Spring 2024 CS 444

Assignment 4: Cat face generation with GANs

Due date: Wednesday, April 17th, 11:59:59PM

Sample images from a GAN trained on the cat face dataset

In this assignment, you will train a generative adversarial network (GAN) on a cat dataset and learn to generate cat face images.

In addition to familiarizing you with generative models and recurrent neural networks, this assignment will help you gain experience with how to implement GANs in PyTorch and how to augment natural images. You will also get familiarized with one of the techniques to improve the quality of images generated by GANs.

This assignment was adapted from and inspired by material from the Stanford CS231n Assignments, the CMU 16-726 Assignment 3, and the PyTorch Tutorials.

Download the starting code here.

Data setup

Once you have extracted the zip file, go to the assignment folder and execute the download script provided:

```
cd assignment4_materials/
./download_cat.sh
```

Alternatively, you can download the cat face dataset from this download link (47.4 MB).

The provided image data are all cropped and resized to the same width and height.

Implementation

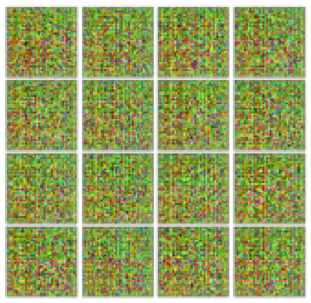
The top-level notebook (`MP4.ipynb`) will guide you through the steps you need to take to implement and train a GAN. You will train two different models, the original GAN and LSGAN, which has a different loss function. The generator and discriminator network architectures you will implement are roughly based on DCGAN. You will also implement data augmentation via PyTorch's built-in transforms.

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. It takes around a five minutes to train the debugging GAN on MNIST and just about one hour to train the main assignment GAN on Cats. Therefore, we highly recommend spending time on the MNIST portion to make sure it is working correctly before progressing to the main assignment.

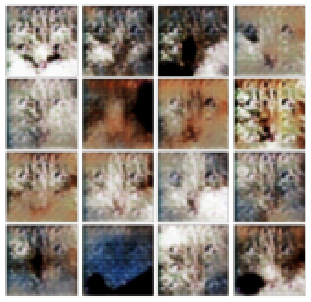
You will need to use a GPU for training your GAN. We recommend using Colab to debug, but a Google Cloud machine once your debugging is finished. Training the GAN should take roughly an hour.

For reference, here are some samples of GAN (left) and LSGAN (right) output at various points during training:

