Lecture 19: Diffusion Models - II

COMP 5801H/4900A: Generative AI and LLMs

2026-03-16

Sriram Subramanian
Assistant Professor & Canada Research Chair, Carleton University
Faculty Affiliate, Vector Institute for Artificial Intelligence
Faculty Affiliate, Schwartz Reisman Institute for Technology and Society

Carleton University

VECTOR INSTITUTE

SCHWARTZ REISMAN INSTITUTE FOR TECHNOLOGY AND SOCIETY

Conditional Generation

- **The Concept**: Up to this point, our model is "unconditional"—it generates a random high-quality image, but we have no control over whether it's a cat, a car, or a sunset. To make Diffusion useful, we must "condition" the U-Net on external information (y).

- **Types of Conditioning** (y)
  - Text Prompts: "A golden retriever wearing a space helmet."
  - Class Labels: "Category: 704 (Canary)."
  - Images: Using an existing sketch or a depth map to guide the structure.
  - In-painting Masks: Telling the model only to regenerate a specific area of an image.

How the U-Net "Sees" the Condition

- The conditioning information (y) is injected into the U-Net alongside the time embedding. There are two primary mechanisms:
  - **Concatenation**: For spatial guidance (like a mask or a sketch), we simply "stack" the condition on top of the noisy image as extra input channels.
  - **Cross-Attention**: For abstract concepts (like text), we use **Attention Layers**. The U-Net "looks" at the text tokens while processing the pixels, asking: "Which pixels in this noisy cloud should become 'Golden Retriever' fur based on this prompt?"
- **The Conditional Objective**
  - The training goal changes slightly. Instead of just predicting noise based on x_t and t, the model now predicts noise based on x_t, t, and y:

$$\epsilon_\theta(x_t, t, y)$$

Guidance: Turning up the Volume

- Simply adding a condition isn't always enough; the model might ignore the prompt. We use **Guidance** to force the model to prioritize the prompt:
  - **Classifier Guidance**: Uses a separate AI "judge" to tell the generator if it's getting closer to the goal.
  - **Classifier-Free Guidance (CFG)**: The modern standard. The model is trained both with and without the prompt, and during generation, we "amplify" the difference between the two to make the prompt's influence much stronger.

Classifier Guidance

- The Concept: In the early days of conditional diffusion, researchers didn't want to retrain the entire U-Net just to add control. Instead, they used a "shortcut": they took a separate, already-perfected Image Classifier (like a ResNet or ViT) and used its "knowledge" to steer the diffusion process in real-time.
- **How it Works: The Feedback Loop**
  - Imagine the U-Net is an artist and the Classifier is a critic.
    - **The Artist (U-Net)** produces a noisy denoised estimate (x_{t-1}).
    - **The Critic (Classifier)** looks at that noisy image and says: "I only see a 20% chance that this is a 'Golden Retriever'."
    - **The Gradient**: The critic calculates the gradient—the exact direction the pixels need to change to make that 20% become a 100%.
    - **The Nudge**: We "nudge" the U-Net's predicted mean in that direction before taking the next step.

The Mathematical "Shift"

- We modify the reverse step by adding a term proportional to the gradient of the classifier's log-probability:

$$\hat{\mu}(x_t) = \mu_\theta(x_t) + s \cdot \sigma_t^2 \nabla_{x_t} \log p_\phi(y \mid x_t)$$

  - $p_\phi(y \mid x_t)$: The probability the classifier assigns to the label y.
  - s **(Guidance Scale)**: The "volume" of the critic. If s is high, the model follows the critic strictly, resulting in very "typical" looking images (high fidelity) but with less variety (low diversity).

Classifier-Free Guidance (CFG)

- **The Concept**: While Classifier Guidance worked, it was clunky. In 2022, Ho & Salimans introduced **Classifier-Free Guidance (CFG)**. Instead of using an external model, we train the U-Net to be its own "critic." This is the industry standard used by DALL-E 3, Midjourney, and Stable Diffusion.
- **The Training Trick: Dropout**
  - During training, we don't always give the model a prompt.
    - **10-20% of the time**: We give the model a "null" prompt (an empty string or a special "unconditional" token).
    - **80-90% of the time**: We give it the actual text prompt.
    - **Result:** The model learns two things simultaneously: how to generate any image (unconditional) and how to generate a specific image (conditional).

