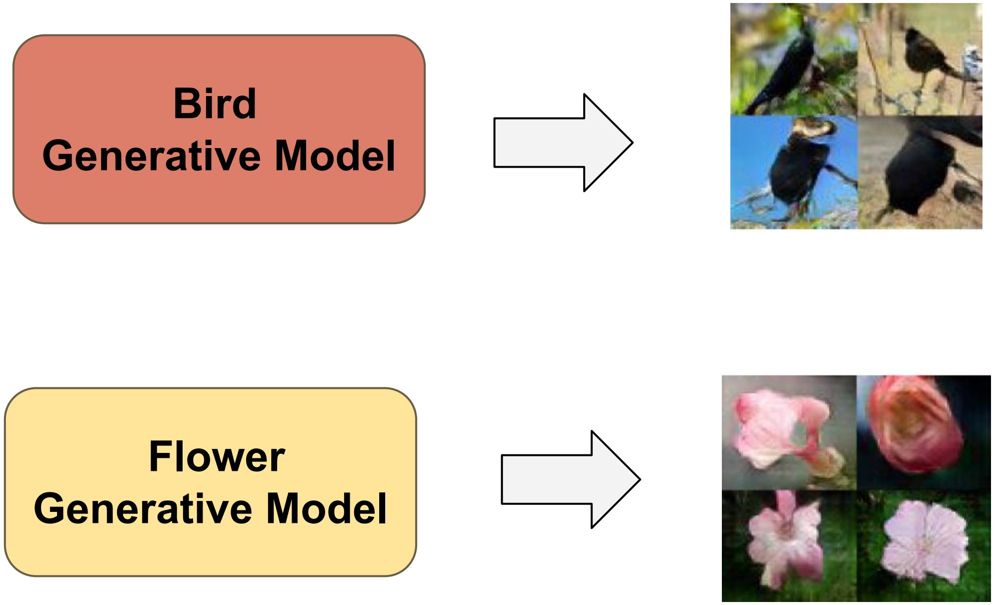
- (3-1) Image Generation
- (3-2) Text-to-Image Generation
- (3-3) Style Transfer

Schedule

- **4/30 or 5/4 :**
    - Release HW3-1
- **5/7 or 5/11 :**
    - Present HW 3-1
    - Release HW 3-2, HW 3-3
- **5/14 or 5/18 : Break**
- **5/21 or 5/25:**
    - Present HW 3-2, HW 3-3

