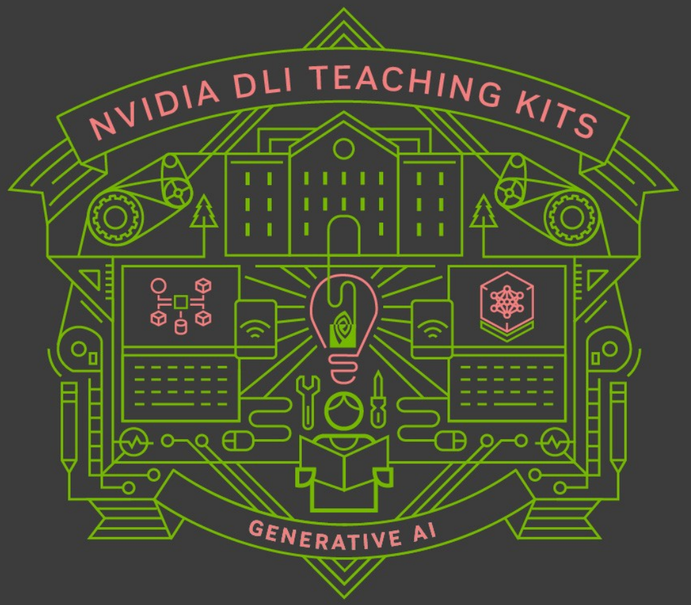
Lecture 6.2 – UNets and Diffusion Models

Generative AI Teaching Kit



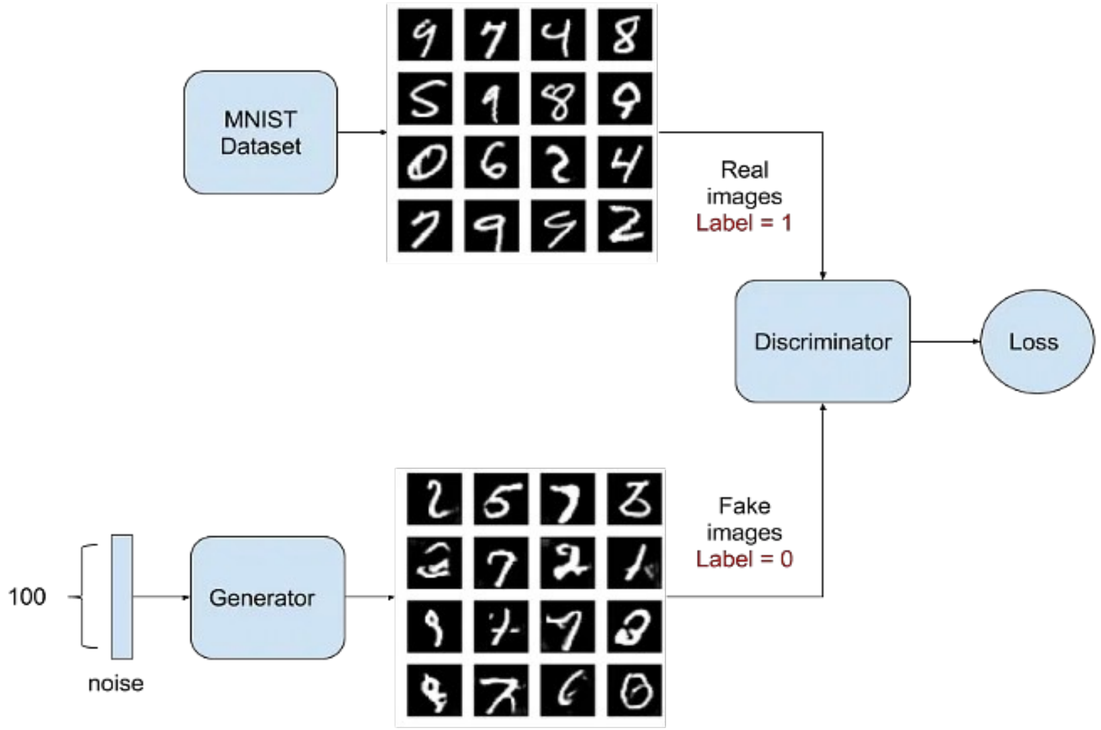
This lecture

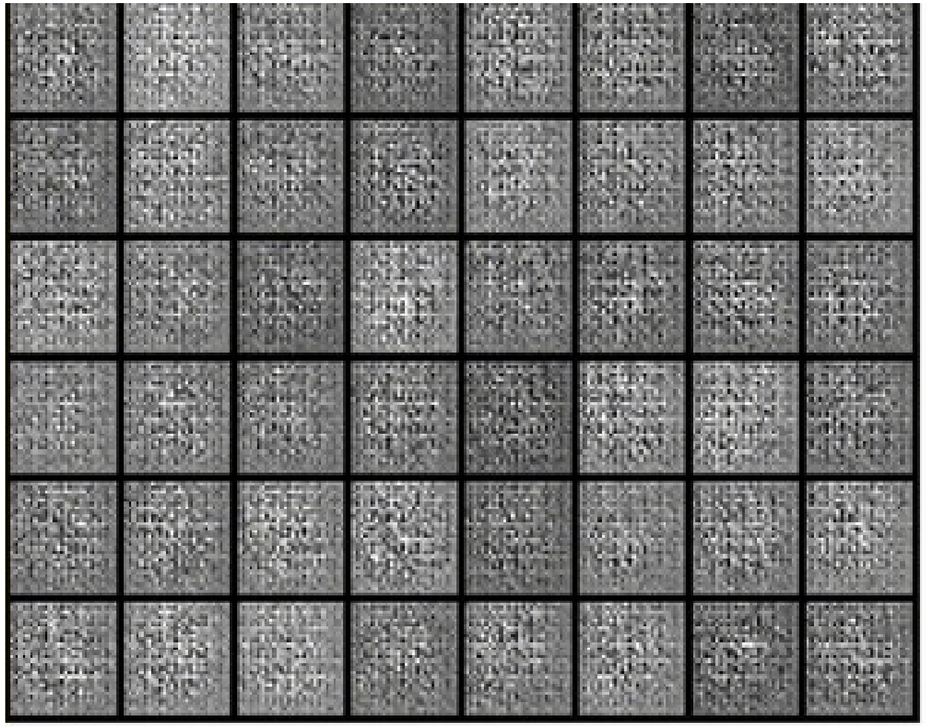
- Recap of issues with GANs
- UNet and Autoencoders
- Denoising Images
- Diffusion Models
- Pros and cons of diffusion models
- Potential improvements

