

EFFICIENT IMAGE SYNTHESIS USING GENETIC ALGORITHM – GENERATIVE ADVERSARIAL NETWORK

**Thesis Submitted
in Partial Fulfillment of the Requirements for the
Degree of**

**MASTER OF TECHNOLOGY
in
Data Science**

**by
Rohit Kumar
(2K22/DSC/13)**

**Under the supervision of
Mr. Sanjay Patidar
Assistant Professor, Department of Software Engineering,
Delhi Technological University**



Department of Software Engineering

**DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daultpur, Bawana Road, Delhi - 110042, India**

May, 2024



DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daultpur, Main Bawana Road, Delhi-42

CANDIDATE'S DECLARATION

I Rohit Kumar hereby certify that the work which is being presented in the thesis entitled "Efficient Image Synthesis Using Genetic Algorithm – Generative Adversarial Network" in partial fulfillment of the requirements for the award of the Degree of Master of Technology in Data Science, submitted in the Department of Software Engineering, Delhi Technological University is an authentic record of my own work carried out during the period from 2022 to 2024 under the supervision of Mr. Sanjay Patidar.

The matter presented in the thesis has not been submitted by me for the award of any other degree of this or any other Institute.

Candidate's Signature

This is to certify that the student has incorporated all the corrections suggested by the examiners in the thesis and the statement made by the candidate is correct to the best of our knowledge.

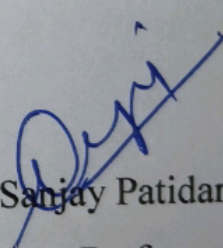
Signature of Supervisor (s)



DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daultapur, Main Bawana Road, Delhi-42

CERTIFICATE BY THE SUPERVISOR(s)

Certified that Rohit Kumar (2K22/DSC/13) has carried out his research work presented in this thesis entitled “Efficient Image Synthesis Using Genetic Algorithm – Generative Adversarial Network” for the award of Master of Technology from Department of Software Engineering, Delhi Technological University, Delhi, under my supervision. The thesis embodies results of original work, and studies are carried out by the student himself and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.


Mr. Sanjay Patidar
Assistant Professor

Department of Software Engineering
Delhi Technological University

Place: Delhi Technological University, New Delhi

Date: 30/05/2024

Efficient Image Synthesis Using Genetic Algorithm – Generative Adversarial Network

Rohit Kumar

ABSTRACT

GAN's have emerged as a powerful technique for generating high-quality images due to their unique characteristics and capabilities. In this report, there is a discussion for the motivation behind using GANs over other generative models for example Restricted Boltzmann Machines (RBMs), Deep Belief Networks (DBMs), and Variational Autoencoders (VAEs). This research highlights the advantages of GAN's in terms of image quality generation. To gain a comprehensive understanding of GAN's and their practical implementations, several studies have been conducted that aided in the creation of a GA-GAN framework. This research provides insights into the theoretical foundations and practical considerations of GANs for image synthesis. This paper introduces unsupervised learning techniques specifically designed for GANs, enabling their effective utilization with small datasets such as MNIST and CIFAR-10. Driven by the knowledge gained from these resources, this report shows a novel implementation on Genetic algorithm-based GAN model which are supported by learning rate schedulers. The approach incorporates various essential concepts and techniques to enhance the quality of image generation using limited datasets. Specifically, methods like normalization, data augmentation, batch normalization, and Adam optimizer were used to enhance the overall accuracy of GAN model. However, this report uses genetic algorithm instead of gradient based approach and also generate high quality image than the real images. For these experiments, the Anime Face Dataset was collected from Kaggle through API integration. This dataset comprises approximately 63,565 anime face images, which is similar in scale to the widely used CIFAR-10 dataset. By employing GA-GAN model with genetic algorithm for optimization, this research work aims to generate high-quality anime face images. The proposed framework employs two performance metrics termed as `real_score` of 0.9722 and `fake_score` of 0.0452. Binary cross entropy loss function was used for both generator and discriminator. These metrics provide valuable insights into the quality and diversity of the generated images. Additionally, the Fréchet inception distance (FID) score was also discussed, which is a widely used metric for evaluating the quality of generated images. The FID score compares the feature embeddings of the generated images and the original dataset using a pre-trained Inception model. A lower FID score indicates a closer similarity among original images and generated images, highlighting the success of the GAN network in capturing the underlying data distribution.



DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daultapur, Main Bawana Road, Delhi-42

ACKNOWLEDGEMENTS

I would like to express my deep gratitude to my project guide Mr. Sanjay Patidar, Assistant Professor, Department of Software Engineering, Delhi Technological University, for his guidance with unsurpassed knowledge and immense encouragement. I am also grateful to Prof. Ruchika Malhotra, Head of the Department, Software Engineering, for providing us with the required facilities for the completion of the Dissertation.

I'd also like to thank our lab assistants, seniors, and peer group for their aid and knowledge on a variety of subjects. I would like to thank my parents, friends, and classmates for their encouragement throughout the project period.

ROHIT KUMAR

2K22/DSC/13

TABLE OF CONTENTS

| | |
|----------------------------------------------|-------------|
| CANDIDATE’S DECLARATION | ii |
| CERTIFICATE BY THE SUPERVISOR(s) | iii |
| ABSTRACT | iv |
| ACKNOWLEDGEMENTS | v |
| TABLE OF CONTENTS | vi |
| LIST OF TABLES | vii |
| LIST OF FIGURES | viii |
| LIST OF ABBREVIATIONS | ix |
| CHAPTER 1: INTRODUCTION | 1 |
| 1.1 Background | 1 |
| 1.2 Motivation | 3 |
| CHAPTER 2: LITERATURE REVIEW | 6 |
| CHAPTER 3: RESEARCH GAP | 13 |
| CHAPTER 4: METHODOLOGY | 15 |
| 4.1 Proposed Work | 15 |
| 4.1.1 Discriminator Network | 18 |
| 4.1.2 Generator Network | 19 |
| 4.1.3 Genetic Architecture | 21 |
| 4.2 Training Process of Discriminator | 22 |
| 4.3 Training Process of Generator | 22 |
| 4.4 Training Process of Full Architecture | 23 |
| CHAPTER 5: EXPERIMENTAL SETUP | 24 |
| 5.1 Tools Used | 24 |
| CHAPTER 6: RESULTS AND ANALYSIS | 25 |
| 6.1 Generation of images from generator | 25 |
| 6.2 Plot scores v/s epochs | 26 |
| 6.3 Plot loss v/s epochs | 27 |
| 6.4 Performance Analysis | 28 |
| CHAPTER 7: CONCLUSION AND FUTURE WORK | 29 |
| REFERENCES | 30 |

LIST OF TABLES

| | |
|--------------------------------------|----|
| Table 2.1 Summary of reviewed paper | 8 |
| Table 4.1 Algorithm of proposed work | 15 |

LIST OF FIGURES

| | |
|----------------------------------------------------------------------------------------------|----|
| Fig. 1.1 Image synthesis [11] | 1 |
| Fig. 1.2 Deep directed generative model flow [10] | 4 |
| Fig. 2.1 LSUN-bedroom dataset [11] | 7 |
| Fig. 2.2 Shows generated images of cats [7] | 10 |
| Fig. 2.3 MNIST Dataset [11][12] | 12 |
| Fig. 4.1 Trainable GAN System | 16 |
| Fig. 4.2 GA-GAN high level system | 17 |
| Fig. 4.3 Shows the original set of images from Anime face dataset | 18 |
| Fig. 4.4 The process of a convolutional operation to generate feature map. | 18 |
| Fig. 4.5 Network architecture of the Discriminator | 19 |
| Fig. 4.6 Working of ConvTranspose2d layer | 20 |
| Fig. 4.7 Network architecture of the Generator | 21 |
| Fig. 4.8 Output of the generator at epoch 0 | 22 |
| Fig. 6.1 New generated images from generator at epoch 1 | 26 |
| Fig. 6.2 New generated images from generator at epoch 16 | 26 |
| Fig. 6.3 New generated images from generator at epoch 25 | 26 |
| Fig. 6.4 New generated images from generator at epoch 30 | 26 |
| Fig. 6.5 Plot of real_score v/s epochs and fake_score v/s epochs | 27 |
| Fig. 6.6 Plot of generator loss(loss_g) v/s epochs and discriminator_loss(loss_d) v/s epochs | 27 |

LIST OF ABBREVIATIONS

| | |
|------|--------------------------------|
| GAN | Generative Adversarial Network |
| GA | Genetic Algorithm |
| FID | Fréchet Inception Distance |
| RBM | Restricted Boltzmann Machine |
| CNN | Convolution Neural Networks |
| RNN | Recurrent Neural Networks |
| GPU | Graphics Processing Unit |
| EDA | Exploratory Data Analysis |
| DNN | Deep Neural Networks |
| ANN | Artificial Neural Networks |
| ReLU | Rectified Linear Unit |

CHAPTER 1

INTRODUCTION

1.1 Background

Deep learning has experienced remarkable success in computer vision, showcasing impressive performance in various practical applications like image classification, object detection, and image segmentation. Unsupervised learning tasks, like image generation, may not consistently achieve the same level of performance as supervised learning tasks, for example detection of objects and classification of images. Image generation aims to acquire the ability to generate images as illustrated in Fig. 1.1.