Task Descriptions

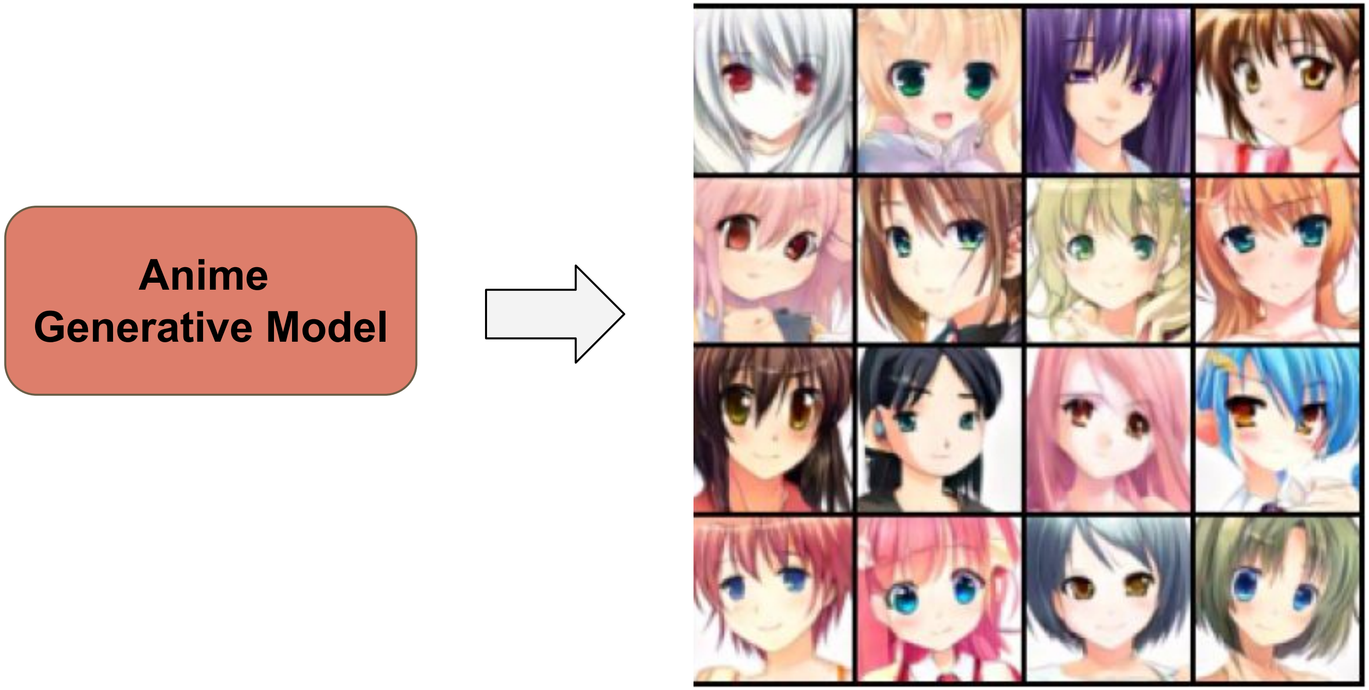
HW3-1: Image Generation 2/2



Bird
Generative Model

Flower
Generative Model

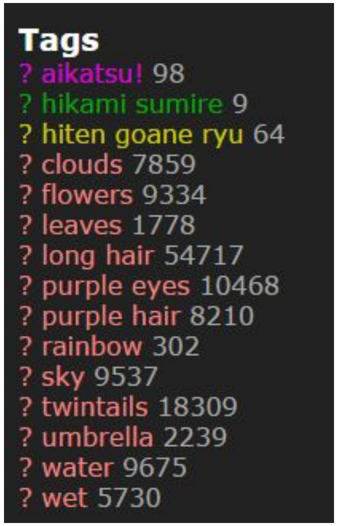
HW3-1: Image Generation 2/2



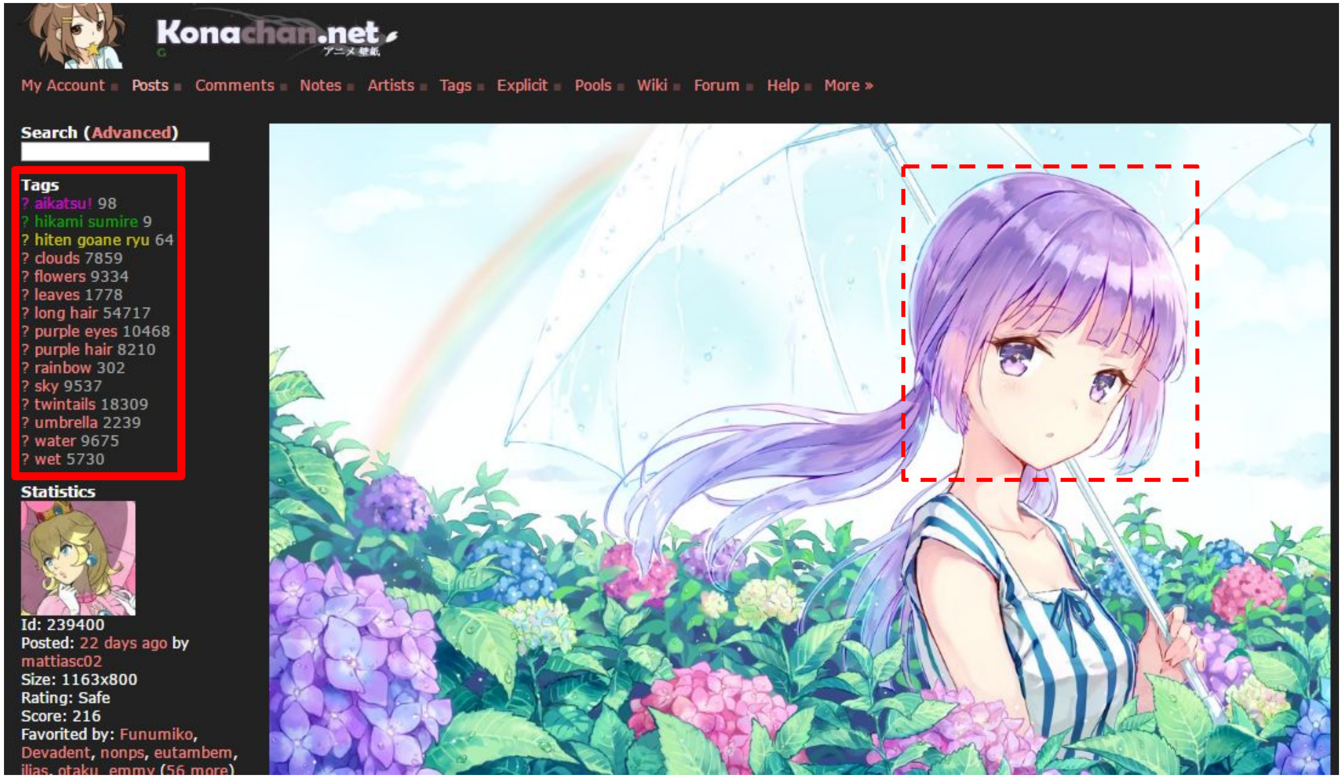
Anime Generative Model

Data Collections 1/2

- Anime dataset



Tags
? aikatsu! 98
? hikami sumire 9
? hiten goane ryu 64
? clouds 7859
? flowers 9334
? leaves 1778
? long hair 54717
? purple eyes 10468
? purple hair 8210
? rainbow 302
? sky 9537
? twintails 18309
? umbrella 2239
? water 9675
? wet 5730



Search (Advanced)

Tags
? aikatsu! 98
? hikami sumire 9
? hiten goane ryu 64
? clouds 7859
? flowers 9334
? leaves 1778
? long hair 54717
? purple eyes 10468
? purple hair 8210
? rainbow 302
? sky 9537
? twintails 18309
? umbrella 2239
? water 9675
? wet 5730

