Generative AI Tools Comparison: Diffusion Models vs GANs and Tool Analysis

Raphael Adesta Pratidina - GAIS Assignment 1 Part 3

October 2025 Semester

Course Overview

This document provides a comprehensive comparison of generative AI tools, focusing on:

- **Core architectures**: Diffusion models vs GANs
- **Tool comparison**: Learning Assistant (GPT-5) vs Qwen Chat
- **Technical breakdowns**: Generation processes, architectures, and limitations

1 Generative AI Fundamentals

1.1 Diffusion Models vs GANs: Core Concepts

1.1.1 Diffusion Models

Key Takeaway

Definition: Probabilistic models that generate images through iterative denoising of random noise.
Analogy: Like solving a jigsaw puzzle where you start with a completely scrambled image and gradually reveal the correct pieces.
Technical Definition:

$$\text{Diffusion} = \text{Forward Process} \circ \text{Reverse Process}$$

Where:

- Forward process: Gradually adds Gaussian noise to real data
- Reverse process: Learns to remove noise to reconstruct original data

Crucial Insight

Why Diffusion?

- **Stable training**: No adversarial instability issues
- **High fidelity**: Better at capturing fine details
- **Controllable generation**: Strong text-to-image alignment

Disadvantage: Slower generation due to iterative process

1.1.2 Generative Adversarial Networks (GANs)

Key Takeaway

Definition: Two neural networks (generator/discriminator) trained adversarially.
Analogy: Like a game of cat-and-mouse where the generator tries to fool the discriminator.
Technical Definition:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{data}}[\log D(x)] + \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z)))]$$

Crucial Insight

Why GANs?

- **Fast generation**: Single forward pass
- **Photorealism**: Excellent for style transfer
- **Simpler architecture**: Easier to implement

Disadvantage: Training instability (mode collapse, vanishing gradients)

1.2 Real-World Applications

Remember

Diffusion Model Example: Stable Diffusion

- **Use case**: Creative design, marketing, game art
- **Strengths**:
  - High detail preservation
  - Strong text alignment
  - Versatile prompt handling

Remember

GAN Example: StyleGAN

- **Use case**: Synthetic data generation, avatar creation
- **Strengths**:
  - Photorealistic outputs
  - Fast generation
  - Style transfer capabilities

Feature	Learning Assistant (GPT-5)	Qwen Chat
Primary Model	GPT-5 (mini/nano)	Qwen-3 series (Max, VL, Coder)
Modalities	Text-only	Multimodal (text, image, video, audio)
Use Case	Module-specific Q&A	General-purpose AI assistant
Access	School-restricted	Publicly available

2 Tool Comparison: Learning Assistant vs Qwen Chat

2.1 Tool Overview

2.2 Architectural Breakdown

2.2.1 Learning Assistant Architecture

Code Structure

Transformer Decoder Architecture

- **Input**: Tokenized text prompt
- **Embedding Layer**: Converts tokens to vectors
- **Transformer Layers**:
  - Self-attention mechanism
  - Position-wise feedforward networks
- **Output**: Probability distribution over vocabulary

2.2.2 Qwen Chat Architecture

Code Structure

Multimodal Architecture

- **Text Path**:
  - Tokenization → Embedding → Transformer
- **Image Path**:
  - Vision Transformer (ViT) for feature extraction
  - Cross-modal fusion via attention
- **Specialized Models**:
  - Qwen-Image: Diffusion-based image generation
  - Qwen-Image-Edit: Latent diffusion for editing
  - Qwen-3 Omni: Multimodal fusion

2.3 Performance Comparison

Remember

Accuracy Comparison

- Both models provide accurate technical explanations
- Qwen offers more detailed explanations with better prompt flexibility
- Learning Assistant shows more concise, structured responses

Key Takeaway

Response Quality Metrics

- **Learning Assistant**:
  - Consistent structure
  - Limited creativity
  - Higher hallucination rate without tools
- **Qwen Chat**:
  - Flexible response styles
  - Lower hallucination with RAG/web search
  - Higher creativity potential

