Text-To-Image Generation Using Generative Adversarial Network

Harsh Sawaji[1], Musab Shaikh[2], Aayush Shah[3], Sumitra Sadhukhan[4]
[4]*Assistant Professor, Department of Computer Engineering, MCT's Rajiv Gandhi Institute of Technology, Andheri, Mumbai*
[1,2,3]*Student, Department of Computer Engineering, MCT's Rajiv Gandhi Institute of Technology, Andheri, Mumbai*

Abstract- **The creation of images is a process that can be achieved using various methods, including manual techniques such as painting and drawing, or digital methods such as computer-generated graphics or image generation using deep learning models. In recent years, there has been a surge of interest in using deep learning algorithms, particularly generative models, to generate realistic and high-quality images.**

One such model is the Generative Adversarial Network (GAN), which is a deep learning architecture composed of two neural networks: a generator and a discriminator. The generator creates synthetic data (images) from random noise, while the discriminator's task is to distinguish between the synthetic data and real data. The two networks are trained together in a zero-sum game, where the generator aims to produce synthetic data that is indistinguishable from the real data, while the discriminator tries to correctly identify the synthetic data. Through this adversarial training process, GANs can generate high-quality synthetic data that can be used for a variety of applications, including image and video generation, data augmentation, and domain adaptation. GANs have shown impressive results in various fields, but they also pose some challenges, such as mode collapse, where the generator produces a limited set of outputs and instability during training.

Keywords: Convolution neural network, discriminator, Generative Adversarial Networks, generator, Unsupervised Learning.

I. INTRODUCTION

Text-to-image generation is a fascinating research area that seeks to bridge the gap between natural language and visual content. It involves creating images that match the textual description provided by a user. The development of text-to-image generation has been made possible by the emergence of Generative Adversarial Networks (GANs), which have revolutionized the field of computer vision. GANs are deep neural networks that can learn to generate realistic images by training on large datasets of real images. They are particularly useful for text-to-image generation because they can learn to generate images that match the semantics of the input text. This paper aims to provide an in-depth review of text-to-image GANs, including their history, current state-of-the-art techniques, and potential applications.

The concept of text-to-image generation has been around for several decades, but it was not until the advent of GANs that it became a viable research area. In the past, researchers used rule-based methods to generate images from textual descriptions. However, these methods were limited in their ability to produce realistic images that accurately reflected the textual description. With the introduction of GANs, researchers have been able to create more advanced models that can learn to generate high-quality images that accurately reflect the input text.

In the context of text-to-image generation, the input to the generator is a textual description, which is converted into a vector representation using techniques such as word embeddings. This vector is then used as input to the generator, which creates an image that matches the textual description. The discriminator is then used to evaluate the quality of the generated image, and the generator is updated accordingly. This process is repeated iteratively until the generator produces images that accurately reflect the input text.

Several techniques have been proposed to improve the performance of text-to-image GANs. One approach is to use attention mechanisms, which allow the generator to focus on specific parts of the input text

when creating the image. Another approach is to use conditional GANs, which incorporate additional information such as object locations or image attributes into the generator. These techniques have been shown to improve the quality of the generated images, making them more realistic and accurate.

Text-to-image GANs have a wide range of potential applications. For example, they could be used to generate images for virtual reality environments or video games. They could also be used in e-commerce applications to generate product images from textual descriptions. Additionally, they could be used in healthcare applications to generate images of medical conditions based on textual descriptions provided by doctors.

II. LITERATURE SURVEY

Generative Adversarial Networks (GANs) are a type of deep learning model that have gained a lot of popularity in recent years due to their ability to generate realistic images, videos, and other types of data.GAN were designed in year 2014 by Ian Goodfellow [1], Since then there have been many modifications and improvements that are done in GAN concept.

This paper introduced the concept of GANs, which consists of a generator and a discriminator network. The generator network generates fake samples, while the discriminator network tries to distinguish between real and fake samples. The two networks are trained together in an adversarial manner to improve the quality of the generated samples [1].

This paper proposed DCGAN, which uses deep convolutional neural networks to improve the quality of generated images. The authors demonstrated that DCGANs can generate high-quality images of faces, bedrooms, and other objects [2].

