ARTIFICIAL NEURAL NETWORKS - WEEK 11

Generative Adversarial Networks (GANs)

Dr. Aamir Akbar

Director of both AWKUM AI Lab and AWKUM Robotics, Final Year Projects (FYPs) coordinator, and lecturer at the department of Computer Science
Abdul Wali Khan University, Mardan (AWKUM)

CONTENTS

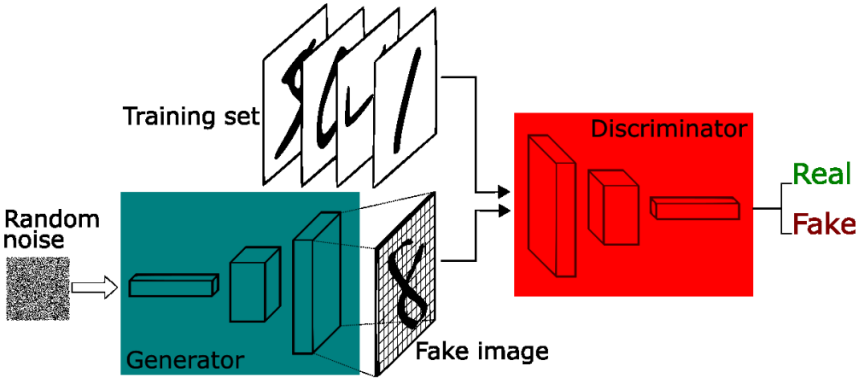
GANS

What are Generative Adversarial Networks (GANs)?

GANs are unsupervised learning ANNs. GANs consist of two neural networks: the `generator` and the `discriminator`, which work in a competitive framework.

1. `Generator:` This network generates new data samples that resemble the training data. It starts with random noise and transforms it into data that aims to look like the real data.
2. `Discriminator:` This network evaluates the authenticity of the data samples, distinguishing between real data from the training set and fake data generated by the generator.

GANS


Training set
Discriminator
Real
Fake
Random noise
Generator
Fake image

GANS VS. AUTOENCODERS

Key Differences:

1. `Objective:` GANs aim to generate new data samples that mimic the training data, while autoencoders aim to reconstruct the input data from a compressed representation.
2. `Architecture:` GANs have two networks (generator and discriminator) working adversarially, whereas autoencoders have a single encoder-decoder pair.
3. `Output:` GANs produce new, realistic data samples. Autoencoders reconstruct existing data.

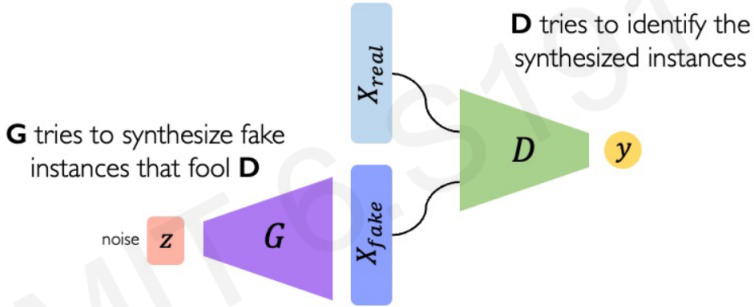
GANS VS. RNNS

Key Differences:

1. `Objective:` RNNs are primarily used for handling sequential data and capturing temporal dependencies, whereas GANs are used for generating new data samples.
2. `Architecture:` RNNs have a recurrent structure to maintain state across time steps. GANs use a generator-discriminator pair with no inherent mechanism for handling sequences.
3. `Output:` RNNs predict or generate sequences based on learned patterns. GANs generate new, standalone data samples.

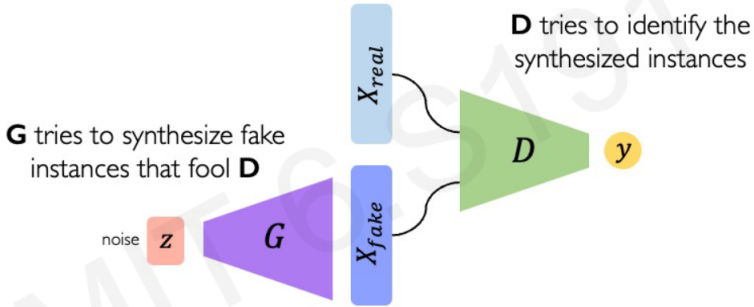
TRAINING GANS

The discriminator (D) tries to tell the difference between real data and the fake data created by the generator (G). The generator tries to make fake data that looks so real it can trick the discriminator.



