Chapter 1

Introduction

In recent years, Generative Adversarial Networks (GANs) have emerged as a powerful tool for generating realistic data. GANs were introduced as a type of generative model based on game theory in which two networks compete in a min-max game to mimic a data distribution using Artificial Neural Networks (ANNs). Both networks must optimize their corresponding objective functions, generating a situation where two players compete for opposites objectives.

GANs have revolutionized the field of AI-generated data and led to an exponential increase in related research such as language generation, image generation, image-to-image translation, image generation in text description, video generation, and other domains achieving state-of-the-art results. GANs are particularly effective in computer vision due to their ability to generate sharper results by replicating a data distribution.

However, GANs suffer from instability during training, leading to issues such as mode collapse. To address these problems, researchers have developed a variety of GAN architectures that are adapted to specific applications. This survey aims to provide an overview of the latest GAN architectures and their applications, as well as how they address the challenges of GAN training. The study will also summarize commonly used metrics for measuring GAN performance and propose a classification of GANs based on their applications.



Figure 1.1: Basic Architecture of GANs.

Chapter 2

Basic Concepts/Literature Review

2.1 – Generative Adversarial Networks

Generative Adversarial Networks (GANs) consists of two models:

* A discriminator D estimates the probability of a given sample coming from the real dataset to be true or false. It works as a critic and is optimized to tell the fake samples from the real ones.
* A generator G outputs synthetic samples given a noise variable input z (z brings in potential output diversity). It is trained to capture the real data distribution so that it can create synthetic samples can be as real as possible, or in other words, can trick the discriminator to offer a high probability.

During the training process, these two models compete against each other. The generator G tries to fool the discriminator, while the discriminator D tries to accurately identify real samples. This creates a zero-sum game where both models are motivated to improve their functionalities. The loss function of this game is formulated as a minimization of the Generator and maximization of the Discriminator's loss.

On one hand, we want to make sure the discriminator D’s decisions over real data are accurate by maximizing . Meanwhile, given a fake sample , the discriminator is expected to output a probability , close to zero by maximizing .

On the other hand, the generator is trained to increase the chances of D producing a high probability for a fake example, thus to minimize . When combining both aspects together, D and G are playing a minimax game in which we should optimize the following loss function:



( has no impact on during gradient descent updates.)

* : Data distribution over noise input
* : Generator’s distribution over data
* : Data distribution over real sample

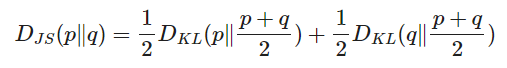
2.2 – Training Problems

Although GAN has shown great success in the realistic image generation, the training is not easy. The process is known to be slow and unstable.

* **Hard to achieve Nash Equilibrium:** The issue with GAN's training process based on gradient descent is that it involves two models learning simultaneously to find a Nash equilibrium in a two-player game. However, each model updates its cost function independently without taking into account the other player's behavior. Updating the gradients of both models concurrently does not guarantee convergence to a stable equilibrium.
* **Mode Collapse:** GANs can sometimes fail to generate diverse and realistic data during training, which is referred to as mode collapse. This means that the generator may produce the same outputs for different inputs, resulting in a limited variety of synthesized data. This problem can occur in a complete form, where all outputs are identical, or in a partial form, where a large number of outputs share similar characteristics.
* **Vanishing Gradient:** This can occur during GAN training when the discriminator becomes too accurate i.e or too inaccurate i.e . This makes the loss function falls to zero and and with no gradient, we cannot update the loss. If the discriminator is too accurate, it can easily differentiate between real and synthesized data, resulting in a loss function that approaches zero and providing little feedback to the generator. This leads to gradients that are close to zero, which slows down the learning process or even causes it to jam. On the other hand, if the discriminator is too inaccurate, it cannot differentiate between real and synthesized data, which also provides the generator with useless information and causes the loss function to fall to zero. In either case, the GAN training faces a dilemma where it becomes very tough to achieve balanced learning between the generator and discriminator.
* **Instability and Stopping Problem:** The objective of both the generator and discriminator networks is opposite, and even small changes in one network can lead to significant changes in the other. This can cause periods of instability during training, which tend to generate further instability, making the problem last longer. Many GAN architectures focus on stabilizing their training to improve their performance. Additionally, GANs do not follow a monotonically decreasing loss function, making it difficult to know when the models have been fully optimized.

