

High Quality Image Generation using Improved Generative Adversarial Networks and Optimized Stable Diffusion Models

Satishkumar Varma

vsat2k@gmail.com

Dwarkadas J. Sanghvi College of Engineering

Kunal Wagh

Pillai College of Engineering

Omkar Raul

Pillai College of Engineering

Sejal Chandgadkar

Pillai College of Engineering

Shreya Belanekar

Pillai College of Engineering

Kanchan Dabre

Dwarkadas J. Sanghvi College of Engineering

Research Article

Keywords: Generative Adversarial Networks, Stable Diffusion, Image Generation, Data Augmentation, Text-to-Image Translation, Super Resolution

Posted Date: September 21st, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-10786845/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

High Quality Image Generation using Improved Generative Adversarial Networks and Optimized Stable Diffusion Models

Satishkumar Varma^{1*}, Kunal Wagh^{2†}, Omkar Raul^{2†},
Sejal Chandgadker^{2†}, Shreya Belanekar^{2†}, Kanchan Dabre^{1†}

¹Department of Computer Science and Engineering (Data Science),
Dwarkanadas J. Sanghvi College of Engineering, Vile Parle, Mumbai,
400056, Maharashtra, India.

²Department of Information Technology, Pillai College of Engineering,
New Panvel, Navi Mumbai, 410206, Maharashtra, India.

*Corresponding author(s). E-mail(s): vsat2k@gmail.com;

Contributing authors: kkwagh.999@gmail.com;

omkarraul264@gmail.com; sejalchandgadkar@gmail.com;

shreyabelanekar24@gmail.com; kanchandabre@gmail.com;

[†]These authors contributed equally to this work.

Abstract

Generative Adversarial Networks (GANs) demonstrated significant promise in generating synthetic images. GANs involve dual deep neural networks: the generator for generating realistic and high-quality new images, whereas the discriminator is for classifying these images. The GAN generator sometimes produces limited image variations and requires hyperparameter tuning. Diffusion models also involve dual processes: the forward process for adding pure noise and the reverse process for reconstructing the original images. Individually, the diffusion models are stable during training as compared to GANs, with less risk of mode collapse. However, training GANs with Stable Diffusion leads to two issues, namely model collapse and non-convergence. In this paper, these two issues are addressed by combining GANs and Stable Diffusion to rework the network architecture to obtain a more powerful model. A GAN and Stable Diffusion-based model architecture is designed and implemented for high-quality image generation and analysis. The GANs and Stable Diffusion frameworks are developed for the interpretation of the images using architectures

such as `stable-diffusion-2-1`, `stable-diffusion-2-base`, GAN, and unsupervised text-to-image translation models. Along with the existing architecture, performance analyses are also carried out by incorporating an additional hybrid architecture. The CLIP scores and Real-or-Fake scores metrics are used to evaluate Stable Diffusion and GAN performance, respectively. The proposed GAN model achieved an accuracy of 81% on the FFHQ dataset and 91% on the Flickr dataset. The proposed Stable Diffusion model achieved a 31.98 CLIP score on the Flickr dataset. Moreover, the Stable Diffusion models generate more realistic images as compared to GANs. These approaches are useful for criminal image generation on the basis of fewer available images of the criminal. Also, this architecture is useful for generating high-quality images in various fields such as data augmentation, medical image processing, agriculture, arts generation, forensics, and text-to-image translation.

Keywords: Generative Adversarial Networks, Stable Diffusion, Image Generation, Data Augmentation, Text-to-Image Translation, Super Resolution

1 Introduction

The ability of Generative Adversarial Networks (GANs) to learn a generative model directly from the data distribution has made them a promising method for image generation tasks. GANs comprise two neural networks, a generator, and a discriminator, which are concurrently trained within a minimax game framework. The generator generates images that are indistinguishable from actual ones, whereas the discriminator differentiates between real and generated images. GANs can learn to create extremely realistic images with a variety of styles, resolutions, and structures through this adversarial process. In recent years, GANs have found widespread application in various image generation tasks, which includes image synthesis [1], style transfer, and image-to-image translation. Image synthesis involves generating new images from scratch, while style transfer transfers one image’s style to another. Image-to-image translation tasks entail transforming an input image into an output image belonging to a different domain, such as converting a black and white image to color or a sketch to a realistic image. Despite the success of GANs in image generation, there are still many challenges and limitations in this field. One of the main challenges is the instability and mode collapse issue, which can lead to the generation of low-quality and repetitive images. To address this issue, various GAN architectures and regularization techniques have been proposed, such as Wasserstein GANs, spectral normalization, and adversarial training with gradient penalty. Moreover, assessing the diversity and quality of images generated by GANs is still challenging, and various metrics and evaluation methods have been proposed to measure the fidelity and diversity of generated images.

In this work, an overview of recent advances in GAN-based image generation techniques are presented. We first introduce the basic architecture and training process of GANs. Following that, we address the various applications of GANs in image generation and present an in-depth analysis of the recent research in GANs, including

the latest developments in GAN architecture, regularization techniques, and evaluation methods. Furthermore, we highlight the challenges and limitations of GAN-based image generation and discuss the future directions and opportunities in this field. Since, there are various different models for image generation, one such is Stable diffusion. Diffusion models are a class of generative models in machine learning that mimic the patterns found in the training data to generate new data, like sounds or images. They get their name by imitating the diffusion process, which is part of their methodology. The process involves repeatedly adding noise to the original data and learning how to undo it, resulting in highly similar new data. The overall study hence proves that stable diffusion holds the ability to generate more realistic and accurate images in comparison to GANs. Comparison of CLIP scores across different versions of the Stable Diffusion model reveals distinct performance patterns, with specific model versions exhibiting superiority for particular datasets. This underscores the importance of selecting appropriate model architectures tailored to dataset characteristics for optimal performance in image generation tasks.

1.1 Contribution of the research

This section highlights the study’s key contributions:

1. The project focuses on applying GANs and Stable Diffusion to obtain realistic images.
2. The comparison between these demonstrates how stable diffusion techniques can be used to overcome the drawbacks of GANs and generate high-quality images.
3. The study highlights how crucial it is to investigate and refine stable diffusion models in order to achieve better picture generating challenges. Moreover, it highlights the potential for further research in this area and offers intriguing applications that might have a big influence across a variety of fields, like style transfer and artistic picture generation.

1.2 Organization of the Paper

This paper outlines the topic of GANs and Stable Diffusion, focusing on their capacity to produce lifelike images through text-based exploration. Section 2 outlines the goals and objectives of the study, while Section 3 delves into a comprehensive literature review and the conducted research. Methodological details are elaborated in Section 4, while Section 5 outlines the findings, and Section 6 provides the paper’s conclusion.

2 Motivation and Objectives

Image generation has significance in many industries due to the fact that it can create images autonomously without human intervention, saving time and resources. This time and cost savings help to streamline processes across industries, encouraging new creativity and innovation by providing a platform for experimentation and exploration. Furthermore, the creation of realistic images benefits a variety of fields, increasing personalization and engagement in the creative process while also enabling advancements in medical imaging, design, and the arts.

The motivation for writing a research paper on Image Generation using GANs and Stable Diffusion using Deep Learning techniques comes from several important factors, including:

1. Medical Imaging: Image generation in medical imaging helps to create detailed representations of anatomical structures, which aids in diagnosis, treatment planning, and medical research.
2. Art Generation: Image generation allows artists and designers to experiment with new creative directions, pushing the boundaries of traditional art forms and encouraging artistic innovation.
3. Data Augmentation: Image generation improves the robustness and diversity of datasets used to train machine learning models, resulting in improved performance and generalization capabilities across a wide range of tasks.
4. Forensics and Simulation: In forensics, image generation helps to reconstruct crime scenes and create facial composites of suspects. Simulation allows for the creation of realistic scenarios for training and analysis in fields such as aviation, defence, and disaster management.

It is also important that we clearly define our study’s objectives. Here are some specific objectives we’d like to highlight:

1. To produce high-quality images, develop a sophisticated image generation model that employs deep learning techniques.
2. Fine-tune various image generation models using domain-specific datasets and compare their accuracy.
3. To optimise the model for fast image generation, reduce the time between user input and image output.
4. To implement image generation by taking text as an input (text to image) or voice as an input (voice to image).
5. To create multiple images of a person from a small amount of data, which can be used to identify thieves in various outfits.
6. To create images that replicate the drawings.