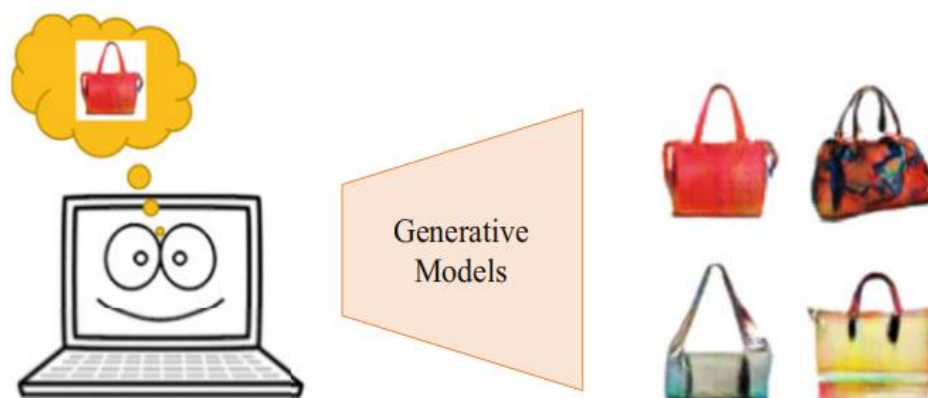


Fig. 1.1 Image synthesis [11]

Generative models are trained to understand the underlying distribution of images, enabling them to generate new images. However, image generation presents several challenges. Firstly, unlike supervised learning problems, image generation lacks explicit correspondence between the input and output. The generative model must therefore learn to effectively capture and represent this mapping. Secondly, image generation involves a significantly larger output dimension compared to many supervised learning tasks. For instance, when dealing with high-resolution images like 1024×1024 , the output dimension can exceed three million, necessitating efficient feature learning. After having these problems, image generation is still a worthy tool which can enhance the performance of various issues related to computer vision, like image-to-image translation [6], image compression [7], image super-resolution [5].

GAN is meant for Generative Adversarial Network in which this specific AI architecture comprises of two neural networks the discriminator and the generator where through a process of competition these networks are trained at a time. A GAN comprises of two main components: a generator and a discriminator. Let's talk about them in depth:

a) Generator: It is a neural network that creates artificial data. It takes input in the form of a random noise and learns to transform it into outputs that match the original dataset. The generator's purpose is to produce data which is very similar to real data, thereby "deceiving" the discriminator.

b) Discriminator: It is another neural network used to evaluate whether data is true or fallacious. It takes input from both the generator's results and real dataset. The discriminator's intention is to correctly distinguish which data is legitimate and which is synthetic.

GANs are trained in such a manner that the generator and discriminator fight against one another like in a game: The purpose generator is to output data which is alike from the original dataset by the discriminator whereas the goal of discriminator is to revamp its potential to disfavour between genuine and fallacious data. In this way this adversarial training permits both networks to improve with the passage of time. Since the generator network outputs more genuine data, the discriminator network has to become better at detecting fakes, giving the generator a chance to provide even better outcomes. This iterative algorithm produces a generator that is capable of producing data that approximately resembles the distribution of the original dataset.

Generative Adversarial Networks (GANs) are widely used picture synthesis technology that can generate realistic images from a diversity of inputs. It's time to see how GANs can be used for image synthesis:

- Random Noise to Images: The generator in GANs starts with taking random noise as input and then it learns to convert this noise into pragmatic images with the passage of time. This method is employed in many models like DCGAN (Deep Convolutional GAN), which translate a plain noise vector into sophisticated, realistic images like animals, human faces or some scenes.

- Image-to-Image Translation: GANs can translate images from one realm to another. Models like Pix2Pix and CycleGAN permit to have transformations like as turning crocodiles to alligator, day to night, and leopard to cheetah. This technique is often used in art, design, and data augmentation.

- Super – Resolution: GANs can convert the images of lower resolution to the images of higher resolution while keeping finer details and enhancing quality. Super-resolution GAN (SRGAN) and Enhanced SRGAN (ESRGAN) are famous

algorithms for increasing the augmentation of vintage images, satellite images, and medical imaging.

- **Style Transfer:** GANs can apply exclusive aesthetic styles to images, resulting in new era graphics. The StyleGAN series allows users to create images in a variety of styles, including paintings and realistic portraits, with authority over some characteristics such as hair, facial expressions, and background.
- **Image Completion:** GANs can impute missing or flawed image segments, a process known as inpainting. This tool can be harnessed to regain old photographs, tweak images, and even create mutations for data augmentation.
- **Text-to-Image Generation:** Text description visuals can be produced by some GANs. DALL-E and VQGAN-CLIP, for example, can accept a spoken directive and generate a picture in response, allowing for innovative image amalgam based on inputs of users.
- **Images Anomaly Detection:** GANs can yield "normal" images which can be used as an assistant to detect abnormalities in a given dataset. This is especially pertinent in medical imaging, as GANs have the ability to generate imitated healthy images to spot issues in scans.

1.2 Motivation

Image generation is the foremost task in the field of computer vision, addressing to gain the knowledge of underlying distribution of images and generate rational images that are identical from real ones. Conventional deep generative models like restricted Boltzmann machines (RBMs) [1], deep Boltzmann machines (DBMs) [2], and variational autoencoders (VAEs) [3] often deal with difficulties due to intransigent functions and interpretation challenges. Nevertheless, generative adversarial networks (GANs) can provide us with an alternative approach that facilitate end-to-end training without having any dependency on approximate inference. GANs are used to train a generator and a discriminator, where the generator generates new images and the discriminator differentiate between genuine and fake images. With the help of adversarial learning, GANs expedite a cutthroat give-and-take between the generator and discriminator to produce highly optimised and efficient realistic images. GANs have shown their strength in propagating authentic images across disparate realms, including faces, animals, objects, and scenes such as landscapes and cityscapes. In spite of their potentiality, GANs can display challenges in terms of nature of image and strength in the training process.

For image generation there are a category of machine learning models which are called as Deep directed generative models. They engage by sampling from a forthright distribution, generally a Gaussian distribution. The obtained samples are

afterwards processed through a neural network, which learns to transform the distribution shown in Fig. 1.2.

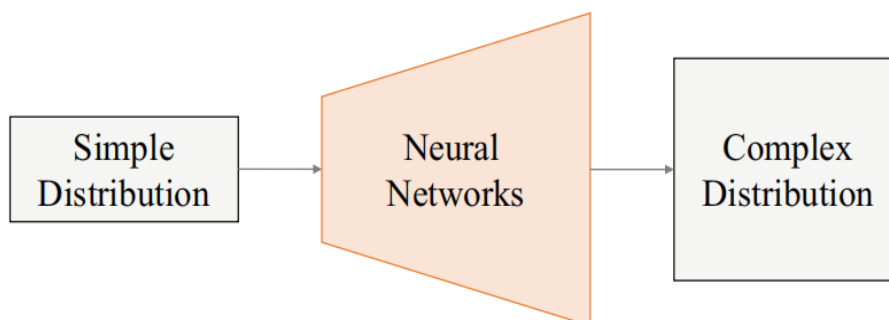


Fig. 1.2 Deep directed generative model flow [10]

Generative Adversarial Networks (GANs) have arisen as a potent class of deep directed generative models well-known for their poignant potential to produce lifelike images. GANs is comprised of two networks. Firstly, the foremost purpose of generator is to generate innovative images, secondly, the discriminator is liable for sagacious between various real and fabricated images. With the help of an adversarial training mechanism, the generator is responsible to create images which are deemed authentic by discriminator, while the discriminator continuously tries to work on its discrimination skills. This iterative training keeps on happening until the generator is gets the capability of producing images that closely matches with genuine ones, leading to the formation of exceedingly realistic generated images.

GANs are endorsed over other generative models including RBMs, DBMs, and VAEs due to the behaviour of them producing greater image quality and diversity. GANs are specifically useful for image-to-image translation, style transfer and super-resolution. In spite of their benefits, GANs have its own challenges such as training instability and the necessity for large-scaled datasets. To improve the training efficiency and stability there is a need to study which combines Genetic Algorithms with GANs, so that the high-quality images can be produced in a shorter amount of time by the goal of creating a strong picture synthesis framework

In conclusion, the boost for this research comes from the need to take advantage of GAN benefits while resolving their challenges using novel methodologies such as Genetic Algorithms. The sole purpose is to create a more optimized and stable image generation framework which has the ability of producing high-quality images while lowering training time and improving resilience.

