A Mathematical Analysis of Diffusion Processes and Generative Adversarial Networks for Image Generation

Akhilan Elangovan
Yale University
akhil.elangovan@yale.edu

Vincent Zhu
Yale University
vincent.zhu@yale.edu

Abstract

Researchers once viewed generative adversarial networks and diffusion models as distinct families. GANs learn a generator and a discriminator by solving a minimax game. Diffusion models learn to reverse a noising process through successive denoising steps. Recent work blends these paradigms. We analyze two hybrid methods: class-guided diffusion, which steers the reverse process with classifier gradients, and Diffusion-GAN, which injects a diffusion process into adversarial training. We derive their core equations and evaluate the approaches. Our results show that hybrid models combine faithful data coverage with stable training. We conclude by discussing the next steps for these hybrid methods.

1 Introduction

Technical Overview of Generative Image Models

Generative image models learn a distribution $p_\theta(x)$ over the pixel space $\mathbb{R}^{H\times W\times C}$. We draw samples from p_θ so they match the true data distribution $p_{\text{data}}(x)$. At inference, we sample a latent vector

$$z \sim \mathcal{N}(0, I) \quad \text{or} \quad x_T \sim \mathcal{N}(0, I).$$

We then apply a learned mapping G_θ or run an iterative denoising procedure to produce a synthetic image $\hat{x}$. The mapping must capture multiple modes—variations in color, texture, and shape—while remaining efficient to train and evaluate.

Most methods fall into two categories. Explicit density models [5] define a latent-variable likelihood $p_\theta(x \mid z)$ with a simple prior $p(z)$, factorize the joint distribution as

$$p_\theta(x) = \prod_i p_\theta(x_i \mid x_{<i}),$$

or learn an invertible map $f_\theta \colon z \to x$ whose Jacobian determinant yields $\log p_\theta(x)$ exactly. These models train by maximizing surrogate likelihoods (like ELBOs) and they provide explicit estimates of $\log p_\theta(x)$. They struggle to produce sharp, high-frequency details in complex images.

Implicit and score-based models focus on sample quality rather than likelihood evaluation. They learn a score function or a generator that transforms noise into images through an iterative process. We will focus on this category in the paper.

Generative Adversarial Networks (GANs)

GANs consist of two neural networks, a generator G and a discriminator D, trained adversa rially. The generator G maps noise vectors $z \sim p_z(z)$ to synthetic images, while the discriminator aims

to distinguish between real and generated images. Mathematically, the training of GANs involves solving a minimax optimization problem [2]:

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

GANs have achieved remarkable success, producing high-quality and realistic images; however, challenges such as mode collapse and training instability remain significant limitations.

Diffusion Models

Diffusion models represent a fundamentally different generative approach. They define a forward noising process, gradually corrupting data points into noise, and subsequently learn a reverse denoising process. Formally, diffusion models are characterized by a forward Markov chain:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}\, x_{t-1}, \beta_t I), \quad t = 1, \ldots, T,$$

and a learned reverse process:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)),$$

[4] where parameters μ_θ and Σ_θ are typically parameterized by neural networks, where optimization is done by maximizing a variational lower bound on the data likelihood. This is equivalent to approximating a reverse-time stochastic differential equation. Generative image models face clear trade-offs: explicit likelihood versus sample quality, training stability versus computational cost, and mode coverage versus image sharpness. Researchers now blend methods from both camps—say, VAE–GAN hybrids or classifier-guided diffusion—to gain the best of each approach. Generative models drive machine learning in image tasks. Two leading methods in today's work are Generative Adversarial Networks (GANs) and diffusion models, which will be our primary focus in the paper.

Our Contributions

In this paper, we aim at determining how the fundamental differences of GANs and diffusion models inform recent innovations that hybridized these approaches for effective image generation. We first review results from two recent papers regarding the integration of elements from both diffusion models and GANs for image generation. Specifically, Dhariwal et al. [1] introduces the idea of assisting diffusion models using classifier guidance, at technique that is essential to GANs. On the other hand, Wang et al. [6] introduces the idea of adding a diffusion process to enhance the training of GANs.

We will then validate the experiments from the papers and conduct our own comparison of the hybridized models to standard diffusion models and GANs.

2 Current Findings

Diffusion Models Beat GANs on Image Synthesis

Dhariwal et al. aim to show that diffusion models are superior in image synthesis by innovating the training process. Specifically, they incorporate the classifier guidance, a concept stemming from GANs (where the classification of an image is important), to improve the diffusion model. Classifier guidance uses a separate image classifier to steer the diffusion process toward a label. At each denoising step t, we form the conditional transition by multiplying the unconditional diffusion kernel with the classifier likelihood:

$$p_{\theta,\phi}(x_t \mid x_{t+1}, y) = Z\, p_\theta(x_t \mid x_{t+1})\, p_\phi(y \mid x_t),$$

where $p_\theta(x_t \mid x_{t+1}) = \mathcal{N}(x_t; \mu, \Sigma)$, y is the class label, and Z normalizes the product.