3 Literature Survey

3.1 Conditionally-Independent Pixel Synthesis

This is a new approach to image generation [2–4] using deep neural networks. There are various existing methods for image generation, such as GANs, which suffer from several limitations, including instability during training and the inability to generate high-resolution images with fine details. A new architecture for image generation called “Conditionally-Independent Pixel Synthesis” (CIPS) [5, 6] can be used to address these limitations.

The primary idea behind this approach is to use a cascade of non-linear transformations to generate each pixel in the image, conditioned on the previous pixels. This approach allows for the generation of high-resolution images with fine details, without the need for convolutions or pooling operations. This approach introduces fresh

opportunities for producing high-fidelity images across numerous domains, including computer vision, graphic design, and artistic expression. The recent research on image generation art creation assisted by deep learning and embedded systems is also carried out in [7]. Also, a diffusion-based multi-style image generation framework with adaptive weighted style fusion is developed by [8].

3.2 Infinite Generative Adversarial Networks

The concept of Infinite Generative Adversarial Networks [9, 10] introduces a fresh methodology for unsupervised image generation [2, 3] through the utilization of GANs. Existing GANs-based methods [10] for image generation are plagued by a phenomenon known as mode collapse, where the generator learns to produce a limited set of samples that do not fully capture the underlying distribution of the data.

To address this limitation, this new type of GAN called the Infinite Generative Adversarial Network (iGAN) [11] is proposed, which is able to generate an infinite number of samples from the underlying distribution. The primary idea behind iGAN is to utilize an infinite mixture of GANs, each of which is trained on a random subset of the data. During generation, iGAN samples from the mixture of GANs, resulting in a diverse set of generated samples that fully capture the underlying distribution of the data.

There has been a comparison of this method to several modern GAN-based approaches to image generation [2, 3, 12], including DCGAN, WGAN, and SNGAN, and demonstrate that iGAN is able to generate high-quality images with greater diversity and coverage of the underlying distribution. They also show that iGAN is able to learn a meaningful latent space that can be used for various image manipulation tasks, such as style transfer and interpolation.

3.3 Image Quality of StyleGAN

StyleGAN [13] architecture proposes several modifications to improve the quality of generated images and is a promising approach because it is able to generate highly realistic images with fine details. When the analysis of StyleGAN architecture is done while focusing on its synthesis network and mapping network, several limitations can be identified in the architecture, including the tendency to produce overly smooth images and the lack of diversity in generated images.

To address these limitations, the authors propose several modifications to the StyleGAN architecture. These modifications include the use of a new normalization technique, the introduction of skip connections, and the incorporation of a noise channel. These modifications lead to improvements in image quality and diversity, as well as faster convergence during training.

3.4 StarGAN v2

StarGAN v2 is a multi-domain image synthesis [1, 14] approach which has the ability to produce diverse images with a single model among various domains. The existing methods for multi-domain image synthesis are limited in their ability to generate

diverse images, which can be a significant bottleneck for applications such as image-to-image translation and virtual try-on. Many of these methods rely on domain-specific models, which can be inefficient and difficult to train. Also, they are often limited in their ability to generate diverse images, since they rely on a fixed mapping between input and output domains.

To address these limitations, “StarGAN v2” [15] can be used, which possesses the capacity to produce varied images across numerous domains utilizing a unified architecture. The key idea behind this approach is to use a style-based generator, in which the model is trained to generate images based on a style vector that is shared across all domains. During training, the model is provided with images from multiple domains, and is trained to generate corresponding images in each domain while preserving the shared style vector.

3.5 High-resolution Image Synthesis

The latest methods for high-resolution image synthesis [1, 14] are limited in their ability to generate images of arbitrary resolutions, which can be a significant bottleneck for applications such as video generation and interactive image editing. Many of these methods rely on iterative refinement, in which low-resolution images are first generated and then upsampled to higher resolutions.

To address this limitation, “Any-resolution Training” [14] method is proposed, which has the ability to generate images of arbitrary resolutions with a single model. The fundamental concept driving this approach involves employing a multi-scale architecture, wherein the model undergoes training to generate images across various resolutions concurrently. This method can help us getting videos of superior quality with smooth transitions between resolutions.

Table 1: Techniques used for Image Enhancement including StarGAN, Simple GAN, Style GAN, PixelCNN, and Stable Diffusion

Literature	Techniques used for Image Enhancement
Choi, Yunjey, et al. 2020 [15]	StarGAN
Karras, Tero, et al. 2020 [13]	Simple GAN, Style GAN, PixelCNN
Anokhin, Ivan, et al. 2021 [5]	StarGAN, PixelCNN, Stable Diffusion
Liu, Ming-Yu, et al. 2021 [16]	Simple GAN
Chai, Lucy, et al. 2022 [14]	Simple GAN
Chenshuang Zhang, et al. 2023 [17]	Stable Diffusion
Divyanshu Mataghare, et al. 2023 [12]	Stable Diffusion
Lvmin Zhang, et al. 2023 [18]	ControlNet
Yang Zhao, et al. 2023 [19]	MobileDiffusion
Ping Chai, et al. 2024 [20]	pix2pix, CycleGAN, StyleGAN, etc.

3.6 Text to Image Generation

Diffusion models [21, 22] generate high-quality images that take into account nearly all relevant details. The Diffusion Model has the advantage of not requiring training from the start. However, because they work directly on individual pixels, Diffusion Model is frequently power and time consuming. Also hierarchical text-conditional image generation with CLIP latents are presented in [23].

To overcome this limitation, the method “Text to Image Generation Using Stable Diffusion” [2, 12] is proposed, with the process divided into compressive and generative learning. Recently, the triplex attention enabled diffusion generative adversarial network for photorealistic image generation from text and images is developed in [24].

4 Methodology

Generative Adversarial Networks (GANs) and Stable Diffusion models are widely used for image generation tasks, with many different goals that can be pursued depending on the specific application. Here are some common goals for image generation using GANs and Stable Diffusion:

1. To generate new images that are similar to the existing training data.
2. To synthesize new images that do not exist in the original dataset.
3. To apply the style of one image to another.
4. To upscale low-resolution images into high-resolution counterparts.
5. To fill in missing or damaged parts of an image.
6. To generate images of a certain object category or images with specific attributes.

4.1 System Architecture

Figure 1 demonstrates the proposed approach, comprising the dataset, data preprocessing, the generator, the discriminator, the adversarial training, and information feedback for optimization.

GANs represent a powerful paradigm in the field of deep learning, enabling the generation of high-quality synthetic data samples across diverse domains. At the core of GANs lies a competitive training framework involving two neural networks: a generator and a discriminator. Through an iterative process, the generator learns to produce data samples that closely resemble those from a target distribution, while the discriminator learns to distinguish between real and fake samples.

This adversarial relationship drives the continual improvement of both networks, leading to the generation of increasingly realistic data. GANs have demonstrated remarkable success in various applications, including image generation, text-to-image synthesis, and style transfer. However, training GANs can be challenging due to issues such as mode collapse and instability. Nonetheless, ongoing research efforts continue to advance GAN architectures and training techniques, paving the way for their widespread deployment in real-world scenarios. This respective GAN training method is presented as Algorithm 1.

The Algorithm 2 outlines the training function for a Stable Diffusion Model, incorporating components such as a Text Encoder, Variational Autoencoder (VAE), and

Table 2: Summary of Literature Survey covering Research Papers, Models used, Accuracy and Datasets used for implementation of models

Literature	Models Used	Accuracy/Score	Dataset
Anokhin, Ivan, et al. 2021 [5]	PixelCNN	2.92 bits/dim 3.51 bits/dim	CIFAR-10 ImageNet
Liu, Ming-Yu, et al. 2021 [16]	DCGAN, WGAN, Progressive GAN, StyleGAN, BigGAN, ProGAN	Highest on CelebA-HQ	CelebA-HQ
Karras, Tero, et al. 2020 [13]	StyleGAN, StyleGAN2	FID 6.80 FID 4.30	FFHQ LSUN
Choi, Yunjey, et al. 2020 [15]	StarGAN v2	FID 2.42 FID 6.57	CelebA-HQ AFHQ
Chai, Lucy, et al. 2022 [14]	StyleGAN3, Anyres-GAN	FID 4.47, pFID 4.28	–
Chenshuang Zhang, et al. 2023 [17]	Denoising diffusion probabilistic models (DDPMs)	FID 6.75 FID 6.75	ERNIE-ViLG 2.0 eDiff-I
Divyanshu Mataghare, et al. 2023 [12]	Perceptual compression model, Dissemination Models, LDM using stable diffusion	IS 5.2 ± 0.05	–

