Spring 2023 CS 747 Deep Learning
Assignment 3

Due date: March 15, 2022, 11:59 pm

Announcement

- Honor Code: Everyone needs to submit their own solution individually for this assignment. When you submit your solution, it must be your own work. Any sources must be cited if it is not from you. Otherwise you may violate the GMU honor code and GMU CS honor code.
- Please note that this assignment requires using Pytorch, and it needs to run on GPUs. You are highly recommended to use Google Colab to finish this assignment, unless you have a local machine with powerful GPUs. Since the process of running one model is time-consuming, **please start the assignment early**.

1 Face Generation with a GAN (50 points)

In this problem, you will train a generative adversarial network (GAN) on the CelebA Dataset[1] and learn to generate face images. You will also implement spectral normalization[2] of discriminator weights to improve the quality of generator images (see Algorithm 1 in Appendix A of the linked paper). You will also get familiarised with one of the techniques to imporve the quality of images generated by GANs.

Data Setup Once you have downloaded the zip file, go to the Assignment folder and execute the CelebA download script provided:

```
$ cd CS747_Assignment3/
$ ./download_celeba.sh
```

The CelebA data provided to you is a preprocessed version which has been filtered using a simple face detector to obtain good images for generation. The images are also all cropped and resized to the same width and height.

[1] `http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html`
[2] `https://arxiv.org/pdf/1802.05957.pdf`

Implementation The top-level notebook (`MP3_P1.ipynb`) will guide you through the steps you need to take to implement and train a GAN. You will train two different models, the original GAN and LSGAN, which has a different loss function. The generator and discriminator network architectures you will implement are roughly based on DCGAN. You will also implement spectral normalization (`spectral_normalization.py`) which you will use in your GAN models (for details, see Algorithm 1 of this paper[3]).

We also provide with a notebook to help with debugging called `GAN_debugging.ipynb`. This notebook provides a small network you can use to train on MNIST. The small network trains very quickly so you can use it to verify that your loss functions and training code are correct.

You will need to use a GPU for training your GAN. We recommend using Colab to debug, but a Google Cloud machine once your debugging is finished as you will have to run the GAN for a few hours to train fully.

For reference, Figure 1 provide some samples of GAN (left) and LSGAN (right) output at various points during training.

2 Extra Credit for Problem 1 (bonus: 20 points)

- Change the network architecture and hyperparameters to train on the full 128x128 resolution CelebA data in the preprocessed dataset we provide rather than the 64x64 resized samples we train on currently.
- Implement one of other the recent GAN modifications like WGAN-GP [4] in PyTorch. Note that the original implementation of WGAN-GP paper was in TensorFlow.

3 Adversarial Attack (50 points)

In this problem, you will implement different adversarial attacks and defenses. To start out, we strongly encourage you to implement Fast Gradient Sign Method (FSGM) described in lecture. We recommend you go through the pytorch tutorial for fast gradient sign: tutorial [5].

3.1 Fast Gradient Sign Method

We want you to run FGSM on ImageNet. Rather than performing your attacks on the full imagenet dataset, which is $>$ 10G of data, we recommend running on Imagenette [6], which is a small subset of ImageNet that contains only 10 classes. The discription of the dataset can be found here [7]. You can

[3] `https://arxiv.org/pdf/1802.05957.pdf`
[4] `https://arxiv.org/pdf/1704.00028.pdf`
[5] `https://pytorch.org/tutorials/beginner/fgsm_tutorial.html`
[6] `https://s3.amazonaws.com/fast-ai-imageclas/imagenette2-320.tgz`
[7] `https://github.com/fastai/imagenette`

use any of the pretrained models [8] as the model under attack. The pretrained models are trained on the whole ImageNet dataset, which has 1000 labels. To attack specific labels in those models, just select the correct output of corresponding classes that appear in the Imagenette. You may find using this [9] helpful to select these classes.

3.2 Implement another attack methods

You are suggested to try one of the following methods:

- Iterative gradient sign
- Least likely class
- Any other adversarial methods

3.3 Implement a defense mechanism for your own attack

You are suggested to try one of the following methods:

- SafetyNet
- Feature denoising
- Input transformations
- Any other defense mechanisms

How you choose to show your work is up to you! Just be thorough with explanations and visualizations and we will be lenient with grading.

Submission Instructions

Please submit all your files on GMU Blackboard Portal before the due date.