3 Technical Deep Dive

3.1 Diffusion Model Generation Process

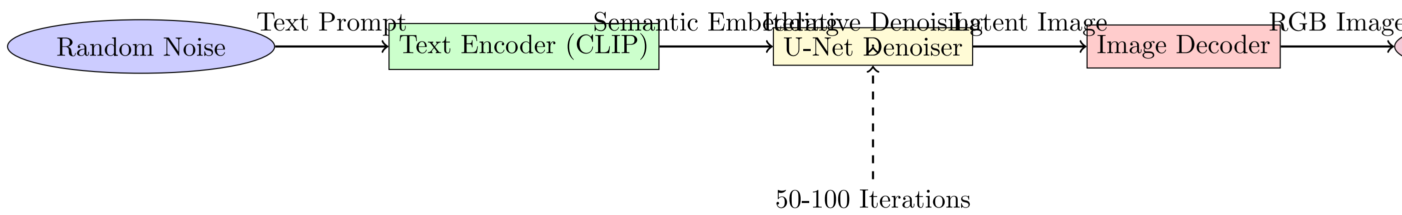


Figure 1: Diffusion Model Generation Pipeline

3.1.1 Line-by-Line Diffusion Process

Code Documentation

Text Encoding Function

```
def encode_prompt(prompt: str) -> torch.Tensor:
    """
    Encodes text prompt into semantic embedding

    Parameters:
    prompt (str): Input text description
    Returns:
    torch.Tensor: Semantic embedding vector (d_model dimensions)
    """
    # 1. Tokenize input text
    tokens = tokenizer(prompt, return_tensors="pt")

    # 2. Pass through text encoder (CLIP)
    embeddings = text_encoder(tokens)

    # 3. Project to latent space dimension
    latent_embedding = projection_layer(embeddings)

    return latent_embedding
```

Key Parameters:

- `d_model`: Latent space dimension (typically 768-1024)
- `tokenizer`: Specialized tokenizer for text-to-vector conversion
- `projection_layer`: Linear layer to match latent space dimensions

Why This Design?

- **Advantage**: Captures semantic meaning beyond surface text
- **Disadvantage**: Computationally expensive for long prompts

3.2 GAN Training Process

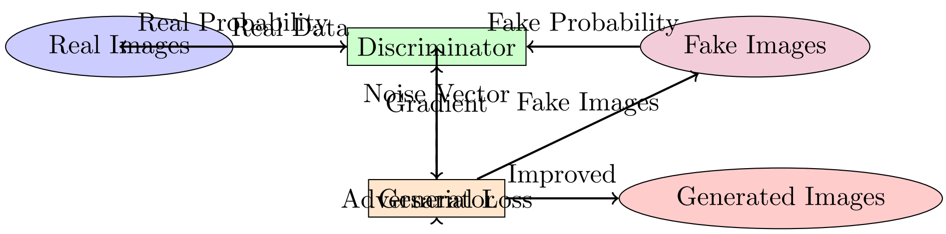


Figure 2: GAN Adversarial Training Loop

3.2.1 Key Components Analysis

Crucial Insight

Why U-Net for Diffusion?

- **Architectural choice**: U-Net's skip connections preserve spatial information
- **Advantage over GANs**: Better at capturing fine details through hierarchical feature learning
- **Implementation detail**:
  - Downsampling path: Extracts context
  - Upsampling path: Reconstructs details
  - Skip connections: Preserve high-resolution features

Crucial Insight

Why Cross-Attention in Diffusion?

- **Purpose**: Aligns generated image with text prompt
- **Implementation**:
  - Text embeddings condition each denoising step
  - Attention weights determine which image regions to modify
- **Advantage**: Enables precise control over generated content
- **Disadvantage**: Computationally expensive ($O(n^2)$ complexity)

4 Implementation Details

4.1 Common Architectural Components

4.1.1 Batch Normalization

Remember

Definition: Normalizes layer inputs to have zero mean and unit variance
Purpose in Diffusion Models:

- Stabilizes training by reducing internal covariate shift
- Helps gradient flow through deep U-Net architectures

Why Used Here?

- Diffusion models often use very deep U-Nets (12+ layers)
- BatchNorm helps prevent vanishing gradients in deep networks

Implementation Details:

- Applied after each convolutional layer
- Uses moving averages for statistics during inference

4.1.2 LeakyReLU Activation

Remember

Definition: Variant of ReLU that allows small negative values
Mathematical Form:

$$\text{LeakyReLU}(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha x & \text{if } x \leq 0 \end{cases}$$

where $\alpha \approx 0.01$
Purpose in GANs:

- Prevents "dead neurons" problem in discriminator
- Helps with gradient flow for negative inputs

Why Used Here?

