Johns Hopkins
Whiting School
of Engineering

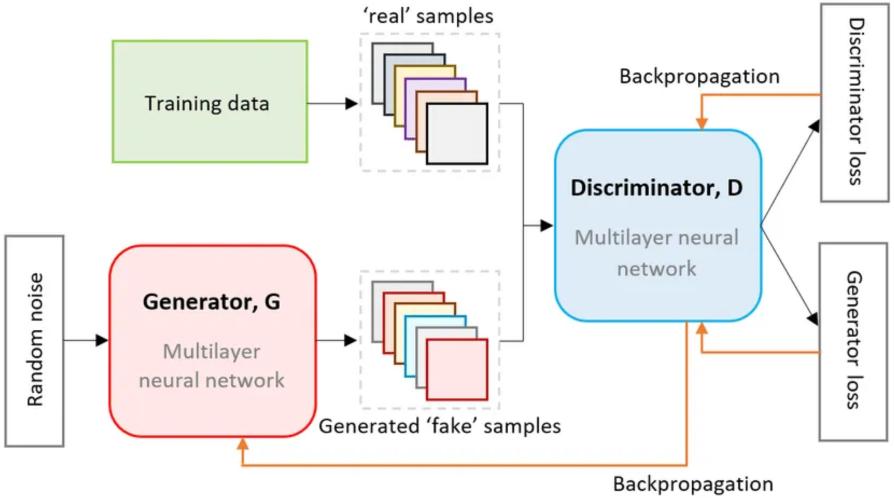
685.621 Algorithms for Data Science

Generative Adversarial Networks (GANs)

Introduction to GANs

GANs are deep learning models that use two competing neural networks to create realistic synthetic data.

- **Basic Structure**
  - Generator (G): takes random noise and generates fake data.
  - Discriminator (D): Distinguishes between real and fake data.
  - Adversarial Training: G and D compete, improve over time.
  - G learns to produce realistic samples, D learns to detect fake samples.



Presidio, 2024

GAN Objective

GANs use a core minimax objective function that is zero sum between G and D.

$$\min_{G} \max_{D} \mathbb{E}x \sim \mathrm{pdata}(x)[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]$$

- D is trying to maximize the function, G is trying to minimize the function. D is looking to correctly classify real vs fake samples. G is looking to generate realistic fake samples that fool D.

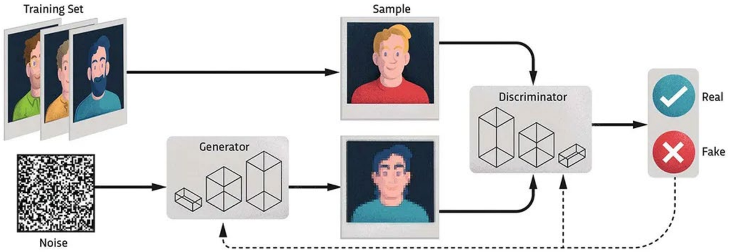
Step 1: Define the D and G

The Generator takes random noise and generates fake data. The Discriminator distinguishes between real and fake data.

- **Example of G**
  - Begin with a random noise vector, transform it into a 784D vector using fully connected layers. ReLU can be used for activations in hidden layers, and Tanh can be used to scale the output to [-1, 1]. The output is a fake image.
- **Example of D**
  - Begin with a 784D vector passed through fully connected layers. ReLU can be used for activations in hidden layers. Final layer is Sigmoid to provide output of 0 or 1. This is a binary classifier.