Recap on GANs and Image GenAI

GAN Recap

Last time, we looked at how GANs can be used to generate new data

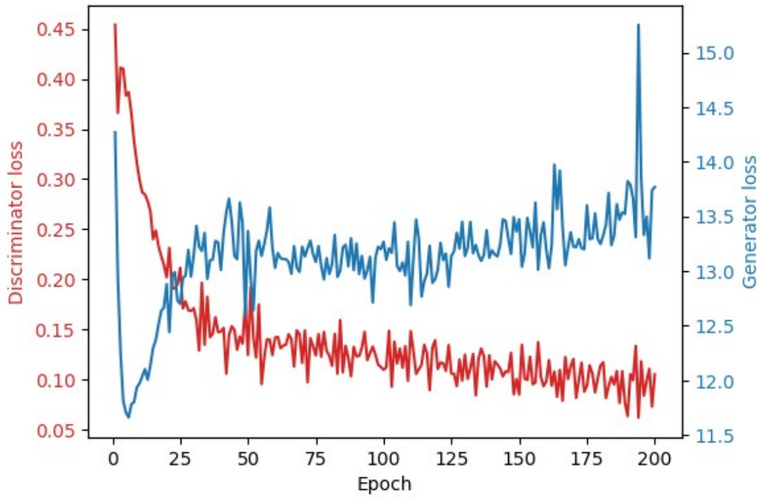




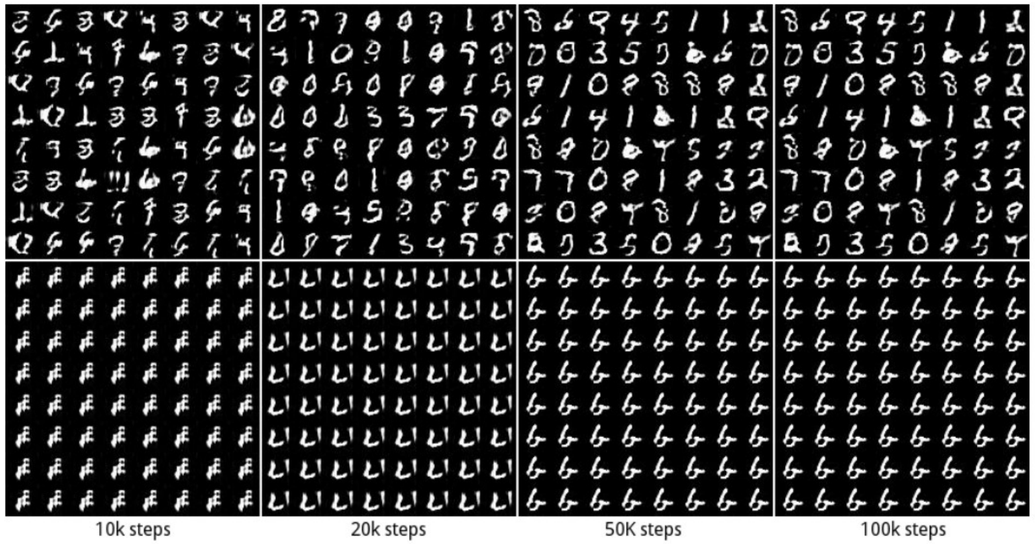
GAN Problems – Mode Collapse and Instability

During training a Generator can find a local minima which produces strong feedback from the discriminator. This is known as **Mode Collapse.**

Another challenge of GAN training is **instability**



If the Discriminator is too strong, then the Generator struggles to train.



UNet Architecture

UNet - A New Architecture for Image Models

The UNet architecture was introduced in 2015 by Ronneberger et.al for Biomedical Image Segmentation

Prior to UNet, many segmentation methods required large amounts of training data and were not well-suited for tasks with limited annotated datasets, which is common in medical imaging.

Existing methods often failed to provide high-resolution segmentation maps and struggled with capturing fine details

