

Homework #3

Deep Learning for Computer Vision

NTU, Fall 2019

108/11/19

Outline

- Task Description
- Dataset
- Grading
- Rules

Task Description

- Image Generation & Feature Disentanglement
 - Problem 1: Generative Adversarial Network (GAN)
 - Problem 2: Auxiliary Classifier GAN (ACGAN)
- Unsupervised Domain Adaptation (UDA)
 - Problem 3: Domain-Adversarial Training of Neural Networks (DANN)
 - Problem 4: Improved UDA Model

Image Generation - GAN

- Architecture of GAN

- Loss: $\mathcal{L}_{GAN}(G, D) = \mathbb{E}[\log(1 - D(G(x)))] + \mathbb{E}[\log D(y)]$

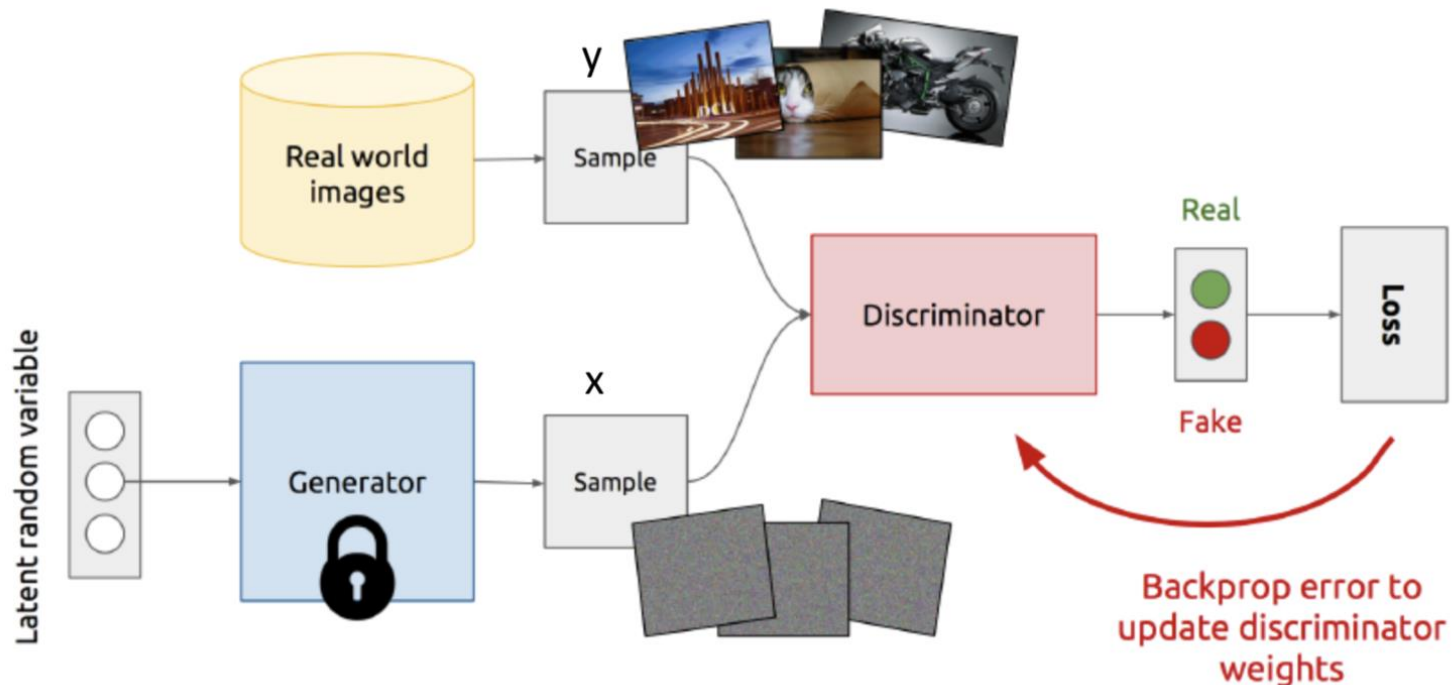
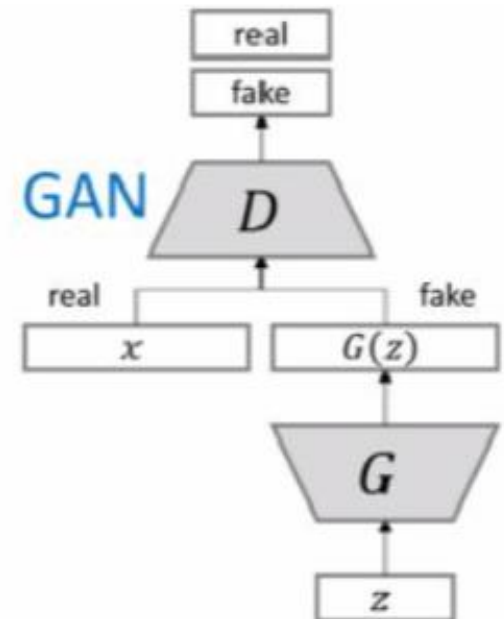