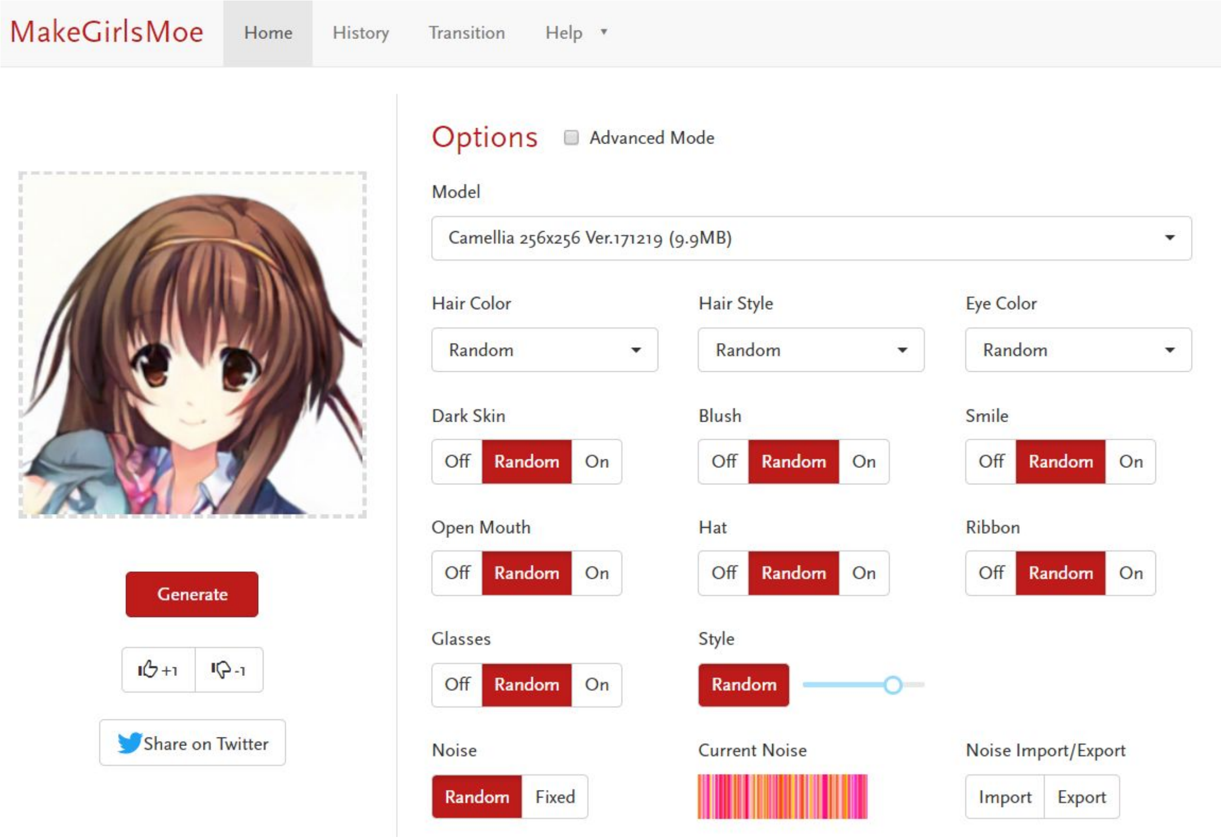
Statistics
Id: 239400
Posted: 22 days ago by mattiasc02
Size: 1163x800
Rating: Safe
Score: 216
Favorited by: Funumiko, Devadent, nonps, eutambem, ilias, otaku_emmy (56 more)

http://konachan.net/post/show/239400/aikatsu-clouds-flowers-hikami_sumire-hiten_goane_r

感謝樊恩宇助教蒐集data

Data Collections 2/2

- Extra data



Options □ Advanced Mode

Model
Camellia 256x256 Ver.171219 (9.9MB)

Hair Color: Random

Hair Style: Random

Eye Color: Random

Dark Skin: Off [✓ Random] On

Blush: Off [✓ Random] On

Smile: Off [✓ Random] On

Open Mouth: Off [✓ Random] On

Hat: Off [✓ Random] On

Ribbon: Off [✓ Random] On

Glasses: Off [✓ Random] On

Style: [✓ Random]