This paper introduced Conditional GANs (cGANs), which can be used for image-to-image translation tasks. The authors showed that cGANs can be used to translate images from one domain to another, such as translating images from day to night or from a sketch to a photorealistic image [3].

This paper proposed StackGAN++ that can generate high-quality images at multiple resolutions. The authors demonstrated that StackGAN++ can generate realistic images of birds and flowers at resolutions up to 1024x1024 pixels [4].

This paper introduced StyleGAN, which uses a style-based generator architecture to generate high-fidelity images with diversity and controllability. They proposed several improvements to the model, resulting in images of unprecedented quality [5].

More recently, this paper proposed Diffusion Models (DIFs) as a new approach to image synthesis. DIFs generate high-fidelity images by iteratively adding noise to a seed image and then diffusing the noise out over time, resulting in images that are difficult to distinguish from real photos [6].

This literature survey highlights the significant advancements made in GANs, with various extensions and improvements proposed for generating high-quality synthetic samples. As the field continues to evolve, it is likely that further developments will be made to improve the performance and stability of these models.

These are just a few examples of the many papers published on GANs. As the field continues to evolve, it is likely that new variations and techniques will be developed to improve the performance and stability of these models.

III. MATHEMATICAL FORMULAE AND FUNCTIONS

1. Probability distributions:
   Probability density function (PDF): $p(x)$ represents the probability density function of the underlying data distribution.
   Generator's probability density function (PDF): $p\_g(x)$ represents the probability density function of the generated data distribution.
2. Loss functions:
   - Generator loss function: $L\_G = log(1 - D(G(z)))$ measures how well the generator is able to fool the discriminator. Here, z is the random noise input to the generator network, $G(z)$ is the generated sample, and $D(G(z))$ is the probability assigned to the generated sample by the discriminator.
   - Discriminator loss function: $L\_D = -[log(D(x)) + log(1 - D(G(z)))]$ measures how well the discriminator is able to distinguish between real

and fake samples. Here, *x* is a real sample from the training data.

3. Adversarial training:
   Adversarial training involves optimizing the generator and discriminator networks in competition with each other. The generator network tries to minimize the generator loss function *L_G*, while the discriminator network tries to minimize the discriminator loss function *L_D*.
4. Gradient descent:
   Gradient descent is used to optimize the generator and discriminator networks during training. The gradients of the loss functions with respect to the network parameters are computed using backpropagation, and the parameters are updated using stochastic gradient descent (SGD).
   Update rule: $\theta \leftarrow \theta - \alpha * \nabla(L(\theta))$ where θ is the network parameters, α is the learning rate, and $\nabla(L(\theta))$ is the gradient of the loss function with respect to the network parameters.
5. Convolutional neural networks:
   Convolutional neural networks (CNNs) are often used as the basis for the generator and discriminator networks in GANs.
   Convolution operation: $h[i,j] = (w * x)[i,j]$ represents the convolution operation between the filter w and the input *x*, where $h[i,j]$ is the output at position (i,j) and * denotes the convolution operator.

IV. METHODOLOGY

The focus of this project is to create an application that utilizes a Generative Adversarial Network (GAN) model to generate images based on text prompts. By inputting a written description, the app will use the GAN to create an image that matches the given text. This process involves training the GAN on a large dataset of images and text descriptions, allowing it to learn how to generate realistic images from textual input. The end result is an application that can be used to generate custom images based on user input.

To begin training a Conditional Generative Adversarial Network (cGAN) model, the first step is to gather a dataset of images and their corresponding labels. In this project, three datasets were selected for this purpose. The first dataset is a collection of cat images obtained from Crawford, the second dataset consists of anime faces and was obtained from Spalcher, and the third dataset is the Celeb-A dataset. Each of these datasets contains a large number of images, along with corresponding labels that describe the contents of the image. These datasets will be used to train the cGAN model to generate images based on specific input conditions.

In a conditional Generative Adversarial Network (cGAN), there are two key components: a generator and a discriminator. The generator takes a noise vector and a label as inputs and generates an image that corresponds to the label. Our generator model consists of an embedding layer and a linear layer to process the label input, as well as a linear layer to process the noise vector input. The outputs of these layers are concatenated and reshaped to form a 4-dimensional tensor. The tensor is then passed through two transposed convolutional layers with batch normalization and leaky ReLU activation functions, and finally through a convolutional layer with a hyperbolic tangent activation function, resulting in the generated image. This architecture allows the generator to learn how to generate images that match the given label.