This paper starts with the introduction of a comprehensive summary of the study's research background, motivation, and contributions. This chapter deals with the introduction of the challenges of efficient image synthesis with the Genetic

Algorithm and Generative Adversarial Networks (GA-GAN) then followed by a thorough evaluation of the existing literature reviews on generative models, specifically GANs. Key articles addressing image quality and training stability challenges are presented, highlighting research gaps and the need for the suggested study. There are limitations of current GAN approaches, particularly in dealing with limited data and including form and texture elements. This chapter emphasizes the need of real-time optimization as well as robustness to variations.

The GA-GAN system used includes the network designs of the discriminator and generator, as well as the use of genetic algorithms to improve training efficiency. The training techniques for both networks are thoroughly described. Then I discussed the tools and datasets utilized in the experiments, such as Jupyter Notebook, TensorFlow, and the Anime Face dataset. This chapter includes a detailed explanation of the experimental setup and implementation. I assessed the performance of the GA-GAN model using a variety of measures, including `real_score`, `fake_score`, and FID score. Visual outcomes of created images at several epochs are provided, as well as score and loss plots over epoch. I summarized the research findings and contributions, including a discussion of the GA-GAN model's effectiveness. This chapter discusses potential future studies, such as hybrid model development and application to larger datasets.

CHAPTER 2

LITERATURE REVIEW

Despite their effectiveness in generating high-quality images, generative adversarial networks (GANs) still encounter challenges, particularly two problems i.e. quality of image they produce and the stability of their training process. In this literature review, I will discuss the following papers that have addressed these challenges. Generative Adversarial Networks for Image Generation [11], Improved Techniques for Training GANs [12], Training Generative Adversarial Networks with Limited Data [13], Comparative Analysis of Deep Convolutional Generative Adversarial Network and Conditional Generative Adversarial Network using Handwritten Digits [14], Auxiliary Conditional Generative Adversarial Networks for Image Data Set Augmentation [15].

In [11], the model's presented in the study were implemented using a TensorFlow implementation of DCGANs available in the public domain. The learning rate was set to 0.0002, except for the LSUN-scenes dataset, where it was adjusted to 0.001. A mini-batch size of 64 was used, and the model's variables were initialized from a Gaussian distribution with a mean of zero and a standard deviation of 0.02. Consistent with DCGANs, the β_1 value for the Adam optimizer was having a value 0.5. The pixel values of the images were scaled to the range of -1 to 1(closed interval) because the generator utilized the Tanh activation function. Both LSGANs and NS-GANs have similar architecture while working on the dataset of LSUN-bedroom.

The network architecture includes different types of layers: convolutional (CONV), transposed convolutional (TCONV), fully-connected (FC), batch normalization (BN), and leaky rectified linear unit (LReLU). Every layer is described with specific parameters, such as kernel size, stride, and the number of output filters. For example, (K3, S2, O256) refers to 3x3 dimension of kernel, with a stride of 2, and there are 256 number of output filters. The images in Fig. 2.1 represents the LSUN dataset. The generated by LSGANs and NS-GANs, are scaled down to a dimension of 112 pixels by 112 pixels. LSGANs produce images with more intricate texture details, particularly in areas like beds, resulting in sharper overall appearance compared to NS-GANs. Additionally, LSGANs were trained on four other scene datasets such as church, dining room, kitchen, and conference room.

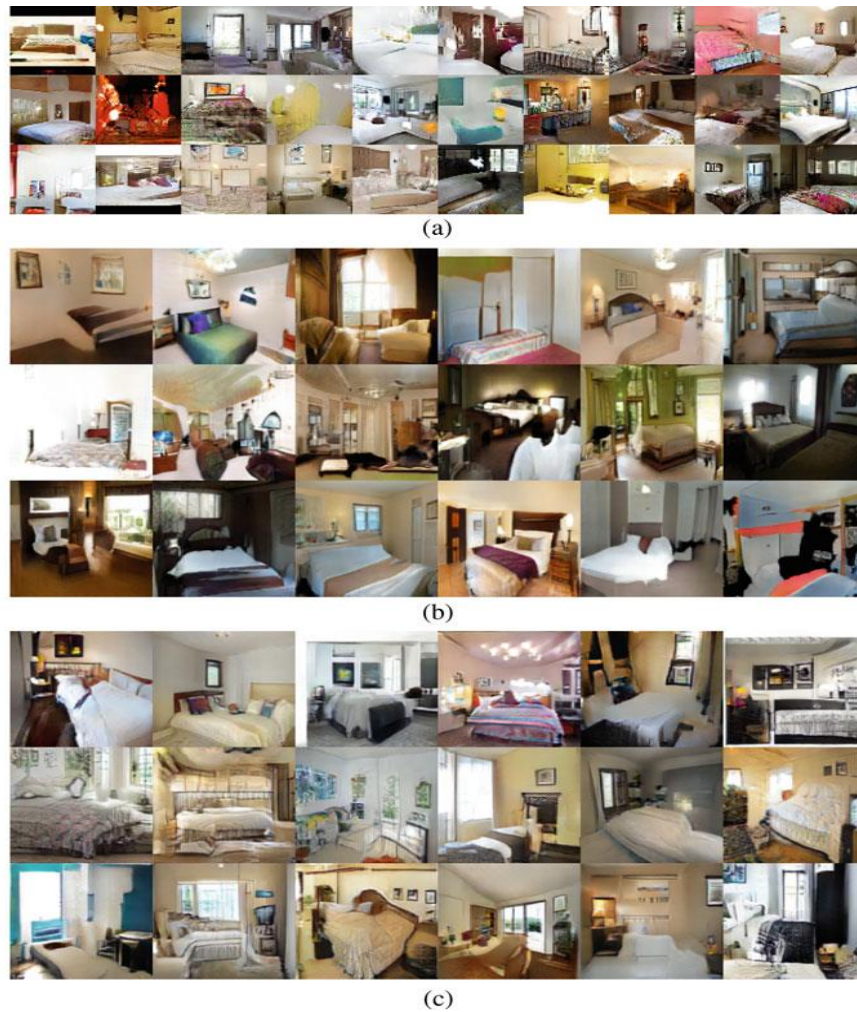


Fig. 2.1 LSUN-bedroom dataset [11]

In Fig. 2.1 The images include: NS-GANs generated images with a resolution of 64×64 , as mentioned in [8]. NS-GANs generated images with a resolution of 112×112 . LSGANs generated images with a resolution of 112×112 .

LSGANs were subjected to further evaluation using the cat dataset [16]. The initial step involved obtaining cat head images with resolutions exceeding 128×128 through pre-processing techniques sourced from a publicly available project. Subsequently, all images were resized.

To facilitate a performance comparison between LSGANs and NS-GANs, both models were trained using an identical architecture on the cat dataset. In between the process of training, checkpoints of the models and batches of generated images were periodically saved every 1000 iterations. The best models among LSGANs and NS-GANs were selected based on the image quality saved at each 1k iteration interval.

At last, the selected model was employed to randomly generate cat images. Fig. 2.2 visually presents the cat dataset, used by LSGANs model and NS-GANs model. Notably, LSGANs exhibited a notable advantage by generating cats with sharper hair in comparison to NS-GANs. To provide a closer examination of the cat eyebrow hair, Fig. 2.2 includes detailed views (parts (c), (d), and (e)) achieved by zooming specific regions of the used image. A summary of all the review done so far can be seen in Table 2.1.

Table 2.1 Summary of reviewed paper

| Paper | Technique Used | Dataset Used | Metrics Used |
|----------------------------------------------------------------|------------------------------------------------------------|---------------------------|----------------------------------------------|
| Restricted Boltzmann Machines for Collaborative Filtering [1] | Restricted Boltzmann Machines (RBMs) | Netflix dataset | Error rates comparison |
| Deep Boltzmann Machines [2] | Deep Boltzmann Machines (DBMs) | MNIST, OCR, NORB datasets | Classification performance, test error rates |
| Auto-Encoding Variational Bayes [3] | Variational Autoencoder (VAE) | Frey Face, MNIST | Log-likelihood estimates |
| Generative Image Models with LAPGAN [4] | Laplacian Pyramid Generative Adversarial Networks (LAPGAN) | CIFAR10, LSUN scenes | MOS scores, realism evaluation |
| Single Image Super-Resolution Using GAN [5] | Generative Adversarial Networks (GANs) | Set5, Set14 datasets | PSNR, SSIM, MOS scores |
| Image-to-Image Translation with cGANs [6] | Conditional GANs (cGANs) | Cityscapes dataset | Per-class accuracy, Class IOU |
| Extreme Image Compression with GANs [7] | Generative Adversarial Networks (GANs) | Custom image datasets | Bitrate savings, user study preferences |
| Deep Generative Stochastic Networks [8] | Deep Generative Stochastic Networks | Unspecified | Qualitative and quantitative evaluations |
| Conditional Image Synthesis with Auxiliary Classifier GANs [9] | Auxiliary Classifier GANs (AC-GANs) | MNIST, CIFAR10 datasets | Classification accuracy, visual fidelity |