Presidio, 2024

Step 2: Training/Evaluating the D and G

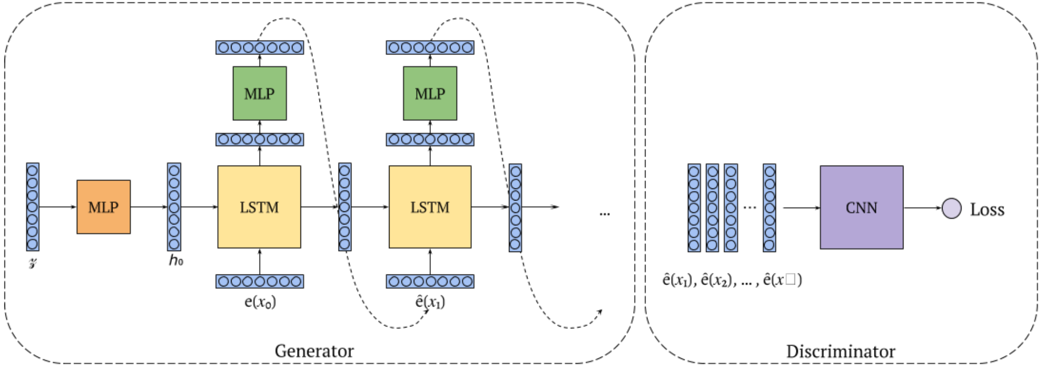


BBC, 2023

1. **Training D**: Generate random noise and feed into Generator to create fake images. Pass fake images into Discriminator to create fake predictions and compute loss (fake predictions vs fake labels). Finally, compute total Discriminator loss, backpropagate the gradients, and update the Discriminator’s weights.
2. **Training G**: Continue to create and pass fake images into the Discriminator and compute loss (fake predictions vs real labels). Finally, backpropagate the gradients and update the Generator’s weights.
3. **Convergence**: Continue iterating through epochs until Generator loss decreases and converges while Discriminator loss increases and stabilizes. This means the Generator is getting good at fooling the Discriminator.

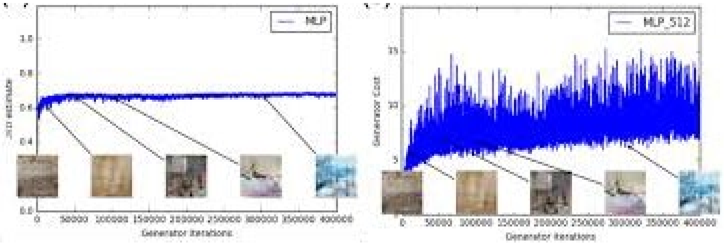
Step 3: Advantages

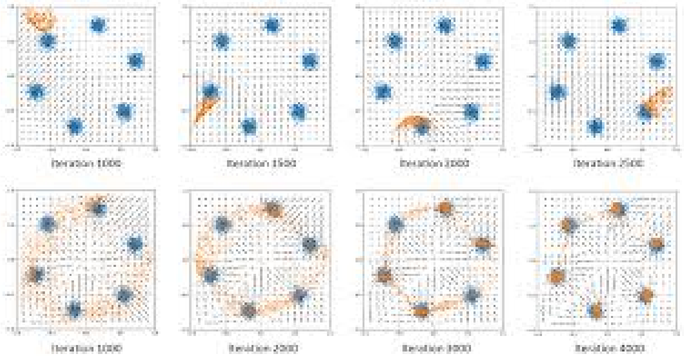
1. **Data Augmentation**: GANs allow for beneficial practical application like data synthesis tasks where generating diverse and coherent text data is essential.
2. **Realistic Generation**: GAN-based models produce flexible, highly realistic, and contextually relevant text sequences for tasks involving NLP.



Step 4: Challenges

1. **Training Instability**: GAN training is unstable with exploding gradients, vanishing gradients, and imbalanced learning due to learning rates/batch sizes/architecture, etc.
   - ○ Solution: Gradient Penalty using WGAN to ensure gradients don't vanish or explode and improve stability. Use different learning rates for G vs D and optimize SGD (Adam) optimization.
2. **Mode Collapse**: Generator produces limited variations of fake data and Discriminator exploits this by easily identifying fake samples.
   - ○ Solution: Minibatch Discrimination to allow D to evaluate G samples in batches. Use of multiple Discriminators so D learns different aspects which forces G to generate more diverse outputs.



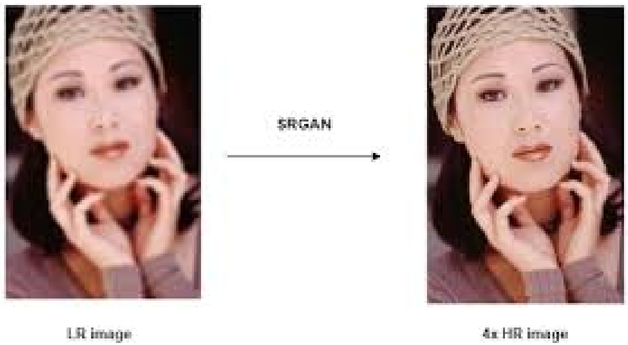


Step 5: Real World Applications

1. **Deepfakes**: Realistic fake videos and images

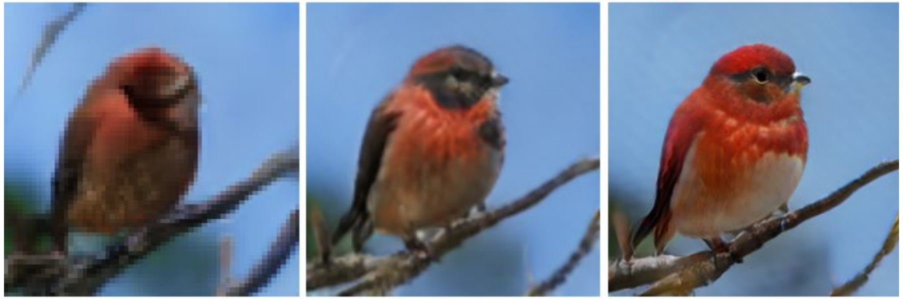
2. **Super-Resolution**: Enhancing image quality