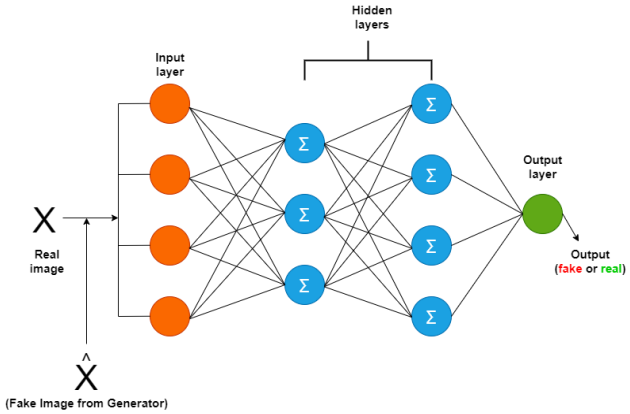

Fig. 01. Generator Architecture

And the discriminator model takes an image and a label as inputs and determines whether the image is real or generated by the generator. The discriminator architecture consists of an embedding layer and a linear layer to process the label input. The resulting output is then reshaped and concatenated with the image tensor to form a 4-dimensional tensor. The tensor is then passed through two convolutional layers with batch normalization and leaky ReLU activation functions, followed by a fully connected layer with a

dropout layer and a sigmoid activation function to produce the output value. This output represents the probability that the input image is real. The architecture is designed to allow the discriminator to learn to distinguish between real and generated images that match the given label.

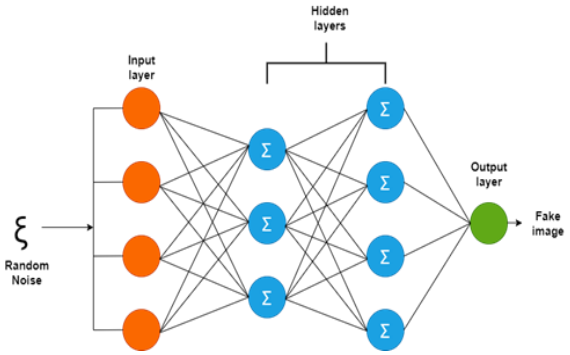

Fig. 02. Discriminator Architecture

To develop a GAN model that can generate realistic images from textual input, we first needed to train the model using the selected datasets.The generator and discriminator are trained in an adversarial manner, with the generator attempting to produce images that fool the discriminator, and the discriminator attempting to distinguish between real and fake images. The generator and discriminator are trained using backpropagation and gradient descent, with the goal of minimizing the adversarial loss function.

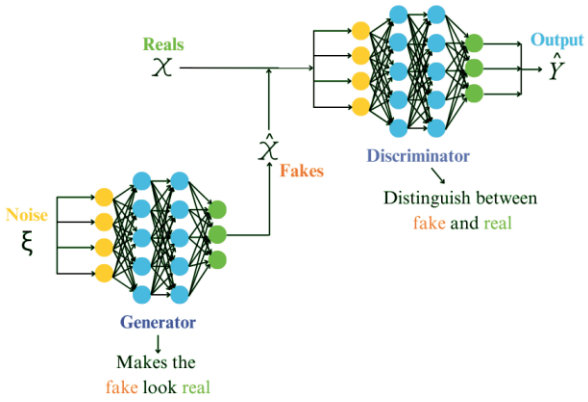

Fig. 03. GAN Architecture

For each of the datasets, we trained the GAN model until it was capable of generating images that closely resembled the training data. This involved running multiple training iterations and fine-tuning the model's parameters to improve its performance. During the training process, the model was fed batches of images and corresponding labels, and it used this data to learn how to generate realistic images based on the given conditions. As the model was trained, we monitored its progress and made adjustments to ensure that it was learning effectively and generating high-quality images. Once the model was capable of generating realistic images similar to the training data, we could move on to using it for image generation based on text prompts.