Image Generation - GAN

- Discriminator
 - Classify if an image is real/fake.
- Generator
 - Fool the discriminator on image reality.



Feature Disentanglement - ACGAN

- **Supervised** feature disentanglement

- **Learning**

- **Overall objective function**

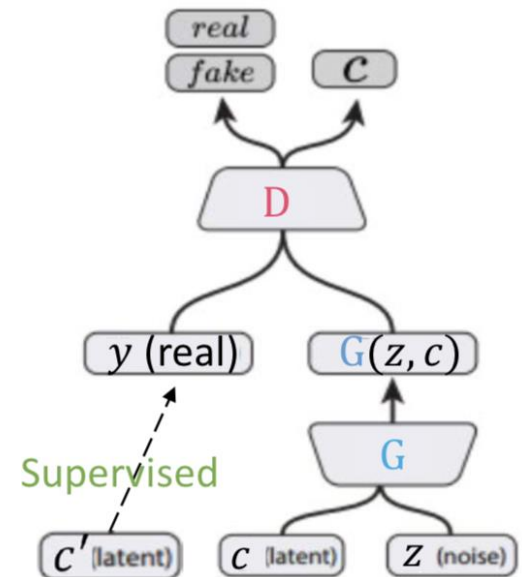
$$G^* = \arg \min_G \max_D \mathcal{L}_{GAN}(\mathbf{G}, \mathbf{D}) + \mathcal{L}_{cls}(\mathbf{G}, \mathbf{D})$$

- **Adversarial Loss**

$$\mathcal{L}_{GAN}(\mathbf{G}, \mathbf{D}) = \mathbb{E}[\log(1 - D(\mathbf{G}(z, c)))] + \mathbb{E}[\log \mathbf{D}(y)]$$

- **Disentanglement loss**

$$\mathcal{L}_{cls}(\mathbf{G}, \mathbf{D}) = \underbrace{\mathbb{E}[-\log D_{cls}(c'|y)]}_{\text{Real data w.r.t. its domain label}} + \underbrace{\mathbb{E}[-\log D_{cls}(c|G(x, c))]}_{\text{Generated data w.r.t. assigned label}}$$



Feature Disentanglement - ACGAN

- Apply auxiliary classifier to disentangle specific attribute.

