Diffusion Models (the tech behind Stable Diffusion, DALL·E, Imagen)

1. What Are Diffusion Models?

- Instead of learning to generate images directly (like GANs/VAEs), diffusion models learn by **gradually destroying data with noise and then learning to reverse the process**.
- Think of it like:
    1. Take a clean image (e.g., a cat).
    2. Add Gaussian noise step by step until it's pure noise.
    3. Train a model to denoise — step by step — until it reconstructs the original image.
- Once trained, you can start from **random noise** and denoise step by step → generating a new image.
- That's how **Stable Diffusion** creates high-quality images from text prompts.

2. Training Process:

- **Forward Process (Diffusion)**:
  Image → add noise over T steps until it's Gaussian noise.
- **Reverse Process (Denoising)**:
  Train a neural network to predict the noise at each step, then remove it gradually.

Equation (simplified): $x_t = \sqrt{\alpha_t} * x_0 + \sqrt{1-\alpha_t} * noise$

where:

- x_t = noisy image at step t
- x_0 = original image
- Model learns to estimate the noise and subtract it.

3. Key Innovations:

- **DDPM (Denoising Diffusion Probabilistic Models)** — base idea (Ho et al., 2020).
- **DDIM (Deterministic Diffusion)** — fewer steps, faster generation.
- **Latent Diffusion (Stable Diffusion)** — instead of generating pixels directly, they generate compressed **latent features**, making it efficient.
- **Conditioning** — add text (via CLIP or Transformer) → allows text-to-image.

4. Applications:

- **Text-to-Image**: Stable Diffusion, DALL·E, Imagen.
- **Image Editing**: Inpainting, super-resolution.
- **Video Generation**: Consistency over frames.
- **Molecule Generation**: Drug discovery.

5. Expected Outputs:

- Unlike GANs, which often fail or collapse, diffusion models consistently produce **ultra-realistic, diverse outputs**.
- Example: Given "a cat in space wearing a spacesuit," Stable Diffusion can generate 100 different unique, high-quality variations.

Diffusion Models — Deep Explanation

1. Forward Diffusion (Noise Process):

We take a real image x_0 (say a handwritten digit) and **gradually add Gaussian noise** over T steps until it becomes pure noise.

Equation:

xt=αt⋅x0+1−αt⋅ϵ,ϵ∼N(0,I)x_t = \sqrt{\alpha_t} \cdot x_0 + \sqrt{1-\alpha_t} \cdot \epsilon, \quad \epsilon \sim N(0, I)xt=αt⋅x0+1−αt⋅ϵ,ϵ∼N(0,I)

- x_t → noisy image at time step t
- α_t → noise schedule (controls how much noise is added at each step)
- ε → random Gaussian noise
- After enough steps, x_T ≈ N(0, I) (pure noise).

2. Reverse Diffusion (Denoising Process):

We train a neural network **εθ(x_t, t)** to predict the noise we added at step t.

- Once we estimate the noise, we can subtract it → reconstruct a cleaner version of the image step by step.
- So, training objective is: L=E[||ϵ−ϵθ(xt,t)||2]L = E[|| \epsilon - \epsilon_\theta(x_t, t) ||^2]L=E[||ϵ−ϵθ(xt,t)||2]
- This is just an **MSE loss** between true noise and predicted noise.

3. Architecture (UNet):

The most common model used in diffusion is **U-Net**:

- Encoder (downsamples image) → captures features.
- Bottleneck with attention layers (captures global info).
- Decoder (upsamples back to image size).
- Skip connections → keep spatial details.

This makes the model very good at image denoising.

4. Noise Schedule

We can't add/remove all noise in one step — too hard. So, we spread it across many steps with a **noise schedule**:

- **Linear schedule**: add small noise increments.
- **Cosine/beta schedules**: more advanced, give better results.

5. Sampling:

- Start from pure noise x_T.
- Iteratively denoise using the trained network.
- After T steps, we get a realistic image.