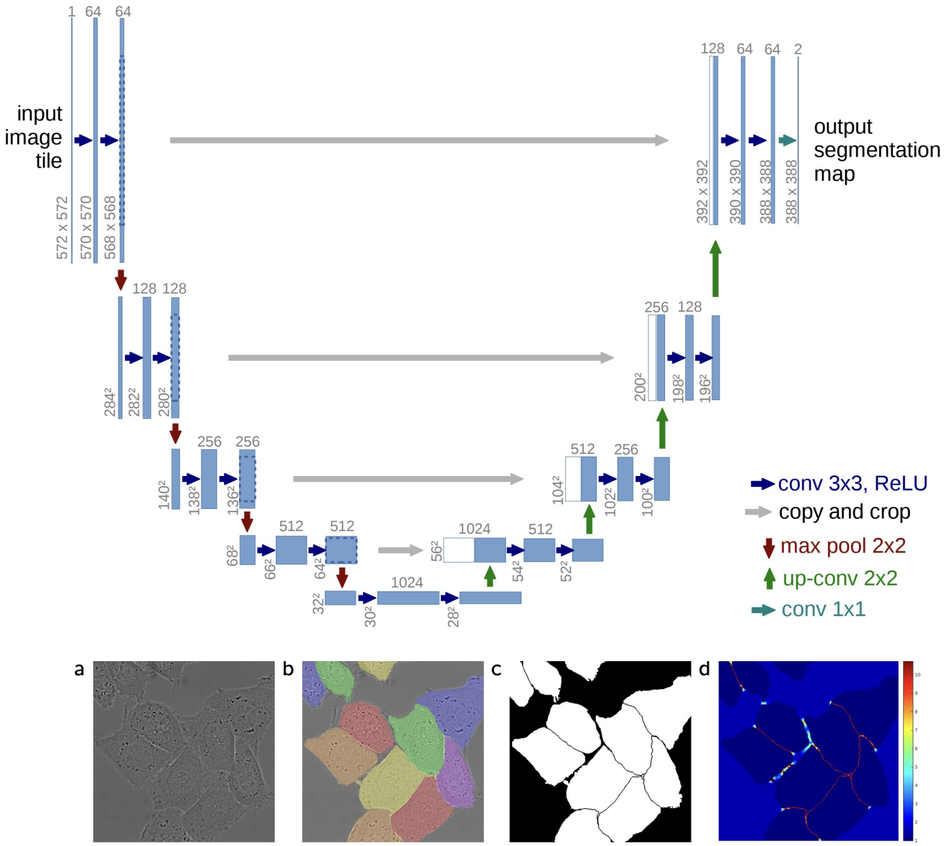
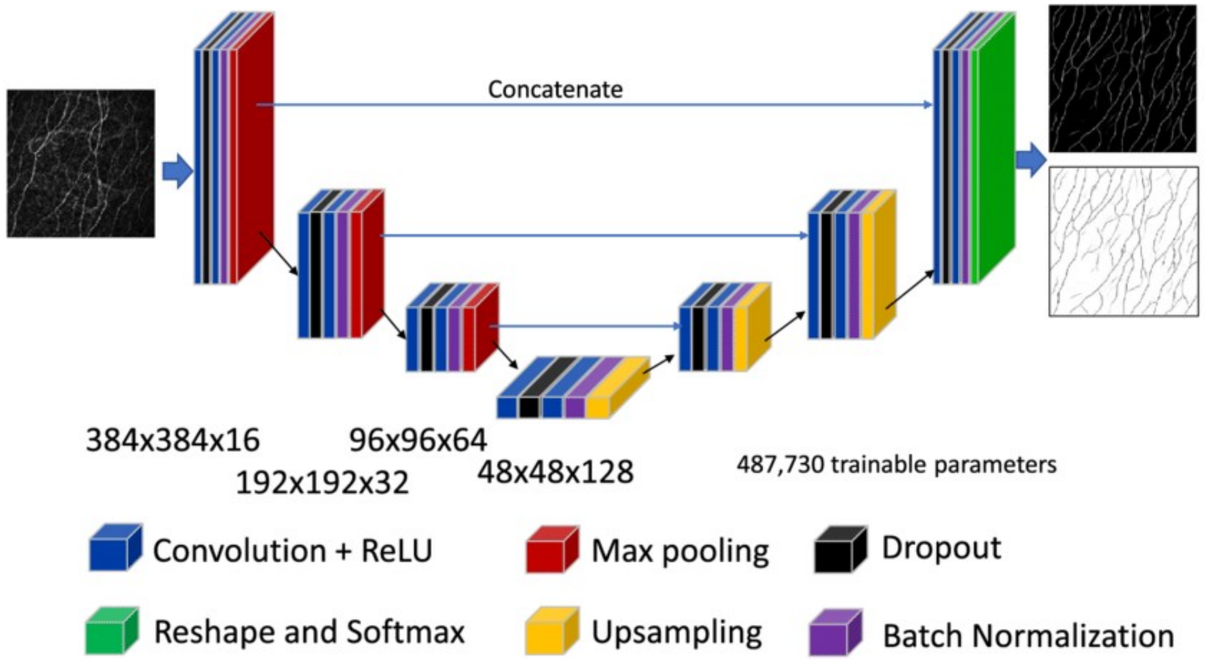


Fig. 3. HeLa cells on glass recorded with DIC (differential interference contrast) microscopy. (**a**) raw image. (**b**) overlay with ground truth segmentation. Different colors indicate different instances of the HeLa cells. (**c**) generated segmentation mask (white: foreground, black: background). (**d**) map with a pixel-wise loss weight to force the network to learn the border pixels.

UNet - Features and Utility

The U-Net is a novel **encoder-decoder** architecture.

- The encoder captures context by down-sampling the input image
- The decoder enables precise localization by up-sampling.
- Skip connections are used to directly transfer high-resolution features from the encoder to the corresponding layers in the decoder

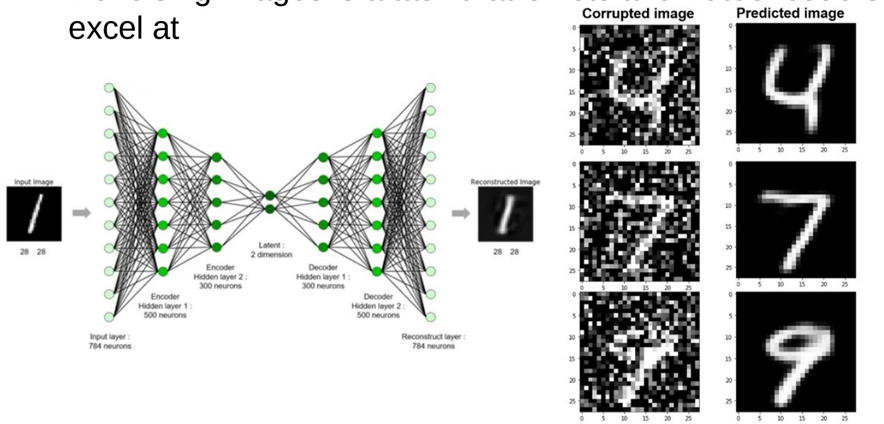


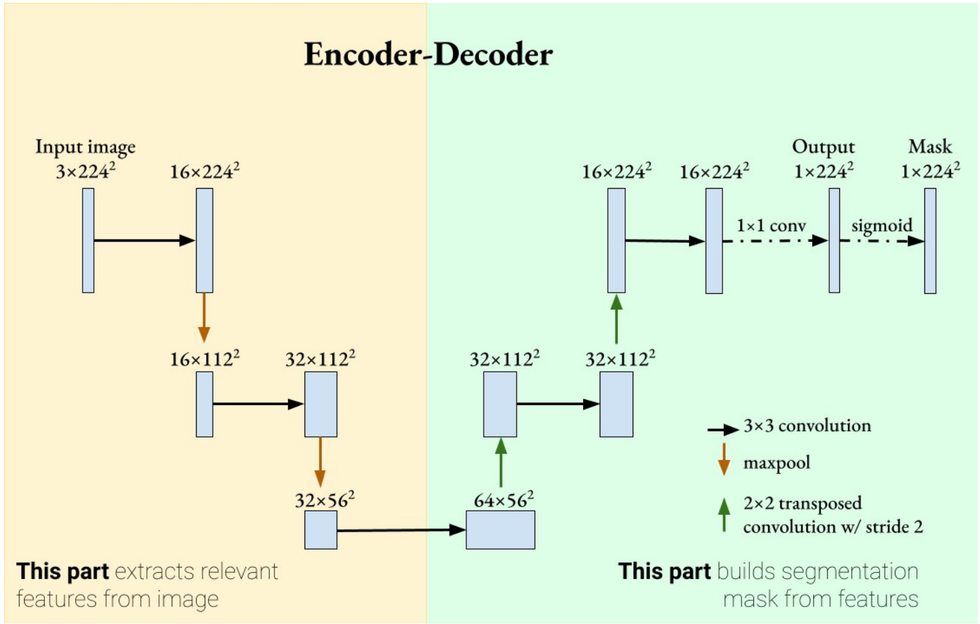
It has been widely adopted and extended for various applications, including medical image analysis, satellite image segmentation, **denoising**, and more.