| | | | |
|----------------------------------------------------------------|---------------------------------------------------------------------------------|-----------------------------------------------------------|--------------------------------------------------------------|
| Generative Adversarial Networks with Applications [10] | Generative Adversarial Networks (GANs) | MNIST, CIFAR-10, SVHN | Classification accuracy, FID scores |
| Advanced Image Generation with DCGANs [11] | Deep Convolutional Generative Adversarial Networks (DCGANs) | CelebA dataset | Visual fidelity, FID scores |
| Improved Techniques for Training GANs [12] | Generative Adversarial Networks (GANs) | MNIST, CIFAR-10, SVHN, ImageNet | Semi-supervised classification, visual Turing test |
| Training GANs with Limited Data [13] | Generative Adversarial Networks (GANs) with adaptive discriminator augmentation | Several datasets | Fréchet Inception Distance (FID) |
| Comparative Analysis of DCGAN and CGAN [14] | Deep Convolutional GANs, Conditional GANs | MNIST | Image quality, generative capabilities |
| Auxiliary Conditional GANs for Image Dataset Augmentation [15] | Auxiliary Conditional Generative Adversarial Networks (AC-GANs) | FMNIST | Accuracy of image classifiers, segmentation techniques |
| Cat Head Detection: Exploiting Shape and Texture [16] | Joint shape and texture detection, Haar-like features on oriented gradients | 10,000 well-labeled cat head images, PASCAL 2007 cat data | Detection rate |
| Multi-Object Detection with Neural Networks [17] | Convolutional Neural Networks (CNNs) | COCO dataset | Intersection over Union (IoU), detection accuracy |
| Improved Techniques for Training GANs [18] | Generative Adversarial Networks (GANs) with new training techniques | MNIST, CIFAR-10, SVHN, ImageNet | Semi-supervised classification accuracy, human error rate in |

| | | | |
|-------------------------------------------|----------------------------------------------------|----------------------------|--------------------------------------|
| | | | visual Turing test |
| Real-Time Object Recognition Systems [19] | Deep Learning, Feature Matching, Real-Time Systems | Custom real-world datasets | System latency, recognition accuracy |

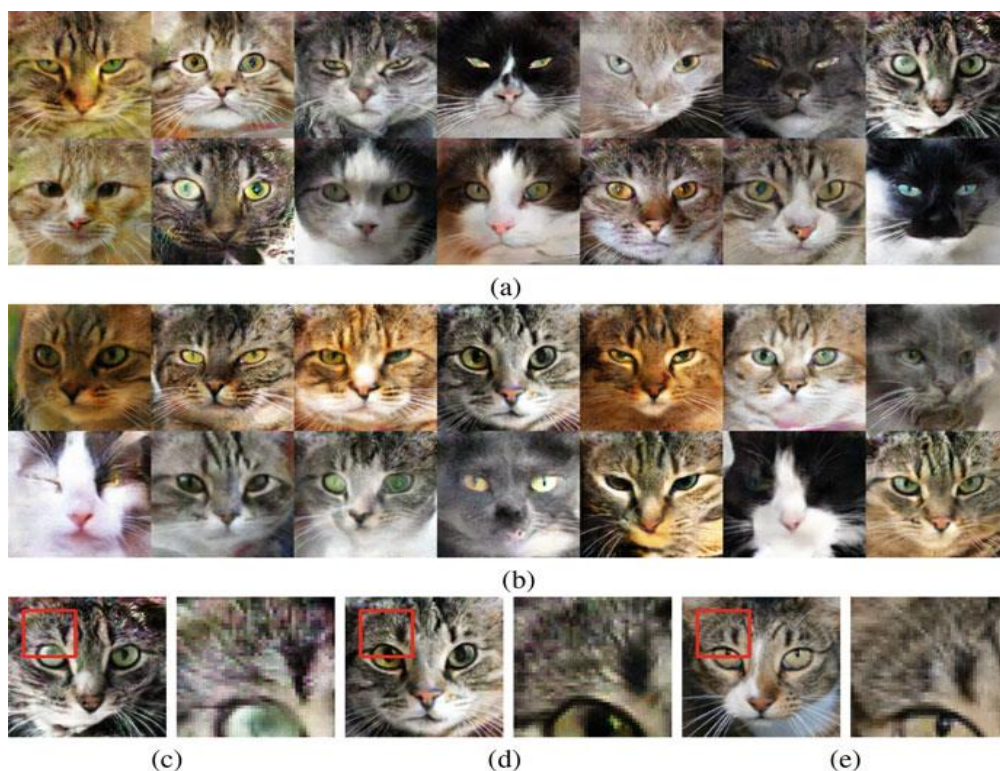


Fig. 2.2 Shows generated images of cats [7]

In Fig. 2.2 the a) part represents cat images used by NS-GANs. model part (b) by LSGANs. part (c) by NS-GANs. part (d) by LSGANs. (e) original images. LSGANs generate cats with more defined, detailed hair compared to the NS-GANs model, resulting in sharper and more exquisite features. Our observations indicate generated hair of cats from NS-GANs exhibits more noise as compared to the cat hair generated by LSGANs. Furthermore, we noticed that LSGANs produce images of higher overall quality compared to NS-GANs.

To quantitatively evaluate LSGANs, we measure the FID metric [17]. It quantifies the dissimilarity among the real original images and generated fake images by estimating feature dimensional space of the model named as inception model like a Gaussian distribution in multiple dimensions. FID has been demonstrated to align more closely with human perception compared to the inception score. A lower FID

value indicates a smaller distance between the generated images and the original images, implying higher similarity in terms of visual quality [18].

FID is utilized to assess the performance of WGANs-GP, NS-GANs, LSGANs and WGANs-GP on various dataset. The comparison includes the performance of LSGANs (-110) and LSGANs (011) using a loss function. The architecture was same for all models and also noise input dimension was same defined in [8], which consists of 4 Conv layer for generator as well as discriminator. In order to ensure a fair comparison, the official implementation of WGANs-GP is used for evaluation. The image dimension of LSUN is 64×64 , Cat is 128×128 , ImageNet is 64×64 , and CIFAR-10 is 32×32 . For each model, 50k images are randomly generated every 4k iterations, and the score of FID is calculated. The following conclusion was discovered: LSGANs (-110) outperforms NS-GANs across all 4 data. LSGANs (-110) also demonstrates superior performance compared to WGANs-GP on three datasets, particularly the dataset of cat images shown in Fig 2.2. Hence, LSGANs (-110) performs way finer than LSGANs (011) on all 4 datasets. LSGANs (011) shows comparable performance to NS-GANs.

LSGAN's and NS-GANs initially exhibit similar score of FID values for the first 25k iterations [17]. However, LSGANs shows improvement by reducing its FID values after 25k iterations and ultimately outperforms NS-GANs. LSGAN's and WGANs-GP eventually achieve comparable optimal FID values. Nonetheless, LSGANs achieve this optimal FID value much faster than WGANs-GP. Specifically, LSGANs requires 1100 minutes to reach an FID value of 22, whereas WGANs-GP takes 4600 minutes. In conclusion, LSGANs is able to achieve better performance than NS-GANs and WGANs-GP, and it is able to do so faster than WGANs-GP. This is because LSGANs does not require multiple updates for the discriminator, and hence for the gradient penalty it does not require extra computational time.

Semi-supervised were performer experiments on MNIST, CIFAR-10, where in [11][12] they performed unsupervised experiments. For the MNIST dataset, which comprises 60k labeled images depicting digits from 0 to 9, a semi-supervised training approach was adopted. In this approach, a small fraction of the dataset was randomly selected, containing 20, 50, 100, or 200 labeled examples. Their network has five hidden layers each. The generator produces samples that are not visually appealing shown on the left side of Fig. 2.3 but when using minibatch discrimination, it can improve the quality of image shown on the right side of Fig. 2.3.



Fig. 2.3 MNIST Dataset [11][12]

[15] This research investigates the use of auxiliary conditional generative adversarial networks (AC-GANs) to improve image datasets, with a particular emphasis on increasing image classifier accuracy through label conditioning and creating new images with global coherence using the FMNIST dataset.

The authors propose various novel architectural features and training approaches to improve GAN performance, particularly in semi-supervised learning and image production. Their algorithms produce cutting-edge results in semi-supervised classification on datasets such as MNIST and CIFAR-10 [18].

[16] This research proposes a method for recognizing cat heads that has the capability of efficiently using both the shape and texture data. The authors describe a two-step strategy that highly involves the use of training form and texture detectors separately so that they can be combined into a combination classifier for enhanced accuracy.

This study presents a two-time scale update rule (TTUR) [19] for training GANs that demonstrates convergence to a stationary local Nash equilibrium. The paper also presents the Fréchet Inception Distance (FID), which is a more consistent assessment metric for GANs.