Next, we approximate the classifier log-probability by a first-order Taylor expansion around $x_t = \mu$. Assuming $\|\Sigma\| \to 0$, we write

$$\log p_\phi(y \mid x_t) \approx \log p_\phi(y \mid \mu) + (x_t - \mu)^\top \nabla_x \log p_\phi(y \mid x)\big|_{x=\mu}.$$

We denote the gradient term by

$$g = \nabla_x \log p_\phi(y \mid x)\big|_{x=\mu}.$$

We add the Gaussian log-density and complete the square:

$$\begin{aligned}
&\log\big[p_\theta(x_t \mid x_{t+1})\, p_\phi(y \mid x_t)\big] \\
&\quad \approx -\tfrac{1}{2}\,(x_t-\mu)^\top \Sigma^{-1}(x_t-\mu) + (x_t-\mu)^\top g + C_2 \\
&\quad = -\tfrac{1}{2}\,(x_t-\mu-\Sigma\, g)^\top \Sigma^{-1}(x_t-\mu-\Sigma\, g) + \tfrac{1}{2}\, g^\top \Sigma\, g + C_2 \\
&\quad = -\tfrac{1}{2}\,(x_t-\mu-\Sigma\, g)^\top \Sigma^{-1}\,(x_t-\mu-\Sigma\, g) + C_3 \\
&\quad = \log p(z) + C_4, \quad z \sim \mathcal{N}(\mu+\Sigma\, g,\ \Sigma).
\end{aligned}$$

Because $p_{\theta,\phi}(x_t \mid x_{t+1}, y) = Z\, p_\theta(x_t \mid x_{t+1})\, p_\phi(y \mid x_t)$, we model the conditional kernel as

$$p_{\theta,\phi}(x_t \mid x_{t+1}, y) = \mathcal{N}\big(x_t;\ \mu+\Sigma\, g,\ \Sigma\big), \quad g = \nabla_x \log p_\phi(y \mid x)\big|_{x=\mu}.$$

This derivation shows that classifier guidance shifts the mean of the Gaussian by $\Sigma\, g$. The shift moves samples toward regions where the classifier assigns higher probability to the target label. Empirical results show that guided diffusion models outperform leading GANs in FID, sFID, precision, and recall.

Furthermore, scaling the classifier gradient by a factor s modifies the conditioning distribution without breaking its probabilistic interpretation. In fact, one shows

$$s\,\nabla_x \log p_\phi(y \mid x) \;=\; \nabla_x \log\big[p_\phi(y \mid x)^s\big] - \nabla_x \log Z \;=\; \nabla_x \log p_s(y \mid x),$$

where

$$p_s(y \mid x) \;=\; \frac{p_\phi(y \mid x)^s}{\displaystyle\int p_\phi(y \mid x)^s\, dx} \;\propto\; p_\phi(y \mid x)^s,$$

and Z is the normalizing constant. Thus classifier guidance with scale s uses the renormalized distribution $p_s(y \mid x)$.

When $s > 1$, raising $p_\phi(y \mid x)$ to the power s accentuates its peaks and diminishes its lower-probability regions. In effect, $p_s(y \mid x)$ becomes sharper than the original $p_\phi(y \mid x)$. The gradient step $\Sigma\, \nabla_x \log p_s(y \mid x)$ pushes samples more strongly toward the classifier's high-confidence modes. This yields higher-fidelity images that match the target label but reduces diversity. The tangible effects of the scaling factor is shown in Figure 1.

Figure 1: From Dhairwal et al.[1], a comparison of images generated under the class "Pembroke Welsh corgi" for a classifier scale factor of 1.0 (left) vs 10.0 (right)

Conversely, choosing $s < 1$ softens the distribution, spreading mass more evenly and producing more varied samples at the cost of precision. In practice, tuning s provides a direct trade-off between sample fidelity and diversity. With this tradeoff, we see in Figure 2 the effects of the scaling factor.

Diffusion-GAN: Training GANs with Diffusion

This paper, by Wang et al. [6], aimed at tackling some of the well known issues of GANs in practice such as non-convergence and training instability. As a result, they proposed Diffusion-GAN, a