2.3 – Jensen-Shannon Divergence & Wasserstein Distance

Jensen-Shannon Divergence (JS divergence) is a measure of similarity between two probability distributions, which ranges from 0 to 1. It is a symmetric and smoother alternative to the Kullback-Leibler Divergence. The use of JS divergence in GANs has contributed to their success, as it replaced the asymmetric KL divergence used in the traditional maximum-likelihood approach.



Wasserstein Distance, also known as Earth Mover's Distance (EM distance), is a measure of the distance between two probability distributions. It can be interpreted as the minimum energy cost required to transform a pile of dirt in the shape of one distribution to the shape of the other, where the cost is determined by the amount of dirt moved and the distance it is moved.

2.4 – Evaluation Metrics

Evaluation metrics are used to measure the quality and performance of GAN models. However, due to the unique nature of GANs, there is no universal metric that can be used to evaluate all GAN applications. Instead, researchers have developed different metrics, each with its own strengths and weaknesses, to measure different aspects of GAN performance. The most widely used evaluation metrics include the Inception Score (IS) and its variants, Multi-scale Structural Similarity for Image Quality (MS-SSIM), and Classifier Two-sample Test (C2ST).

The IS measures the quality and diversity of the generated samples of a GAN by using a pretrained neural network classifier called Inception v3. It calculates the probabilities of the generated samples and assumes that low entropy and high quality data are correlated. However, it cannot handle mode collapse, where all generated samples by the GAN will be practically the same.

The MS-SSIM provides a metric that compares the similarity between the real and synthesized dataset based on the geometry and structure of the image. It is commonly used with IS or its variations to provide a wider view of the generated data quality.

C2ST measures the distance between the synthesized and real data by using a binary classifier that divides the samples into synthesized and real ones, judging whether different samples belong to the same data distribution. The 1-Nearest Neighbor classifier (1-NN) and Neural Networks can be used as C2ST.

Overall, researchers use a combination of these metrics to evaluate GAN performance and to have a wider view of the model's strengths and weaknesses.

2.5 – Improved Approaches to GAN Training

The following suggestions are recommended to improve the training of Generative Adversarial Networks (GANs). The first five techniques aim to achieve faster convergence of GAN training and are proposed in a paper titled "Improve Techniques for Training GANs". The last two methods are proposed in another paper titled "Towards principled methods for training generative adversarial networks" and address the problem of disjoint distributions.

* **Feature Matching**, involves optimizing the discriminator to compare the generator's output with the expected statistics of the real samples. This is done by defining a new loss function that calculates the difference between the two sets of statistics.
* **Minibatch Discrimination**, allows the discriminator to consider the relationship between data points in one batch instead of processing each point independently. This is done by approximating the closeness between every pair of samples in a minibatch and summarizing the overall relationship of each data point.
* **Historical Averaging**, penalizes the training speed when the model parameters change too rapidly over time. This is done by adding a term to the loss function that considers the past configuration of the model parameters.
* **One-Sided Label Smoothing**, replaces the binary labels (1 and 0) used to train the discriminator with softened values such as 0.9 and 0.1. This is shown to reduce the network's vulnerability.
* **Virtual Batch Normalization (VBN)**, normalizes each data sample based on a fixed reference batch instead of within its minibatch. This reference batch is chosen once at the beginning of training and remains the same throughout.
* **Adding Continuous Noises** to the inputs of the discriminator to artificially spread out the distribution and increase the chances of overlapping between the probability distributions.
* **Using the Wasserstein metric instead of the Jensen-Shannon divergence** to measure the similarity between probability distributions. The Wasserstein metric has a smoother value space and provides a more meaningful value when two distributions are disjoint.