Algorithm 1 Training Generative Adversarial Networks (GANs)

```
1: Input: Real data samples  $x$ , Random noise vectors  $z$  from a prior distribution
2: Output: Trained generator  $G$  and discriminator  $D$ 
3: procedure TRAINING GANS
4:   Initialize:
5:     Initialize the weights of  $G$  and  $D$  with random values.
6:     Define loss functions for  $G$  and  $D$ .
7:   Training Loop:
8:   while not converged do
9:     Batch sampling of real data samples  $x$ .
10:    Generate a batch of fake samples  $\hat{x} = G(z)$ , where the value for  $z$  is sampled
    from the prior distribution.
11:    Update the discriminator:
12:      Compute the discriminator loss using both real and fake samples.
13:      Update the discriminator weights to minimize the loss.
14:    Update the generator:
15:      Generate a batch of fake samples  $\hat{x} = G(z)$ .
16:      Compute the generator loss using the discriminator's output.
17:      Update weights of the generator to maximize the loss.
18:  end while
19:  Loss Functions:
20:    Discriminator Loss (Binary Cross-Entropy):
21:     $L_D = -\mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x)] - \mathbb{E}_{z \sim p(z)}[\log(1 - D(G(z)))]$ 
22:    Generator Loss (Binary Cross-Entropy):
23:     $L_G = -\mathbb{E}_{z \sim p(z)}[\log D(G(z))]$ 
24:  Optimization:
25:    Use stochastic gradient descent (SGD) or its variants to optimize the weights
    of  $G$  and  $D$ .
26:  Evaluation:
27:    Evaluate the trained GAN on a separate validation or test dataset.
28:    Use evaluation metrics like Inception Score, Fréchet Inception Distance, or
    human evaluation.
29:  Convergence Check:
30:    Monitor the convergence of the GAN by observing the discriminator and
    generator losses over epochs.
31:    Stop training when the losses stabilize or reach a satisfactory level.
32:  Post-Processing:
33:    Optionally, perform post-processing techniques like smoothing, denoising, or
    style transfer on the generated samples.
34:  Deployment:
35:    Deploy the trained GAN model for generating new samples in real-world
    applications.
36: end procedure
```

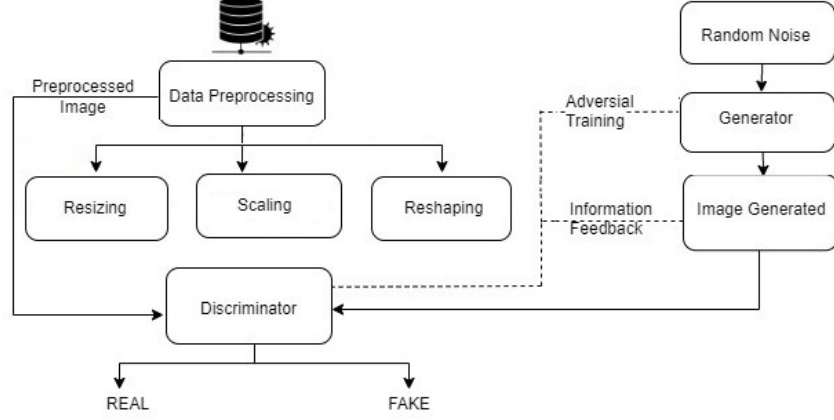


Fig. 1: Proposed system architecture diagram for GAN and Stable Diffusion image generation.

UNet architecture. The goal is to train a model capable of generating stable and high-quality samples. Key steps in the training function include setting hyperparameters such as training batch size, gradient steps accumulation, learning rate (lr), maximum training steps, output directory, and gradient checkpointing. If gradient checkpointing is enabled, specific components like the text encoder and UNet are configured accordingly. A dataloader is created for the training dataset, and if scale learning rate is enabled, the learning rate is adjusted accordingly. The optimizer, typically AdamW, is initialized, and the model components are prepared for training using the accelerator. Training progresses through epochs and batches, with the model updated based on accumulated gradients. Input data undergoes encoding through the VAE, with noise injected using a noise scheduler. The UNet processes the latent representations along with additional information from the text encoder to predict noise for diffusion. Overall, this procedure lays the groundwork for training a Stable Diffusion Model using the provided components and hyperparameters, aimed at achieving stable and effective sample generation. The proposed system architecture for Stable Diffusion Model is presented in Figure 2

This Algorithm 3 as Stable Diffusion Model 2 pertains to the calculation of loss and gradient updates based on the prediction type specified in the noise scheduler configuration. If the prediction type is “epsilon,” the target value for the loss calculation is set to the noise itself. If the prediction type is “v_prediction,” the target value is obtained from the noise scheduler’s velocity prediction function. If neither condition is met, a **ValueError** is raised indicating an unknown prediction type.

The loss is calculated using mean squared error (MSE) loss between the predicted noise and the target value. The resulting loss is then propagated back through the model using the accelerator’s backward function. If multiple processes are utilized, gradients are obtained from the module or model’s parameters accordingly. Additionally, gradient values corresponding to the placeholder token ID are zeroed out to prevent updates to those tokens.

Algorithm 2 Training Stable Diffusion Model 1

```
1: Input: Text Encoder, VAE, UNet
2: Output: Trained Stable Diffusion Model
3: procedure TRAINING FUNCTION(text encoder, vae, unet)
4:   training batch size  $\leftarrow$  hyperparameters[“training batch size”]
5:   gradient steps accumulation  $\leftarrow$  hyperparameters[“gradient steps accumulation”]
6:   lr  $\leftarrow$  hyperparameters[“lr”]
7:   max training steps  $\leftarrow$  hyperparameters[“max training steps”]
8:   output direc  $\leftarrow$  hyperparameters[“output direc”]
9:   grad checkpointing  $\leftarrow$  hyperparameters[“grad checkpointing”]

10:  if grad checkpointing then
11:    text encoder.grad checkpointing enable()
12:    unet.enable grad checkpointing()
13:  end if

14:  training dataloader  $\leftarrow$  create dataloader(training batch size)

15:  if hyperparameters[“scale learning rate”] then
16:    lr  $\leftarrow$  (lr  $\times$  steps gradient accumulation  $\times$  batch size train  $\times$ 
    accelerator.num processes)
17:  end if

18:  optimizer  $\leftarrow$  AdamW(text encoder.get input embeddings().parameters(), lr =
    lr)

19:  (text encoder, optimizer, training dataloader)  $\leftarrow$ 
    accelerator.prepare(text encoder, optimizer, training dataloader)

20:  updated steps per epoch  $\leftarrow$  math.ceil(len(training dataloader)/gradient accumulation steps)
21:  training epochs  $\leftarrow$  math.ceil(training steps max/updated steps for each epoch)

22:  batch size total  $\leftarrow$  training batch size  $\times$  accelerator.num processes  $\times$ 
    gradient accumulation steps

23:  progress bar  $\leftarrow$  tqdm(range(training steps max), disable =
     $\neg$ accelerator.is this local main process)
24:  progress bar.set the description(“Steps”)
25:  global steps  $\leftarrow$  0

26:  for epoch in range(training epochs) do
27:    text encoder.train()
28:    for steps, batch in enumerate(training dataloader) do
29:      with accelerator.accumulate(text encoder) do
30:        latents  $\leftarrow$  vae.encode(batch[“pixel values”].to(dtype =
    weight dtype)).latent dist.sample().detach()
31:        latents  $\leftarrow$  latents  $\times$  0.18215
32:        noise  $\leftarrow$  torch.randn_like(latents)
33:        bsz  $\leftarrow$  latents.shape[0]
34:        timesteps  $\leftarrow$  torch.randint(0, noise scheduler.num training timesteps, (bsz, ),
    device = latents.device).long()
35:        noisy latents  $\leftarrow$  noise scheduler.add noise(latents, noise, timesteps)
36:        hidden encoder states  $\leftarrow$  text encoder(batch[“input ids”])[0]
37:        noise prediction  $\leftarrow$  unet(noisy latents, timesteps,
    hidden encoder states.to(weight dtype)).sample
38:        hidden encoder states.to(weight dtype).sample
39:      end for
40:    end for
41:  end for
42: end procedure
```