CHAPTER 3

RESEARCH GAP

1. Efficiency in Limited Data Scenarios:

Current methodologies, such as Generative Adversarial Networks (GANs), demonstrate challenges in handling datasets with inadequate samples. Research could focus on developing models that maintain performance robustness even with limited data availability.

2. Integration of Shape and Texture Features:

While existing studies explore the integration of shape and texture for specific detection tasks (e.g., cat head detection), there is a broader potential for applying these combined features across various domains. Future research could investigate the generalization of these features for improved and optimized object recognition.

3. Optimization for Real-Time Processing:

Real-time object recognition systems demand further optimization to work effectively in dynamic environments where decision speed remains critical without relaxing accuracy.

4. Expanding Semi-Supervised Learning with GANs:

Even though it seems promising, the application of GANs in semi-supervised learning area needs further inspection to boost model stability and to comprehend the mechanics affecting their training dynamics.

5. Enhanced Robustness to Variations:

Despite advancements in handling intra-class variation and environmental conditions, there remains a substantial need for developing detection systems that are more resilient to changes in lighting, pose, and background.

6. Innovative Augmentation Techniques:

Data augmentation strategies that are specifically tailored to address the challenges of highly imbalanced datasets and rare object occurrences require development to better represent complex real-world scenarios.

7. Cross-modal Data Integration:

Integrating diverse data types (such as audio, text, or sensor data) with visual data to enrich the contextual understanding of object detection systems is under-explored and represents a significant opportunity for research.

8. Feature Extraction and Optimization:

There is potential for innovation in the discovery of new feature extraction techniques or the optimization of existing features tailored to specific

applications, which could lead to improved performance in object detection systems.

9. Addressing Ethical and Bias Considerations:

As object detection technologies find broader application, it is crucial to address potential biases and ensure that these systems operate fairly and equitably across diverse user groups.

These gaps underscore the necessity for continued research to overcome current limitations and to propel the field of computer vision towards more sophisticated and practical solutions. Addressing these challenges will not only enhance the theoretical understanding but also improve the practical applications of machine learning and object detection technologies.

CHAPTER 4

METHODOLOGY

4.1 Proposed Work

While DNN are commonly associated with supervised learning tasks like regression and classification, GAN's employ neural nets for distinct purposes such as generative modeling. Unlike supervised learning, generative modeling belongs to the domain of unsupervised learning in ML. It considers the exploration and acquisition of patterns and regularities within input data without explicit labels or guidance. By leveraging these learned patterns, GANs have the capability to generate novel examples that closely match with the real dataset, expanding the model's capacity beyond mere prediction tasks. It is known that there are various techniques of generative modeling, a Generative Adversarial Network (GAN's) uses the Fig. 4.1 system to generate images.

Table 4.1 Algorithm of proposed work

Algorithm 1 Training process of GANs

for number of training iterations **do**

- Sample a batch of real images x from training data.
- Sample a batch of noise vectors z from Gaussian distribution.
- Use z to generate a batch of fake images x^* from the generator.
- Update the discriminator using $\text{UPDATE_DISCRIMINATOR}(x, x^*)$.
- Update the generator using $\text{UPDATE_GENERATOR}(x^*)$.

end for

function $\text{UPDATE_DISCRIMINATOR}(x, x^*)$

- Compute the discriminator's prediction for x and x^* .
- Compute the classification error for x and x^* .
- Update the discriminator's parameters to minimize the classification error.

end function

function $\text{UPDATE_GENERATOR}(x^*)$

- Compute the discriminator's prediction for x^* .
- Compute the classification error for x^* .
- Update the generator's parameters to maximize the classification error of the discriminator.

end function

To evaluate the ability of discriminator to differentiate b/w real images and images that are generated, the binary cross entropy loss/objective function (as illustrated in Equation 4.1) is commonly employed. This loss function quantifies the performance of the discriminator as a binary classification model, measuring its effectiveness in accurately classifying the input images. By minimizing this objective function, the discriminator network is trained to improve its discriminatory capabilities and enhance its ability to differentiate between the two types of images.

$$Loss = -\frac{1}{output\ size} \sum_{i=1}^{output\ size} y_i \cdot \log \hat{y}_i + (1 - y_i) \cdot \log(1 - \hat{y}_i)$$

Equation 4.1 Binary cross entropy loss function

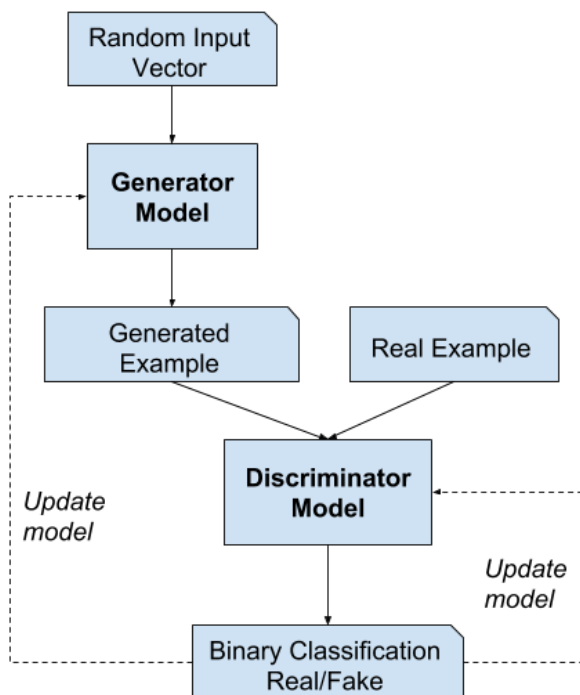


Fig. 4.1 Trainable GAN System

Within the GAN framework, two architectures of neural nets coexist: the generator and the discriminator in which generator's primary function is to generate synthetic samples, also known as "fake" samples, based on a random vector or matrix input. Discriminator is used to differentiate between real images and fake images. The process for training of these networks occurs concurrently, but in alternating epochs.

First, the discriminator undergoes training for a few epochs, followed by training of the generator for a few epochs. This iterative process is repeated to enhance the capabilities of both networks over time. The training of GANs is challenging and highly dependent on various factors such as hyperparameters, activation functions, and regularization techniques. They require careful tuning to achieve desirable results. In this tutorial, our goal is to be able to train generator and discriminator simultaneously that helps us to generate images of faces of some anime characters. To accomplish this, I will utilize the nature inspired genetic Algorithm technique which updates the parameters of both discriminator and generator in order to reach a global optimum. Genetic Algorithm has many benefits, one such is that it does not allow the parameters to get stuck on a local optimum for long generation/epochs.

Genetic Algorithms are based on natural evolution and genetics of living beings. This algorithm does not require gradient information hence it is computationally inexpensive and also takes less epochs to converge. It was also found during the experimentation that GA based GAN takes less training time, unlike in my previous work of GAN where it took me around 1 week to train for 25 epochs, the new proposed algorithm took almost only 2 days to train for 30 epochs. Fig. 4.2 shows the High-level system of GA-GAN used for this research work.

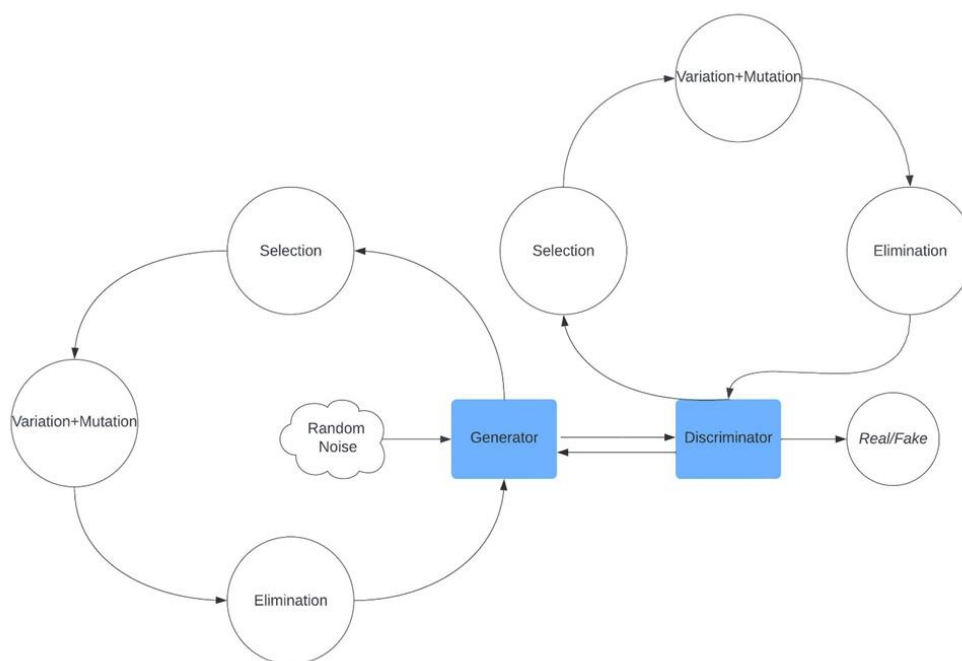


Fig. 4.2 GA-GAN high level system

The Dataset of Anime Faces, which comprises around 63k cropped out faces of anime characters, is illustrated in Fig. 4.3. It is crucial to emphasize that

generative modeling falls under the realm of unsupervised learning, meaning that the dataset does not come with any pre-existing labels or annotations.



