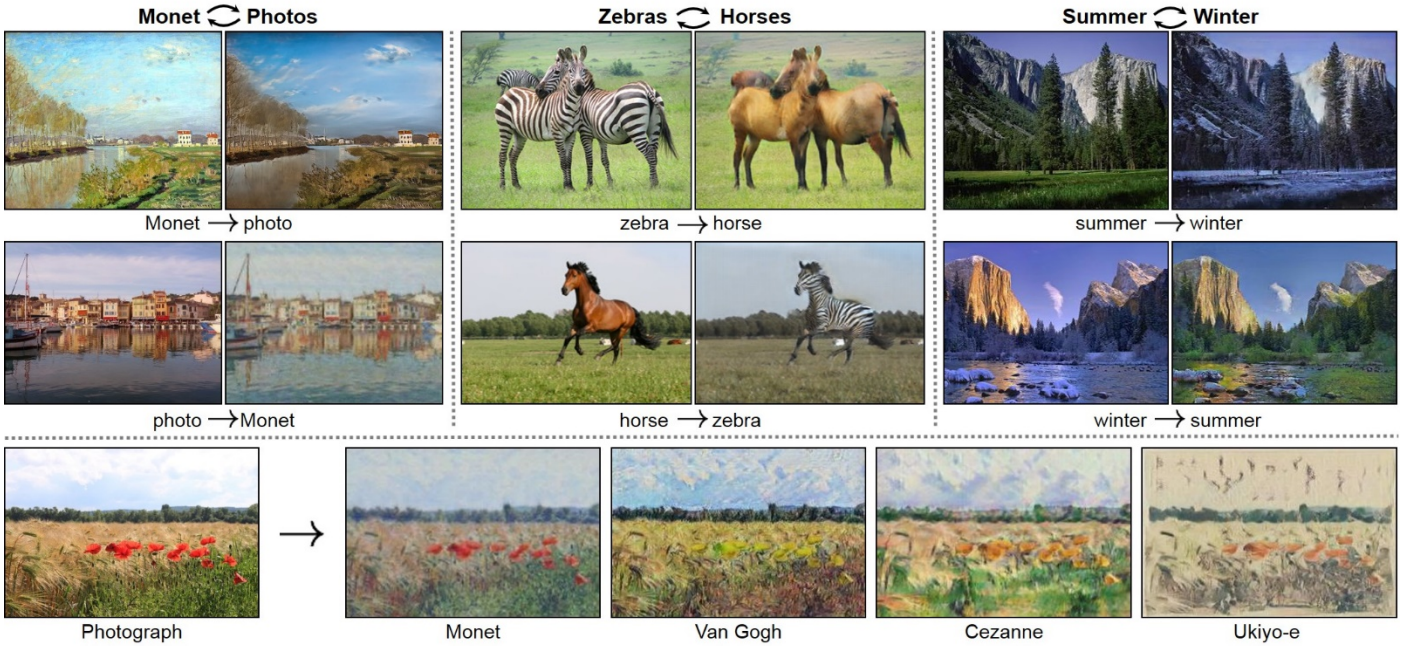
EE898 Programming Assignment 2

CycleGAN : Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks

Minjun Kang (kmmj2005 [at] kaist.ac.kr)

Programming Assignment 2

- In PA2, you will implement CYCLE GAN that translates image to image.
- You have to implement the missing parts in the given code. The missing part is depicted as "Your Implementation" in skeleton code.