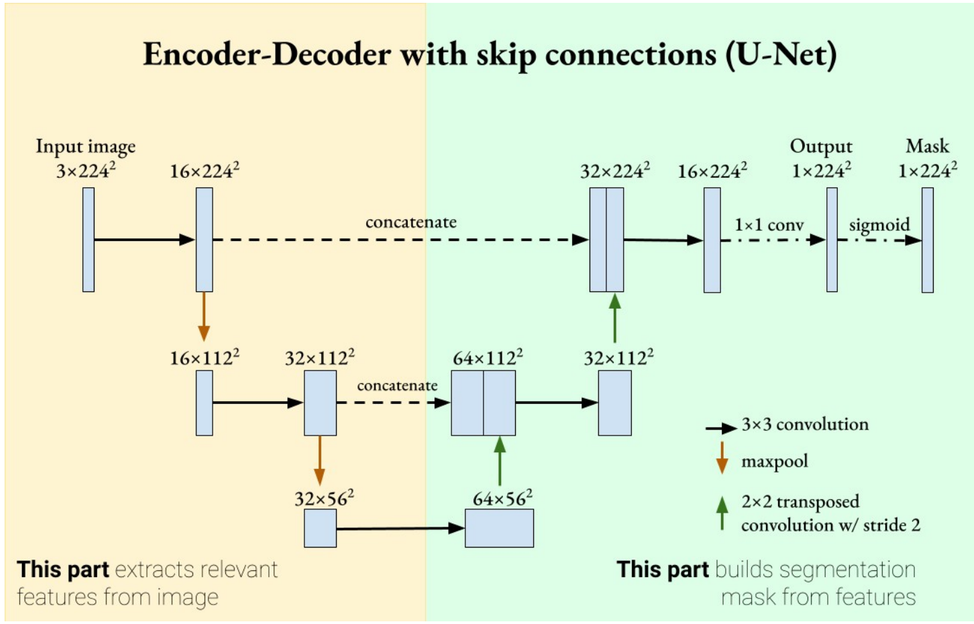
UNets and AutoEncoders

- UNets share many of the same features as traditional encode-decoder or autoencoder networks
- This architecture has become ubiquitous in computer vision

Denoising images is a task that UNets and Autoencoders excel at







DARTMOUTH ENGINEERING | NVIDIA

Denoising Images

Image Denoising

Denoising is a task in image processing and computer vision.

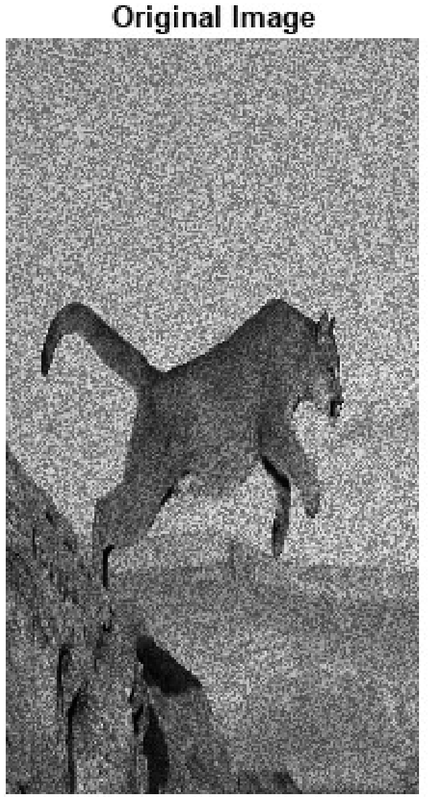
Aim:

To remove or reduce noise from an image.

Real-world Application:

Noise can be introduced into an image due to various reasons, such as camera sensor limitations, lighting conditions, and compression artifacts.

The goal of denoising is to recover the original image, which is considered to be noise-free, from a noisy observation.



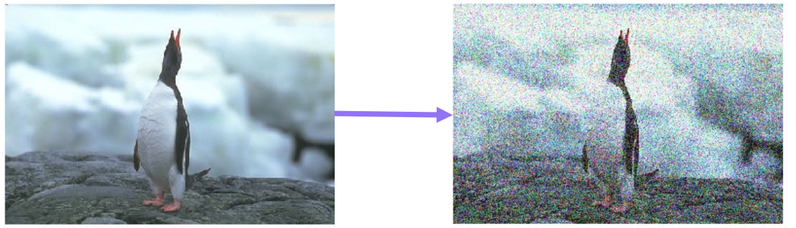
Training a Denoising Model

Training Data:

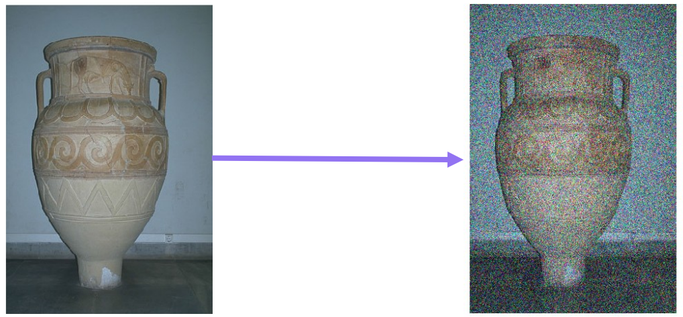
- Input: Gaussian Noise added to clear images
- Output: Noise-free images