Fig. 4.3 Shows the original set of images from Anime face dataset

4.1.1 Discriminator Network

The main work of discriminator is to receive a whole dataset of images as input and determine whether it is a real original image from the original set or a fake image produced by the generator. In essence, the discriminator operates like a standard neural network. In our implementation, we employ a CNN that produces only one numerical output for each input image. By utilizing a stride of 2, we gradually decrease the size of the derived feature map, following the process shown in Fig. 4.4.

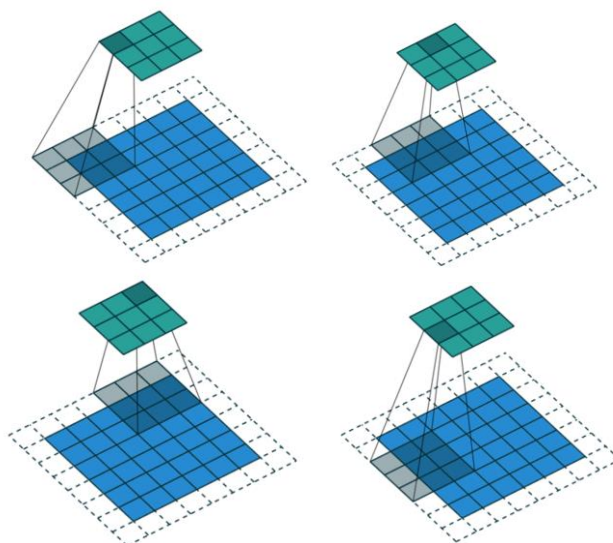


Fig. 4.4 The process of a convolutional operation to generate feature map.

The discriminator architecture created is a CNN having seven convolutional layers, followed by a flattening layer and also a sigmoid layer. The convolutional layers are responsible for extracting all the features from the image, while the layer for flattening converts the extracted matrix of features into a vector. The sigmoid layer then classifies the input image as genuine or fake. The discriminator takes an input image of size 64x64x3 and generates a value from 0 to 1 denoting a probability of being a real image or a fake image. Discriminator is trained to increase the chances of probability of the classification between real images as real and fake images as fake. The block diagram of the network architecture of the discriminator illustrated in Fig. 4.5.

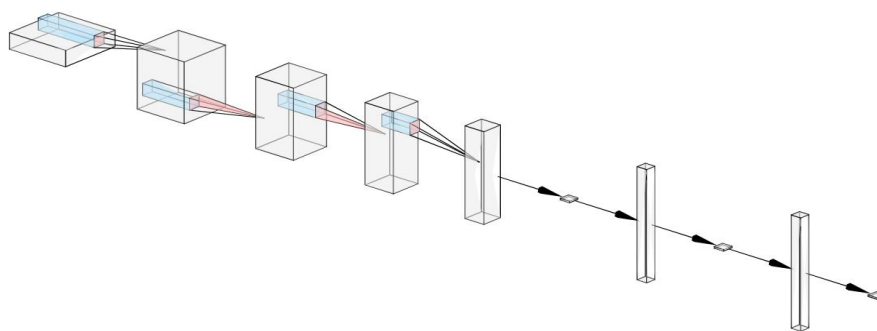


Fig. 4.5 Network architecture of the Discriminator

4.1.2 Generator Network

The generator is fed with a random matrix or vector of random numbers as input (referred to as a latent tensor/space). The generator then transforms this vector into an image. The generator transforms a latent space (or mathematically tensor) with dimensions $128 \times 1 \times 1$ to a tensor of image with dimensions $3(\text{channels}) \times 28 \times 28$. This conversion is achieved by employing the ConvTranspose2d (or transpose convolution layer) layer in PyTorch library, which performs an inverse convolution operation (also known as a transposed convolution or deconvolution).

The ConvTranspose2d layer is a convolutional layer that works in reverse. Instead of taking an input image and extracting features from it, the ConvTranspose2d layer is fed with an input vector that in response generates an output image. The ConvTranspose2d layer uses a transposed convolution kernel to do this.

The ConvTranspose2d layer is used to generate images from random vectors. The ConvTranspose2d layer takes a random vector and creates an output image as shown in Fig. 4.6. The output image is a realistic image that looks like it was generated by a human. The ConvTranspose2d layer is a powerful tool for generating images. It can be used to generate images of any size and any complexity.

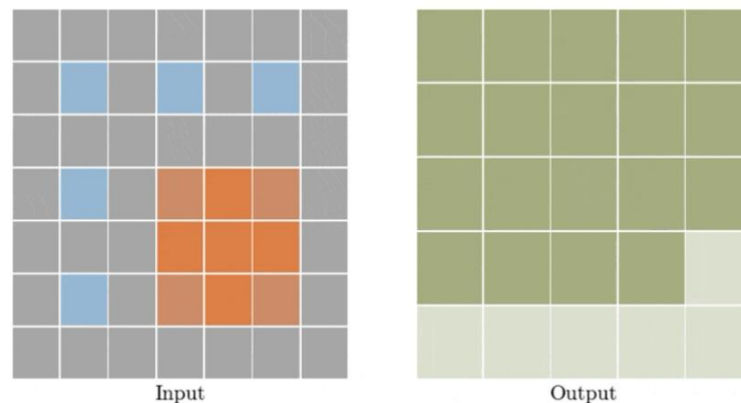


Fig. 4.6 Working of ConvTranspose2d layer

The generator that designed is a CNN with five transpose convolutional layers, with 4 layers of batch normalization layer just after each transpose convolutional layer, similarly 4 ReLU activation function is used just after each transpose convolutional layer, at last a activation function known as hyperbolic tangent Tanh function. The convolutional layers are responsible for upsampling the input vector into an image. The layer of batch normalization transformation helps us to stabilize the training process. The ReLU activation functions allow the generator to learn non-linear features from the input vector. The hyperbolic tangent activation function ensures that the generator' output is a real number between -1 and 1, which is the range of values for an RGB image.

1. The first transpose convolutional layer consists of 512 filters, 4 as kernel size, with a stride of 1. The filters are initialized randomly. The output of this layer is a tensor of size 512x4x4.
2. The second transpose convolutional layer consists of 256 filters, 4 as kernel size, with a stride of 2. The filters are initialized randomly. The output of this layer is a tensor of size 256x8x8.
3. The third transpose convolutional layer consists of 128 filters, 4 as kernel size, with stride of 2. The filters are initialized randomly. The output of this layer is a tensor of size 128x16x16.
4. The fourth transpose convolutional layer consists of 64 filters, 4 as kernel size, with a stride of 2. The filters are initialized randomly. The output of this layer is a tensor of size 64x32x32.
5. The fifth transpose convolutional layer consists of 3 filters, 4 as kernel size, with a stride of 2. The filters are initialized randomly. The output of this layer is a tensor of size 3x64x64 as shown in Fig. 4.7.

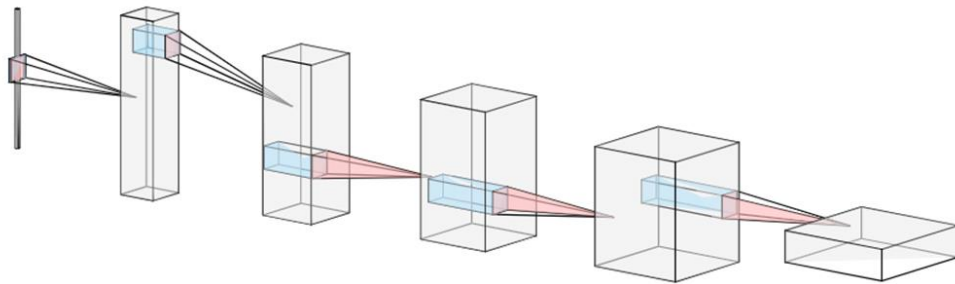


Fig. 4.7 Network architecture of the Generator

4.1.3 Genetic Architecture

A single Genetic algorithm is implemented for both discriminator as well as generator. A Fitness values in the genetic algorithm is calculated to perform survival of the fittest in order to eliminate the bad solution and more forward with the good solution. Genetic algorithm is a population-based technique hence we will denote a random but fixed initial population size. Crossover was performed then mutate function after creating a mating pool generated by selection layer. Hence the above algorithm will be in a loop until it finds optimal parameters for GAN models. The parameters are updated for one model at a time i.e. there is synchronous communication between models.

Now, we can generate our first set of images and visualize them by applying transformations and denormalization techniques, as depicted in Fig. 4.8. At this stage, generator has not undergone any training. Consequently, the generated outputs are expected to resemble random noise rather than meaningful images.