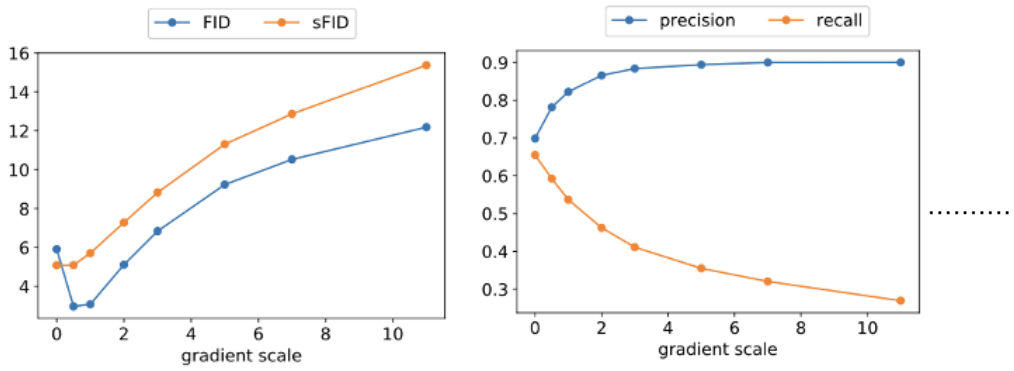

Figure 2: From Dhairwal et al., the FID (and sFID) is minimized at a gradient scale of about 1 for their class-conditional ImageNet 128×128 model. A comparison across classifier scaling factors which shows the tradeoff between precision and recall.

novel framework that integrates a forward diffusion process into GAN training to generate Gaussian-mixture distributed instance noise. This frameworks adds 3 main components to a normal GAN. First, the main addition is the diffusion process during training, which consists of adding Gaussian-mixture noise to both real and fake data (images), which the discriminator is trained to distinguish. Next, because the amount of noise increases at each time step, there is a separate discriminator for each time step. Finally, the generator is trained using gradients that are backpropogated through the diffusion process. An overview of these components can be seen in Figure 3.

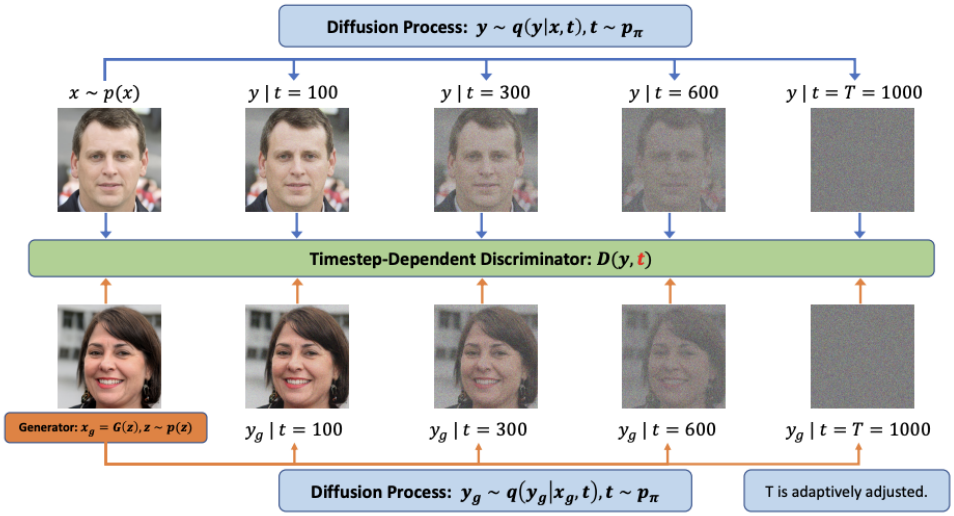

Figure 3: From Wang et al., a flowchart for Diffusion-GAN demonstrating changes of adding in the diffusion process to the training. In the top row are the real images and the bottom row are the fake images. As time moves from left to right, more noise is added to both images, and the discriminator seen in the middle learns to distinguish the images at each time step t.

We will now continue with a mathematical analysis of each step of the Diffusion-GAN and a theoretical analysis on the convergence of training with noise injection. Let $x \sim p(x)$ be a sample from the real data and $x_g \sim p_g(x)$ be a sample from the generated data. With the added noise, we let $y \sim q(y \mid x)$ be a noisy sample from the real data, and $y_g \sim q(y_g \mid x_g)$ be a noisy sample from the

generated distribution. Specifically, the paper defined the noisy distribution as

$$q(y \mid x) := \sum_{t=1}^{T} \pi_t q(y \mid x, t), \qquad q(y_g \mid x_g) := \sum_{t=1}^{T} \pi_t q(y_g \mid x_g, t)$$

where t represents the time step, and T is the total number of steps. Moreover, π_t are the mixture weights and are non-negative and sum to one. Essentially, the overall noisy distribution is a weighted average of the noisy distribution at each step t. Specifically, using properties of diffusion, they get $q(y \mid x, t)$ can be expressed as a Gaussian,

$$q(y \mid x, t) = \mathcal{N}(y; \sqrt{\bar{\alpha}_t}x, (1 - \bar{\alpha}_t)\sigma^2 I)$$

where β_t is a pre-defined variance schedule, $\alpha_t = 1 - \beta_t$, and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$. Next, to actually train the generator and discriminator, we have the following min-max objective,

$$V(G, D) = \mathbb{E}_{x \sim p(x), t \sim p_\pi, y \sim (y|x,t)}[\log(D_\phi(y, t)] + \mathbb{E}_{z \sim p(z), t \sim p_\pi, y_g \sim q(y|G_\theta(z),t)}[\log(1 - D_\phi(y_g, t))]$$