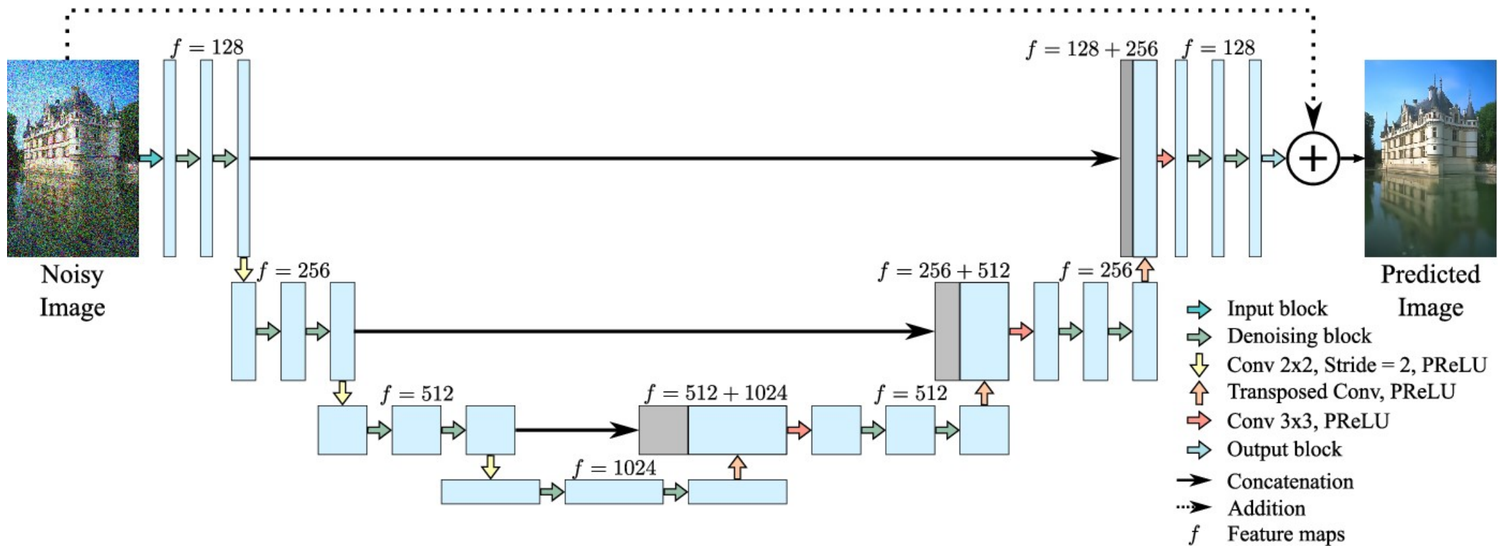
Metrics:

- Typically mean-squared error
- Can be more sophisticated such as using loss networks for image perception loss



Adding Noise





Adding noise to images

Step 1: Define Parameters

Mean, μ, of the Gaussian noise.

Standard deviation, σ, of the Gaussian noise.

Step 2: Generate Gaussian Noise

For each pixel value x, generate a noise value, n, from a Gaussian distribution with mean μ and standard deviation σ:

$$n \sim \mathcal{N}(\mu, \sigma^2)$$

Step 3: Add Noise to Pixel Values:

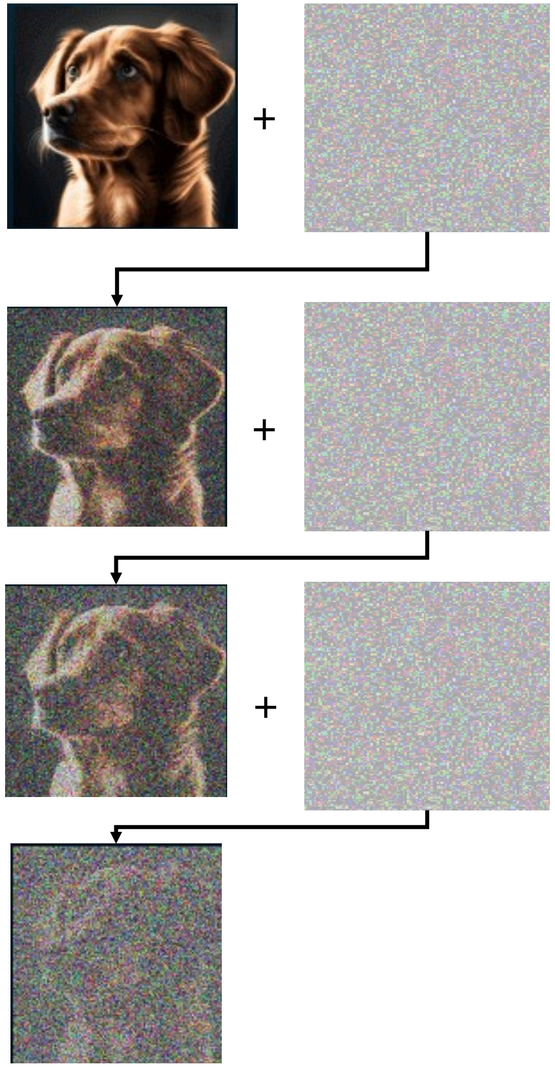
Add the generated noise value n to the original pixel value x:

$$x' = x + n$$

Step 4: Normalize Pixel Values

Ensure that the resulting pixel value x' is within the valid range (e.g., [0, 255] for 8-bit images):