- All of your code (python files and ipynb file) in a single ZIP file. The filename should be `netid_mp3_code.zip`.
- Your ipython notebooks with output cells converted to PDF format. The filenames should be `netid_mp3_part1_output.pdf`, `netid_mp3_part2_output.pdf` and `netid_mp3_part3_output.pdf`.
- A brief report in PDF format using this template[10], with name `netid_assignment3_report.pdf`.

[8] `https://pytorch.org/vision/stable/models.html`

[9] `https://gist.github.com/yrevar/942d3a0ac09ec9e5eb3a`

[10] `https://docs.google.com/document/d/12Qp4jm6AUeCjd2fY5kYJT7ITTD9NFK4MEFIbQiFkGA8/edit?usp=sharing`

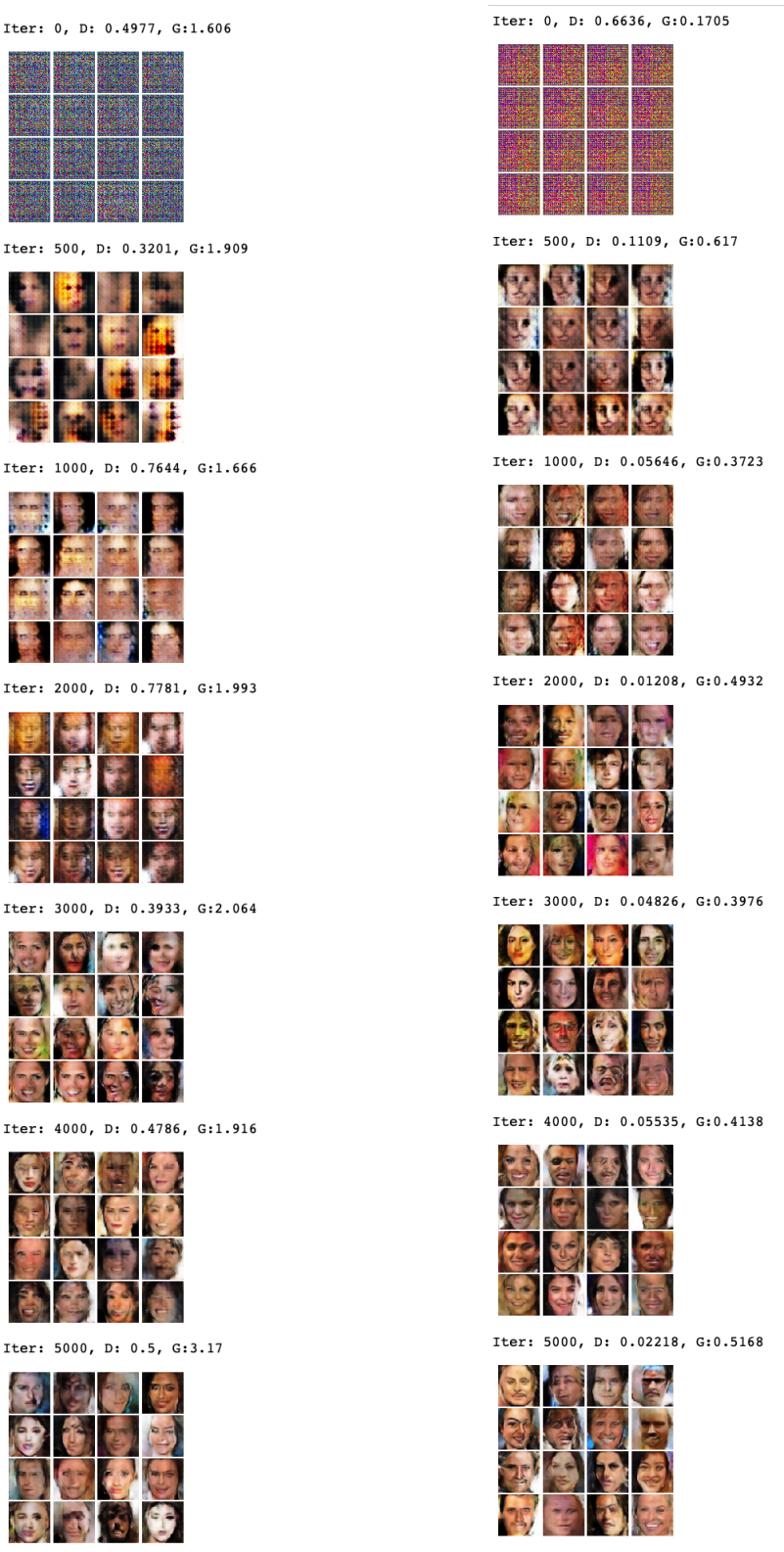

Figure 1: Sample generations of GAN (left) and LSGAN (right)