which the generator G tries to minimize, and the discriminator D tries to maximize. Note that $D(y, t)$ gives the probability that the discriminator believes y is a real sample. Thus, D maximizes the above by have large $D(y, t)$ and small $D(y_g, t)$. On the other hand, G tries to minimize the above by creating $y_g \sim q(y \mid G_\theta(x), t)$ samples that the discriminator mistakes for real images, thus making $D(y_g, t)$ large which decreases the overall function. Note here that t is drawn from a discrete probability distribution $p_\pi := \text{Discrete}(\pi_1, \ldots, \pi_T)$, meaning that t is sampled from $\{1, \ldots, T\}$ with probability π_t. Moreover, we can write the noisy generated sample $y_g \sim q(y \mid G - \theta(z), t)$ as

$$y_g = \sqrt{\bar{\alpha}_t} G_\theta(z) + \sqrt{(1 - \bar{\alpha}_t)}\sigma\epsilon, \epsilon \sim \mathcal{N}(0, I)$$

which shows that the objective function $V(G, D)$ above is differentiable with respect to the generator parameters. Thus, in the final step of the training, we can use gradient descent to optimize the generator with back-propagation. Thus, so far, we have shown the mathematical effects of adding a diffusion process to GAN training.

Next, the paper introduces some theoretical properties of Diffusion-GAN which demonstrate the improvements it makes over a baseline GAN. We will review some of these properties and proofs. First, observe that the objective function for Diffusion-GAN is similar to the original GAN objective function discussed in the introduction. They then show that this objective function minimizes an approximation of the Jensen-Shannon (JS) divergence with respect to the noisy samples and time step t,

$$\text{JS}(p(y, t) \| p_g(y, t)) = \mathbb{E}_{t \sim p_\pi}[\text{JS}(p(y \mid t) \| p_g(y, t))]$$

Recall that JS divergences is defined in terms of KL divergence as follows, for probability distributions $p : \mathbb{R}^d \to \mathbb{R}$ and $q : \mathbb{R}^d \to \mathbb{R}$,

$$\text{JS}(p \| q) = \frac{1}{2}\,\text{KL}(p \| r) + \frac{1}{2}\,\text{KL}(q \| r), \quad \text{where} \quad r = \frac{1}{2}(p + q)$$

Like all other measures of divergence, the JS divergence is zero if and only if the two distributions are the same. However, this only shows that the JS divergence is minimized with respect to the noisy distributions. A more useful and concrete result would be that the objective function minimizes the JS divergence with respect to the original distributions. This paper uses the following theorem to reach this result:

Theorem 1 *Let $x \sim p(x), y \sim q(y \mid x)$ and $x_g \sim p_g(x), y_g \sim q(y_g \mid x_g)$, where $q(y \mid x)$ is the transition density. Given certain $q(y \mid x)$, if y could be reparameterized into $y = f(x) + h(\epsilon), \epsilon \sim p(\epsilon)$, where $p(\epsilon)$ is a known distribution, and both f and g are one-to-one mapping functions, then we could have $p(y) = p_g(y) \iff p(x) = p_g(x)$.*

Sketch of Proof. Recall that

$$p(y) = \int p(x)q(y \mid x)dx \qquad p_g(y) = \int p_g(x)q(y \mid x)dx$$

$\underline{\Longleftarrow}$: By the definitions above, if $p(x) = p_g(x)$, then $p(y) = p_g(y)$.

$\underline{\implies:}$ Let $y \sim p(y)$ and $y_g \sim p_g(y)$. Since we assume that $q(y \mid x)$ is the transition density, we get that

$$y = f(x) + g(\epsilon), \quad x \sim p(x), \quad \epsilon \sim p(\epsilon)$$
$$y_g = f(x_g) + g(\epsilon_g), \quad x_g \sim p_g(x), \quad \epsilon_g \sim p(\epsilon)$$

where f and g are one-to-one functions, and $f(x) \overset{D}{=} f(x_g) \implies x \overset{D}{=} x_g$. Then, using properties of moment-generating functions (MGF) and the moment-generating function uniqueness theorem, we can get that $M_{f(x)} = M_{f(x_g)}$. If then follows that

$$M_{f(x)} = M_{f(x_g)} \implies f(x) \overset{D}{=} f(x_g) \implies p(x) = p(x_g)$$

as desired. ■

To satisfy the conditions of this theorem, we need to find functions f and h that are one-to-one. Earlier in the paper, the authors showed that given the setup of the theorem (that the entire paper has been using) y could be reparameterized as

$$y = \sqrt{\bar{\alpha}_t}x + \sqrt{1 - \bar{\alpha}_t}\sigma\epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

Since $\sqrt{\bar{\alpha}_t}$ and $\sqrt{1 - \bar{\alpha}_t}\sigma$ are constants, we get that f and h are both linear and thus one-to-one. Since the conditions of the theorem are now satisfied, it can be used to show that minimizing divergence between the noisy samples also minimizes divergence of the original samples. As a result, adding the diffusion process to the training does not affect the quality of the generated samples.

