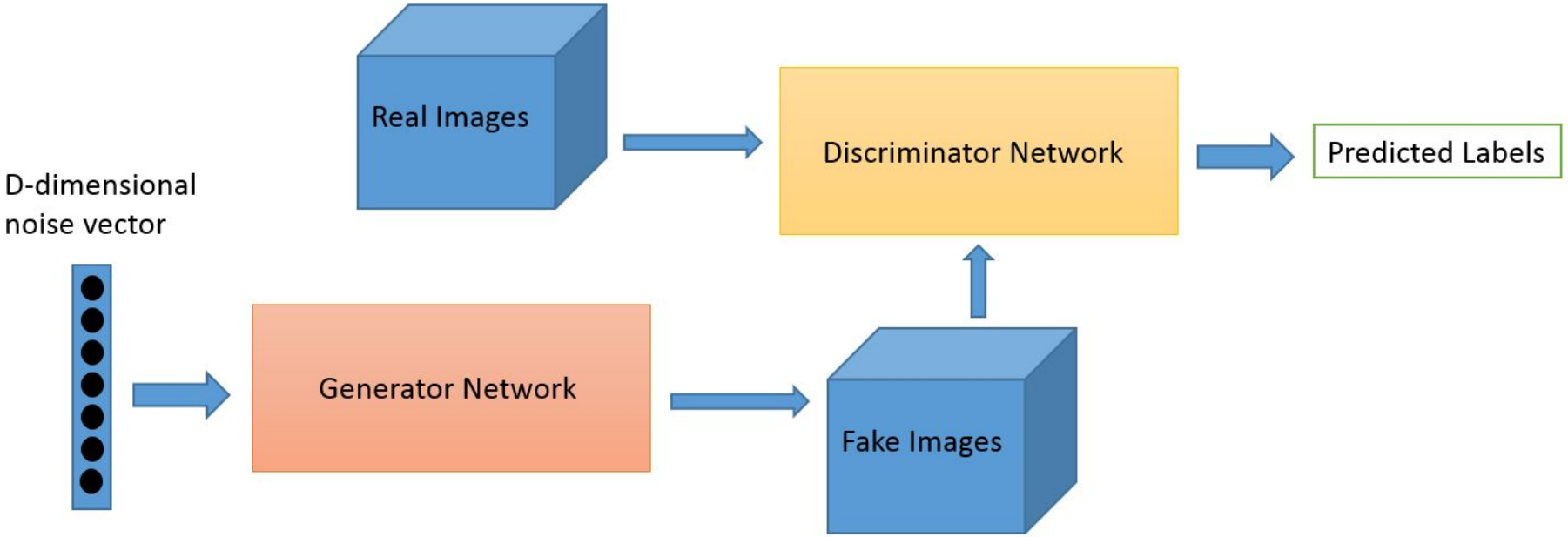
GANs Tutorial

Huimin Ren
hren@wpi.edu

Agenda

- Target
- Demo
- Deliverables & Grading
- Tutorial

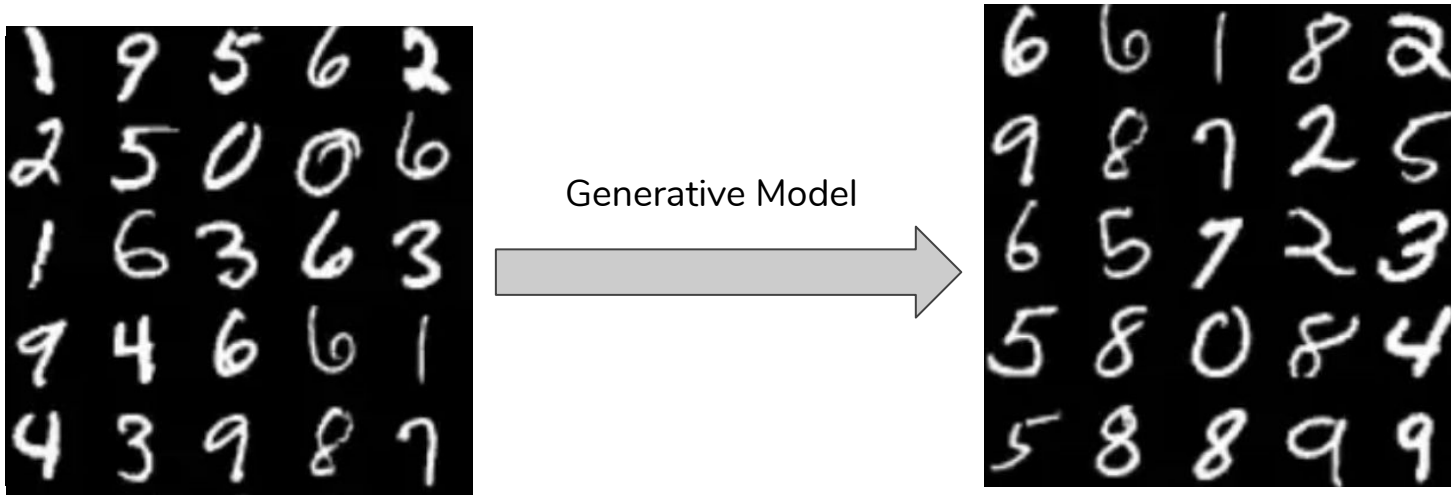
GAN



https://skymind.ai/wiki/generative-adversarial-network-gan