The Inference Magic: The Vector Push

- During sampling, we ask the U-Net for two noise predictions at every single step:
  - ϵ_{cond}: What the noise should look like with the prompt.
  - ϵ_{uncond}: What the noise should look like without any prompt.
- We then calculate the final noise by pushing the "unconditional" prediction away from the "conditional" one:

$$\epsilon_{final} = \epsilon_{uncond} + w \cdot (\epsilon_{cond} - \epsilon_{uncond})$$

- w **(Guidance Scale)**: This is the "volume" slider.
- If $w = 1$: Standard conditional generation.
- If $w > 1$: The model "exaggerates" the prompt features, making the image extra vibrant and strictly aligned with your words.

Why CFG Won the War

Feature	Classifier Guidance	Classifier-Free Guidance (CFG)
Setup	Requires 2 separate models.	1 model does everything.
Noise Robustness	Classifier often breaks on noisy data.	Inherently understands noise.
Perceptual Quality	Often creates "noisy" artifacts.	Produces cleaner, more saturated images.
Adversarial Issues	Easy to "fool" the classifier.	Harder to fool; more "honest" to the data.

CLIP (Contrastive Language-Image Pretraining)

- The Problem: A U-Net is great at moving pixels, but it doesn't "speak" English. To make Text-to-Image work, we need a bridge that can translate a sentence like "A cat in a hat" into a mathematical language the U-Net can understand. That bridge is CLIP (OpenAI, 2021).
- **Contrastive Learning: The "Match Game"**
  - CLIP was trained on 400 million pairs of images and their captions from the internet. Instead of trying to "draw," it was trained to **match**:
    - It has a **Text Encoder** and an **Image Encoder**.
    - **The Goal**: If you feed it a picture of a dog and the word "Dog," the mathematical vectors (embeddings) for both should be nearly identical.
    - **The Contrast**: If you feed it a picture of a dog and the word "Car," the vectors should push as far away from each other as possible.

Embedding Space: The Map of Human Concepts

- CLIP creates a "Map" (Latent Space) where every concept is a coordinate.
  - In this space, the vector for the word "**Cold**" is physically near the pixels for "**Blue," "Ice," and "Snow.**"
  - Because this map is multidimensional, CLIP understands nuances: it knows that "Apple" can mean a fruit or a computer company depending on the context of the other words around it.
- **Integration: CLIP meets Diffusion**
  - In a model like **Stable Diffusion**, CLIP acts as the "Director":
    - **Input**: You type a prompt.
    - **CLIP Text Encoder**: Turns your words into a series of high-dimensional vectors (Embeddings).
    - **Cross-Attention**: These vectors are fed into the U-Net. At every layer, the U-Net "consults" the CLIP embeddings to decide: "Do these pixels look enough like the 'Golden Retriever' vector CLIP described?"

Why CLIP is the Secret Sauce

Feature	Without CLIP	With CLIP
Understanding	Struggles with complex descriptions.	Understands composition, style, and lighting.
Zero-Shot	Can only draw things it was specifically labeled for.	Can draw "A dinosaur riding a unicycle" even if it never saw that specific image.
Style Transfer	Hard to replicate artist styles.	Knows the "Vibe" of Van Gogh or Pixar.

The Text-to-Image Pipeline

- **The Big Picture**: Creating an image is not a single calculation; it is a relay race. Information is passed from a language model to a noise predictor, and finally to an image decoder. This is the standard architecture used by **DALL-E, Midjourney, and Stable Diffusion.**
- The Four Pillars of the Pipeline
  - **The Prompt (User Input)**: The human-readable intent (e.g., "An astronaut riding a horse on Mars").
  - **The Text Encoder (CLIP)**: Translates the prompt into a "Language Embedding", a mathematical map that the U-Net can understand.
  - **The Denoiser (U-Net)**: The engine that looks at a noisy canvas and uses the CLIP embedding as a guide to "carve out" the requested shapes.
  - **The Decoder (VAE)**: Takes the final "blueprint" generated by the U-Net and turns it into high-resolution, viewable pixels.

The Step-by-Step Flow

- **Tokenization**: Your text is broken into tokens and converted into a vector by CLIP.
- **Initial Noise**: A random "seed" is generated. This is your x_T (the blank, noisy canvas).
- **The Iterative Loop**:
  - The **U-Net** looks at the noise.
  - It checks the **CLIP embedding** to see what it's supposed to draw.
  - It subtracts a little bit of noise.
  - **Repeat**: This loop runs 20–50 times (using the **DDIM/Scheduler** logic).
- **The Final Reveal**: Once the noise is gone, the latent representation is passed through a **VAE Decoder** to produce the sharp, coloured image you see on your screen.