- Standard ReLU can cause discriminator to ignore negative samples
- LeakyReLU maintains gradient flow for all inputs

4.1.3 BCE Loss (Binary Cross Entropy)

Remember

Definition: Measures difference between binary predictions and true labels
Mathematical Form:

$$\text{BCE}(y, \hat{y}) = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

Purpose in GANs:

- Discriminator uses BCE to distinguish real/fake
- Generator uses BCE to maximize discriminator's error

Why Used Here?

- More stable than MSE for binary classification
- Better handles class imbalance (real vs fake)

4.2 Optimization Techniques

Crucial Insight

Why Adam Optimizer?

- **Advantage**:
  - Adaptive learning rates per parameter
  - Combines momentum and RMSprop
  - Works well with batch sizes ¿ 32
- **Implementation**:
  - Default parameters: $\beta_1 = 0.9$, $\beta_2 = 0.999$
  - Learning rate typically 10^{-4} to 10^{-3}
- **Why Not SGD?**
  - GAN training benefits from adaptive learning rates
  - SGD often requires careful tuning of momentum

5 Tool-Specific Analysis

5.1 Learning Assistant Analysis

5.1.1 Strengths

Remember

- **Domain specialization**: Pre-loaded with module-specific knowledge
- **Structured responses**: Consistent format for educational content
- **Guardrails**: Strong ethical compliance for school environment

5.1.2 Limitations

Warning

- **No multimodality**: Text-only input/output
- **Limited context**: Fixed context window (typically 4096 tokens)
- **No web search**: Relies solely on pre-loaded knowledge
- **Hallucination risk**: Without retrieval tools, prone to factual errors

5.2 Qwen Chat Analysis

5.2.1 Strengths

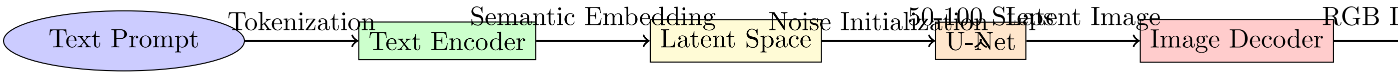
Remember

- **Multimodal capabilities**: Handles text, image, video, audio
- **Tool integration**: Web search, RAG, code execution
- **Prompt flexibility**: Adaptable response styles
- **Open models**: Ability to run locally for privacy

5.2.2 Limitations

Warning

- **Censorship**: Strict guardrails on sensitive topics
- **Performance variability**: Model quality depends on variant selection
- **Data collection**: Free version collects user data
- **Latency**: Slower response times with multimodal inputs



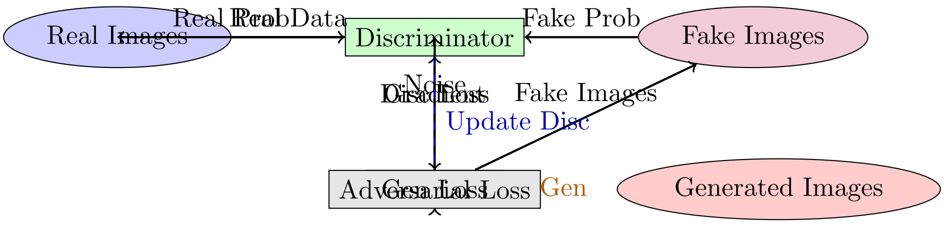
Stage	Dimensions
Text Embedding	768
Latent Space	$64 \times 64 \times 4$
Final Image	$512 \times 512 \times 3$