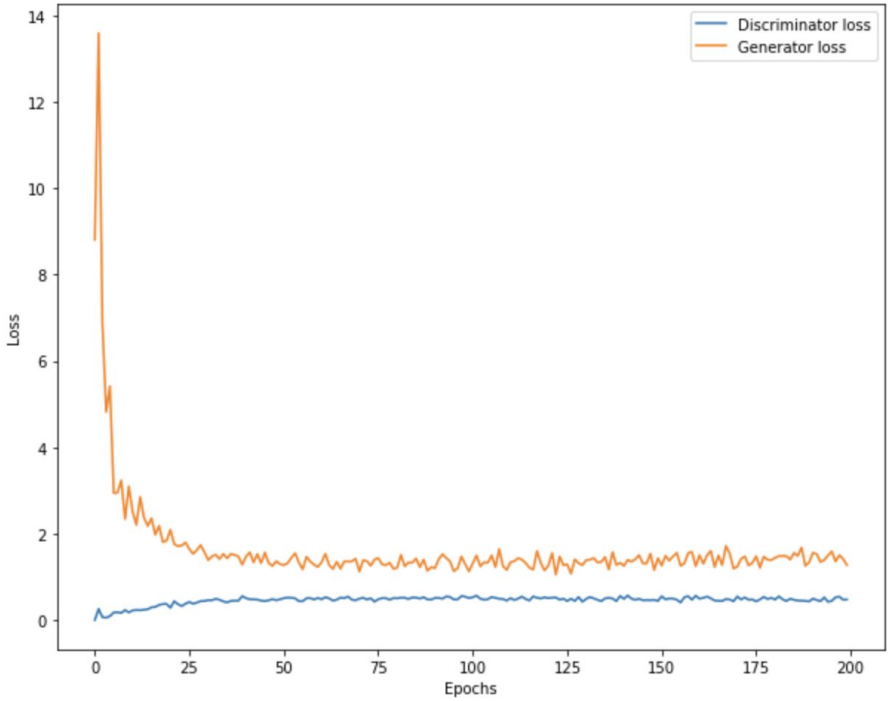
Assignment Target

MNIST generation: http://yann.lecun.com/exdb/mnist/



Requirement: https://github.com/huiminren/DS504_GAN_HW

Demo


Discriminator loss
Generator loss
Loss
Epochs
0
2
4
6
8
10
12
14
0
25
50
75
100
125
150
175
200