Noise: [✓ Random] Fixed

Current Noise

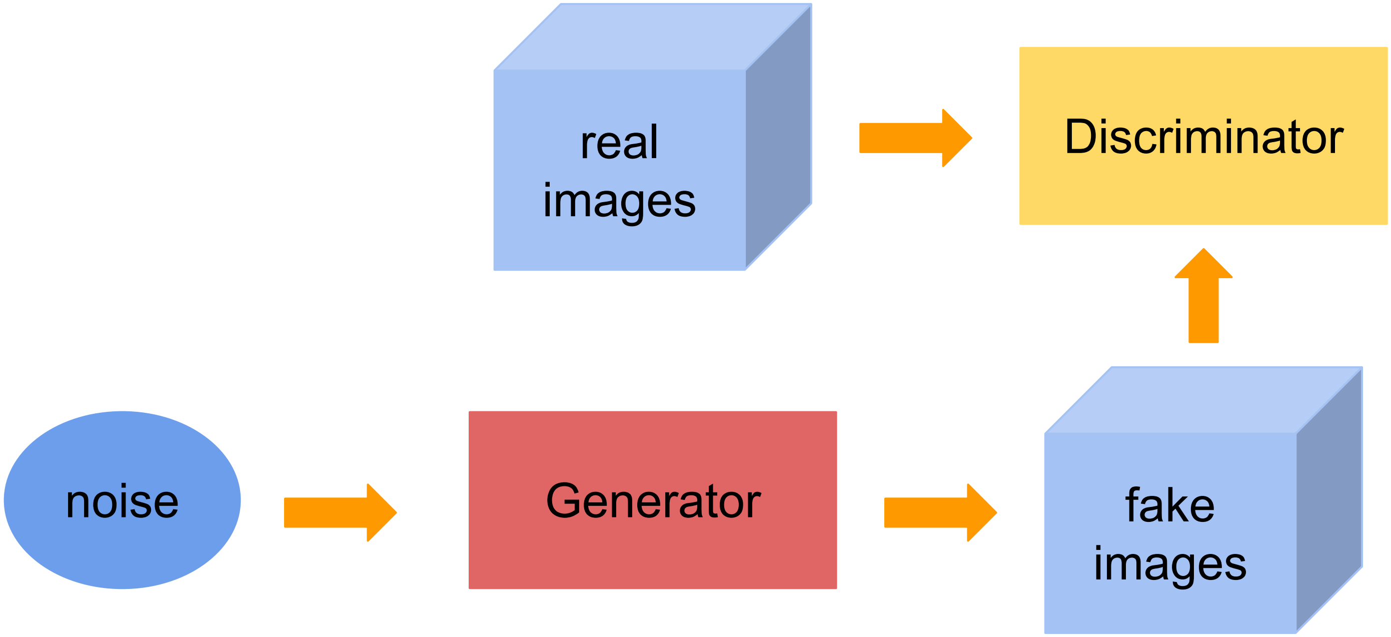
Noise Import/Export: Import Export

https://make.girls.moe/#/ 感謝吳宗翰同學收集並處理資料

Model & Training Tips

GAN 1/5

- Overview



GAN 2/5

- Discriminator
  - input: images (batch_size, height, width, channels)
  - output: scores (batch_size,)
  - arthitecture: CNN, DNN
- Generator
  - input: noises (batch_size, noise_dim)
  - output: images (batch_size, height, width, channels)
  - arthitecture: CNN, DNN
  - use deconvolution (transpose convolution) layer in CNN

Baseline Model

- Generator
  - input = (100,)
  - Dense(128*16*16, ‘relu’)
  - Reshape((16, 16, 128))
  - Upsampling
  - Conv2D(128, kernel = 4)
  - Relu
  - Upsampling
  - Conv2D(64, kernel = 4)
  - Relu
  - Conv2D(3, kernel = 4)
  - tanh

- Training
  - Adam(lr = 0.0002, beta = 0.5)

- Discriminator
  - input = (64, 64, 3)
  - Conv2D(32, kernel = 4)
  - Relu
  - Conv2D(64, kernel = 4)
  - ZeroPadding
  - Relu
  - Conv2D(128, kernel = 4)
  - Relu
  - Conv2D(256, kernel = 4)
  - Relu
  - Flatten
  - Dense(1, sigmoid)

GAN 3/5

- Training procedure

  ```
  repeat max_iteration times
      repeat d_update times
          discriminator = update_discriminator(training_data, generator)
      repeat g_update times
          generator = update_generator(discriminator)
  ```

- Testing procedure

  ```
  noise = sample_batch_noise(batch_size, noise_dim)
  output_images = generator(noise)
  ```

GAN 4/5

- Update discriminator
    - real_images = sample_batch_data(training_data, batch_size)
    - noise = sample_batch_noise(batch_size, noise_dim)
    - fake_images = generator(noise)
    - real_predicts = discriminator(real_images)
    - fake_predicts = discriminator(fake_images)
    - d_loss = loss_d_fn(real_predicts, real_labels, fake_predicts, fake_labels)
    - d_grad = gradients(d_loss, d_params)
    - d_params = updates(d_params, d_grad)
    - # do not update the parameters of generator

GAN 5/5

- Update Generator
    - noise = sample_batch_noise(noise_dim, batch_size)
    - fake_images = generator(noise)
    - fake_predicts = discriminator(fake_images)
    - g_loss = loss_g_fn(fake_predicts, real_labels)
    - g_grad = gradients(g_loss, g_params)
    - g_params = updates(g_params, g_grad)
    - # do not update the parameters of discriminator

Wasserstein GAN 1/3

The output of D is thus not probability anymore.
The D loss turn to be a measure of distance.