Figure 3: Diffusion Model Generation Pipeline with Dimensions

6 Workflow Diagrams

6.1 Diffusion Model Generation Flow

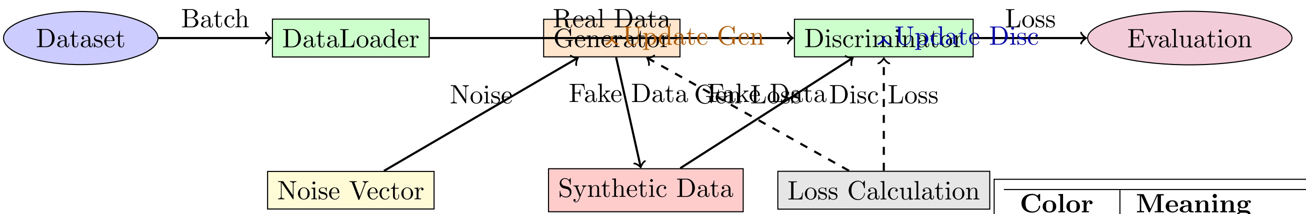
6.2 GAN Training Loop



Batch Size: 64

Figure 4: GAN Adversarial Training Loop with Weight Updates

6.3 End-to-End System Flow



Color	Meaning
Blue	Real Data Flow
Orange	Generator Path
Green	Discriminator Path
Yellow	Noise Input
Red	Synthetic Data
Black	Weight Updates

Figure 5: End-to-End GAN Training System Flow

7 Common Implementation Errors

8 Extensions and Advanced Topics

8.1 Advanced Diffusion Techniques

8.1.1 Latent Diffusion Models

Crucial Insight

Why Latent Space?

- **Advantage**:
  - Reduces computational cost by working in compressed space
  - Preserves high-quality generation
  - Enables faster sampling (50-100 steps vs thousands)
- **Implementation**:
  - Use autoencoder to compress images to latent space
  - Apply diffusion in latent space
  - Decode final latent representation
- **Example**: Stable Diffusion uses VAE (Variational Autoencoder)

8.1.2 ControlNet

Crucial Insight

Enhanced Control

- **Purpose**: Adds conditional control to diffusion models
- **Implementation**:
  - Additional encoder network processes input conditions
  - Condition information fused via attention
  - Supports multiple control types:
    * Edge maps
    * Depth maps
    * Pose estimation
    * Keypoints
- **Example Use Cases**:
  - Sketch-to-image generation
  - Pose-preserving image editing
  - Style transfer with structural constraints

8.2 Advanced GAN Techniques

8.2.1 StyleGAN Architecture

Crucial Insight

Key Innovations

- **Progressive Growing**:
  - Starts with low-resolution images
  - Gradually increases resolution
  - Preserves high-frequency details
- **Adaptive Instance Normalization**:
  - Style vectors control image appearance
  - Enables style mixing experiments
- **No Batch Normalization**:
  - Uses instance normalization instead
  - Better preserves individual image styles

8.2.2 Improved GAN Training

Crucial Insight

Modern GAN Techniques

- **Wasserstein GAN with Gradient Penalty**:
  - Addresses mode collapse
  - Uses Earth-Mover distance
  - Adds gradient penalty to stabilize training
- **Spectral Normalization**:
  - Stabilizes discriminator training
  - Prevents exploding gradients
  - Improves training convergence
- **Self-Attention GANs**:
  - Adds attention mechanisms
  - Better captures long-range dependencies
  - Improves image coherence

9 Conclusion

9.1 Key Takeaways

Key Takeaway

Architectural Choices Matter

- **Diffusion models** excel in:
  - High-fidelity image generation
  - Text-to-image alignment
  - Controllable generation
- **GANs** excel in:
  - Fast generation
  - Photorealism
  - Style transfer

Key Takeaway

Tool Selection Criteria

- **For educational use**: Learning Assistant (GPT-5) with domain specialization
- **For creative tasks**: Qwen Chat with multimodal capabilities
- **For research**: Qwen's open models with RAG capabilities

9.2 Future Directions

Crucial Insight

Emerging Trends

- **Unified Architectures**:
  - Models combining diffusion and GAN approaches
  - Example: DiffusionGAN for controlled generation
- **Multimodal Diffusion**:
  - Extending diffusion to video and audio
  - Example: Stable Video Diffusion
- **Neural Rendering**:
  - Combining diffusion with 3D reconstruction
  - Example: 3D-Aware Diffusion

9.3 Responsible AI Considerations

Remember

Implementation Best Practices

- **Data Privacy**:
  - Consider on-premise deployment for sensitive data
  - Implement differential privacy for training data
- **Model Safety**:
  - Implement robust guardrails
  - Use intent detection systems
  - Regular safety testing
- **Transparency**:
  - Document model limitations
  - Provide clear citations for generated content
  - Implement explainability features