Deliverables & Grading

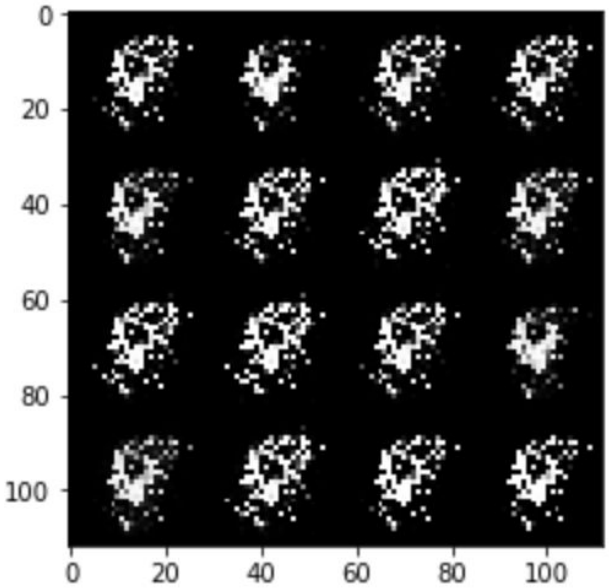
Due Date: Thursday April 18(23:59)

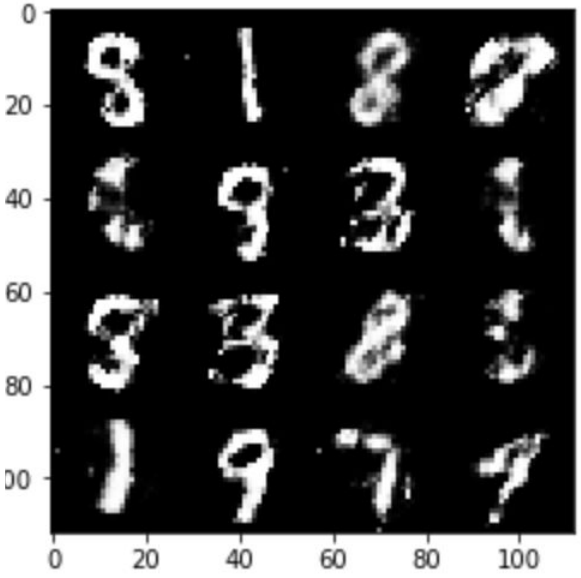
- Reports (70%)
  - **Set of Experiments Performed:** Include a section describing the set of experiments that you performed (reproducibility). [30’]
  - **Special skills:** Include the skills which can improve the generation quality. [20’]
  - **Visualization:** Include 25 (5*5) final generated images, a loss plot of the generator and discriminator during your training. [20’]
- Code (20%)
  - No errors with small test data set.
- Generator model (10%)
  - Each generated images can be recognized by human, otherwise you may loss some points.

Bad Generation

Good Generation

---------- Epoch 1 ---------- Epoch 20 -------





Requirements

- Python 3.x
- Keras 2.+

Other packages are not allowed for generating handwrite numbers.

Bonus (10%)

Generate images from other data source.

- Data set
  - Face Dogs and Cats Anime ...
- Package
  - You are allowed to use any deep learning package, such as Tensorflow, Pytorch, etc.
- Deliverable
  - Code
  - Model
  - README file (How to compile and load the model to generate images)
  - 25 generated images

How to use CPU version locally

- Install Anaconda https://docs.anaconda.com/anaconda/install/
- Create virtual environment
  https://conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html
  https://uoa-eresearch.github.io/eresearch-cookbook/recipe/2014/11/20/conda/
- Install packages (pip install or conda install)

How to use Keras

A template for building GAN model is provided.
https://github.com/huiminren/DS504_GAN_HW/blob/master/template.py

```
generator = Sequential()

# Add layers with Dense, Conv2D, etc.
# Multiple activation functions are available, such as relu, sigmoid, tanh etc
generator.add(Dense(128, input_dim=z_dim, activation='sigmoid')) # The first layer should state input dimension
generator.add(Dense(384))
generator.add(LeakyReLU(alpha=0.2))
generator.add(Dense(512))
generator.add(LeakyReLU(alpha=0.2))
generator.add(Dense(784, activation='sigmoid'))  # Values between 0 and 1

# compile model
generator.compile(loss='binary_crossentropy', optimizer=Adam(lr=0.0002, beta_1=0.5), metrics=['accuracy'])
generator.summary()
```

How to use GPU version on Turing?