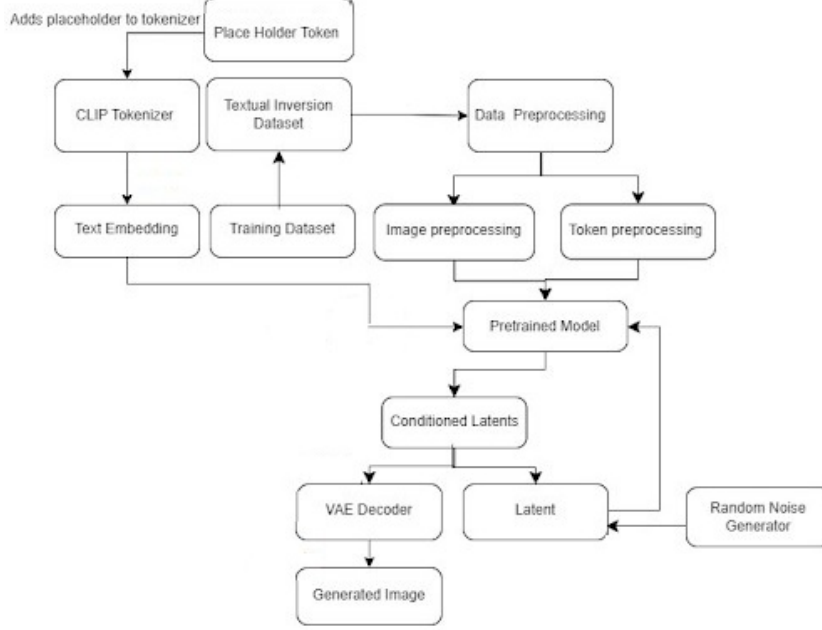


Fig. 2: Proposed System architecture for Stable Diffusion Model consisting the data preprocessing step, placeholder token creation, adding the token to the vocabulary, embedding the token, stable diffusion architecture consisting VAE, Decoders, Latent Conditioning Step and the final image generation.

After updating the optimizer’s parameters, gradients are reset, and if gradient synchronization is enabled, progress is updated accordingly. The global steps counter is incremented, and if the specified number of steps for saving progress is reached, the model’s current state is saved. The procedure continues until the maximum training steps are reached, at which point the training loop terminates, and model training is completed. Finally, the trained model is saved to the specified output directory.

4.2 Experimental Setup

This section provides detailed information about the hardware and software prerequisites, the datasets utilized, and the evaluation metrics employed.

4.2.1 Hardware and Software

Table 3 summarizes the essential requirements for implementing image generation using Generative Adversarial Networks (GANs) and the Stable Diffusion model. It specifies an Intel i5 11th generation processor for computational power, a 1TB hard disk drive (HDD) for storage, and 8GB of RAM for efficient data processing. The operating system needed is Windows 11, with Python chosen as the programming language

Algorithm 3 Training Stable Diffusion Model 2 (Loss, Gradients, and Saving)

```
1: if noise_scheduler.config.prediction_type == "epsilon" then
2:   target_value  $\leftarrow$  noise
3: else if noise_scheduler.config.prediction_type == "v_prediction" then
4:   target  $\leftarrow$  noise_scheduler.get_velocity(latents, noise, timesteps)
5: else
6:   raise ValueError("Unknown prediction type ") +
   noise_scheduler.config.prediction_type
7: end if

8: loss  $\leftarrow$  F.mse_loss(noise_pred, target, reduction = "none").mean([1, 2, 3]).mean()
9: accelerator.backward(loss)

10: if accelerator.number_processes > 1 then
11:   gradients  $\leftarrow$  text_encoder.module.get_the_input_embeddings().weight.gradient
12: else
13:   grads  $\leftarrow$  text_encoder.get_the_input_embeddings().weight.gradient
14: end if

15: index_gradient_to_zero  $\leftarrow$  torch.arange(len(tokenizer))  $\neq$  placeholder_token_id
16: gradients.data[index_gradients_to_zero, :]  $\leftarrow$  gradients.data[index_gradients_to_zero, :].fill(0)

17: optimizer.step()
18: optimizer.zero_grad()

19: if accelerator.sync_gradients then
20:   progress_bar.update(1)
21:   global_steps  $\leftarrow$  global_steps + 1
22:   if global_step (mod hyperparameters["save_the_steps"]) == 0 then
23:     saving_path  $\leftarrow$  os.path.join(output_dir, "learned_embeddings_steps_" +
   global_steps + ".bin")
24:     save_progress(text_encoder, placeholder_token_id, accelerator, save_path)
25:   end if
26: end if

27: logs  $\leftarrow$  {"loss" : loss.detach().item()}
28: progress_bar.set_postfix(**logs)

29: if global_steps  $\geq$  max_training_steps then
30:   break
31: end if

32: accelerator.wait_for_everyone()

33: if accelerator.is_main_process then
34:   model  $\leftarrow$  StableDiffusionModel.from_pretrained(pretrained_model_name_or_path, ...)
35:   text_encoder = accelerator.load_model(text_encoder), tokenizer =
   tokenizer, vae = vae, unet = unet
36:   model.save_pretrained(output_dir_path)
37:   save_path  $\leftarrow$  os.path.join(output_dir, "learned_embeddings.bin")
38:   save_progress(text_encoder, placeholder_token_id, accelerator, save_path)
39: end if
```

for its versatility and extensive machine learning libraries. These specifications ensure a capable setup for successful image generation tasks with GANs and Stable Diffusion.

Table 3: Hardware and Software Specifications for the implementation of image generation using GANs and Stable Diffusion

Requirement Type	Specification
Processor	Intel i5 11th generation
HDD	1TB
RAM	8GB
Operating System	Windows 11
Programming Language	Python

4.2.2 Datasets

Table 4 summarizes the datasets employed in the implementation of image generation using GANs and Stable Diffusion. It includes FFHQ with 3,000 images, Flickr with 6,000 images, and Celeb with 100 images. These diverse datasets offer a rich variety of images crucial for training and evaluating the effectiveness of the GANs and Stable Diffusion model.

Table 4: Datasets Used for the implementation of Image Generation using GANs and Stable Diffusion

Dataset	Count	Type
FFHQ	3000	Images
Flickr	6000	Images
Celeb	100	Images

4.2.3 Evaluation Metrics

The Inception Score (IS) and Fréchet Inception Distance (FID) scores are the appropriate evaluation metrics for GANs according to [16].

Inception Score (IS):

This metric assesses the quality and diversity of generated images by comparing their class distribution with that of a pre-trained classifier network. The IS is defined as:

$$\text{IS} = \exp(\mathbb{E}_x [D_{\text{KL}}(p(y|x) \parallel p(y))]) \quad (1)$$

In this equation, $p(y|x)$ denotes the class distribution of the generated images predicted by the classifier network, $p(y)$ is the dataset’s marginal class distribution, and D_{KL} is the Kullback-Leibler divergence.

Fréchet Inception Distance (FID):

This metric evaluates the disparity between the feature representations of the generated images and those of the real images, calculated by a classifier network that has been pre-trained. The FID is defined as:

$$\text{FID} = \|\mu_{\text{real}} - \mu_{\text{fake}}\|^2 + \text{tr}(\Sigma_{\text{real}} + \Sigma_{\text{fake}} - 2(\Sigma_{\text{real}}\Sigma_{\text{fake}})^{0.5}) \quad (2)$$

In this equation, μ_{real} and μ_{fake} denote the mean feature representations of the real and fake images, while Σ_{real} and Σ_{fake} denote the covariance matrices of the real and fake feature representations, respectively.

5 Result and Analysis

The project aims to generate high resolution images using GAN and Stable Diffusion and below are the results obtained from each technique.

5.1 Result Analysis of GAN

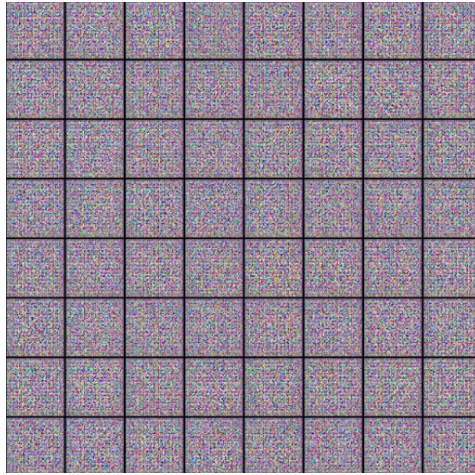
Before training, an image generated by a simple GAN produces random noise as output. As a result, images generated before training are typically noisy, blurry, and do not closely resemble the target data distribution. These generated images contain no meaningful patterns or structures.

After training, the generator network has learned to generate data that closely matches the distribution of the real data it was trained on. The images generated by the GAN become increasingly realistic, with improved clarity, detail, and resemblance to the actual data. The discriminator’s feedback during training helps the generator refine its output, making it more coherent and closer to the desired output distribution.