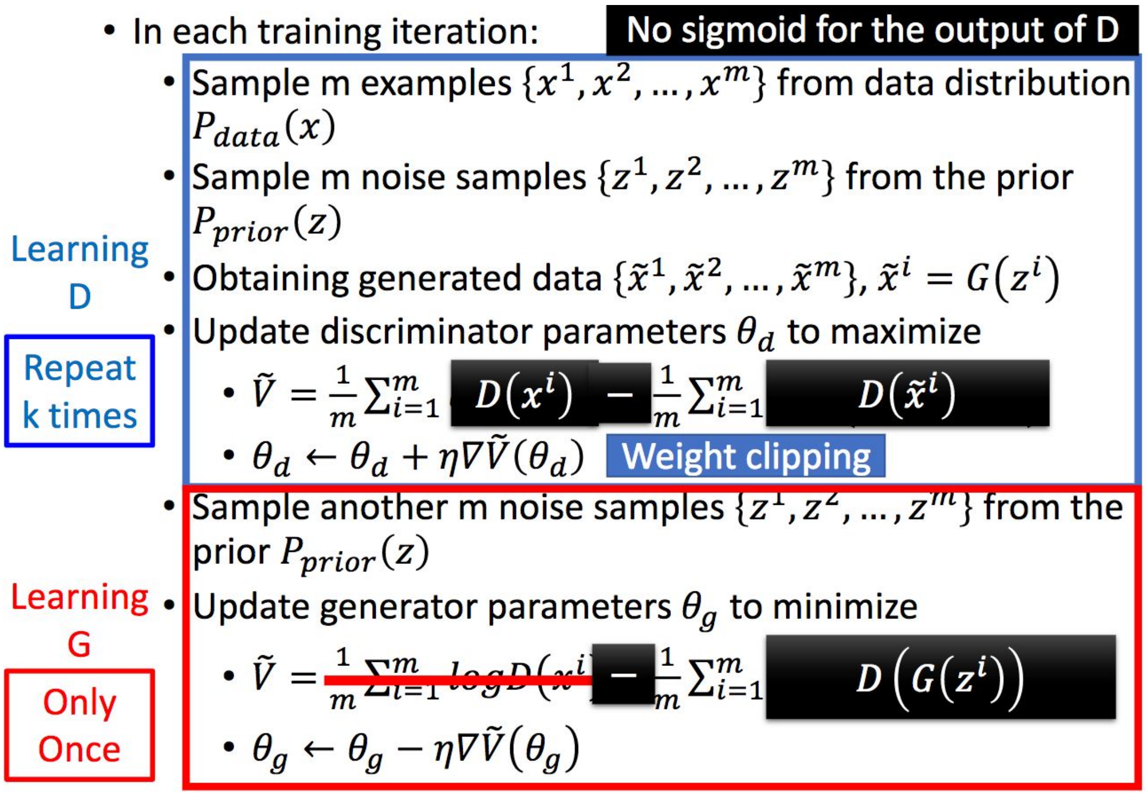
$$L_D^{WGAN} = E[D(x)] - E[D(G(z))]$$

$$L_G^{WGAN} = E[D(G(z))]$$

$$W_D \leftarrow clip\_by\_value(W_D, -0.01, 0.01)$$

ref:https://arxiv.org/abs/1701.07875

Wasserstein GAN 2/3



- In each training iteration:

No sigmoid for the output of D

Learning D

Repeat k times

- Sample m examples $\{x^1, x^2, \ldots, x^m\}$ from data distribution $P_{data}(x)$
- Sample m noise samples $\{z^1, z^2, \ldots, z^m\}$ from the prior $P_{prior}(z)$
- Obtaining generated data $\{\tilde{x}^1, \tilde{x}^2, \ldots, \tilde{x}^m\}$, $\tilde{x}^i = G\left(z^i\right)$
- Update discriminator parameters θ_d to maximize
  - $\tilde{V} = \frac{1}{m}\sum_{i=1}^{m} D\left(x^i\right) - \frac{1}{m}\sum_{i=1}^{m} D\left(\tilde{x}^i\right)$
  - $\theta_d \leftarrow \theta_d + \eta \nabla \tilde{V}(\theta_d)$ Weight clipping

Learning G

Only Once

- Sample another m noise samples $\{z^1, z^2, \ldots, z^m\}$ from the prior $P_{prior}(z)$
- Update generator parameters θ_g to minimize
  - $\tilde{V} = \cancel{\frac{1}{m}\sum_{i=1}^{m} \log D\left(x^i\right)} - \frac{1}{m}\sum_{i=1}^{m} D\left(G\left(z^i\right)\right)$
  - $\theta_g \leftarrow \theta_g - \eta \nabla \tilde{V}(\theta_g)$

Wasserstein GAN 3/3

- Implementation Notes:
  - Do not apply sigmoid at the output of D
  - Clip the weight of D
  - Use RMSProp instead of Adam
  - Train more iteration of D (the paper use 5)

Improved WGAN (WGAN-GP) 1/2

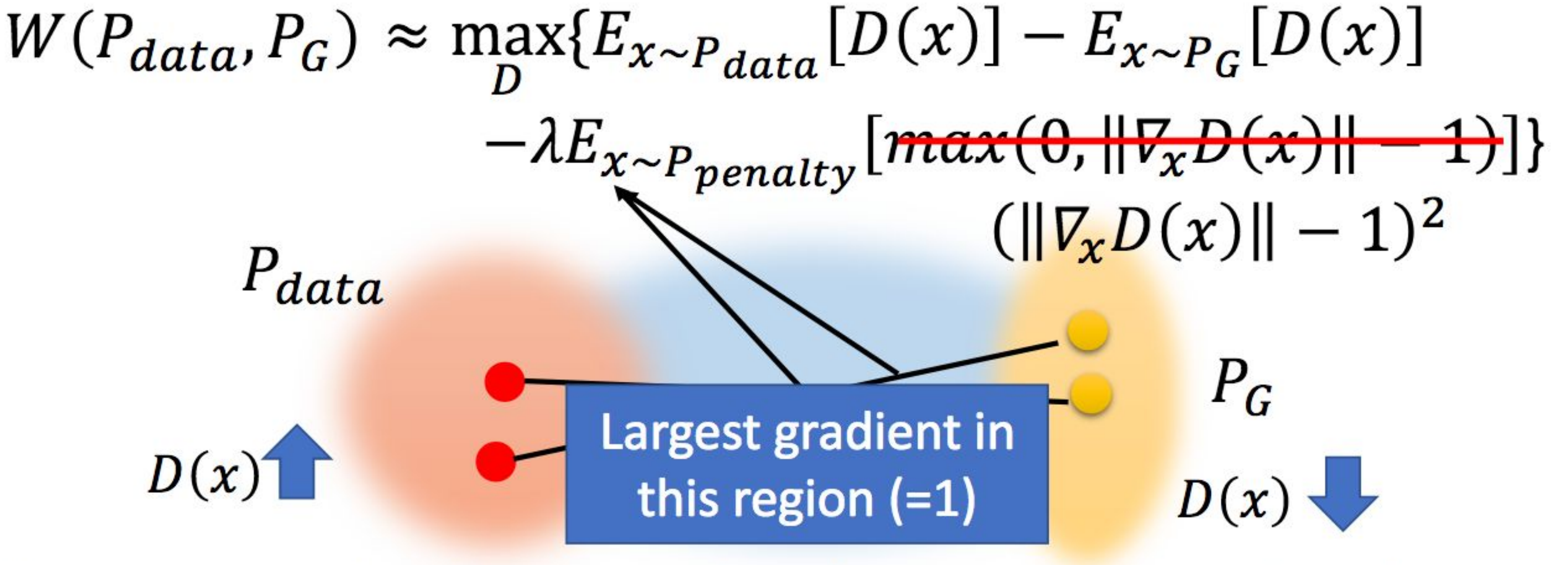
Do not clip the weight of D but to add a new objective called “Gradient Penalty”.