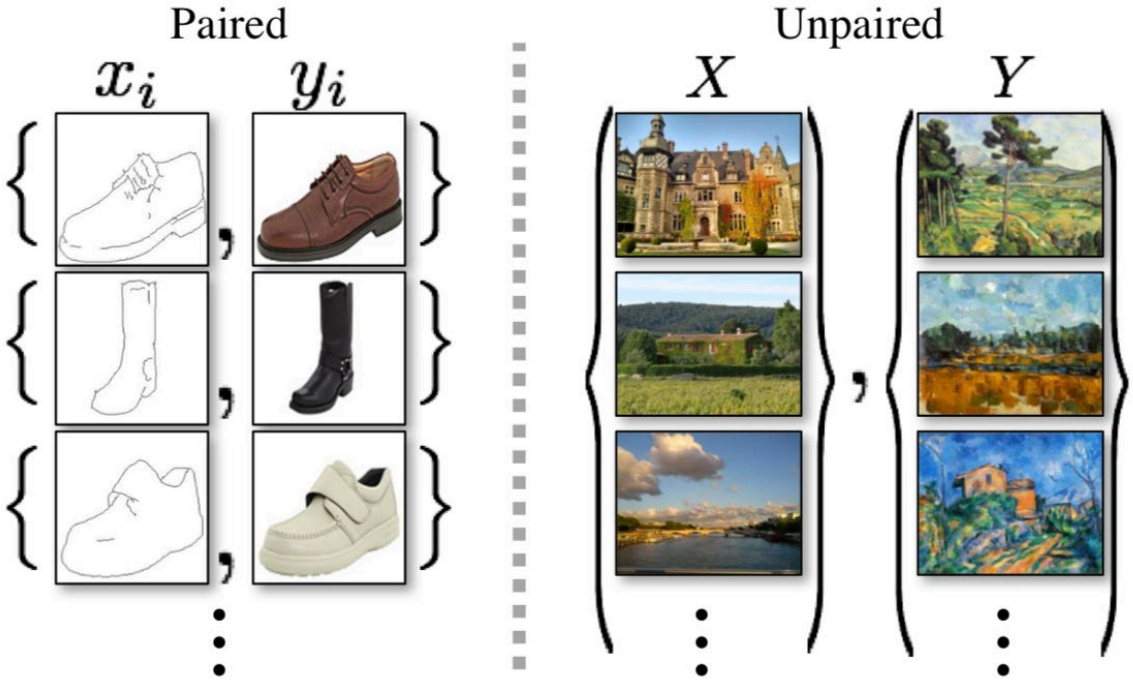
CycleGAN : Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks (ICCV 2017)

CycleGAN

[1] CycleGAN : Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks (ICCV 2017)