In Figure 3, the image generated by our GAN model, fine-tuned on the Flickr dataset, exhibits a remarkable level of realism and visual coherence. This synthetic image is a product of the GAN’s ability to transform a random noise vector into a compelling and photorealistic composition. The generated image captures the essence of the Flickr dataset, seamlessly blending various elements such as people and vehicles, and successfully mimicking the diverse scenes and objects found in the training data.

Figure 4 shows the image generated by our GAN model, fine-tuned on the Celebs-HQ dataset featuring the renowned actor Brad Pitt, is a striking testament to the GAN’s capability in generating Brad Pitt-like images. This synthetic portrayal exudes an exceptional likeness to the actor, effectively capturing his unique facial features, expressions, and visual identity.

Figure 5 shows an image generated by our GAN model fine-tuned on the expansive and diverse FFHQ dataset, is a testament to the GAN’s remarkable capacity to produce novel faces. This synthetic image is the product of the GAN’s ability to transform a random noise vector into a portrait that appears strikingly genuine. The generated

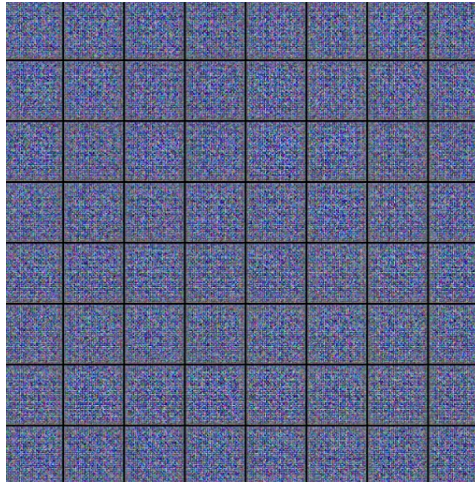


(a) Before Training.

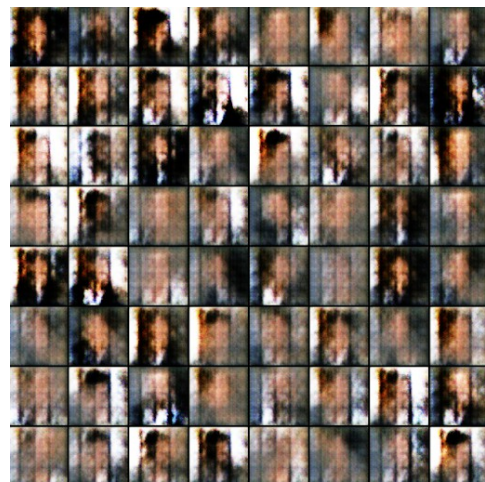


(b) After training on 50 epochs.

Fig. 3: Synthetic image composition generated by the fine-tuned GAN model on the Flickr dataset: (a) Before Training generating the noisy images, (b) After training on 50 epochs generating images similar to Flickr Dataset images.



(a) Before training.



(b) After training on 50 epochs.

Fig. 4: Image generated by proposed fine-tuned GAN using Celebs-HQ Dataset: (a) before training generating noisy images, (b) after training on 50 epochs generating images similar to Celebs-HQ Dataset images.

image showcases the model’s proficiency in generating an array of random faces, each imbued with a unique combination of facial features, expressions, and identities.

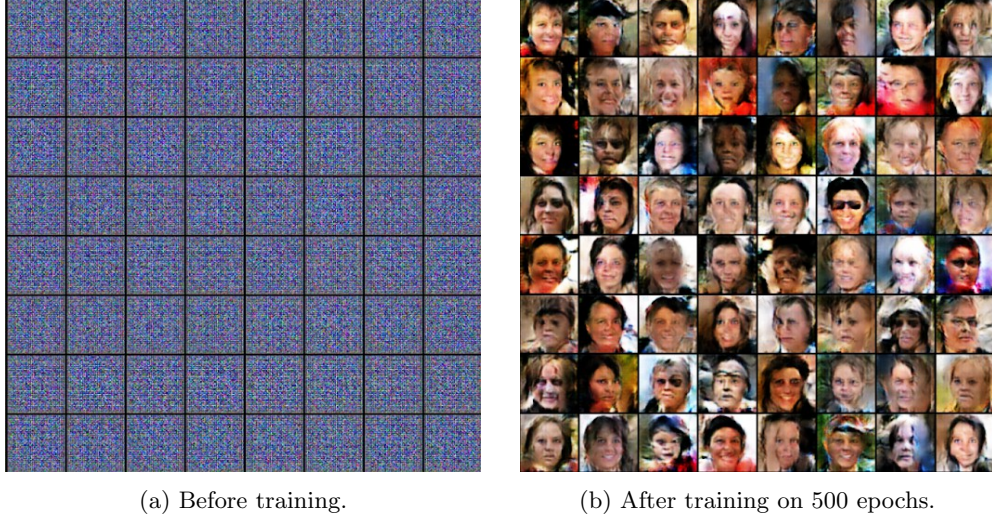


Fig. 5: Image generated by proposed fine-tuned GAN using FFHQ Dataset: (a) Before training generating noisy images, (b) After training on 500 epochs generating images similar to FFHQ Dataset images.

The training process of our Generative Adversarial Network on three distinct datasets, namely Flickr, Celebs-HQ, and FFHQ, is vividly depicted in the accompanying loss graphs. As shown in Figure 6, each graph represents the evolution of the GAN’s training loss over a set number of epochs, shedding light on the model’s convergence and generative performance.

Notably, the GAN fine-tuned on the Celebs-HQ dataset exhibits an initial phase of a gradual decline in the loss curve, indicating rapid early-stage learning. In contrast, both the FFHQ and Flickr datasets display a gradual incline in the loss curves during the initial epochs, suggesting a slower learning process. As the training progresses, all three datasets ultimately converge towards lower loss values, reflecting the GAN’s increasing proficiency in generating high-quality content. These loss graphs offer valuable insights into the distinct learning dynamics of the GAN across varied datasets and training durations, underscoring the adaptability of the model to different data sources and the impact of training duration on its generative capabilities.

5.2 Result Analysis of Stable Diffusion

In Figure 7, the fine-tuned model demonstrates significant improvements, producing images that are more visually convincing. The images obtained by the fine-tuned Stable Diffusion model exhibit a higher level of visual realism. Details, textures, and structures in these images closely resemble the characteristics of the real-world objects

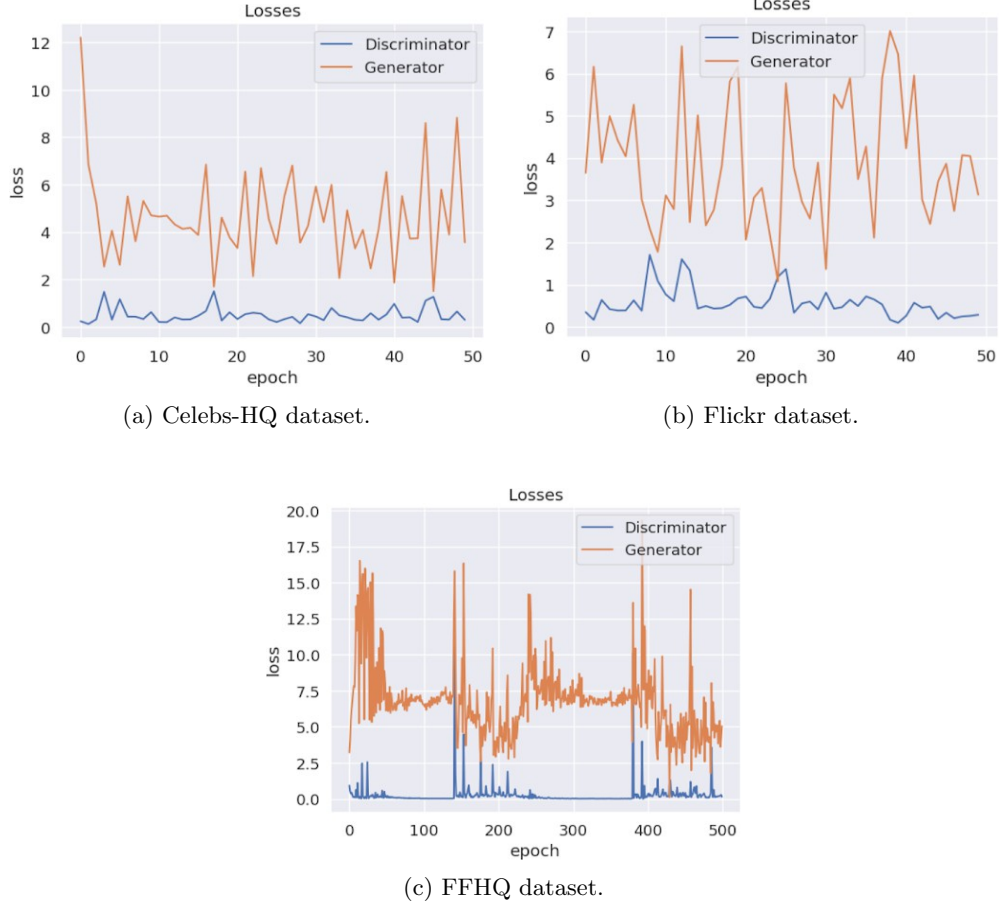


Fig. 6: Result Analysis Of GAN using Epoch vs. Loss Graphs of (a) Celebs-HQ dataset, (b) Flickr Dataset, and (c) FFHQ Dataset for both Generator and Discriminator Networks of GAN.

or scenes. The fine-tuned model appears to produce images with greater consistency in terms of color, lighting, and object proportions. This results in a more coherent and visually pleasing output.