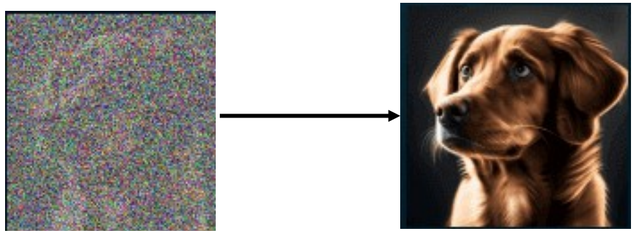
$$x' = \min(\max(x + n, 0), 255)$$



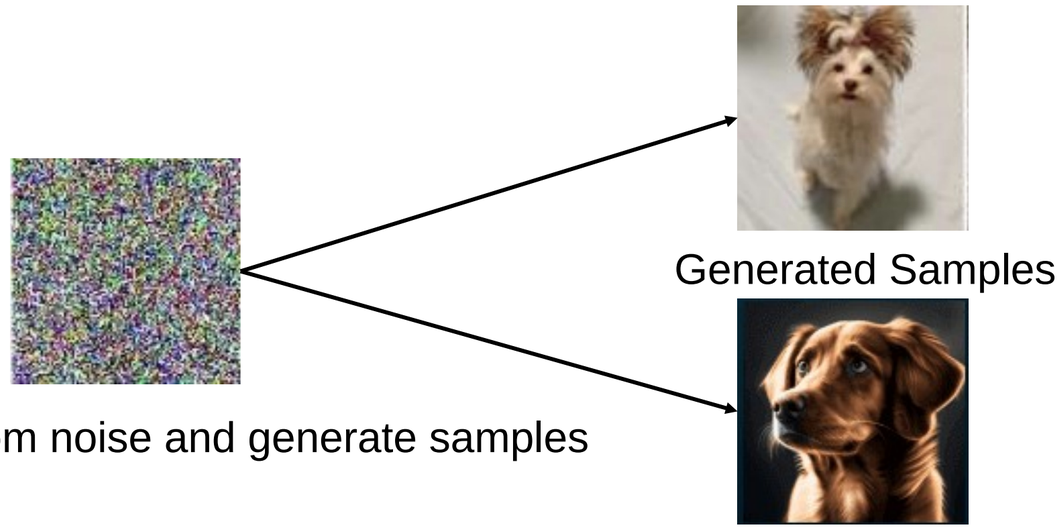
What if there was only noise?

- Denoising Models can very successfully remove this gaussian noise from an image
- We've also seen that with techniques like GANs, we can take a random vector (i.e a vector of noise) and generate new samples

What if we combine these two ideas… A generative architecture based on a denoising model… ?



Instead of just denoising..



Take random noise and generate samples

Diffusion Models

The Forward Diffusion Process – Adding noise to an image

The forward diffusion process is a Markov chain starting from the original data x and ending at a noise sample ϵ.

- At each step t, the data is progressively corrupted by adding Gaussian noise.
- Data is progressively corrupted until it becomes pure noise.
- By step T, the data x_t is entirely random and independent of x.

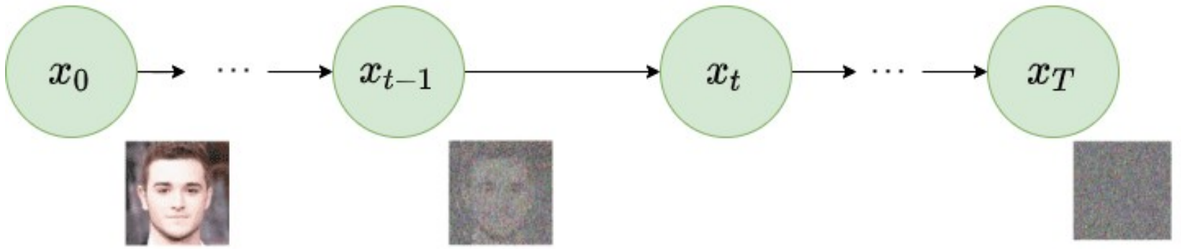
Forward Diffusion Equation

$$x_t = \sqrt{1-\beta_t} \cdot x_{t-1} + \sqrt{\beta_t} \cdot \eta_t$$

x_t : Data at step t

β_t : Noise level at step t

η_t : Standard Gaussian random variable



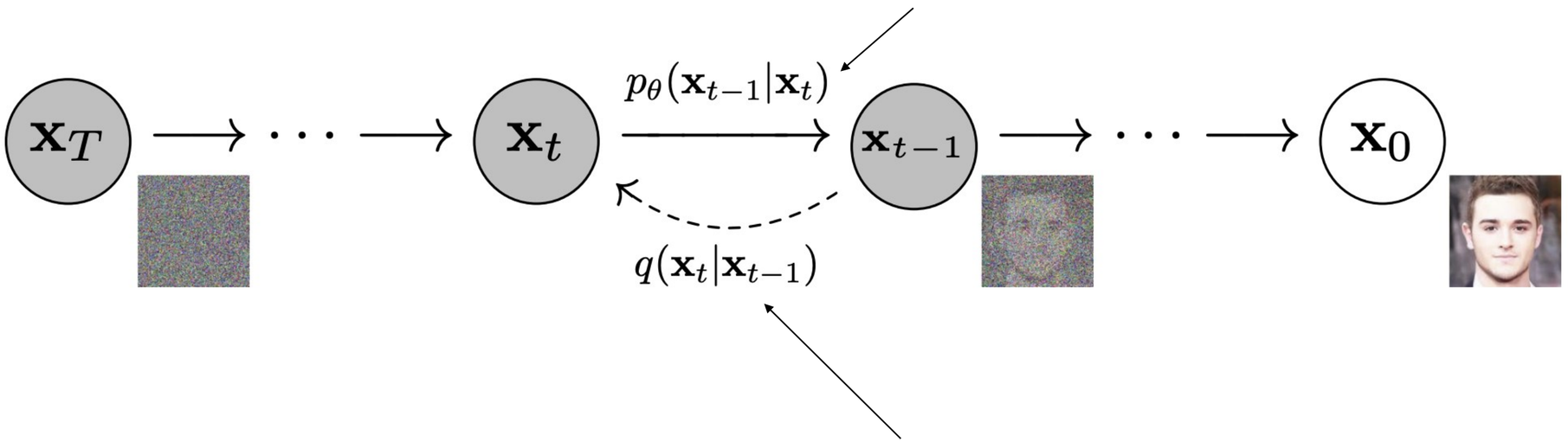
Note: A Markov chain is a process whereby each state depends only on the previous state.

The Reverse Diffusion Process – Removing Noise

The forward process is fairly simple and straight forward:
Take an image and progressively add noise until the structure of the image is gone and only pure noise remains.

But we want to ***reverse*** this process. This will allow us to generate images from noise. But how?