CycleGAN

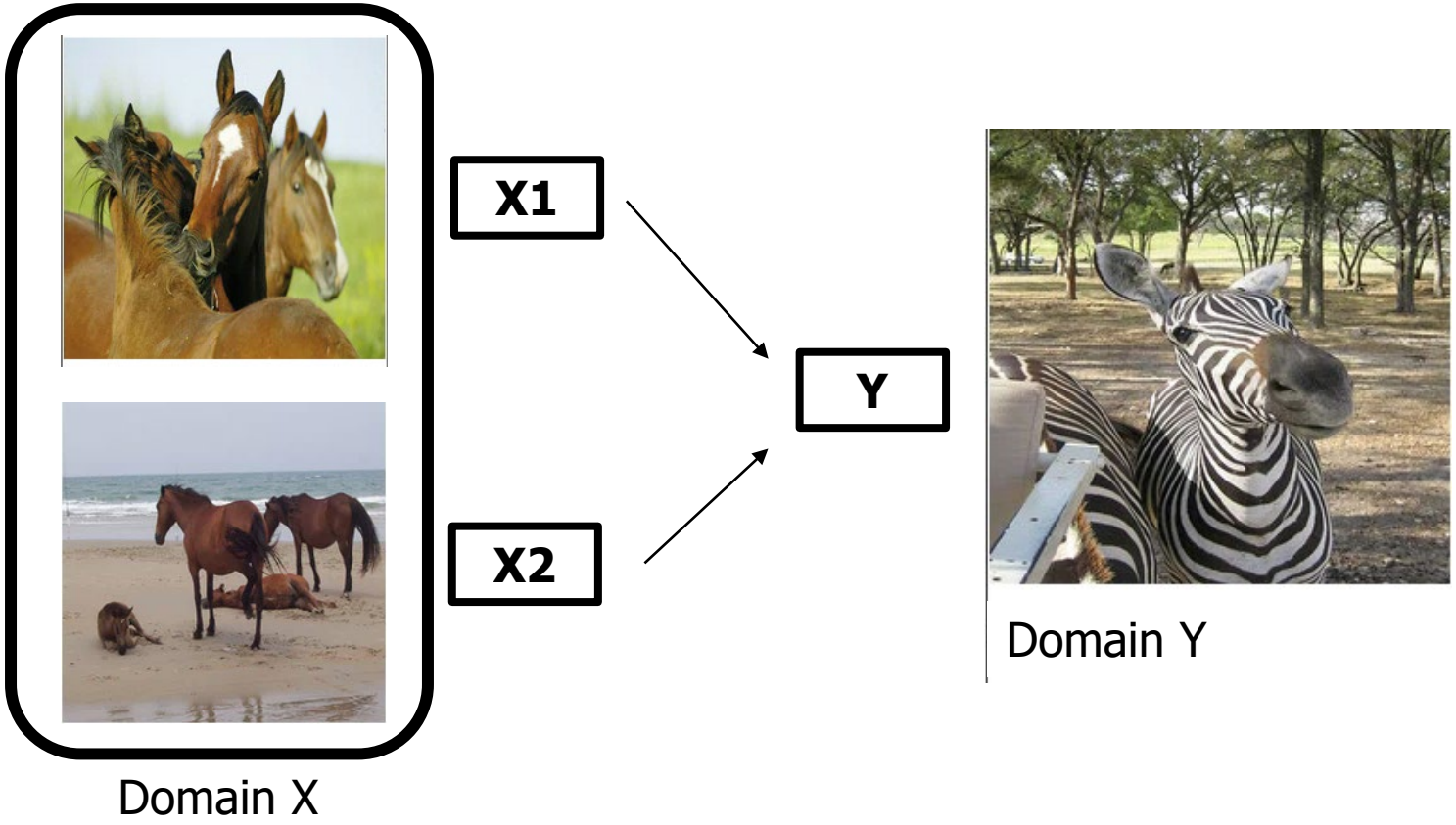
Unpaired Dataset



CycleGAN used unpaired dataset for domain translation.
These unpaired datasets are rather easy to be obtained.

CycleGAN

Pix2pix, mode collapse



In unpaired dataset, two different images with the same style but have different contents can be translated to the target style with same contents. There is no constraint for generator to avoid this.

CycleGAN

Cycle Consistency Loss (Forward Cycle)

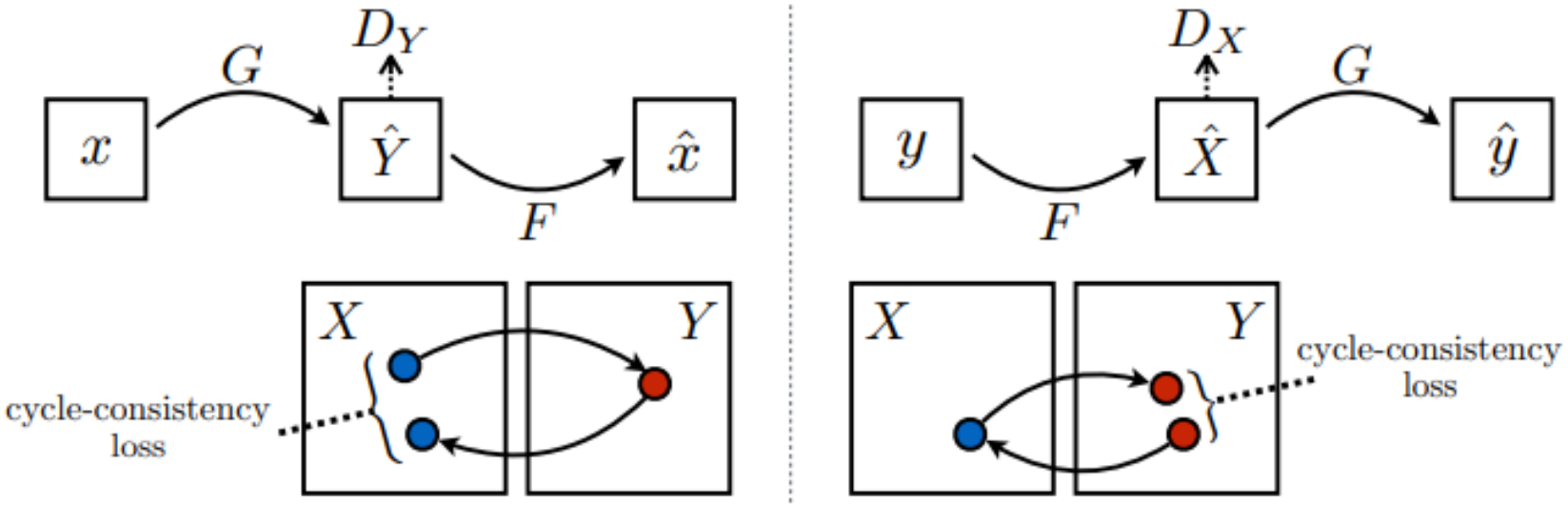
$$L_{cyc}(G,F)=E_{x\sim p_{data}(x)}[\|F(G(x))-x\|_1]$$

To prevent mode collapse in image to image translation, cycle consistency loss restrains the generator to produce the image with the same content as input with the target domain's style.