TRAINING GANS

Training GANs involves a unique process where two neural networks, the generator (G) and the discriminator (D), are trained simultaneously in a competitive setting.



ALGORITHM: TRAINING GANS

Initialize generator G and discriminator D with random weights
while not converged **do**
    `Training Discriminator` D:
        Sample a batch of real data x from the training set
        Sample a batch of random noise z
        Generate fake data $\hat{x} = G(z)$
        Compute discriminator loss: $\mathcal{L}_D$
        Update discriminator D's weights to minimize $\mathcal{L}_D$
    `Training Generator` G:
        Sample a batch of random noise z
        Generate fake data $\hat{x} = G(z)$
        Compute generator loss: $\mathcal{L}_G$
        Update generator G's weights to minimize $\mathcal{L}_G$
    `Feedback Loop`:
        Discriminator D gives feedback to Generator G
        Generator G uses feedback to improve generated data $\hat{x}$
end while

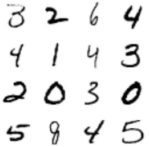
SUMMARY: TRAINING GANS

1. `Training Cycle:` The discriminator and generator are trained alternately.
2. `Discriminator Loss:` Measures the accuracy in distinguishing real data from fake data.
3. `Generator Loss:` Measures the success in generating realistic data that fools the discriminator.
4. `Goal:` The generator improves in creating realistic data, while the discriminator improves in detecting fake data, pushing both networks to enhance their performance continuously until a balance is achieved.

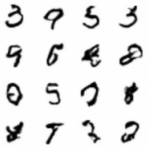
GANS IN PRACTICE

GAN: example on MNIST

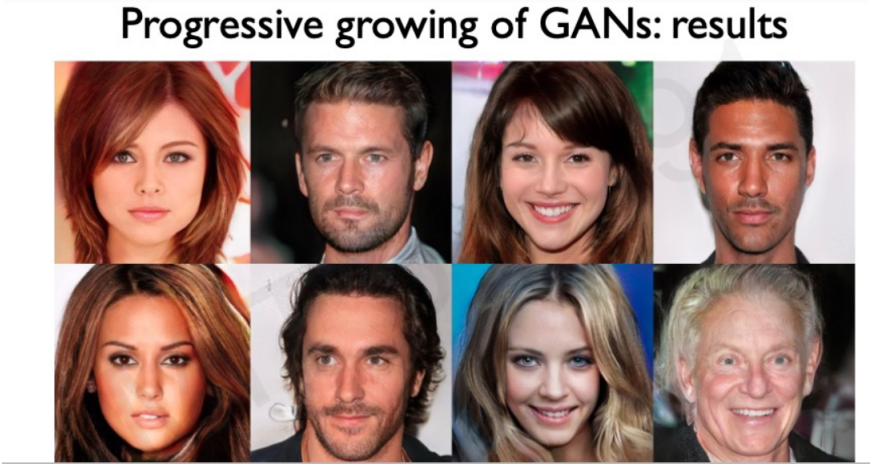
MNIST training data:



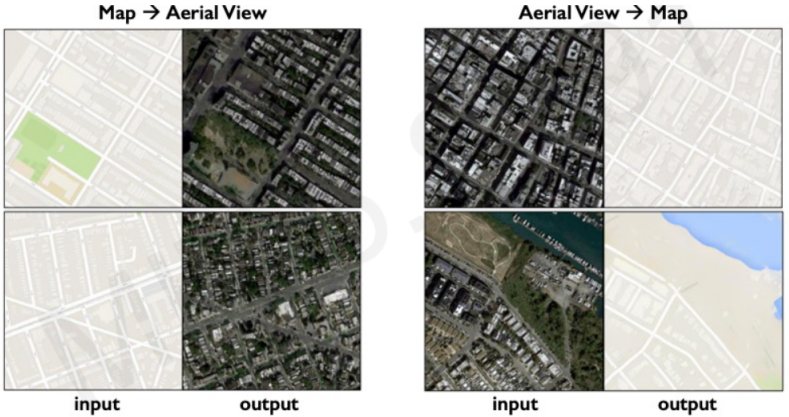
GAN generated examples:



GANS IN PRACTICE



GANS IN PRACTICE



RESOURCES

To download the source codes used in the previous slides, follow the link:

▸ Download Source Codes

Import the codes into your preferred development environment, such as Visual Studio Code (VS Code), to practice and explore further.

To learn programming in Python, follow my comprehensive 15-week Programming in Python course at:

▸ Programming in Python