Why this Pipeline is Revolutionary

Step	What it solves
CLIP	Solves the "Meaning" problem (understanding the prompt).
U-Net	Solves the "Structure" problem (arranging the pixels).
Iterative Refinement	Solves the "Quality" problem (fixing errors step-by-step).
VAE (Latent Space)	Solves the "Speed" problem (working on compressed data).

Pixel Space vs. Latent Space

- **The Challenge**: High-resolution images contain a staggering amount of data. A single 1024×1024 RGB image has over 3 million data points. Trying to run a U-Net on this raw "Pixel Space" at every one of the 50+ diffusion steps requires massive amounts of VRAM and processing power, far more than a consumer GPU can handle.
- **The "Pixel Space" Problem**
  - **Redundancy**: Most pixels in an image are redundant. If you know a pixel is part of a clear blue sky, the 100 pixels next to it are likely also blue.
  - **Cost**: The computational complexity of self-attention layers in the U-Net scales **quadratically** with the number of pixels. Doubling the resolution makes the model 4x slower and 4x more memory-hungry.
  - **Noise**: At high resolutions, the model spends too much time trying to denoise "imperceptible" high-frequency jitter rather than focusing on the actual shapes.

The Solution: Latent Space

- Instead of diffusing the pixels, we use a **Variational Autoencoder** (VAE) to "squish" the image into a mathematically dense, lower-dimensional representation called **Latents**.
  - **Compression**: A 512×512 image is compressed into a 64×64 latent grid.
  - **Meaning over Mess**: The VAE discards the "boring" redundant data and keeps only the essential semantic information (shapes, colors, textures).
  - **The Math**: We run the Diffusion process **entirely within this small** 64×64 **space.**

Pixel Space vs. Latent Space

Feature	Pixel Diffusion (DALL-E 2)	Latent Diffusion (Stable Diffusion)
Resolution	Works directly on 256 X 256 or higher.	Works on 64 X 64 (then upscales).
GPU Requirement	Enterprise-grade (A100/H100)	Consumer-grade (RTX 3060/ Laptops)
Training Speed	Very slow.	Much faster (compact data)
Detail Focus	Struggles with global structure.	Excellent at global structure and composition.

Stable Diffusion (LDM)

- **The Breakthrough**: Stable Diffusion is a **Latent Diffusion Model (LDM)**. Its secret is that it doesn't work with pixels directly. Instead, it performs the entire "noisy" diffusion process in a hidden, mathematical space that is much smaller and more efficient.
- **Stage 1: Perceptual Compression (The VAE)**
  - Before any noise is added, a Variational Autoencoder (VAE) acts as a high-powered "zipper":
    - **The Encoder (E)**: Compresses a 512×512 pixel image into a 64×64 **Latent Grid**.
    - **Compression Factor**: This is an $8 \times$ **spatial reduction** (and a $64 \times$ total pixel reduction).
    - **The Goal**: To strip away "imperceptible" high-frequency noise (like individual grain) and keep only the "semantic" essence (shapes, lighting, and concepts).

Stable Diffusion (LDM)

- **Stage 2: Latent Diffusion (The U-Net)**
  - Now that the image is small (64×64), the U-Net does its work.
    - **Efficient Denoising**: The U-Net predicts noise in this compressed space. Because the data is $64 \times$ smaller, the math is significantly faster.
    - **Semantic Focus**: Because the VAE has already removed the "pixel jitter," the U-Net can focus entirely on the **logic** of the image (e.g., ensuring a face has two eyes) rather than worrying about individual sub-pixels.

Summary of the Flow

- **Pixel Space**: High-resolution image (512×512).
- **Encoder**: "Compresses" it to a Latent (64×64).
- **Diffusion**: Add/Remove noise in the **Latent Space**.
- **Decoder**: "Decompresses" the final latent back into a high-res image.

Inpainting & Outpainting

- **The Concept**: Because Diffusion is an iterative process of "filling in" noise, we can provide the model with a **Mask** to tell it exactly which parts of an image to change and which parts to keep frozen. This transforms the AI from a simple generator into a surgical editor.
- **Inpainting: The "Eraser" Tool**
  - Inpainting is the process of replacing or repairing a specific area within an existing image.
    - **The Mask:** The user paints a white area (the mask) over the part of the image they want to change (e.g., changing a shirt's colour or removing a person).
    - **The Process:** The U-Net is told to only "denoise" the pixels under the white mask. It uses the surrounding unmasked pixels (the context) to ensure the new content matches the lighting, style, and shadows of the original.