CycleGAN

Cycle Consistency Loss (Forward and Backward)

$$L_{cyc}(G,F)=E_{x\sim p_{data}(x)}[\|F(G(x))-x\|_1]+E_{y\sim p_{data}(y)}[\|G(F(x))-y\|_1]$$

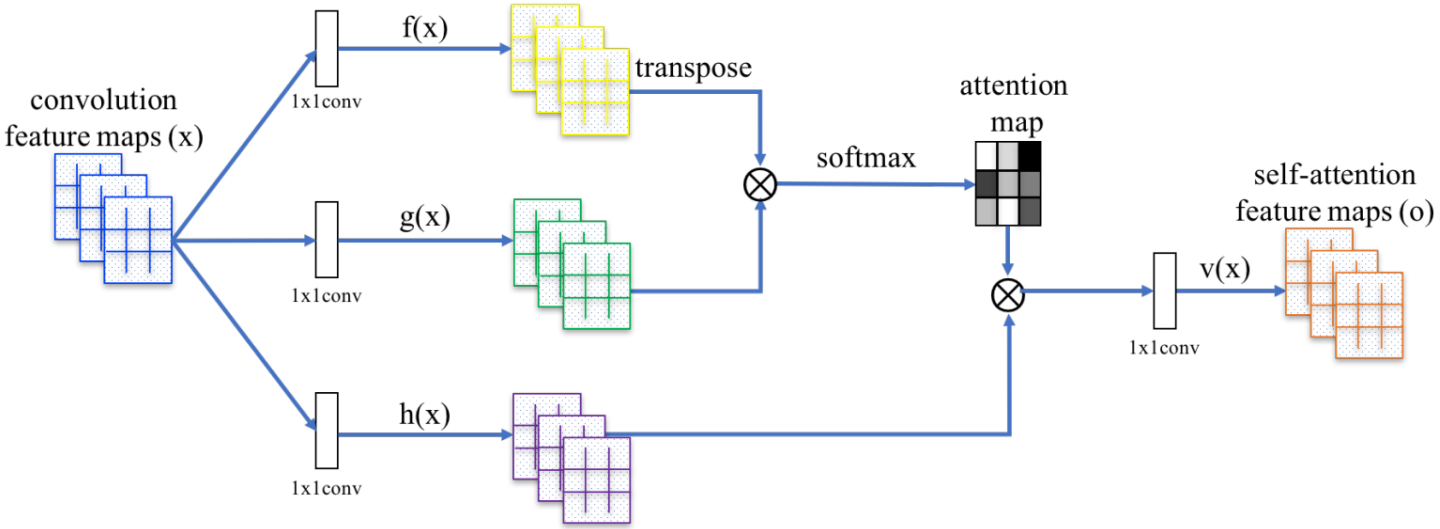


There can be two different cycles between source domain(X) and target domain(Y). The first case indicates the forward cycle consistency and the second case shows the backward cycle consistency.

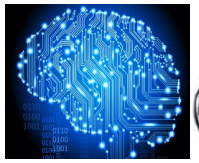
Self Attention GAN

[1] Self-Attention Generative Adversarial Networks (ICML 2019)

Self Attention GAN



With self attention layer, the generator can draw images in which fine details at every location are carefully coordinated with fine details in distant portions of the image.



Metric for GAN

[1] Improved Techniques for Training GANs (NIPS 2016)

[2] GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium (NIPS 2017)

Metric for GAN

Inception Score(IS)

- Apply Inception model to every generated image to get conditional label distribution $p(y|x)$.
- Images that contain meaningful objects should have $p(y|x)$ with low entropy
- Generator needs to generate varied images with latent vectors z, so the marginal $\int p(y|x = G(z))dz$ should have high entropy.
- Combining the above two properties, metric of Inception Score is defined as

$$\mathrm{Exp}(\mathrm{E}_{x}\mathrm{KL}(p(y|x)||p(y)))$$

[1] Improved Techniques for Training GANs (NIPS 2016)

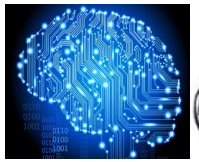
Metric for GAN

Fréchet Inception Distance(FID)