This is the reverse process that we need to solve for



This is the forward process (aka the approximate posterior that added noise)

The Reverse Diffusion Process – Learned Gaussian Transitions

We know that the final state of the data is a Gaussian distribution:

$$p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$$

We can then define the joint distribution over the whole reverse process, from noise at t=T to the noise-free image at t=0:

$$p_\theta(\mathbf{x}_{0:T}) := p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$$

To sample an image along this process, $\mathbf{x}_{t-1} \sim p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ we can use:

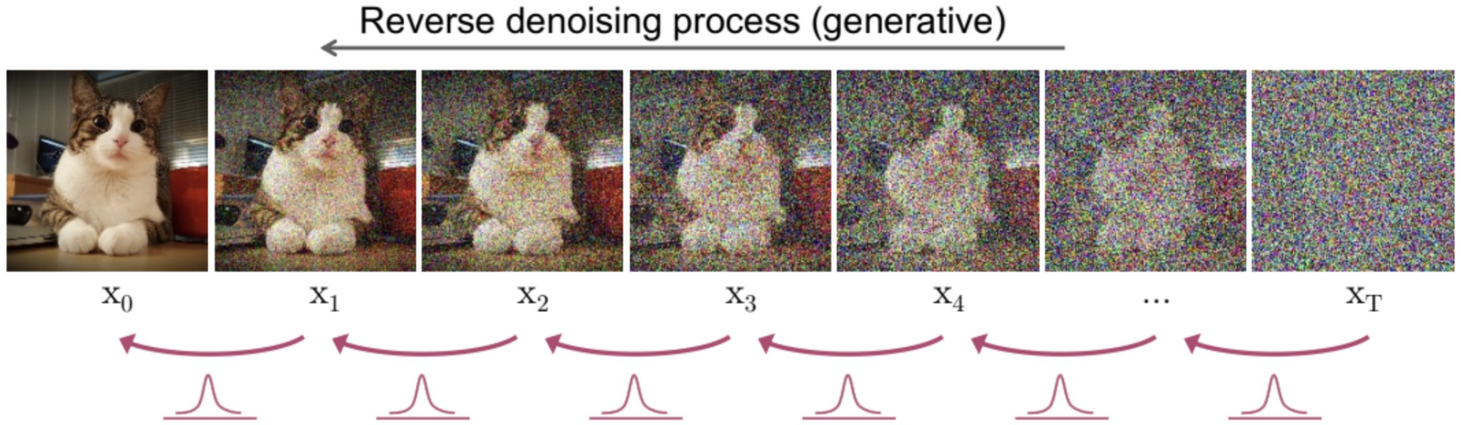
$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}, \text{ where } \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

Where $\alpha_t := 1 - \beta_t$ and $\bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$ and $\boldsymbol{\epsilon}_\theta$ is the approximated noise at step t

The Denoising Diffusion Probabilistic Model

The DDPM paper, the first diffusion model, utilized a UNet architecture to take as input a noisy image, and was trained to remove the right amount of noise for each step.

In training, the model had access to the exact amount of noise at each step for each training sample. The mean squared error between the noise predicted and the true noise to remove was used as the loss function. Having access to the exact amount of noise is one of the benefits of the closed form of the joint distributions that the forward process affords.



$$p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$$
$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \underbrace{\mu_\theta(\mathbf{x}_t, t)}_{\text{Trainable network (U-net, Denoising Autoencoder)}}, \sigma_t^2\mathbf{I})$$

$$\Rightarrow \quad p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$$

Pros and Cons of Diffusion Models

Pros of Diffusion Models

High-Quality Outputs:

Diffusion models can generate high-quality, detailed images, often rivaling or surpassing the output quality of GANs (Generative Adversarial Networks).

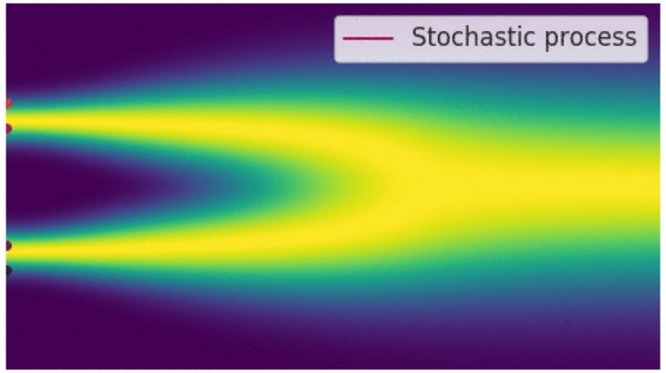
Rich Sampling Process:

The iterative nature of the denoising process allows for fine-grained control over the generation, potentially leading to better sample diversity and quality.

Pros of Diffusion Models

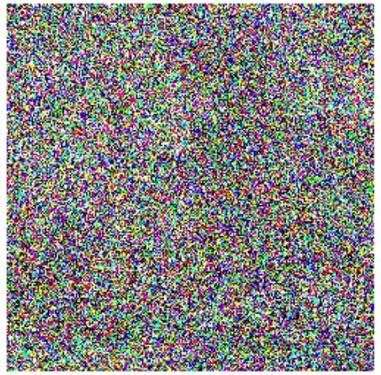
Stable Training

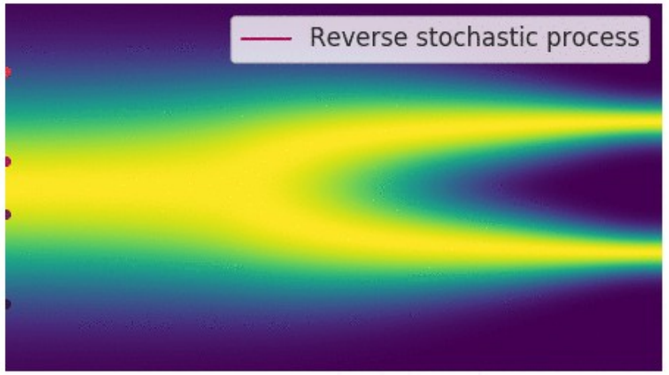
Diffusion models are generally more stable to train compared to GANs, which often suffer from issues like mode collapse and unstable training dynamics.



Theoretical Soundness

The probabilistic framework of diffusion models is well-understood, providing a solid theoretical foundation. This framework allows for explicit control over the generation process.