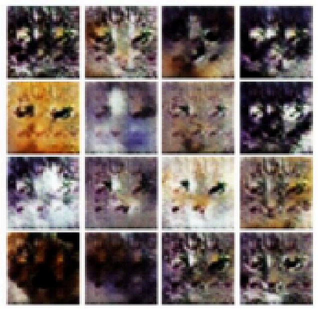
Iter: 0, D: 1.431, G:1.785



Iter: 1000, D: 0.3443, G:3.141

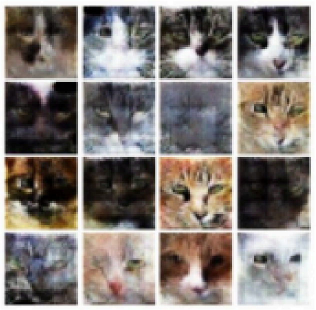


Iter: 2000, D: 0.6244, G:6.43

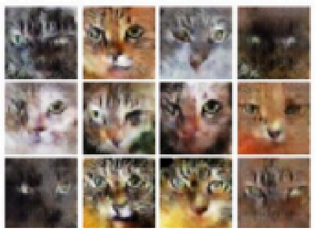


Iter: 3000, D: 0.5287, G:4.122

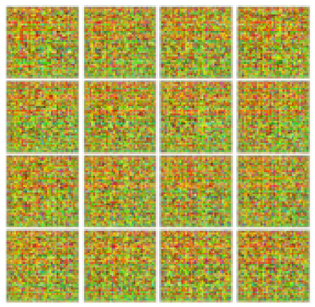
Iter: 4000, D: 0.3922, G:5.622



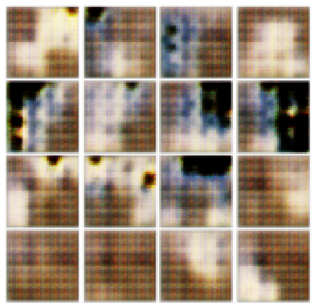
Iter: 5000, D: 0.2895, G:5.237



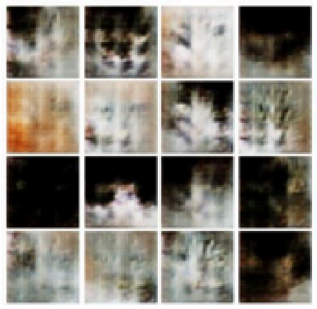
Iter: 0, D: 0.6724, G:9.398



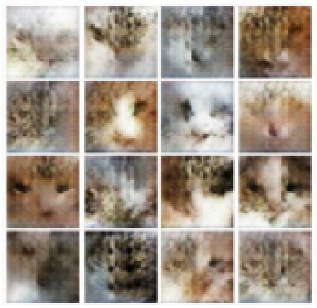
Iter: 1000, D: 0.3329, G:0.7512



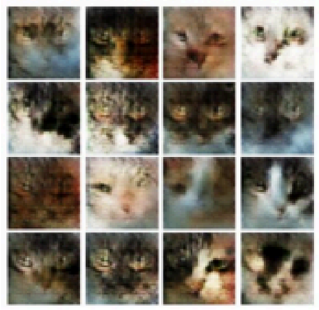
Iter: 2000, D: 0.1918, G:0.169



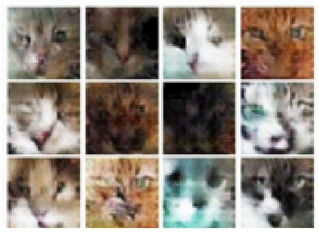
Iter: 3000, D: 0.2504, G:0.2977

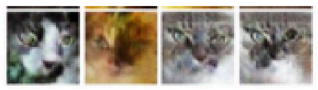


Iter: 4000, D: 0.1581, G:0.2812



Iter: 5000, D: 0.2837, G:0.4201





Extra Credit

- Experiment with an alternative GAN loss such as margin-based WGAN/WGAN-GP, DRAGAN, or BEGAN.
- Implement an advanced normalization scheme such as spectral normalization -- see paper (esp. Algorithm 1 in Appendix A) or this blog post.
- Try a fancier GAN architecture such as Progressive GAN or self-attention GAN.
- Starting with your trained model, play around with GAN inversion or latent space traversal techniques to perform image manipulation.
- Train your GAN on a different dataset.

Useful Pointers

- Lecture material on GANs
- A nice blog post series including introductory posts along with posts on DCGAN and LSGAN
- Generative Adversarial Nets paper
- LSGAN paper
- DCGAN paper

Environment Setup (Local)

If you will be working on the assignment on a local machine then you will need a python environment set up with the appropriate packages. We suggest that you use Conda to manage python package dependencies (https://conda.io/docs/user-guide/getting-started.html).

IPython

The assignment is given to you in the `MP4.ipynb` file. To open the files on a local machine, ensure that ipython is installed (https://ipython.org/install.html). You may then navigate the assignment directory in terminal and start a local ipython server using the `jupyter notebook` command.

Submission Instructions

This assignment is due on Canvas on the due date specified above. You must upload the following files:

1. All of your code (python files and ipynb file) **in a single ZIP file**. The filename should be **netid_mp4_code.zip**.
2. Your MP4 ipython notebook with output cells converted to **PDF format**. The filename should be **netid_mp4_output.pdf**.
3. A brief report in PDF format using **this template**. The filename should be **netid_mp4_report.pdf**.

Please refer to course policies on collaborations, late submission, etc.