Fine-tuning enhances the model’s ability to generate recognizable objects or scenes. It reduces instances of abstract or distorted shapes, making the generated content more relatable. Artifacts, such as noise or blurriness, are notably reduced in the images produced by the fine-tuned model. This leads to cleaner and crisper results.

As shown in Figure 8, the drawing generated by fine-tuning the model on the original artistic drawing exhibits an impressive degree of high fidelity. It faithfully captures the essence and core features of the original artwork. The shapes and artistic style in the generated drawing closely mirror those in the original artwork. This shows

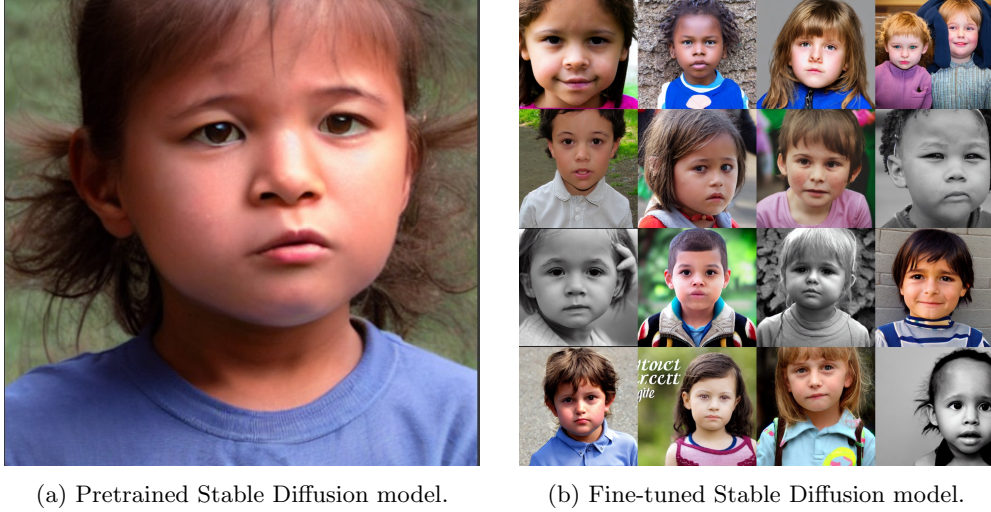


Fig. 7: Generated images comparison of Pretrained and Fine-tuned Stable Diffusion model. The pretrained model is fine-tuned on the FFHQ dataset. Comparing image generated by (a) Pretrained Stable Diffusion model, and (b) Fine-tuned Stable Diffusion model.

that the fine-tuned model has successfully retained the artistic qualities of the source material.

Fine-tuning has allowed the model to preserve intricate details in the original artwork, including fine lines, textures, and intricate patterns. The level of detail in the generated drawing is strikingly similar to the source. Fine-tuning has not resulted in any loss of clarity regarding the subject matter of the drawing. Objects or scenes within the generated image remain recognizable and coherent with the original content.

We have used actor Brad Pitt images for fine-tuning the Stable Diffusion model. As shown in Figure 9, the images generated by the fine-tuned model for the specific actor demonstrate a striking level of accuracy in capturing their current appearance. Facial features, expressions, and visual characteristics closely resemble the real-life subject, showcasing the model’s proficiency in mimicking the actor’s likeness.

Perhaps the most noteworthy result is the model’s capacity to produce childhood images of the actor. Even though childhood images were not incorporated in the dataset, the fine-tuned model effectively extrapolates the actor’s earlier appearance, demonstrating a sophisticated understanding of facial aging and transformation over time.

In Figure 10, the bar graph presents the comparison of CLIP scores across different models for three datasets: FFHQ, Flickr, and Celeb-HQ. The models are represented on the x-axis, while the y-axis denotes the CLIP scores.

The `stability/stable-diffusion-2-base` model outperforms other models for the FFHQ and Flickr datasets, achieving the highest CLIP scores. This suggests that this model is particularly effective for these two datasets. On the other hand, the

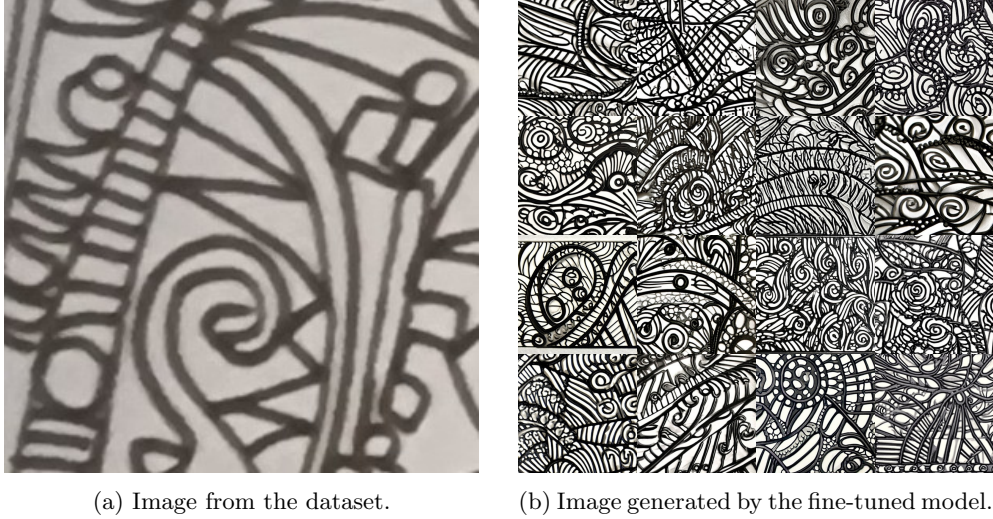


Fig. 8: The fine-tuning of artist images data on Stable Diffusion model and the comparison of dataset image and image generated by fine-tuned Stable Diffusion model: (a) Image from the dataset, and (b) Image generated by the fine-tuned Stable Diffusion model.

CompVis/stable-diffusion-v1-4 model exhibits superior performance on the Celeb-HQ dataset, securing the highest CLIP score among all models. This indicates that this model is more suited to the Celeb-HQ dataset.

In Figure 11, the batch size (one of the hyperparameters) is compared with the CLIP score of the proposed model. From the graph, it can be concluded that the highest CLIP score of 31.9821 is achieved for a batch size of 5.

In Figure 12, the bar graph compares the CLIP scores of existing models to those of the fine-tuned models. The figure offers a visually compelling representation of the model performance improvements. Each bar on the graph corresponds to a specific model, with the existing models represented by one set of bars and the fine-tuned models by another.

The height of each bar reflects the CLIP score, providing a clear and quantifiable measure of the models' capabilities. The steeper or higher bars among the fine-tuned models indicate a significant improvement in their CLIP scores, demonstrating the efficacy of the fine-tuning process.

6 Discussion

The findings of this research project underscore the effectiveness of both GANs and Stable Diffusion models in generating realistic images. While GANs exhibit notable improvements in image quality post-training, the comparison with Stable Diffusion models reveals a slight disparity, with the latter producing higher-quality images. This highlights the importance of exploring alternative techniques like Stable Diffusion,



(a) Brad Pitt image from the dataset.



(b) Childhood image generated by model.