The next theoretical guarantee that they show is that the f-divergence between the marginal distributions of the noisy real and fake samples is differentiable and thus can be optimized using methods such as gradient descent. Recall that the f-divergence between two probability distributions $p : \mathbb{R}^d \to \mathbb{R}$ and $q : \mathbb{R}^d \to \mathbb{R}$ is defined by

$$\mathcal{D}_f(p\|q) = \int_{\mathbb{R}^d} f\left(\frac{p(x)}{q(x)}\right) q(x)dx$$

for a convex function $f : [0, +\infty) \to (-\infty, +\infty]$ that is finite for all $x > 0$, $f(1) = 0$, and $f(0) = \lim_{t\to 0^+} f(t)$. They present the following theorem,

Theorem 2 *Let $p(x)$ be a fixed distribution over $\mathcal{X}$ and z be a random noise over another space $\mathcal{Z}$. Denote $G_\theta : \mathcal{Z} \to \mathcal{X}$ as a function with parameter θ and input z and $p_g(x)$ as the distribution of G_θ. Let $q(y \mid x, t) = \mathcal{N}(y; \sqrt{\bar{\alpha}_t}x, (1 - \bar{\alpha}_t)\sigma^2 I)$, where $\bar{\alpha}_t \in (0, 1)$ and $\sigma > 0$. Let $q(y \mid t) = \int p(x)q(y \mid x, t)dx$ and $q_g(y \mid t) = \int p_g(x)q(y \mid x, t)dx$. Then, $\forall t$m if function G_θ is continuous and differentiable the f-divergence $\mathcal{D}_f(q(y \mid t)\|q_g(y \mid t))$ is continuous and differentiable with respect to θ.*

Sketch of Proof. For each of notation, let $x \sim \mathbb{P}_r$, $x_g \sim \mathbb{P}_g$, $y \sim \mathbb{P}_{r',t}$, $y_g \sim \mathbb{P}_{g',t}$, $a_t = \sqrt{\bar{\alpha}_t}$, and $b_t = (1 - \bar{\alpha}_t)\sigma^2$. We then have the following set up:

$$p_{r',t}(y) = \int_{\mathcal{X}} p_r(x)\mathcal{N}(y; a_t x, b_t I)dx$$
$$p_{g',t}(y) = \int_{\mathcal{X}} p_g(x)\mathcal{N}(y; a_t x, b_t I)dx$$
$$z \sim p(z), x_g = g_\theta(z), y_g = a_t x_g + b_t\epsilon, \epsilon \sim p(\epsilon)$$

Then, using the definition of f-divergence and the equations above, we can write

$$\mathcal{D}_f(p_{r',t}(y)\|p_{g',t}(y)) = \mathbb{E}_{z\sim p(z),\epsilon\sim p(\epsilon)}\left[f\left(\frac{p_{r',t}(a_t g_\theta(z) + b_t\epsilon)}{p_{g',t}(a_t g_\theta(z) + b_t\epsilon)}\right)\right] \tag{1}$$

The proof then follows considering only the uni-variate case, but can be easily extended to the multivariate case. The next step is to show that $p_{r',t}(y)$ and $p_{g',t}(y)$ are continuous over y which can be easily verified by using the definition of the Gaussian distribution and showing that

$$\lim_{\Delta y\to 0} p_{r',t}(y - \Delta y) = p_{r',t}(y)$$

and similarly for $p_{g',t}(y)$. Then, since g_θ is also a continuous function, it follows from (1) that

$$D_f(p_{r',t}(y)\|p_{g',t}(y))$$

is continuous over θ. Finally, to show that the f-divergence is differentiable, by the chain rule, it suffices to show that $p_{r',t}(y)$, $p_{r',t}(y)$, and f are differentiable. Since f-divergence is usually defined with f differentiable, we can assume this. Then, It is simple to show that

$$\nabla_\theta p_{r',t}(a_t g_\theta(z) + b_t\epsilon) = \int_{\mathcal{X}} p_r(x)\frac{1}{C_1}\nabla_\theta \exp\left(\frac{1}{C_2}\|a_t g_\theta(z) + b_t\epsilon - a_t x\|_2^2\right) dx$$

$$\nabla_\theta p_{g',t}(a_t g_\theta(z) + b_t\epsilon) = \mathbb{E}_{z'\sim p(z')}\left[\frac{1}{C_1}\nabla_\theta \exp\left(\frac{1}{C_2}\|a_t g_\theta + b_t\epsilon - a_t g_\theta(z')\|_2^2\right)\right]$$

where C_1 and C_2 are constants and thus, $p_{r',t}(y)$ and $p_{r',t}(y)$ are differentiable. Thus, we have shown that $D_f(p_{r',t}(y)\|p_{g',t}(y))$ is continuous and differentiable with respect to θ. ■

The result of this theorem is that $\forall t$, y and y_g are defined on the same support space, and $\mathcal{D}_f(q(y \mid t)\|q_g(y \mid t))$ is continuous and differentiable everywhere. As a result, we can take the gradients of the f-divergence and optimize it using gradient descent. Since JS divergence is an f-divergence with $f(u) = -\log(2u) - \log(2-2u)$, we get that $\mathrm{JS}(q(y \mid t)\|q_g(y \mid t))$ is differentiable and can be optimized. The result of these two theorems show that injecting noise into the training process can facilitate learning.