$$L_D^{WGAN\_GP} = L_D^{WGAN} + \lambda E[(\|\nabla D(\alpha x + (1-\alpha)G(z))\|_2 - 1)^2]$$

$$L_G^{WGAN\_GP} = L_G^{WGAN}$$

ref:https://arxiv.org/pdf/1704.00028.pdf

Improved WGAN (WGAN-GP) 2/2



$$W(P_{data}, P_G) \approx \max_D \{E_{x\sim P_{data}}[D(x)] - E_{x\sim P_G}[D(x)]$$

$$-\lambda E_{x\sim P_{penalty}}[\cancel{max(0, \|\nabla_x D(x)\| - 1)}]\}$$

$$(\|\nabla_x D(x)\| - 1)^2$$

ref:https://arxiv.org/pdf/1704.00028.pdf

Least Squares GAN

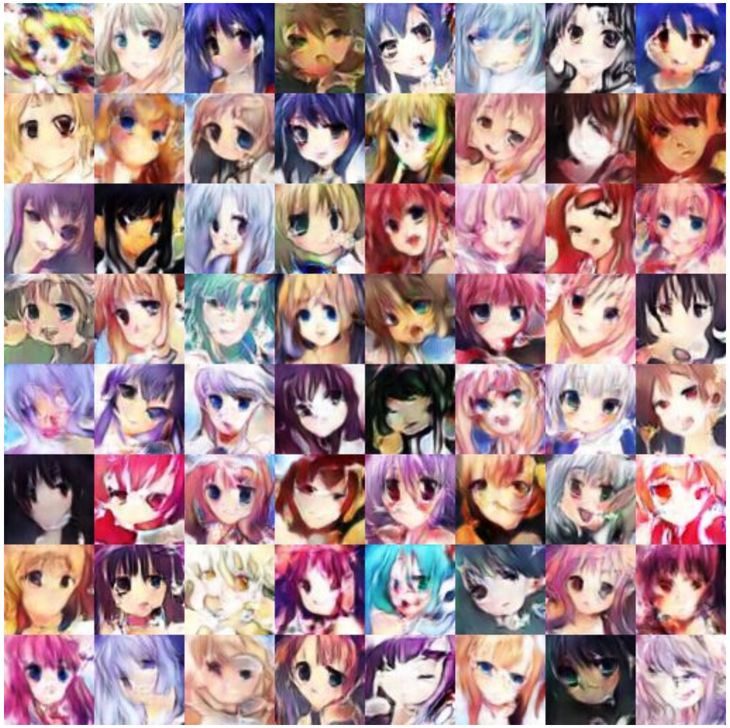
🟧 1 🟥 1 🟧 0 🟥 -1

$$\begin{aligned} \min_D V_{\text{LSGAN}}(D) &= \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}\big[(D(\boldsymbol{x})-b)^2\big] + \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\big[(D(G(\boldsymbol{z}))-a)^2\big] \\ \min_G V_{\text{LSGAN}}(G) &= \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\big[(D(G(\boldsymbol{z}))-c)^2\big], \end{aligned} \tag{2}$$