- Measure the distance of real data and fake data in feature space by using pretrained network on IMAGENET. IS score doesn't consider the relationship between real and fake data.
- After extracting feature of real and fake data, calculate mean and variance of the gaussian distribution of features.
- With mean($\mathrm{m_r}$, $\mathrm{m_f}$) and variance($\mathrm{C_r}$, $\mathrm{C_f}$) of real and fake data's feature, FID is defined as

$$FID^2 = ||m_f - m_r||_2^2 + \mathrm{Tr}(C_f + C_r - 2(C_f C_r)^{1/2})$$

[2] GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium (NIPS 2017)



CycleGAN for PA2

Dependency

- Requirements
  - Linux with Python ≥ 3.6 (Tested with Ubuntu 18.04, Python=3.6.10)
  - PyTorch ≥ 0.4
  - torchvision that matches the PyTorch installation.
  - Requirements.txt is provided with skeleton code.

  (Required Version number might be different with different environment)

- Skeleton code is provided for quick start

Unpaired Dataset for CycleGAN

- Dataset Provided
  - ae_photos
  - apple2orange
  - summer2winter_Yosemite
  - horse2zebra
  - monet2photo
  - cezanne2photo
  - ukiyoe2photo
  - vangogh2photo
  - maps
  - cityscapes
  - facades
  - iphone2dslr_flower
- You can download and use the above datasets by

  bash ./datasets/download_cyclegan_dataset.sh [dataset_name]

CycleGAN setting

- All the control parameters are in options/base_options.py, options/train_options.py.
- You can change any variables in options except size of input image(--crop_size) but be aware to use the same --batch_size, --epochs and parameters related to learning rate schedule for fair comparison.
- If you want, you can add additional parameters in options.

CycleGAN train/test

- If you implement all the details of CycleGAN in given skeleton code, you can train and test your model with the following code in bash
- Train

```
Python train.py --dataroot ./datasets/[dataset_name] --name [workspace name] --model cycle_gan
```

- Test

```
Python test.py --dataroot ./datasets/[dataset_name] --name [workspace name] --model cycle_gan --num_test 250
```

- If you finished training your model, checkpoint file and some sample images (results) will be in your checkpoint/[workspace name] directory.
- If you run the test code with trained model, results will be in your results/[workspace name].
- You can simply add other control parameters in 'base_options.py' and 'train_options.py' in options directory by simply adding --[control parameter] value
- You have to complete Step 1-1 and 1-2 to run the code.

Training CycleGAN

- Default hyperparameters related to training schedules are set assuming that we train the model with 4-GPUs(GTX 1080 TI), with total batch size 8. We recommend to change batch size if you have 2-GPUs.

- Training time for naïve CycleGAN model takes about 12 ~ 15 hours for maps dataset and 18 ~ 20 hours for horse2zebra dataset. (within a day) Please start early.

- If you need to change the total batch size by some reasons (i.e., #GPUs, #imgs/GPU), you should modify the initial learning rate value and total training iterations (including what iterations to down-scale the learning rate), following the **linear scaling rule** specified in [1].

[1] P. Goyal et al., Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour, arXiv'17

Step 1-1 CycleGAN Design

models.networks.py

- Tips
  - You can implement any generator/discriminator models that you want to use as baseline.
  - 'PatchGAN' in Pix2pix[1] is good starting point.