Fig. 9: Generated images comparison of Pretrained and Fine-tuned Stable Diffusion model. The pretrained model is fine-tuned on the Celeb dataset: (a) Brad Pitt image from the dataset, and (b) Childhood image of Brad Pitt generated by fine-tuned Stable Diffusion model.

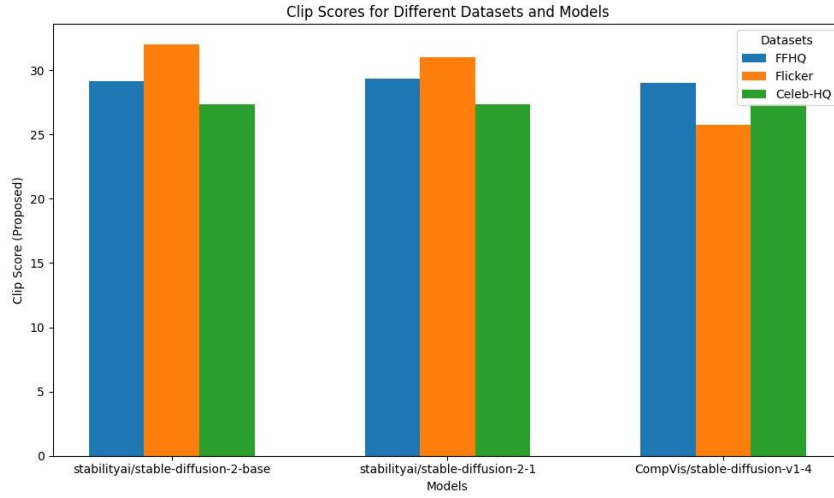


Fig. 10: Result Analysis of various versions of Stable Diffusion Models. The x-axis indicates the different versions of the Stable Diffusion model for three different datasets (FFHQ, Flickr, and Celeb). The y-axis indicates the CLIP scores of the fine-tuned models.

Table 5: Performance Evaluation of Pretrained and Fine-tuned models on various Datasets using Real or Fake Accuracy matrix for GANs and Clipscore matrix for Stable Diffusion models

Dataset	Model	Real or Fake	Clipscore (existing)	Clipscore (proposed)	Placeholder token
FFHQ	stabilityai/stable-diffusion-2-base	–	27.2277	29.1549	<batch2 proj 0>
	stabilityai/stable-diffusion-2-1	–	26.7496	29.3114	<batch2 proj 4>
	CompVis/stable-diffusion-v1-4	–	27.3863	28.9897	<batch2 proj 2>
	GAN	Real: 0.8404 Fake: 0.1018	–	–	–
Flickr	stabilityai/stable-diffusion-2-1	–	28.9321	31.9821	<batch2 proj 7>
	CompVis/stable-diffusion-v1-4	–	30.1758	31.0349	<batch2 proj 9>
	GAN	Real: 0.9125 Fake: 0.1562	–	–	–
Celeb	stabilityai/stable-diffusion-2-1	–	25.6839	25.7778	<batch2 proj 11>
	CompVis/stable-diffusion-v1-4	–	26.6171	27.3278	<batch2 proj 11>
	GAN	–	–	–	–

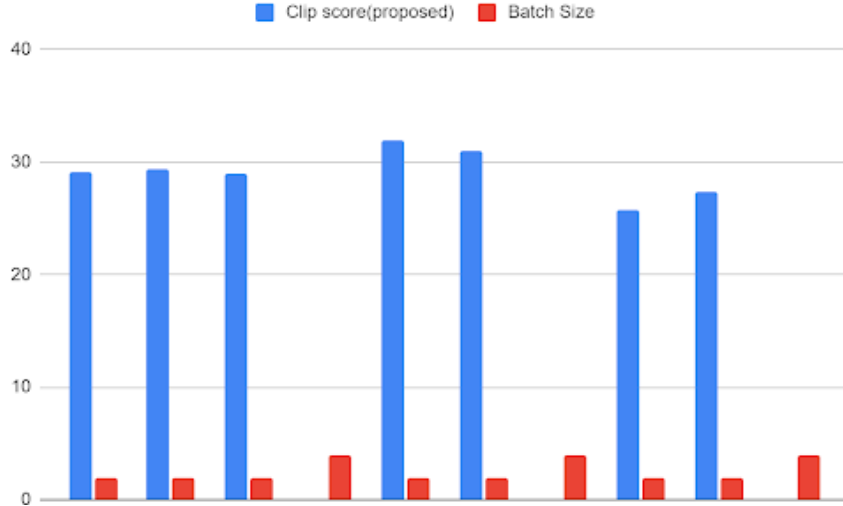


Fig. 11: Comparison of batch size and CLIP score proposed for Stable Diffusion Models over FFHQ, Flickr, and Celeb Datasets. (The blue colour indicates the CLIP score and the red colour indicates the batch size.)

which address the inherent limitations of GANs, such as mode collapse and non-convergence issues.

Moreover, the emergence of Stable Diffusion as a viable alternative emphasizes the significance of diversifying approaches in image generation tasks. By optimizing Stable Diffusion models, researchers can harness their potential to produce high-fidelity images across various domains. This indicates a promising avenue for further research and development, with opportunities to explore techniques like style transfer, image diversity generation from limited examples, and artistic image generation.

Moving forward, continued exploration and refinement of both GANs and Stable Diffusion models hold the key to unlocking innovative use cases in domains such as computer vision, digital art, and multimedia content generation. By leveraging the strengths of each approach and addressing their respective limitations, researchers can pave the way for groundbreaking advancements with far-reaching implications in diverse domains.

7 Conclusion

To summarize, the project focused on creating realistic images using Generative Adversarial Networks. During the project, an image dataset was used to train a GAN model. The results showed that the generated images visually resembled real images. However, when compared to images generated by Stable Diffusion, the quality of GAN-generated images was slightly lower. Stable Diffusion emerged as a technique capable of producing high-quality images by addressing the shortcomings of GANs. This explains how to use and optimize Stable Diffusion models.

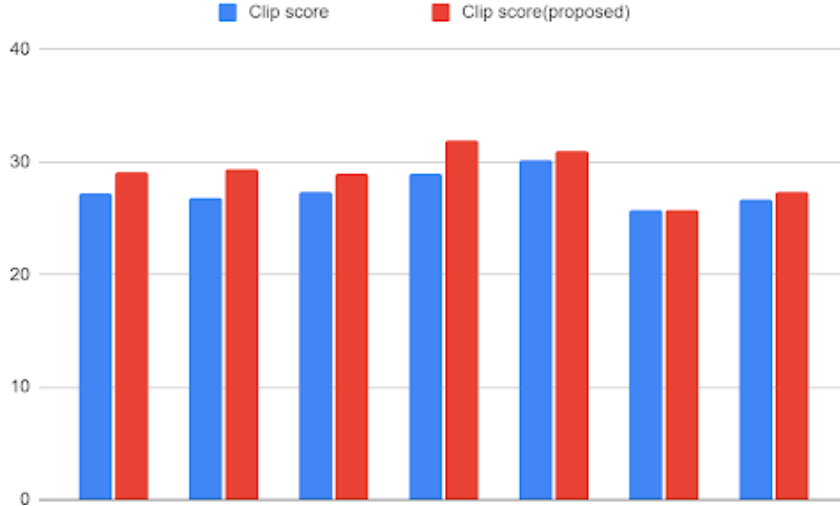


Fig. 12: Comparison of CLIP scores between existing baseline models and the proposed fine-tuned models across the evaluation datasets.

Fine-tuning the Stable Diffusion model demonstrates significant improvements in visual realism, texture consistency, and the reduction of artifacts across various datasets. Furthermore, the fine-tuned model exhibits the capacity to extrapolate realistic childhood representations of subjects even when such imagery is absent from the training data.

Further research and development in this field has the potential to yield exciting applications in a variety of domains. It could, for example, lead to techniques such as style transfer, generating a diverse range of images from a small set of examples, and even artistic image generation.