Next, the authors of the paper conducted experiments to compare Diffusion-GAN to state-of-the-art GAN baselines. Specifically, they compared these models on 6 different datasets that varied in resolution and diversity. They measured FID (fidelity) and Recall (diversity) benchmarks, with lower FID indicating better fidelity and higher Recall indicating better diversity. First, they built Diffusion-GAN on top of StyleGAN2, which was the state-of-the-art GAN backbone at the time (2022). They also compare it to two other alterations, StyleGAN2 + DiffAug, and StyleGAN2 + ADA. The results of the experiment are summarized in Figure 4. As we can see, Diffusion StyleGAN2 outperformed

Methods	CIFAR-10 (32 × 32)		CelebA (64 × 64)		STL-10 (64 × 64)		LSUN-Bedroom (256 × 256)		LSUN-Church (256 × 256)		FFHQ (1024 × 1024)	
	FID	Recall	FID	Recall	FID	Recall	FID	Recall	FID	Recall	FID	Recall
StyleGAN2 (Karras et al., 2020a)	8.32*	0.41*	2.32	0.55	11.70	0.44	3.98	0.32	3.93	0.39	4.41	0.42
StyleGAN2 + DiffAug (Zhao et al., 2020)	5.79*	0.42*	2.75	0.52	12.97	0.39	4.25	0.19	4.66	0.33	4.46	0.41
StyleGAN2 + ADA (Karras et al., 2020a)	**2.92***	0.49*	2.49	0.53	13.72	0.36	7.89	0.05	4.12	0.18	4.47	0.41
Diffusion StyleGAN2	3.19	**0.58**	**1.69**	**0.67**	**11.43**	**0.45**	**3.65**	**0.32**	**3.17**	**0.42**	**2.83**	**0.49**

Figure 4: From Wang et al., results of experiments of image generation on 6 benchmark datasets. The best result for each column is highlighted using **bold** with the second best is underlined.

the other models in almost all benchmarks and measurements. Moreover, they also tested Diffusion-GAN when trained on very small amounts of data. For limited data, the best model at the time was InsGen, which outperfoemd both StyleGAN2 + ADA and StyleGAN2 + DiffAug for data-efficient GAN training. The results can be seen in Figure 5.

Models	FFHQ (200)	FFHQ (500)	FFHQ (1k)	FFHQ (2k)	FFHQ (5k)	Cat	Dog	Wild
InsGen (Yang et al., 2021)	102.58	54.762	34.90	18.21	9.89	2.60*	5.44*	1.77*
Diffusion InsGen	**63.34**	**50.39**	**30.91**	**16.43**	**8.48**	**2.40**	**4.83**	**1.51**

Figure 5: From Wang et al., comparison of Diffusion InsGen and baseline InsGen for data-efficient training. Training sample sizes range fro 200 to 5000. All values seen are FID measurements. The best result in each column is highlighted in **bold**.

Again, we can clearly see that Diffusion-GAN outperforms the state-of-the-art data-efficient GAN model. Finally, it is important to consider the memory and time costs of Diffusion-GAN, as it adds a possibly time and memory significant step in the training process. However, the authors of the paper found that the memory and time costs of Diffusion-GAN were comparable to that of the baseline GAN. For example, for one of the benchmark datasets, the training time with four NVIDIA V100 GPUs was around 8 seconds for the baseline StyleGAN2 and 9.5 seconds for Diffusion-StyleGAN2. For reference, see Figure 6 for examples of images generated by Diffusion-StyleGAN2.