[1] Image-to-Image Translation with Conditional Adversarial Nets, CVPR 2017

Step 1-2 CycleGAN Loss

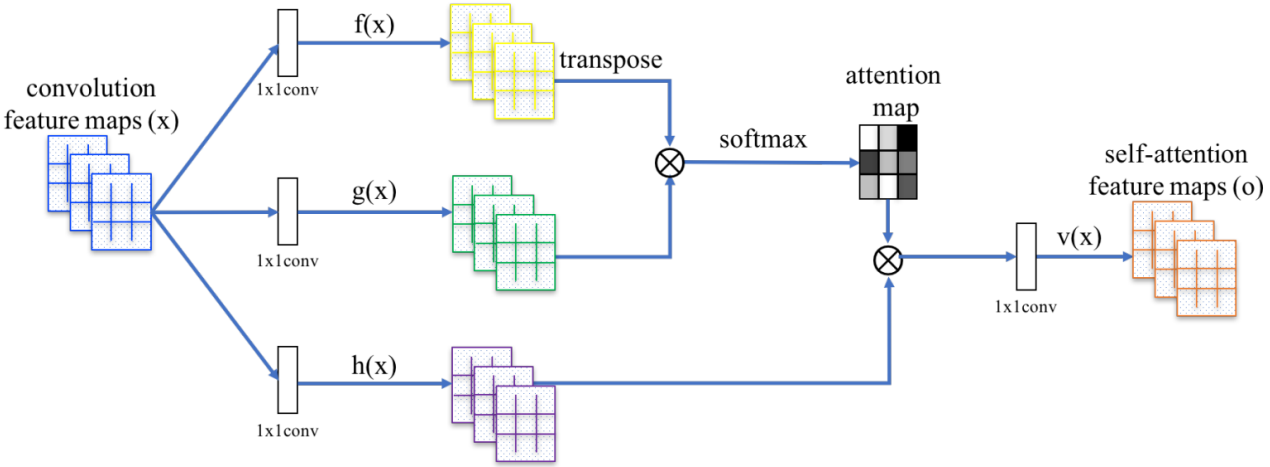
models.cycle_gan_model.py

- Tips
  - Implement LSGAN loss.
  - Implement forward and backward cycle loss.
  - Train generator first and then train discriminator iteratively.
  - You have to complete this Step 1-1 and 1-2 to run the code.

Step 2. Self attention for CycleGAN

models.networks.py

- Design new layer for Self Attention module.
- Check out the paper **Self-Attention Generative Adversarial Networks, ICML 2019.**
- Design new generator and discriminator with self attention module.
- Discuss the influence of self attention module by using attention map.



Step 3. FID Score for comparison

metric.metric.py

- **Use Inception V3 network (Given in skeleton code)** for FID score implementation.

- Check out the paper **GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium, NIPS 2017** for FID score implementation.

- Please report your FID score of test datasets (horse2zebra and maps) for Step1 and Step2 model. Report two FID scores of features from (between target like source and target) and (source like target and source).

- Discuss the relationship between the qualitative results and FID score.

$$FID^2 = ||m_f - m_r||_2^2 + \mathrm{Tr}(C_f + C_r - 2(C_f C_r)^{1/2})$$

for mean and variance of real and fake data's feature, $(\mathrm{m_r}, \mathrm{C_r})$ and $(\mathrm{m_f}, \mathrm{C_f})$

Step 3. FID Score for comparison

metric.metric.py

At num_test = 50

MAPS DATASET (8 batch, 250 epoch)	CycleGAN (GITHUB Baseline)	+Self Attention module
FID score (target like source and target)	74	70
FID score (source like target and source)	103	102

Your CycleGAN's FID score can be different from this reported FID score depending on your implementation of generator and discriminator.

Step 4. Extra Credit

- Design your own generator and discriminator that has better FID score than the baseline(Step 1 and 2). (Lower the better)
- Creative idea and intuition will be highly graded.
- Ablation study and logical explanation for your model is recommended.
- You can use any (at least) **two** different datasets to compare with the Step 1 and 2.

Cautions and Comment

- There will be partial points on every steps, so try to do your best.
- Collaboration with other students is prohibited. Please do it yourself and comment on your code in order to show your understanding.
- **Do not copy and paste from github code.** You will be strongly punished for plagiarism.

PA2 Submission

- **Due**: **June 25th 11:59PM** (No deadline extension)
- **To**: Minjun Kang (kmmj2005 [at] kaist.ac.kr)
- **Submission should include**:
  - Report (`.docx` or `.pdf`)
  - Source code (do not include dataset folder)
  - Only one `.pth`checkpoint that yields the best FID score (lowest).
  - Zip your all includings named as `PA2_{StudentID}_{NAME}.zip`, i.e., `PA2_20191111_MinjunKang.zip`.
- **The report should include**:
  - Your understanding of each step of implementation.
  - Visualization of attention map from the Step 2 model.
  - FID score table for the baseline model, Step 2 model trained with "maps" dataset and "horse2zebra" dataset.
  - Qualitative results of all the implemented models. (Choose any two different datasets)
  - (Extra) Implement your own generator/discriminator that has lower FID score and report FID score with qualitative results. (Choose any two different datasets)
  - (Extra) Ablation Study and detailed explanation.

Revision History

Revision	Date	Description
0	2020-06-10	Release
1	2020-06-11	15pg dataset name is revised --num_test is set to 250, revised in slide 17.