Inpainting & Outpainting

- **Outpainting: Extending the Borders**
  - Outpainting (or "Generative Fill") allows the model to imagine what exists outside the original crop of a photo.
    - **The Workflow**: We take the original image and place it on a larger, empty canvas. The empty space is filled with pure noise.
    - **The Extension**: The U-Net "bleeds" the patterns from the edges of the real photo into the noisy void. It maintains the horizon line, perspective, and colour palette to create a wide-angle version of a narrow shot.

How it Works (Under the Hood)

- There are two primary ways the U-Net handles this:
  - **Conditioning on Masks**: The mask is passed as an additional channel (like an extra "colour" layer) so the U-Net knows exactly where the "edit zone" begins.
  - **Latent Blending**: At every step of the diffusion loop, the model takes the unmasked pixels from the original image and "pastes" them back over the generation. This prevents the AI from accidentally drifting away from the parts of the photo you wanted to keep.

Use Cases

Feature	Creative Application
Object Removal	Deleting a "Photo-bomber" or a power line from a landscape.
Style Swap	Changing a character’s clothing while keeping their face identical.
Aspect Ratio Fix	Turning a vertical phone photo into a cinematic horizontal shot.
Restoration	Repairing scratches or missing pieces in historical photographs.

Video & 3D Diffusion

- **The Concept**: Once we mastered 2D images, the next frontier was adding more dimensions. For **Video**, we add a temporal dimension (T). For **3D**, we optimize spatial volumes using 2D knowledge.
- **Video Diffusion: Expanding the T Dimension**
  - To turn an image generator into a video generator, we don't just generate 24 independent images (which would flicker like crazy). We must ensure Temporal Consistency.
  - **VideoLDM (Video Latent Diffusion)**:
    - It takes a pre-trained **Stable Diffusion** model and "inflates" it.
    - We insert **Temporal Layers** (1D convolutions or temporal attention) between the existing spatial layers.
    - These layers "look" across frames to ensure that a cat's ear in Frame 1 stays in the same place in Frame 2.

Video & 3D Diffusion

- **3D Diffusion: DreamFusion & NeRFs**
  - There isn't enough 3D data in the world to train a "3D U-Net" from scratch. Instead, **DreamFusion** (Google, 2022) uses a 2D diffusion model as a "Critic" to create 3D objects.
    - **The Process (Score Distillation Sampling - SDS):**
      - Start with a random 3D shape (a **Neural Radiance Field** or **NeRF**).
      - Take a "snapshot" of it from a random angle.
      - Add noise to that snapshot and ask a 2D Diffusion model (like Imagen or Stable Diffusion) to denoise it.
      - The 2D model says: "*This looks like a blurry blob, it should look more like a Viking ship.*"
      - **The Nudge**: We update the 3D shape so that its snapshots look more like what the 2D model expects.
      - **Repeat** from thousands of angles until a 3D model emerges.

Dimensionality Comparison

Feature	Image Diffusion (2D)	Video Diffusion (3D)	3D Diffusion (Volumetric)
Grid Size	H x W	H x W x T	X x Y x Z
Main Challenge	Sharp Details	Temporal Smoothness	Multi-view Consistency
Key Tech	2D U-Net	3D U-Net/Temporal Attention	Score Distillation

Summary & Key Takeaways

- **The Essence**: Diffusion models represent a shift from "One-Shot" generation (like GANs) to **Iterative Refinement**. By breaking the complex task of creation into 1,000 tiny steps of denoising, we achieve unprecedented stability, quality, and control.
- **The Core Architecture (The Trinity)**
  - Modern Text-to-Image (like Stable Diffusion) relies on three interconnected systems:
    - **CLIP (The Brain)**: Connects human language to visual concepts.
    - **U-Net (The Engine)**: An iterative denoiser that uses **Skip Connections** for spatial precision and **Cross-Attention** to listen to the prompt.
    - **VAE (The Efficiency)**: Compresses high-res pixels into **Latent Space** so the math can run on a consumer GPU.

Summary & Key Takeaways

- **The Generative Lifecycle**
  - **Forward Process**: Adding Gaussian noise to data until it becomes pure static (Fixed/Physics-based).
  - **Reverse Process**: Learning to "predict the noise" at any given timestep to recover the data (Trainable/Neural).
  - **Inference**: Starting with random noise and "carving" an image out through an **ODE/SDE Solver** (The Sampler).

Final Thought: Why Diffusion Won

"Before Diffusion, AI generation was a 'black box' that often collapsed or glitched. Diffusion turned generation into a predictable, physical process. It is the first technology that allows us to treat pixels like clay—something we can mold, mask, extend, and refine until it perfectly matches our imagination."