The frontend of our app is developed using the Flutter technology, which is capable of creating mobile applications for both iOS and Android. One of the key features of our app is a dropdown list that allows users to select from three different models, namely Cat model, Human model, and Anime model. Users have the freedom to choose any of these models as per their preference. Once the user selects a model, they can enter the text prompt which describes the image they want to generate. A generate button is provided, which remains disabled until the user completes the above-mentioned two steps. After completing these steps, users can click on the generate button, which triggers an HTTP POST request to be sent to the backend. The backend then processes the request and generates the desired image, which is converted into the base64 encoding.

Once the backend generates the image, it sends it back to the frontend in the form of a base64 encoded string. The frontend then processes this string and converts it into a .JPG format image. This conversion is carried out to minimize the load on the backend, as it is more efficient to send text data than images, due to lower bandwidth consumption.

V. RESULT

In this study, we implemented Generative Adversarial Networks (GANs) to generate images based on given prompts. The first prompt was "Cat with blue hat, realistic, cute" and the generated image shows a realistic and adorable cat wearing a blue hat. The second prompt was "Beautiful women with red hair, portrait, 4k" and the generated image depicts a high-resolution portrait of a stunning woman with red hair. These results demonstrate the potential of GANs to create realistic and visually appealing images based on specific prompts.

Fig. 04. Cat image using prompt “cat with blue hat,realistic,cute”

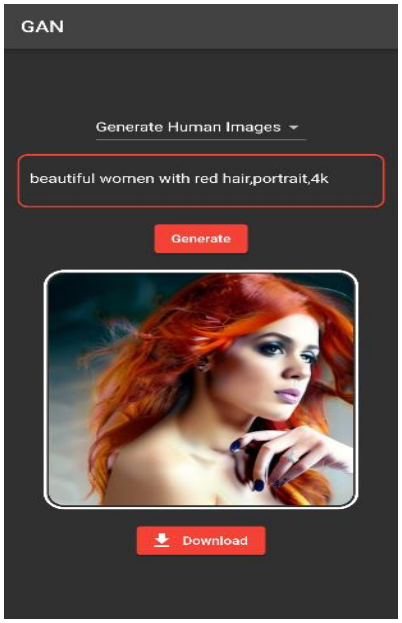

Fig.05. A woman portrait using prompt “beautiful woman with red hair, portrait,4k”

VI. Conclusion

Hence, the proposed system consists of three Conditional Generative Adversarial Network (cGAN) models: the Cat model, the Face model, and the Anime Art model. Each of these models has been trained on a specific dataset - the Cat model on cat images obtained from Crawford, the Face model on the Celeb-A dataset, and the Anime Art model on anime faces obtained from Spalcher. The goal of these models is to generate realistic images based on textual input prompts.

Our app enables users to generate and utilize images without any restrictions. Unlike human images, which are often subject to copyright laws, our GAN-generated images do not exist in the real world, meaning users are not required to pay for them or use them with watermarks. This also eliminates any possibility of copyright infringement, allowing users to freely use the generated images without any legal implications.Our app provides a novel approach to image generation, which has the potential to benefit various industries such as advertising and design, by providing access to high-quality images without any copyright limitations. Overall, the app offers a user-friendly platform for generating and using images, which is both efficient and cost-effective.

REFERENCES

[1] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio,Y.(2014): "Generative Adversarial Nets". In Advances in Neural Information Processing Systems (NIPS) (pp. 2672-2680).

[2] Radford, A., Metz, L., & Chintala, S. (2015): "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks". arXiv preprint arXiv:1511.06434.

[3] Isola, P., Zhu, J. Y., Zhou, T., & Efros, A. (2017): "Image-to-Image Translation with Conditional Adversarial Networks". In IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 5967-5976).

[4] Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., & Metaxas, D. (2018): "StackGAN++: Realistic Image Synthesis with Stacked Generative Adversarial Networks". IEEE Transactions on Pattern Analysis and Machine Intelligence, 41(8), 1947-1962.

[5] Karras, T., Aila, T., Laine, S., & Lehtinen, J. (2019): "A Style-Based Generator Architecture for Generative Adversarial Networks". In IEEE

Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 4401-4410).

[6] Brock, A., Donahue, J., & Simonyan, Karen (2018): "Large Scale GAN Training for High Fidelity Natural Image Synthesis". arXiv preprint arXiv:1809.11096.