- Apply Turing account http://arc.wpi.edu/computing/accounts/turing-accounts/
  - Turing documentation http://arc.wpi.edu/cluster-documentation/build/html/batch_manager.html
  - ssh wpi_email@turing.wpi.edu
- Install anaconda
  - Download https://www.anaconda.com/distribution/#linux
  - Upload to your Turing account (scp -r LocalFilePath TargetFilePath)
  - bash **.sh // install anaconda
  - source ~/.bashrc // activate
- Create virtual environment
  - conda create -n myenv python=3.6 https://conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html#creating-an-environment-with-commands
  - conda activate myenv
- Install packages on virtual environment
  - Eg. `conda install -c anaconda tensorflow-gpu`

How to use GPU version on Turing?

Example:
https://github.com/aymericdamien/TensorFlow-Examples/blob/master/examples/6_MultiGPU/multigpu_cnn.py

https://github.com/pytorch/examples/blob/master/mnist/main.py

Shell script:

module load cuda90/toolkit/9.0.176

Basic Linux commands

Tutorial for Turing: https://github.com/huiminren/DS504_GAN_HW/blob/master/TuringTutorial.pdf

Log in Turing Account

ssh wpi_email_user_name@turing.wpi.edu
password same as your wpi email

Basic Commands for Linux

pwd //print the name of current working directory
ls // show all the files under current working dicrectory
cd ~ // change directoryto your root
cd ~/ds504/ // change directory to ds504
mkdir ds504 // create a folder named ds504
mv folder_1 folder_2 // move a folder to another folder or change folder_1's name to folder_2
cp -r folder_1 folder_2 // copy folder_1 to folder_2
rm -r folder_name // delete a folder or a file
vi file // to view and modify a file or create a file if the file name is not exist
type a/i to insert
type esc to exit
wp // write and quit
q // quit (when you didn't change anything)
q // quit without saving
scp -r local_file_path target_file_path // upload or download files [Must be used in your local terminal]

Basic Commands for Turing

sbatch **.sh // submit your job
squeue // view all the jobs
scancel jobID // cancel your job
module list // view your model
module load ** // load module you need, such as cuda

Virtual Environment

conda create -n myenv python=3.6 //create a new virtual environment
conda activate myenv // activate the virtual environment
conda deactivate // deactivate the virtual environment
conda install ** // install some packages you need
[You need to activate your virtual environment before running your code]

Shell Script for Turing

Eg. [you can copy the following script, and make sure this shell script is in the same directory as the python file you want to run.]
#!/bin/bash
#SBATCH -N 1 // number of nodes
#SBATCH -n 2 // number of CPUs
#SBATCH --mem=16G // memory as you need
#SBATCH -p short //long 7days, or short 24 hours
#SBATCH -C K80 // GPU, you can choose K40, K20 as you need
#SBATCH -o ds504.out // output file name
#SBATCH --gres=gpu:2 // number of GPUs
python GAN.py // the python file you want to run
[Please remove the commands before you use the script.]

Algorithm Initialize θ_d for D and θ_g for G

- In each training iteration:

Learning D

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

Learning G

- Sample m noise samples $\{z^1, z^2, \dots, z^m\}$ from a distribution
- Update generator parameters θ_g to maximize
  - $\tilde{V} = \frac{1}{m}\sum_{i=1}^{m} log\left(D\left(G(z^i)\right)\right)$
  - $\theta_g \leftarrow \theta_g - \eta \nabla \tilde{V}(\theta_g)$

http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2018/Lecture/GAN%20(v2).pdf