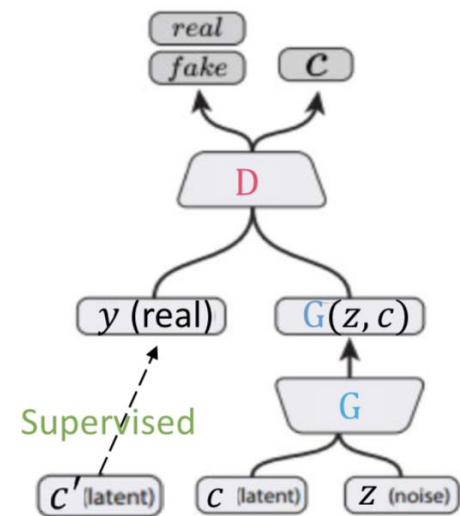
- Discriminator

- Classify if an image is real/fake.
 - Classify attribute of image (minimize cross-entropy)

⇒ They should be jointly trained!

- Generator

- Fool the discriminator on image reality.
 - Satisfy the discriminator by preserving the inputting attribute.



GAN Training Tips

- Architecture Guideline proposed by Radford et al. in [DCGAN](#)

Architecture guidelines for stable Deep Convolutional GANs

- Replace any pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).
- Use batchnorm in both the generator and the discriminator.
- Remove fully connected hidden layers for deeper architectures.
- Use ReLU activation in generator for all layers except for the output, which uses Tanh.
- Use LeakyReLU activation in the discriminator for all layers.

Well known [tips and tricks](#) published on GitHub.

It is suggested that you take a look before you start training.

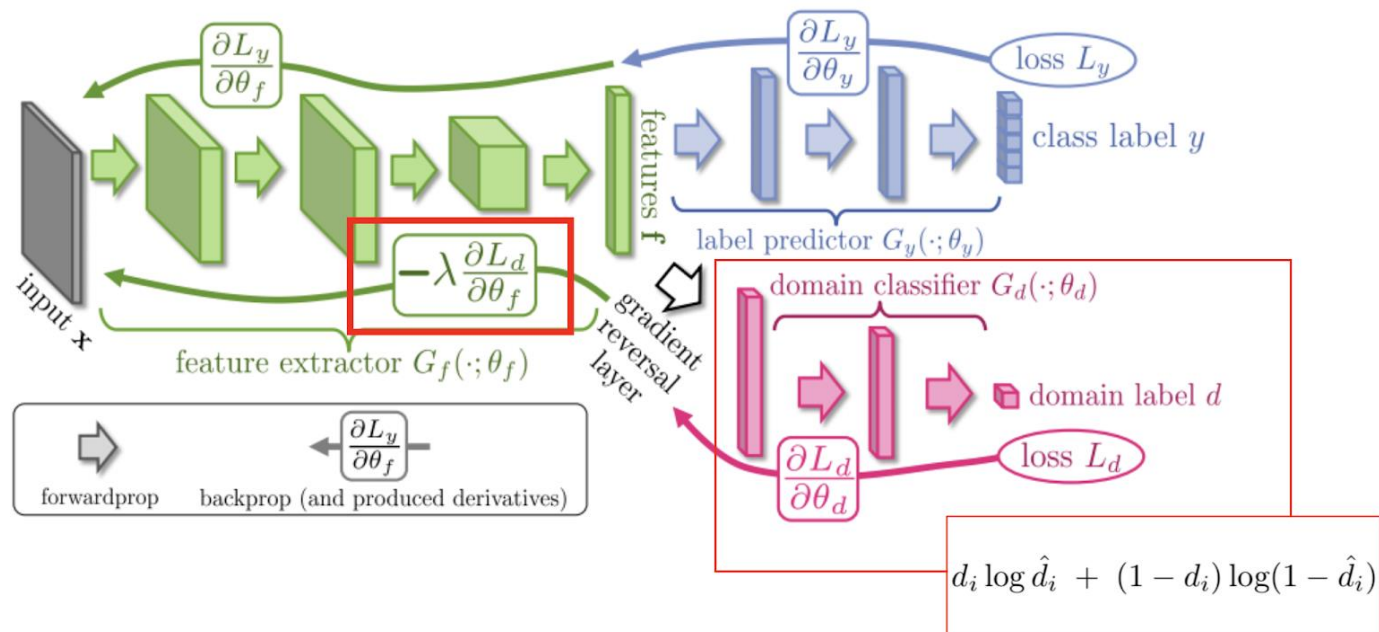
TA's experience and hints

- Surveying related papers and using similar architectures may help.
- Trace the accuracy of discriminator network to see if G_{net} and D_{net} performance matches.
- Improved GAN algorithm is harder to implement but easier to train (e.g. WGAN, WGAN-GP)
- Use your VAE's decoder/encoder as the generator/discriminator prototype.