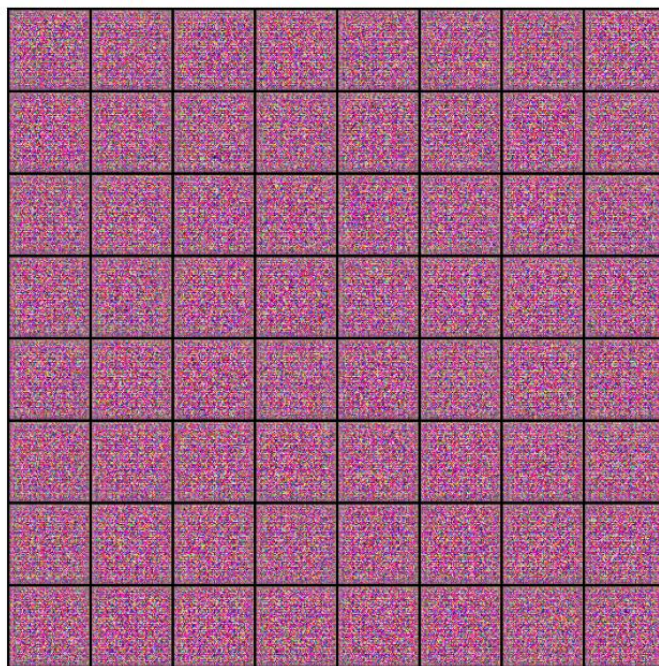


Fig. 4.8 Output of the generator at epoch 0

4.2 Training Process of Discriminator

The discriminator is trained to output 1 when presented with real images from the Anime Face Dataset and 0 when presented with images generated by network of generator. Initially, a mini batch of the original real set of images is fed into the discriminator, and the loss is calculated by setting the target labels to 1. This step helps the discriminator learn to correctly classify real images. Next, a mini batch of fake generated set of images that are generated by generator is passed through discriminator. The loss is computed by setting a value of 0 to the target class labels. The losses from both real and fake image batches are combined, and the overall loss is used to perform gradient descent. Updating of weights is done in this phase to improve its performance to discriminate between real and generated images.

4.3 Training Process of Generator

During the training of the generator, a unique approach is employed to optimize its performance. Since the generator produces image outputs, it is not immediately apparent how we can train it effectively. Here is an overview of how this process works:

1. First, a mini batch/set of images are generated from the generator. These generated images are then passed through the discriminator.

2. After that loss function is calculated by setting 1 to the target class labels for the generated images, that tells us that they are actual real images.
3. Gradient descent is applied using the calculated loss, resulting in updates to the weights of the generator. This process aims to enhance the generator's ability to produce images that closely resemble real images and deceive the discriminator.

This approach ensures that the generator learns to generate more authentic and realistic images by continuously improving its performance based on the feedback from the discriminator.

4.4 Training Process of Full Architecture

We will implement a complete training loop to simultaneously train the discriminator and generator using the fit function. Instead of a gradient based learning technique to update the parameters we will use GA. It was seen that the training time was very less as compared to previous works. it was seen also seen in [19] that almost 30 hours were taken on a single GPU to train LSUN dataset. Additionally, we will periodically save a selection of generated images during the training process for visual examination and analysis.

The training method begins by assigning random weights to the populations of both the discriminator and generator. The Genetic Algorithm will then evolve these populations using selection, crossover, and mutation processes. Each individual's fitness in the population is estimated using a bespoke fitness function that considers the discriminator's accuracy as well as the quality of the images reproduced by the generator.

To improve the stability of the training process, we will use a dual-objective method for the fitness function. The fitness function will assess the ability of discriminator to distinguish between genuine & produced images. The fitness function for the generator prioritizes creating images that are progressively alike from genuine images as determined by the discriminator.

Each generation, the top performers will be chosen to pass on their genes to the following generation. This selection procedure ensures that only the best-performing weights are carried forward, which improves the discriminator and generator's overall performance across future generations.

By using Genetic Algorithms for training, we hope to accomplish a more efficient and effective method, lowering overall training time while preserving or improving image quality. This method opens up new opportunities for alternate optimization strategies in GAN training, with generated images saved at regular intervals for qualitative evaluation and early issue discovery.

CHAPTER 5

EXPERIMENTAL SETUP

5.1 Tools Used

- Jupyter Notebook: It is used for a wide variety of tasks of data science, including exploratory data analysis (EDA), data wrangling (cleansing) and transformation, data visualization, predictive modelling, machine learning, and deep learning.
- Pandas: It is a Python toolkit for working with data collections. It includes functions for analysing, wrangling, cleansing, and modifying data. The word "Pandas" refers to both "Panel Data".
- Matplotlib: Matplotlib is a comprehensive Python package that permits you to create static and interactive visualizations. Matplotlib allows for both easy and difficult tasks. It helps us create plots that are suitable for visualisation. It helps us create reciprocal figures that can zoom, pan, and update.
- Seaborn: It is a Python package for plotting statistical graphs. It is built atop of matplotlib and combines seamlessly with Pandas data structures.
- Scikit-learn: It is one of the most helpful machine learning libraries in Python. The sklearn package includes several beneficial methods for machine learning and statistical modelling, like classification, regression, clustering, and dimensionality reduction.
- TensorFlow: It is an open-source library created by Google, mainly used for deep learning applications. It also reinforces conventional machine learning. It was primarily built for huge numerical computations without taking deep learning into consideration.

The Anime Face Dataset, often found on Kaggle, contains a comprehensive collection of images focused on anime-style faces. The dataset contains roughly 63,565 photos, making it an excellent resource for training and testing machine learning projects. All of the photographs depict anime-style faces, with a variety of expressions, haircuts, and character designs typical of anime and manga art genres. Images are often delivered in standard formats such as JPEG or PNG, allowing for simple integration into machine learning pipelines and frameworks.

CHAPTER 6

RESULTS AND ANALYSIS

6.1 Generation of images from generator

Below are some sample sets of images generated as outputs using the generator during different training epochs. We then tried to compare those sets of images (from a human standpoint) from the original set of images given in Fig. 4.3. At this stage, the output is essentially random noise since the generator has not undergone any training yet shown in Fig. 4.8. As a result, the generated images lack meaningful structure or resemblance to the target data. At epoch 1 the generated set of images from the generator can be seen in Fig. 6.1, similarly at epoch 16,25,30 the generated set of images from the generator can be seen in Fig. 6.2, Fig. 6.3 and Fig. 6.4 respectively.



Fig. 6.1 New generated images from generator at epoch 1



Fig. 6.2 New generated images from generator at epoch 16



Fig. 6.3 New generated images from generator at epoch 25



Fig. 6.4 New generated images from generator at epoch 30

6.2 Plot scores v/s epochs

After training both the model (generator and discriminator) simultaneously for 30 epochs we found the `real_score` as 0.9722 and `fake_score` as 0.0452 that can be observed in Fig. 6.5.

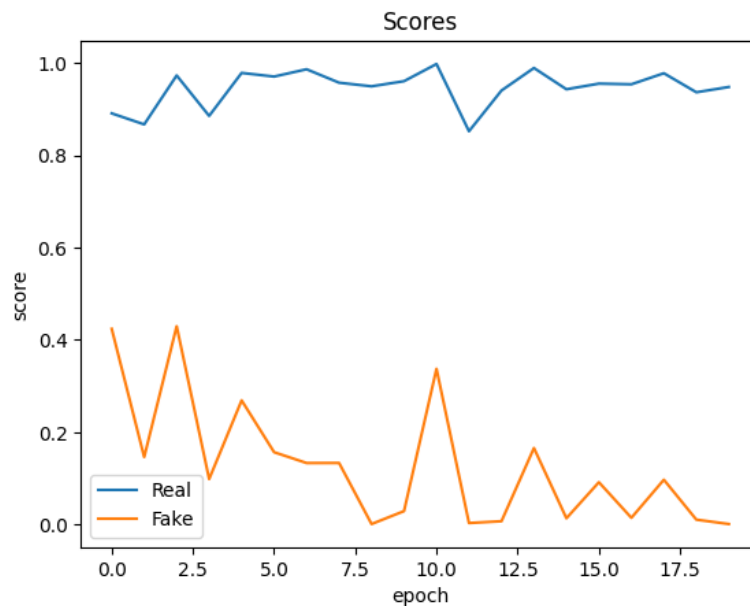


Fig. 6.5 Plot of real_score v/s epochs and fake_score v/s epochs

6.3 Plot loss v/s epochs

After training both the model (generator and discriminator) simultaneously for 30 epochs we found the results as loss_g: 5.9757, loss_d: 0.0781 that can be seen in Fig. 6.6.