Declarations

- Funding: This study was carried out without any financial assistance.
- Conflict of Interests: Not applicable.
- Ethics approval and consent to participate: Not applicable.
- Consent for publication: Not applicable.
- Consent for participate: Not applicable.
- Data Availability: The datasets generated and analysed in this work are publicly available as given below. To evaluate the performance of the proposed GAN and Stable Diffusion models, two primary datasets were utilized for experimentation, training, testing, and validation:
 - Flickr-Faces-HQ Dataset (FFHQ): A subset of 3,000 random images was sampled from this dataset to conduct training and validation procedures.
 Source URL: <https://www.kaggle.com/datasets/arnaud58/flickrfaceshq-dataset-ffhq>

- Flickr Image Dataset: A subset of 6,000 random images was selected and processed for the experimental pipeline.
Source URL: <https://www.kaggle.com/datasets/hsankesara/flickr-image-dataset>
- Materials availability: The datasets generated and analysed in this work are publicly available as given below.
- Code availability: All the preprocessing, model training scripts, and evaluation tools used in this study are available below. The complete implementation code for the Stable Diffusion Models is hosted and executed on Google Colab. The source code, environment configurations, and model checkpoints can be accessed directly at: https://colab.research.google.com/drive/1lQqP8ZWcyqUPV_Zj1vR8um7dwxPXP6pk
- Author contribution:
 - Satishkumar Varma was instrumental in guiding the team, encouraging them to be critical thinkers and select a research direction that met scientific relevance and impact. His constant input regarding reviewing major milestones, enhancing the scope of research, and suggesting prudent insights influenced the overall framework and depth of the study. He also closely tracked the progress of the team and made useful comments during the course of the project. Her insightful analyses and constructive critique helped keep things clear, ensured academic precision, and refined the final work. He, also carried out the corrections, timely technical responses to the journal including revisions and LaTeX file modification and execution.
 - Kunal Wagh identified the research gap, set up the experiment and carried out important parts of the analysis. He was also directly involved in the core research and manuscript writing and did the visualisation so as to keep everything coherent from one stage to the next. He was instrumental in refining methodology and incorporating feedback from several rounds of reviews.
 - Omkar Raul managed the technical execution, including monitoring performance, preprocessing, and model training. He collaborated on ensemble development, hyperparameter tuning, and model selection.
 - Sejal Chandgadker managed the technical execution, including monitoring performance, preprocessing, and model training. She collaborated on ensemble development, hyperparameter tuning, and model selection.
 - Shreya Belanekar managed the technical execution, including monitoring performance, preprocessing, and model training. She collaborated on ensemble development, hyperparameter tuning, and model selection.
 - Kanchan Dabre managed the corrections, timely technical responses to the journal including revisions and LaTeX file modification and execution. Also prepared publicly available links.

References

- [1] Zhu, P., Abdal, R., Qin, Y., Wonka, P.: Sean: Image synthesis with semantic region-adaptive normalization. In: Proceedings of the IEEE/CVF Conference on

- Computer Vision and Pattern Recognition (CVPR), pp. 5116–5125 (2020). <https://doi.org/10.1109/CVPR42600.2020.00515>
- [2] Rajesh, B., Dusa, N., Javed, M., Dubey, S.R., Nagabhushan, P.: T2ci-gan: Text to compressed image generation using generative adversarial network. In: Proceedings of IAPR’s 6th International Conference on Computer Vision and Image Processing (CVIP) (2022). <https://doi.org/10.48550/arXiv.2210.03734>
 - [3] Nguyen, L.X., Aung, P.S., Le, H.Q., Park, S.-B., Hong, C.S.: A new chapter for medical image generation: The stable diffusion method. In: 2023 International Conference on Information Networking (ICOIN), pp. 483–486 (2023). <https://doi.org/10.1109/ICOIN56518.2023.10049010> . IEEE
 - [4] Zhang, S.: Dreambooth-based image generation methods for improving the performance of cnn. In: 2023 IEEE 3rd International Conference on Electronic Technology, Communication and Information (ICETCI), pp. 1181–1184 (2023). <https://doi.org/10.1109/ICETCI57876.2023.10176568> . IEEE
 - [5] Anokhin, I., Demochkin, K., Khakhulin, T., Sterkin, G., Lempitsky, V., Korzhenkov, D.: Image generators with conditionally-independent pixel synthesis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14236–14245 (2021). <https://doi.org/10.48550/arXiv.2011.13775>
 - [6] Yi, D., Guo, C., Bai, T.: Exploring painting synthesis with diffusion models. In: 2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence (DTPI), pp. 332–335 (2021). <https://doi.org/10.1109/DTPI52967.2021.9540115> . IEEE
 - [7] Niu, Y.: Research on image generation art creation assisted by deep learning and embedded systems. Discover Artificial Intelligence **720**(6) (2026) <https://doi.org/10.1007/s44163-026-01686-2>
 - [8] Gong, H.: A diffusion-based multi-style image generation framework with adaptive weighted style fusion. Discover Artificial Intelligence (2026) <https://doi.org/10.1007/s44163-026-01921-w>
 - [9] Yu, B., Ding, Y., Xie, Z., Huang, D.: Stacked generative adversarial networks for image compositing. EURASIP Journal on Image and Video Processing **2021**(1), 1–17 (2021) <https://doi.org/10.1186/s13640-021-00550-w>
 - [10] Chavdarova, T., Fleuret, F.: Sgan: An alternative training of generative adversarial networks. arXiv preprint arXiv:1712.02330 (2017) <https://doi.org/10.48550/arXiv.1712.02330>
 - [11] Ying, H., Wang, H., Shao, T., Yang, Y...., Zhou, K.: Unsupervised image generation with infinite generative adversarial networks. In: Proceedings of the

- IEEE/CVF International Conference on Computer Vision (ICCV) (2021). <https://doi.org/10.48550/arXiv.2108.07975>
- [12] Baviskar, V., Sable, R., Kokane, D., Shelke, P., Malwadkar, S., Shinde, T., Verma, M., Gaikwad, R.: Imaginera: Text-to-image generation using stable diffusion. In: Chaki, N., Debnath, P., Das, N., Yang, X.-S. (eds.) Proceedings of International Conference on Data Analytics and Insights, pp. 415–425 (2026). https://doi.org/10.1007/978-3-032-14772-1_33
 - [13] Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8110–8119 (2020). <https://doi.org/10.48550/arXiv.1912.04958>
 - [14] Chai, L., Gharbi, M., Shechtman, E., Isola, P., Zhang, R.: Any-resolution training for high-resolution image synthesis. In: Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XVI, pp. 150–168 (2022). <https://doi.org/10.48550/arXiv.2204.07156>
 - [15] Choi, Y., Uh, Y., Yoo, J., Ha, J.-W.: Stargan v2: Diverse image synthesis for multiple domains. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8188–8197 (2020). <https://doi.org/10.48550/arXiv.1912.01865>
 - [16] Liu, M.-Y., Huang, X., Yu, J., Wang, T.-C., Mallya, A.: Generative adversarial networks for image and video synthesis: Algorithms and applications. Proceedings of the IEEE **109**(5), 839–862 (2021) <https://doi.org/10.48550/arXiv.2008.02793>
 - [17] Zhang, C., Zhang, C., Zhang, M., Kim, I.K.: Text-to-image diffusion model in generative ai: A survey. arXiv preprint arXiv:2303.07909 (2023) <https://doi.org/10.48550/arXiv.2303.07909>
 - [18] Zhang, L., Rao, A., Agrawala, M.: Adding conditional control to text-to-image diffusion models. arXiv preprint arXiv:2302.05543 (2023) <https://doi.org/10.48550/arXiv.2302.05543>
 - [19] Zhao, Y., Xu, Y., Xiao, Z., Hou, T.: Mobilediffusion: Subsecond text-to-image generation on mobile devices. arXiv preprint arXiv:2311.16567 (2023) <https://doi.org/10.48550/arXiv.2311.16567>
 - [20] Chai, P., Hou, L., Zhang, G., Tushar, Q., Zou, Y.: Generative adversarial networks in construction applications. Automation in Construction **158** (2024) <https://doi.org/10.1016/j.autcon.2024.105265>
 - [21] Ivanovs, M., *et al.*: Synthetic image generation with a fine-tuned latent diffusion model for organ on chip cell image classification. In: 2023 Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA), pp. 148–153

- (2023). <https://doi.org/10.23919/SPA59660.2023.10274460> . IEEE
- [22] Hamazaspyan, M., Navasardyan, S.: Diffusion-enhanced patchmatch: A framework for arbitrary style transfer with diffusion models. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 797–805 (2023). <https://doi.org/10.1109/CVPRW59228.2023.00087>
 - [23] Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., Chen, M.: Hierarchical text-conditional image generation with clip latents. arXiv preprint arXiv:2204.06125 (2022) <https://doi.org/10.48550/arXiv.2204.06125>
 - [24] Sonal S. Fatangare, P.P.G.: Triplex attention enabled diffusion generative adversarial network for photorealistic image generation from text and images. Engineering Applications of Artificial Intelligence **166** (2026) <https://doi.org/10.1016/j.engappai.2025.113588>