GAN is often difficult to train and tune, starting this part early may help a lot.

Unsupervised Domain Adaptation - DANN

- Domain Classifier
 - Maximize domain classification loss to confuse the feature between *source* and *target*
- Label Predictor
 - Minimize source-domain data classification loss



Outline

- Task Description
- **Dataset**
- Grading
- Rules

Dataset — Generation & Disentanglement

A subset of human face dataset [CelebA](#)

Images are cropped and downscaled to 64x64

40000 training samples (about 21% of complete CelebA)

13 out of 40 attributes are provided for feature disentanglement experiments

Dataset — Generation & Disentanglement

Format

data/

— face/

— train/

— **train.csv**

40000 images for training (00000.png ~ 39999.png)

training file attributes

— digits/

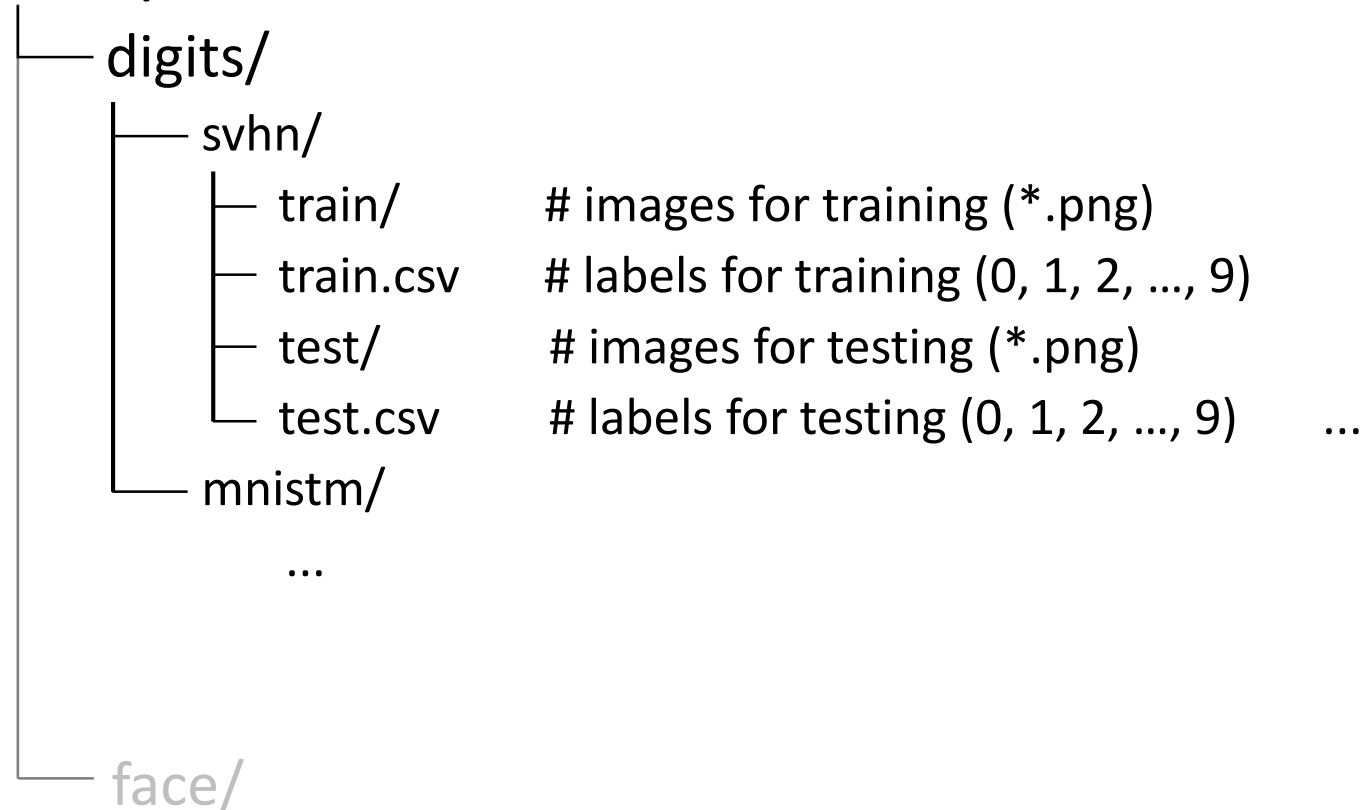
Header in first row, <img_filename>, <attribut_1>, <attribut_2>, ..., <attribut_13> afterwards

```
image_name,Bangs,Big_Lips,Black_Hair,Blond_Hair,Brown_Hair,Heavy_Ma
keup,High_Cheekbones,Male,Mouth_Slightly_Open,Smiling,Straight_Hair
,Wavy_Hair,Wearing_Lipstick
00000.png,0.0,0.0,0.0,0.0,0.0,1.0,0.0,1.0,0.0,1.0,1.0,0.0,0.0,0.0
00001.png,0.0,0.0,0.0,0.0,0.0,0.0,1.0,1.0,0.0,0.0,0.0,0.0,1.0,1.0
00002.png,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,1.0,1.0,0.0,0.0,0.0,0.0
00003.png,0.0,0.0,1.0,0.0,0.0,0.0,1.0,1.0,1.0,1.0,1.0,0.0,0.0,0.0
...
```