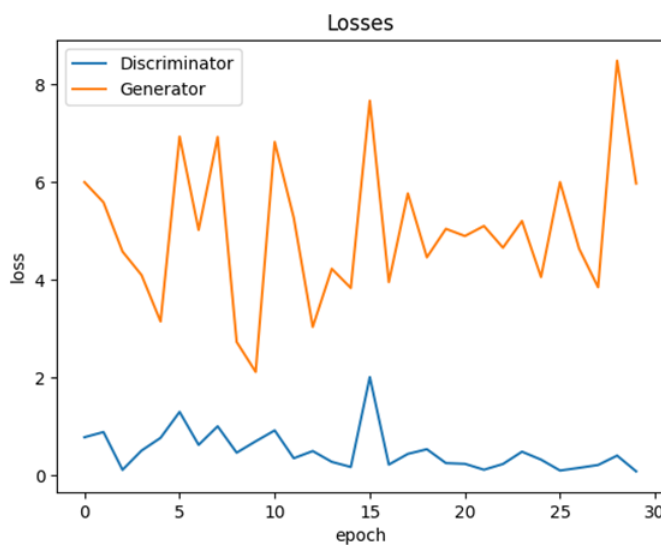


Fig. 6.6 Plot of generator loss(loss_g) v/s epochs and discriminator_loss(loss_d) v/s epochs

6.4 Performance Analysis

In a GAN, the discriminator evaluates images, assigning scores to genuine (`real_score`) and created (`fake`) images. Here's a condensed description of these metrics:

- `Real_score` is the discriminator's level of confidence in predicting real images. A high score (around 1) implies that the discriminator correctly recognizes actual data.
- `Fake_score` indicates the discriminator's confidence in classifying created (`fake`) images. A low score (around zero) indicates that the discriminator correctly detects fabricated data.

A good discriminator has a high `real_score` and a low `fake_score`. When training the generator, the goal is to mislead the discriminator by minimizing the `fake_score`. Effective training produces realistic generated images, as evidenced by a low `fake_score`. After 30 epochs, the `real_score` was 0.9722 whereas the `fake_score` was 0.0452. This implies that the discriminator properly identifies actual photos while recognizing fraudulent ones, indicating a good training procedure. To fully evaluate the GAN's performance, a final assessment should take into account visual quality and binary cross-entropy loss.

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

In this study, I explored various techniques such as data normalization, data augmentation, batch normalization, and the genetic algorithm to generate a set of images that closely resemble the original set of images. After 30 epochs of iteration, I achieved promising results with a `real_score` of 0.9722 and a `fake_score` of 0.0452. These results indicate a high similarity between the generated and real images, as a higher `real_score` and a lower `fake_score` contribute to a lower Frechet Inception Distance (FID) score. Furthermore, the fact that significant image quality improvement was achieved in just 30 epochs demonstrates the potential of our approach. As part of future work, I plan to perform more meta heuristic algorithm to make a hybrid model that further enhance the model's performance, aiming to achieve even higher `real_scores` and lower `fake_scores` compared to the existing implementation.

Additionally, due to resource limitations in terms of GPU compute units, I was unable to train the model on large datasets such as LSUN, Cat, and CIFAR100 datasets. Therefore, another future endeavor will involve implementing our own Hybrid GAN architecture on these datasets and comparing the results using FID as a metric. By customizing the architecture and employing the LSGAN objective/loss function (-110), we aim to develop a Hybrid GAN model that excels in generating high-quality images from limited image datasets.

Furthermore, future study will investigate the integration of sophisticated approaches such as attention mechanisms and progressive GAN growth to improve image quality and training efficiency. Implementing transfer learning to use pre-trained models on huge datasets may also improve model performance with fewer resources.

The findings of this study highlight the potential for further advancements in GANs and image generation techniques, providing valuable insights for future research in the field. This study lays the groundwork for more advanced generative models capable of producing highly realistic images efficiently.

REFERENCES

- [1] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. 2007. Restricted Boltzmann machines for collaborative filtering. In Proceedings of the 24th international conference on Machine learning (ICML '07). Association for Computing Machinery, New York, NY, USA, 791–798.
- [2] Salakhutdinov, Ruslan, and Hugo Larochelle. "Efficient learning of deep Boltzmann machines." Proceedings of the thirteenth international conference on artificial intelligence and statistics. JMLR Workshop and Conference Proceedings, 2010.
- [3] Kingma, Diederik P., and Max Welling. "Auto-encoding variational bayes." arXiv preprint arXiv:1312.6114 (2013).
- [4] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'14). MIT Press, Cambridge, MA, USA, 2672–2680.
- [5] Ledig, Christian, et al. "Photo-realistic single image super-resolution using a generative adversarial network." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
- [6] Kushwaha, R.S., Rakhra, M., Singh, D. and Singh, A., 2022, December. An overview: super-image resolution using generative adversarial network for image enhancement. In 2022 5th International Conference on Contemporary Computing and Informatics (IC3I) (pp. 1243-1246). IEEE.
- [7] P. Isola, J. -Y. Zhu, T. Zhou and A. A. Efros, "Image-to-Image Translation with Conditional Adversarial Networks," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 5967-5976, doi: 10.1109/CVPR.2017.632.
- [8] Agustsson, Eirikur, et al. "Extreme learned image compression with gans." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2018.
- [9] Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." arXiv preprint arXiv:1511.06434 (2015).
- [10] Denton, Emily L., Soumith Chintala, and Rob Fergus. "Deep generative image models using a [math] \sigma^2] laplacian pyramid of adversarial networks." Advances in neural information processing systems 28 (2015).
- [11] Metz, Luke, et al. "Unrolled generative adversarial networks." arXiv preprint arXiv:1611.02163 (2016).
- [12] Mao, Xudong, and Qing Li. Generative adversarial networks for image generation. Springer, 2021.

- [13] Salimans, Tim, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. "Improved techniques for training gans." *Advances in neural information processing systems* 29 (2016).
- [14] Karras, Tero, et al. "Training generative adversarial networks with limited data." *Advances in neural information processing systems* 33 (2020): 12104-12114.
- [15] Vishwakarma, Dinesh Kumar. "Comparative analysis of deep convolutional generative adversarial network and conditional generative adversarial network using hand written digits." *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*. IEEE, 2020.
- [16] Mudavathu, Kalpana Devi Bai, MVP Chandra Sekhara Rao, and K. V. Ramana. "Auxiliary conditional generative adversarial networks for image data set augmentation." *2018 3rd International Conference on Inventive Computation Technologies (ICICT)*. IEEE, 2018.
- [17] Zhang, W., Sun, J., Tang, X. (2008). Cat Head Detection - How to Effectively Exploit Shape and Texture Features. In: Forsyth, D., Torr, P., Zisserman, A. (eds) *Computer Vision – ECCV 2008*. *ECCV 2008. Lecture Notes in Computer Science*, vol 5305. Springer, Berlin, Heidelberg.
- [18] Heusel, Martin, et al. "Gans trained by a two time-scale update rule converge to a local nash equilibrium." *Advances in neural information processing systems* 30 (2017).
- [19] Salimans, Tim, et al. "Improved techniques for training gans." *Advances in neural information processing systems* 29 (2016).
- [20] Wang, Chaoyue, Chang Xu, Xin Yao, and Dacheng Tao. "Evolutionary generative adversarial networks." *IEEE Transactions on Evolutionary Computation* 23, no. 6 (2019): 921-934.
- [21] Brock, A., Donahue, J. and Simonyan, K., 2018. Large scale GAN training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*.
- [22] Dhawan, S. and Kumar, S., 2020, November. Improving resolution of images using Generative Adversarial Networks. In *2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA)* (pp. 880-887). IEEE.
- [23] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y., 2014. Generative adversarial nets. *Advances in neural information processing systems*, 27.
- [24] Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z. and Shi, W., 2017. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4681-4690).
- [25] Li, Y., Sixou, B. and Peyrin, F., 2021. A review of the deep learning methods for medical images super resolution problems. *Irbm*, 42(2), pp.120-133.

- [26] Shim, S., Kim, J., Lee, S.W. and Cho, G.C., 2022. Road damage detection using super-resolution and semi-supervised learning with generative adversarial network. *Automation in construction*, 135, p.104139.
- [27] Liu, Q.M., Jia, R.S., Liu, Y.B., Sun, H.B., Yu, J.Z. and Sun, H.M., 2021. Infrared image super-resolution reconstruction by using generative adversarial network with an attention mechanism. *Applied Intelligence*, 51, pp.2018-2030.
- [28] Zhang, K., Hu, H., Philbrick, K., Conte, G.M., Sobek, J.D., Rouzrokh, P. and Erickson, B.J., 2022. SOUP-GAN: Super-resolution MRI using generative adversarial networks. *Tomography*, 8(2), pp.905-919.
- [29] Yun, J.U., Jo, B. and Park, I.K., 2020. Joint face super-resolution and deblurring using generative adversarial network. *IEEE Access*, 8, pp.159661-159671.
- [30] Dong, C., Loy, C.C., He, K. and Tang, X., 2015. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2), pp.295-307.
- [31] Lim, B., Son, S., Kim, H., Nah, S. and Mu Lee, K., 2017. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops* (pp. 136-144).