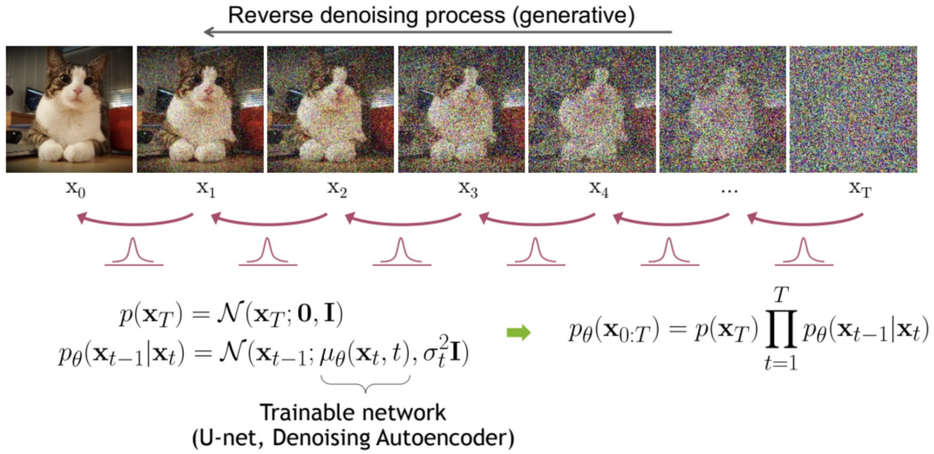




Cons of Diffusion Models

Computationally Intensive

The iterative denoising process requires a large number of forward passes through the network, making it computationally expensive and time-consuming compared to models like GANs that generate images in a single forward pass.



Complex Training Process

The training process can be complex and requires careful tuning of hyperparameters, such as the noise schedule and the loss function, to achieve optimal performance.

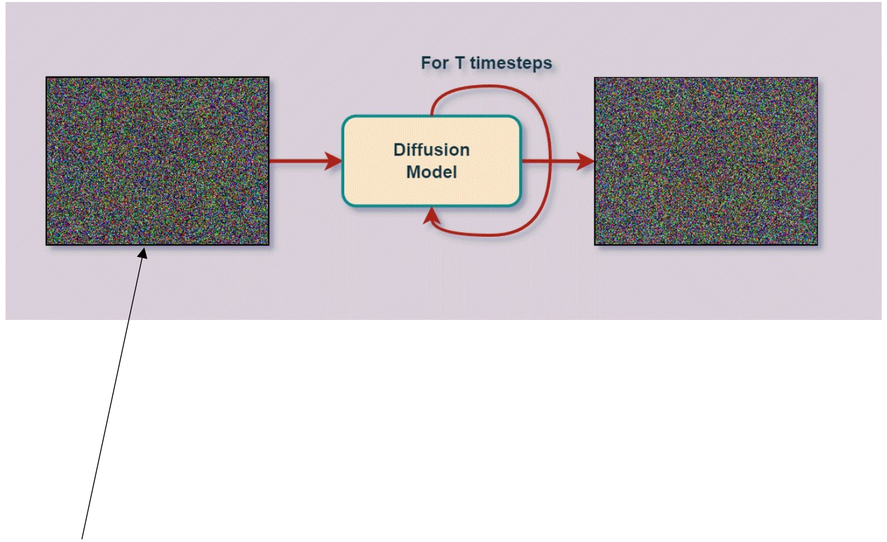
Cons of Diffusion Models

Slow Inference

Due to the iterative nature, generating a single image can take significantly longer compared to other generative models.

Difficulty in Direct Manipulation

Traditional diffusion models do not offer straightforward ways to manipulate specific attributes of the generated data.



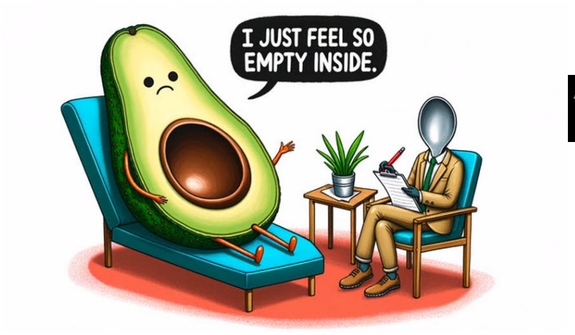
Only input is a noise field

Potential Improvements

Improvements to traditional Diffusion Methods

- In the next lesson, we'll look at Latent Diffusion Models, like Stable Diffusion

- These models:
  - Incorporate other information to guide the creation of images
  - Utilize diffusion on the latent space to improve speed and performance



An illustration of an avocado sitting in a therapist's chair, saying 'I just feel so empty inside' with a pit-sized hole in its center. The therapist, a spoon, scribbles notes.

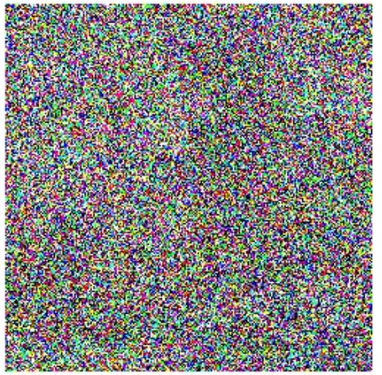
Next time…

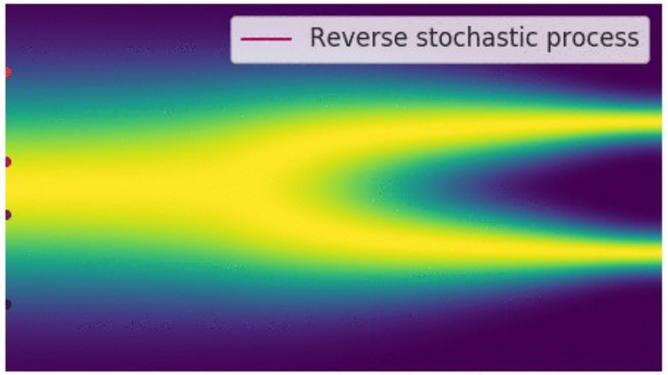
Wrap Up

UNets and Diffusion Models

- Today we covered new topics in generative image AI
- We introduced the UNet architecture
- Discussed the problem of denoising images
- Discovered the diffusion process to reverse the denoising process to create new images

---





In the next lesson we will start our discussion on latent diffusion models and how they can overcome some of the traditional diffusion model limitations.

Thank you!