Dataset — Unsupervised Domain Adaptation

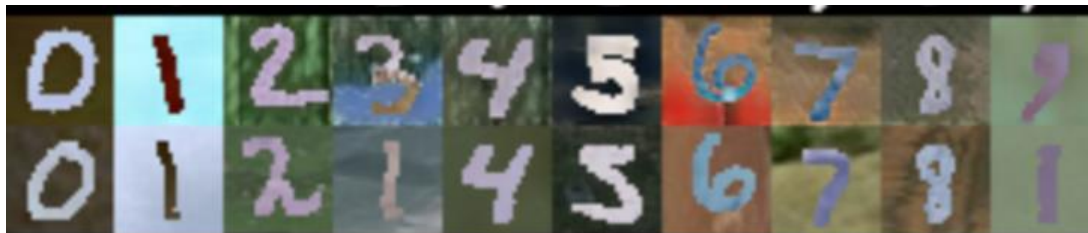
Format

data/



Dataset — Unsupervised Domain Adaptation

- MNIST-M Dataset
 - # of data: 60,000 / 10,000 (testing)
 - Generated from MNIST
 - It is a subset of a larger set available from MNIST. The digits have been size-normalized and centered in a fixed-size image: **28 x 28 x 3** pixels.
 - Total number of class: **10** (0~9)



Dataset — Unsupervised Domain Adaptation

- SVHN Dataset

- # of data: 73,257 / 26,032 (testing)
- Real-world image dataset for developing machine learning.
- **10** classes
- MNIST-like (size: **28 * 28 * 3**) images centered around a single character

You need to resize the above
datasets to **28 * 28 * 3** !