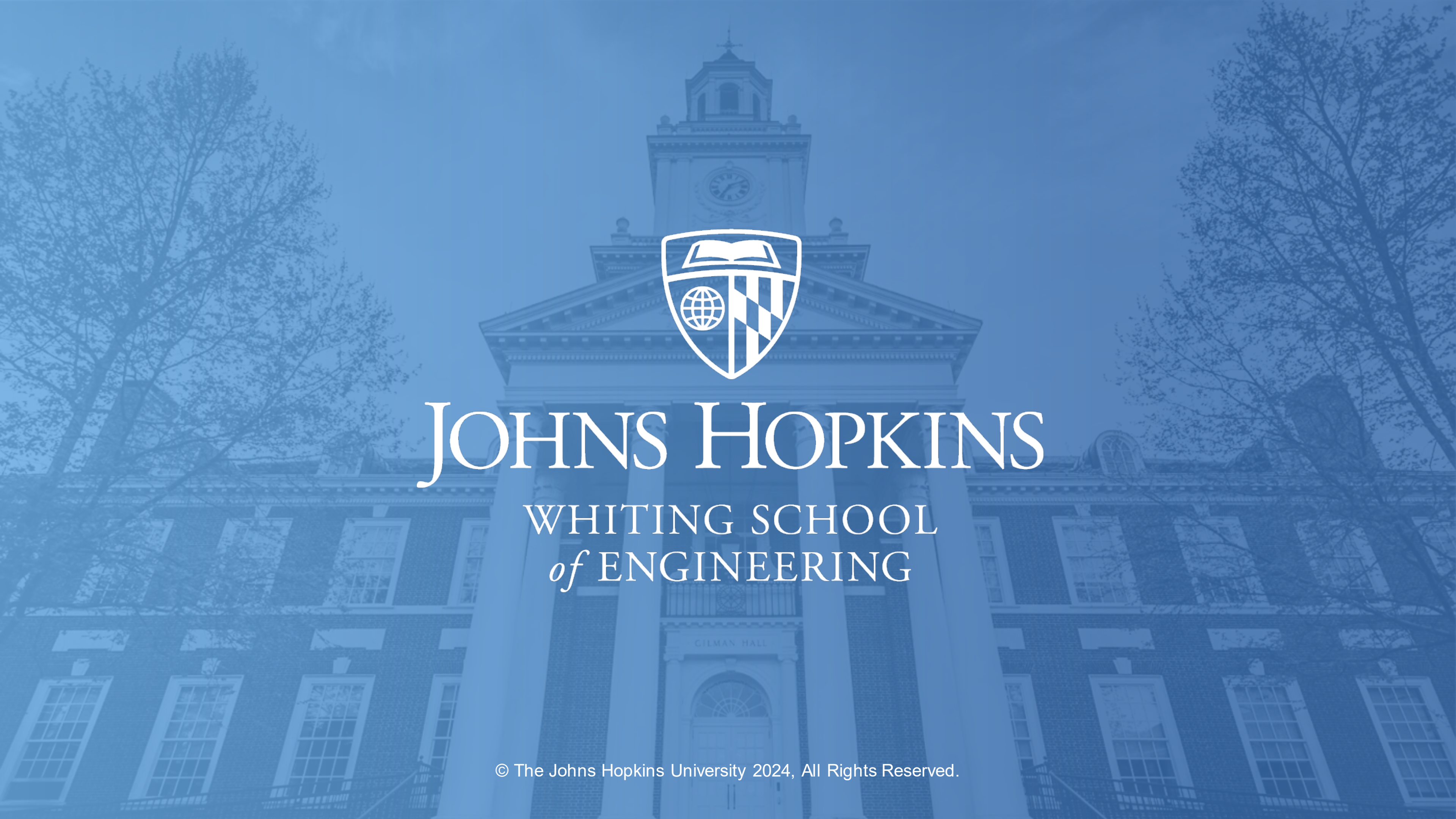
3. **Text-to-Image Generation**: DALL-E



GAN

this bird is red with white and has a very short beak





JOHNS HOPKINS

WHITING SCHOOL
of ENGINEERING