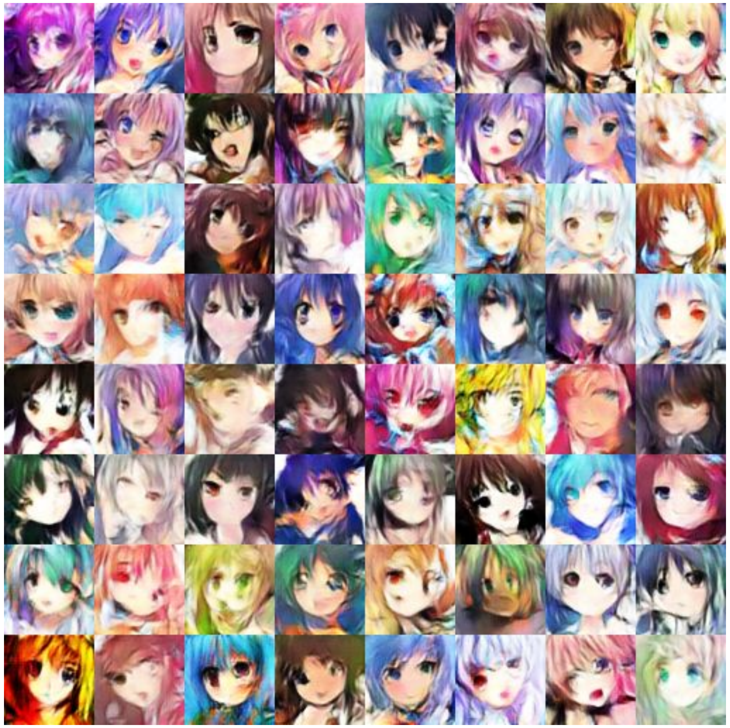
🟧 1 🟥 0

ref:https://arxiv.org/abs/1611.04076

Little Results

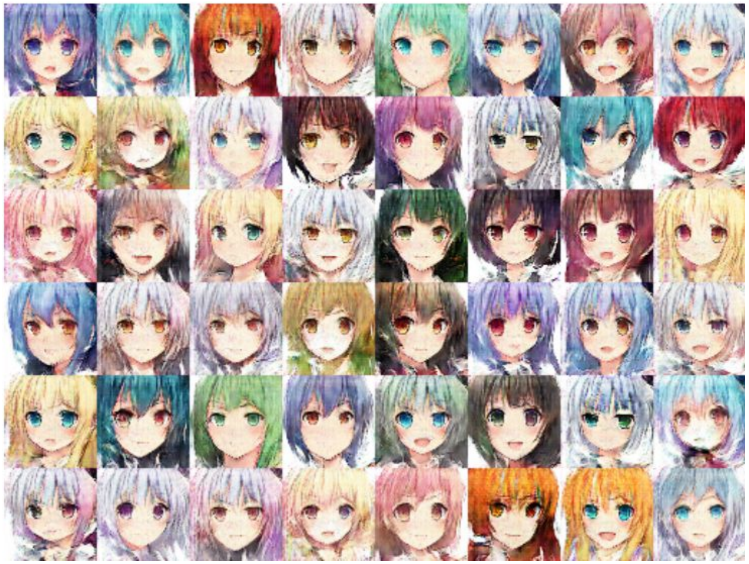


GAN result

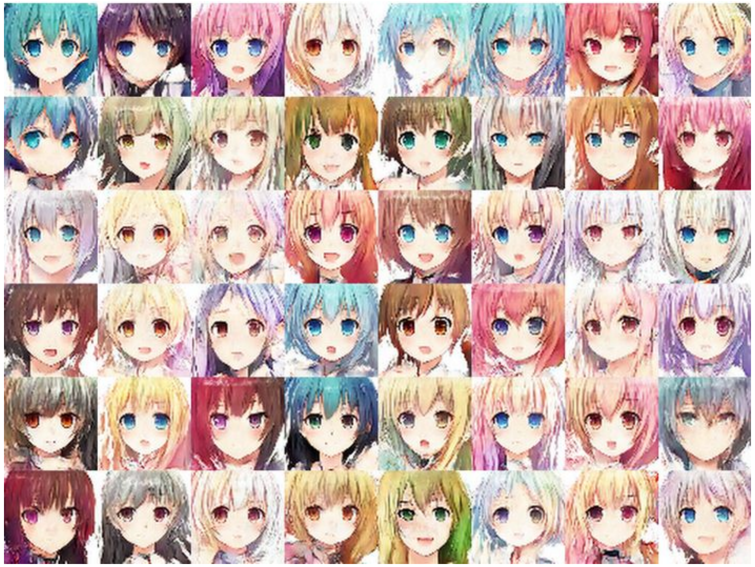


WGAN result

TA Results



DCGAN



DCGAN w/ GP

dataset : https://crypko.ai/#/

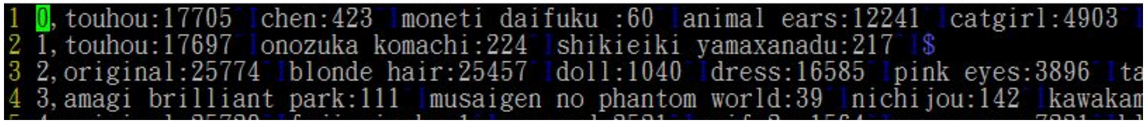
Submission & Grading

Data & format

- Anime Dataset
  - ○ training data: 33.4k (image, tags) pair
  - ○ **faces/**, tags.csv, sample_testing_text.txt

blue eyes
red hair
short hair

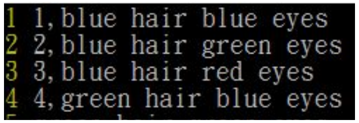
- training tags file format
  - ○ img_id <comma> tag1 <colon> #_post <tab> tag2 <colon> #_post
  - ○ tags.csv is **not** used in HW3-1



```
0,touhou:17705	chen:423	moneti daifuku :60	animal ears:12241	catgirl:4903
1,touhou:17697	onozuka komachi:224	shikieiki yamaxanadu:217	$
2,original:25774	blonde hair:25457	doll:1040	dress:16585	pink eyes:3896	ta
3,amagi brilliant park:111	musaigen no phantom world:39	nichijou:142	kawakam
```

tags.csv

- testing text file format
  - ○ testing_text_id <comma> testing_text
  - ○ testing text only includes **'color hair'** and **'color eyes'**, only alphabetic char involved.
  - ○ sample_testing_text.txt is **not** used in HW3-1



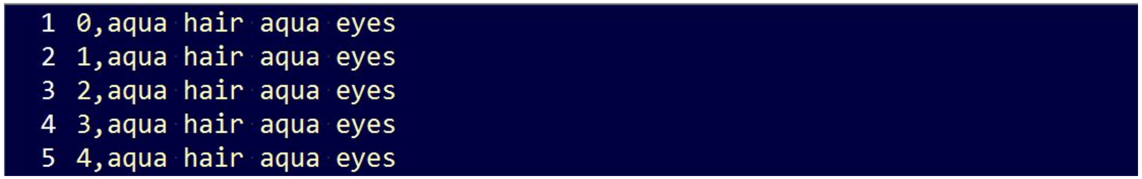
```
1,blue hair blue eyes
2,blue hair green eyes
3,blue hair red eyes
4,green hair blue eyes
```

sample testing_text.txt

Data & format

- Extra data
  - training data: 36.7k (image, tags) pair
  - **images/**, tags.csv
- training tags file format
  - img_id <comma> hair tag <space> eyes tag
  - tags in extra data only includes ‘color hair’ and ‘color eyes’
  - tags.csv is **not** used in HW3-1

black eyes
red hair



```
0,aqua hair aqua eyes
1,aqua hair aqua eyes
2,aqua hair aqua eyes
3,aqua hair aqua eyes
4,aqua hair aqua eyes
```

tags.csv

Data Link

- Anime Dataset
- Extra Data
- Extra Data2 (no tag)