Outline

- Task Description
- Dataset
- **Grading**
- Rules


Grading – Overview

- **Problem 1:** GAN (20%)
- **Problem 2:** ACGAN (20%)
- **Problem 3:** DANN (35%)
- **Problem 4:** Improved UDA model (35%)

Grading – Problem 1

Problem 1: GAN (20%)

1. Describe the architecture & implementation details of your model. (5%)
2. Plot 32 random images generated from your model. [fig1_2.jpg] (10%)
3. Discuss what you've observed and learned from implementing GAN. (5%)


 You should **fix the random seed** in your program such that the [fig1_2.jpg] produced by your script is exactly the same as the one in your report.

Grading – Problem 2

Problem 2: ACGAN (20%)

For this problem, you should pick “**Smiling**” attribute provided in **face/train.csv** to train your model (e.g. *smiling*). The remaining 12 attributes that are not chosen can be ignored during training.

1. Describe the architecture & implementation details of your model. (5%)
2. Plot 10 random **pairs** of generated images from your model, where each **pair** should be generated from the same random vector input but with opposite attribute. This is to demonstrate your model’s ability to disentangle features of interest. [fig2_2.jpg] (10%)
3. Discuss what you’ve observed and learned from implementing ACGAN. (5%)

 You should **fix the random seed** in your program such that the [fig2_2.jpg] produced by your script is exactly the same as the one in your report.

Grading – Problem 3

Problem 3: DANN (35%)

In this problem, you need to implement DANN and consider the following 2 scenarios: (Source domain → Target domain)

(1) MNIST-M → SVHN, (2) SVHN → MNIST-M

1. Compute the **accuracy** on **target** domain, while the model is trained on **source** domain only. (lower bound) (3%)
2. Compute the **accuracy** on **target** domain, while the model is trained on **source and target** domain. (domain adaptation) (3+7%)
3. Compute the **accuracy** on **target** domain, while the model is trained on **target** domain only. (upper bound) (3%)
4. Visualize the latent space by mapping the *testing* images to 2D space (with t-SNE) and use different colors to indicate data of (a) different digit classes 0-9 and (b) different domains (source/target). (6%)
5. Describe the architecture & implementation detail of your model. (6%)
6. Discuss what you've observed and learned from implementing DANN. (7%)

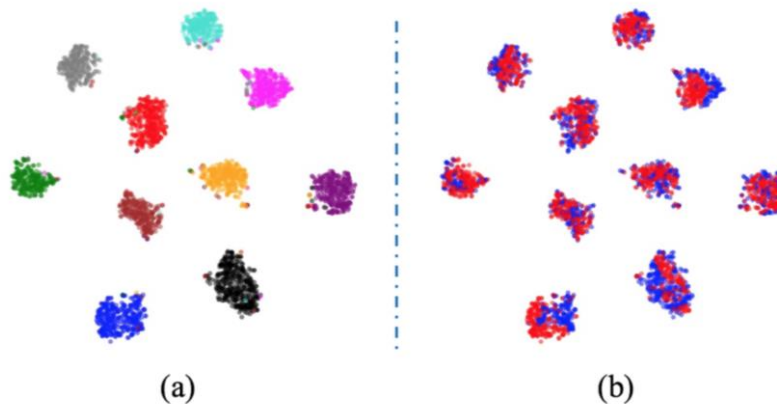
Grading – Problem 3 & 4

- Expected results on report:

Accuracy: e.g.,

	SVHN \rightarrow MNIST-M	MNIST-M \rightarrow SVHN
Trained on source	(lower bound)	
Adaptation (DANN/Improved)		
Trained on target	(upper bound)	

t-SNE: e.g.,



Grading – Problem 4

Problem 4: Improved UDA model (35%)

In this problem, you are asked to implement an improved model and consider the following 2 scenarios: (Source domain → Target domain)

(1) MNIST-M → SVHN, (2) SVHN → USPS

1. Compute the **accuracy** on **target** domain, while the model is trained on **source and target** domain. (domain adaptation) (6+10%)
2. Visualize the the latent space by mapping the *testing* images to 2D space (with t-SNE) and use different colors to indicate data of (a) different digits classes 0-9 and (b) different domains (source/target). (6%)
3. Describe the architecture & implementation detail of your model. (6%)
4. Discuss what you've observed and learned from implementing your improved UDA model. (7%)