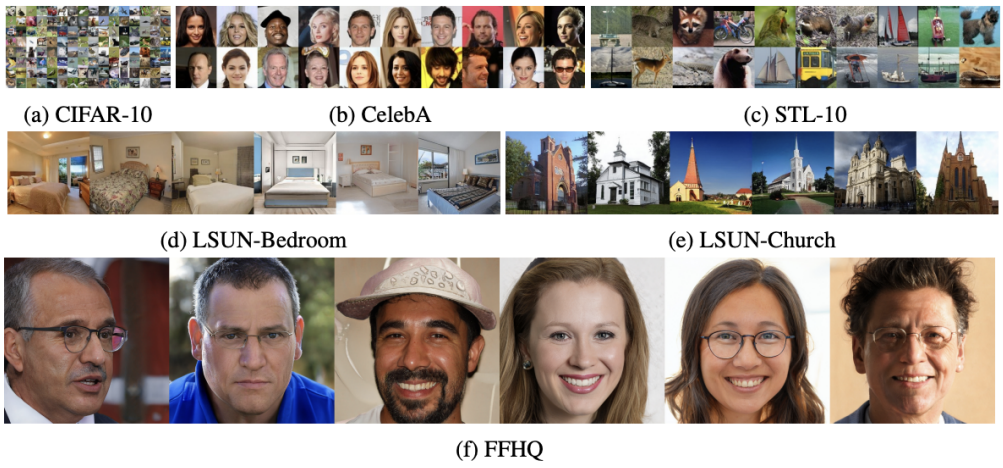

Figure 6: From Wang et al., examples of images generated by Diffusion-StyleGAN2 from the six benchmark datasets.

3 Method

3.1 Effects of Classifier Guidance and Scaling in the MNIST Dataset

We load the MNIST dataset (contains handwritten digits) and convert each image to a tensor.

We represent each diffusion step $t \in \{0, \dots, T-1\}$ by a 128-dimensional vector $\mathrm{emb}(t)$. For dimensions $0 \le i < 64$, we define

$$\mathrm{emb}_i(t) = \sin\Big(t \cdot \exp\big(-\ln(10000)\,\tfrac{i}{63}\big)\Big),$$

and for $64 \le i < 128$,

$$\mathrm{emb}_i(t) = \cos\Big(t \cdot \exp\big(-\ln(10000)\,\tfrac{i-64}{63}\big)\Big).$$

This sinusoidal embedding gives the network a continuous sense of position in the noise schedule.

Our denoiser uses a U-Net architecture with residual and attention blocks that is adapted from Wang et al. [3]. An initial 3×3 convolution lifts the single-channel input to 64 feature maps. We then apply two downsampling stages, each consisting of a residual block (two 3×3 convolutions with GroupNorm and SiLU activations) followed by a strided convolution that halves spatial resolution and doubles the number of channels. We inject the time embedding into each downsampling block via learned linear projections, and after the first downsampling we insert a multi-head self-attention layer to capture long-range dependencies. In the bottleneck we apply two residual blocks, then reverse the process with transpose convolutions to upsample. We add skip connections from corresponding downsampling layers and again inject the time embedding. A final 3×3 convolution maps back to one channel and yields the predicted noise.

We use a cosine noise schedule with $T = 1000$ steps and offset $s = 0.008$. We set

$$\bar{\alpha}_i = \cos^2\Big(\frac{i/T + s}{1+s}\,\frac{\pi}{2}\Big), \qquad \beta_i = 1 - \frac{\bar{\alpha}_i}{\bar{\alpha}_{i-1}},$$

with $\bar{\alpha}_{-1} = 1$. We clamp $\beta_i \le 0.999$, define $\alpha_i = 1 - \beta_i$, and compute the cumulative products $\bar{\alpha}_i = \prod_{j=0}^{i} \alpha_j$. This schedule controls the variance of added noise at each diffusion step.

We train the denoiser by sampling a batch of clean images x_0, drawing random times t, and forming noisy inputs

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1 - \bar{\alpha}_t}\, \epsilon, \quad \epsilon \sim \mathcal{N}(0, I).$$

The U-Net predicts $\hat{\epsilon}$, and we minimize the mean squared error $\|\epsilon - \hat{\epsilon}\|^2$ using the Adam optimizer with learning rate 10^{-4}. After each update, we update an exponential moving average of the network

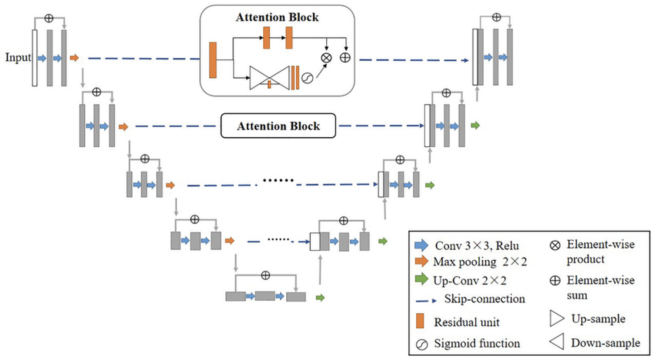

Figure 7: An overview of the residual UNET with attention we referenced in creating our UNET, from Wang et al. [3]

parameters with decay 0.9999 to stabilize training. In parallel, we train a noise-aware classifier on clean images ($t = 0$). We extract features via two convolutional layers and one residual block, apply adaptive average pooling, concatenate the 128-dimensional time embedding, and use a linear layer to predict class logits. Both networks train for 60 epochs.