HW3 Grading Policy:

- HW3-1 Code (image generation) : 5%
- HW3-2 Code (text-to-image generation): 5%
- Report : 15%
- HW3-3 (Bonus, style transfer): 2%
- 分工表:0.5%

HW3-1 Report Questions

- Model Description
  - Describe the models you use to, including the model architecture, objective function for G and D. (1%)
- Experiment settings and observation
  - Show generated images (1%)
- Compare your model with WGAN, WGAN-GP, LSGAN (choose 1)
  - Model Description of the choosed model (1%)
  - Result of the model (1%)
  - Comparison Analysis (1%)
- Training tips for improvement (4%)
- Mode Collapse (2%)

Training Tips for improvement

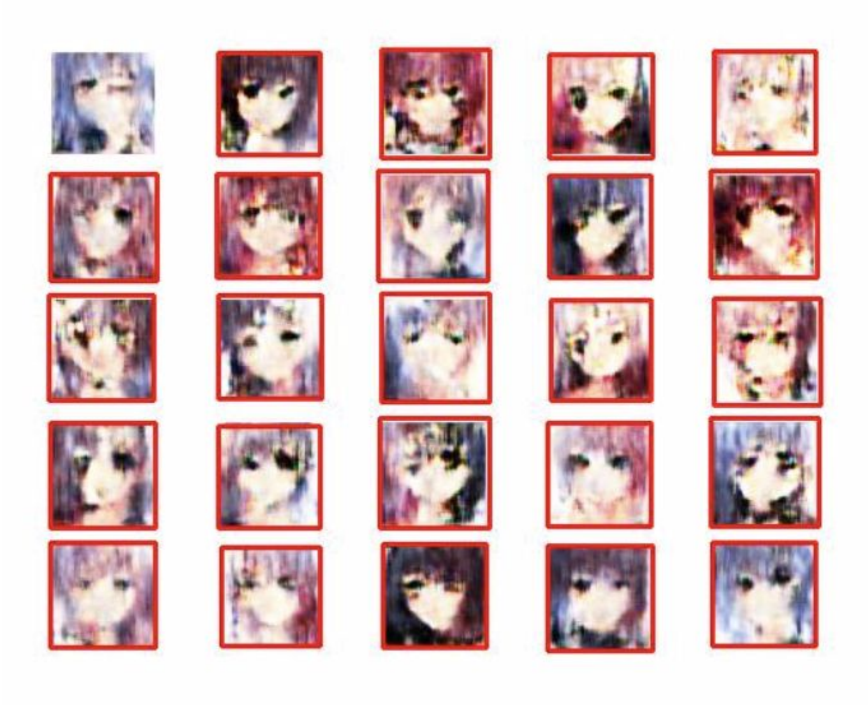
- Pick three tips in the following website
  - https://github.com/soumith/ganhacks
  - Please implement these tips on image generation
- Total : 3%, 1% for each
  - Which tip & implement details (0.5%)
  - Result (image or loss...etc.) and Analysis (0.5%)
- Only the following tips are accepted
  - 1, 2, 3, 4, 5, 6, 9, 13, 14, 17

Mode Collapse

- First, train a model that leads to mode collapse and record its model architecture as well as hyperparameters (especially iterations and lr)
- Then, find ways to alleviate mode collapse phenomenon without modifying model architecture, lr and iterations

HW3-1 Code Grading

- Reproduce Score : 2%
- Baseline Score : 2%
- TA Review : 1% (**mode collapse**)



Output Format Requirement

- The generated images should be in Directory **samples/**
  - **請大家繳交時就將產生的結果傳到samples/, 助教利用script reproduce時也請同學將結果輸出到這個資料夾。為保證reproduce結果相同, 請同學將random的部份固定**
  - **批改作業會在azure上, 請同學繳交前在機台上檢查**
  - 已經在github裡的image -> samples/gan_original.png
  - run_gan.sh -> samples/gan.png
- Each generated image must be resized to **64 x 64**
- Generate 25 image into one png
  - sample code is in baseline.py
  - 為防止同學產生的圖片不一致, 請同學使用baseline.py裡的**save_imgs()**

Baseline Model

- Anime Face Recognition
- Github for baseline.py
- How to run:
  - pip install opencv-python
  - Download pre-trained model in Here.
  - python baseline.py --input <input_image>
- Generate 25 images in one png
  - if faces > 20: pass

Recommened Packages

- Python 3.6
- **Tensorflow**
- **PyTorch**
- Keras
- MXNet
- matplotlib
- skimage
- Python Standard Library

Submission on Github

- Only one branch **master** is needed
- Only **generator** and **inference** mode is needed
- Remember to put your **pre-trained models or download scripts** so that we can run your code successfully

Q&A

b05902013@csie.ntu.edu.tw
b05902127@csie.ntu.edu.tw