Grading – Problem 4

Related UDA papers

- [Adversarial Discriminative Domain Adaptation](#). CVPR 2017
- [Domain Separation Network](#). NIPS 2016
- [Unsupervised Pixel-Level Domain Adaptation with Generative Adversarial Networks](#). CVPR 2017
- [Generate To Adapt: Aligning Domains using Generative Adversarial Networks](#). CVPR 2018
- [From source to target and back: Symmetric Bi-Directional Adaptive GAN](#). CVPR 2018

You may select any one of the papers above, or search for other related papers, or design a method by yourself to implement your improved model.

Grading – Problem 3 & 4

- If your accuracy of problem 3-2 does not pass the baseline accuracy, you only get 3 points in this sub-problem.
- Baseline score for DANN:
 - SVHN → MNIST-M: **40%**
 - MNIST-M → SVHN: **40%**
- Your accuracy of problem 4-1 should **surpass** the accuracy reported in problem 3-2. If not, you only get 6 points in this sub-problem.

Outline

- Task Description
- Dataset
- Grading
- **Rules**

Rules – Deadline & Policy

- Report and source code deadline : **108/12/03 (Tue.) 03:00 AM (GMT+8)**
- Late policy : Shown on Github.
- **Academic Honesty:** Plagiarism/cheating is strictly prohibited and against school regulations. If any form of plagiarism/cheating is discovered, all students involved would receive an F score for the course (which is NOT negotiable).
- Self-learning by researching on the Internet or discussing with fellow classmates is highly encouraged. However, you must obtain and write the final solution by yourself. Please specify, if any, the references for each of your answers (e.g., the name and student ID of your collaborators and/or the Internet URL you consult with) in your report. If you complete the assignment all by yourself, you must also specify “no collaborators”.
- We will adopt relative grading for Problems 1 & 2. That is, your scores will depend on the quality of your report compared to others’.
- Using external dataset is forbidden for this homework.
- Violation of any format/execution specification will result in zero points for the corresponding problem. Please follow the homework specs carefully.

Rules – Submission

- Click the following link and sign in to your GitHub account to get your submission repository:

https://classroom.github.com/a/rHar_GWe

- After getting your GitHub repository (which should be named "hw3-
<username>"), be sure to **read the README carefully** before starting your work.
- By default, we will only grade your last submission *before the deadline* (**NOT** your *last submission*). Please e-mail the TAs if you'd like to submit another version of your repository.

Rules – Submission

- Your GitHub repository should include the following files:
 - hw3_<studentID>.pdf
 - hw3_p1p2.sh (for Problems 1 & 2)
 - hw3_p3.sh (for Problems 3)
 - hw3_p4.sh (for Problems 4)
 - your python files (train.py and others)
 - your model files (can be loaded by your python file)
- **For more information, please check the README.md in your repository.**
- **Don't upload your dataset.**
- If any of the file format is wrong, you will get zero point.
- If your model is larger than GitHub's maximum capacity (100MB), you can upload your model to another cloud service (e.g., Dropbox). However, your script file should be able to download the model automatically. (Dropbox tutorial: [link](#))

Rules - Packages

- python3.6
- pytorch==1.2.0
- scipy==1.2.0
- tensorboardX==1.8
- torchvision==0.2.1
- Other python3.6 standard library
- For more details, please refer to the requirements.txt in your homework package.
- **E-mail or ask TA first if you want to import other packages.**

Q & A

- If you have any question, you can:
 - Use TA hours (please check [course website](#) for time/location)
 - Contact TAs by e-mail (ntudlcvta2019@gmail.com)
 - Useful website: [link](#)
 - **DO NOT** directly message TAs (we will ignore any direct message)