We adapted the methodology from Dhairwal et al. [1] for image generation from the models. To generate images unconditionally, we sample $x_T \sim \mathcal{N}(0, I)$ and reverse through steps T to 1 by computing

$$\mu_{t-1} = \frac{1}{\sqrt{\alpha_t}}\Big(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\,\hat{\epsilon}\Big), \qquad x_{t-1} = \mu_{t-1} + \sqrt{\beta_t}\,z, \quad z \sim \mathcal{N}(0, I).$$

For classifier-guided sampling, at each step we also compute the gradient

$$g = \nabla_{x_t} \log p_\phi(y \mid x_t),$$

and adjust the posterior mean as

$$\mu'_{t-1} = \mu_{t-1} + \eta\,\beta_t\,g,$$

before adding noise, where η is the guidance scale.

We assess sample quality using the Frechet Inception Distance (FID). We compute metrics between real MNIST images and generated samples, and we save grids of generated images as PNG files for visual inspection.

4 Experiments

After training the model, we evaluated the efficacy of the technique by comparing the performance of the class-guided diffusion (at different scale) and standard diffusion. We report a similar trend to Dhariwal et al. [1] with respect the scale of the gradient in class-conditioned diffusion (Figure 8). There is any optimal scale (of around 1.5), after which the FID increases as excessive guidance drives the model far from the training distribution. Furthermore, we confirm that classifier-guidance can improve model performance by comparing the optimal-scaled class-guided diffusion standard diffusion in the FID score of the generated images. Note that the magnitude of the FID score is smaller when compared to the results in the paper. This is because the MNIST dataset often have less complex feature distributions when compared to datasets ImageNet. The black and white nature of the MNIST dataset along with the simplicity of most numbers allow for closer approximations from the diffusion models, leading to lower FID magnitudes.

Furthermore, in Figure 9 we see the examples of generated digits from both techniques. In our sample, the classifier guidance seems to generate sharper images where in most cases the intended number is clear. In contrast, the unconditional samples include more cases where the digits are ambiguous or break down into white blobs.

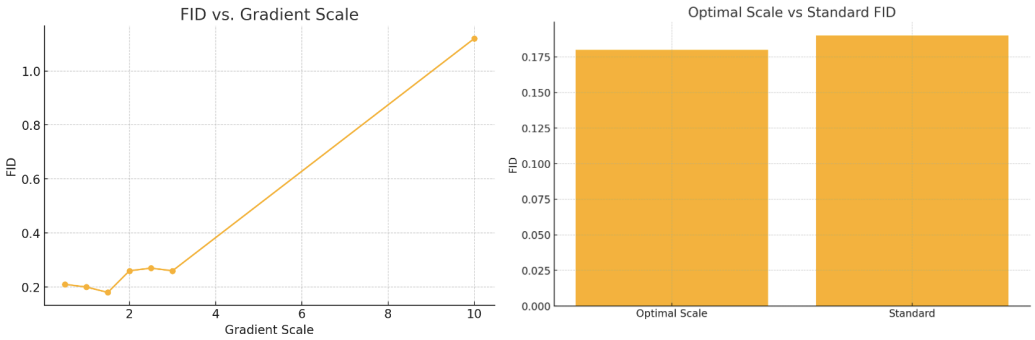

Figure 8: Left: FID vs Scale for classifier guidance, Right: Comparison of the optimal-scaled class-guided diffusion vs standard diffusion

Figure 9: Examples of numbers generated in the MNIST dataset for the optimal classifier guided diffusion vs standard image diffusion

5 Conclusion

In this paper, we summarized and compared diffusion models, GANs, and hybridized models for image generation. In the past decade, both diffusion models and GANs were at the forefront of image generation techniques. However, more recently, innovations have been able to combine various aspects of diffusion models and GANs into hybridized models. We first reviewed the work of Dhariwal et al., which showed that adding classifier guidance, a technique from GANs, to diffusion models, helped improved performance. Next, we reviewed the work of Wang et al., which introduced Diffusion-GAN, a novel framework that incorporates a forward diffusion process during training. For both of these models, we analyzed the mathematics and theoretical results that were proven. Then, we also confirmed some of the theoretical results by implementing our own diffusion model with classifier guidance and trained it on the MNIST dataset, which showed improvement over the standard diffusion process. Moving forward, future work could include verifying the theoretical results on a larger-scale, and applying these hybrid techniques to other domains of image generation.

References

[1] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis, 2021.

[2] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[3] Kan He, Xiaoming Liu, Rahil Shahzad, Robert Reimer, Frank Thiele, Julius Niehoff, Christian Wybranski, Alexander Bunck, Huimao Zhang, and Michael Perkuhn. Advanced deep learning approach to automatically segment malignant tumors and ablation zone in the liver with contrast-enhanced ct. *Frontiers in Oncology*, 11:669437, 07 2021.

[4] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020.

[5] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2022.

[6] Zhendong Wang, Huangjie Zheng, Pengcheng He, Weizhu Chen, and Mingyuan Zhou. Diffusion-gan: Training gans with diffusion, 2023.