Chapter 3

Problem Statement / Requirement Specifications

In this section, we discuss the problem statement and the requirements for the proposed system.

3.1 – Problem Statement

The problem addressed in this survey is the challenge of keeping up with the constantly evolving field of Generative Adversarial Networks (GANs) in deep learning, which have had a significant impact on society. With new research and advancements in GANs being published frequently, it is difficult for researchers to stay up-to-date with the latest architectures, optimization techniques, validation metrics, and application areas of GANs. The objective of this survey is to provide a comprehensive overview of GANs, evaluate the efficiency of different variants of the model architecture, and analyse the metrics and loss functions commonly used for evaluating GAN performance. The survey ultimately aims to guide future researchers in the field by summarizing the evolution and performance of the most promising GAN variants.

3.2 – Project Planning

* We studied various literature on **Generative Adversarial Networks** (GANs)
* We studied various problems based on GAN trainings and we observed problems like **MOD collapse**, **slow convergence rate** of discriminator and generator networks which lead to oscillation or divergence from the min-max point.
* We further studied about **Nash equilibrium** and **proximal convergence** of GANs for finding stable min-max points.
* We learnt about the mathematics leading to **Gradient Descent Ascent** and about the zero-sum game optimizations in non-convex contours of the loss functions.
* We further study about various evaluation metrics and various improved architectures.
* We summarise the performance of various GAN variants which aims future researchers in this field.

3.3 – Project Analysis

3.3.1 – Feasibility Analysis

Before proceeding with the project, a feasibility analysis was be conducted to assess if the project is feasible from technical, financial, and operational perspectives. The required technology and expertise are available to develop and implement the study on GANs and it does not outweigh the budget. It is compatible with existing tools and technologies used by the target audience, as well as the willingness of the target audience to participate in the survey.

3.3.2 – Risk Analysis

Risk analysis involves identifying potential risks that could negatively impact the project and developing a plan to mitigate those risks. Here are some potential risks for the project of conducting a survey on Generative Adversarial Networks (GANs):

* Research Complexity: The research on GANs is constantly evolving and expanding, making it difficult to keep up with the latest developments. This could result in incomplete information being presented in the survey.
* Technical Issues: The development of the survey platform and data analysis tools may encounter technical issues such as software bugs or compatibility issues, which could delay the project and impact the accuracy of the results.
* Availability of Resources: The project may face challenges in terms of access to funding, research materials, and expert knowledge, which could impact the quality and accuracy of the survey results.

To mitigate these risks, the project team implemented measures such as conducting frequent checks on the accuracy and completeness of the research, implementing rigorous testing and quality assurance procedures for the survey platform and data analysis tools, ensuring adequate resources are available for the project's success.

3.3.3 – Scope Analysis

The scope of the problem statement is to provide a comprehensive study on the latest architectures, optimization of loss functions, validation metrics, and application areas of Generative Adversarial Networks (GANs). The survey aims to evaluate the efficiency of different variants of the GAN model architecture and their best application areas. Additionally, the survey aims to analyse the different metrics for evaluating the performance of GANs and frequently used loss functions. The final objective of the survey is to provide a summary of the evolution and performance of GANs to guide future researchers in the field. The scope of the study does not include the development or implementation of a specific GAN system or model.

3.4 – System Design Constraints

3.4.1 – Software and Hardware Specifications

We utilized TensorFlow and Keras, which are popular machine learning libraries, to create and train neural networks. For cloud-based computation, we chose Google Colab as our platform, and imageio and matplotlib libraries were used for visualizing the data by creating graphs and gifs or videos. The implementation of the code was done on Google Colab on a Chrome Browser using the T4 GPU, which has 16 GB of GPU memory, 320 Turing Tensor Cores, and 2,560 CUDA cores. Its boost clock frequency can go up to 1590 MHz, and it supports CUDA compute capability 7.5.

3.4.2 – Dataset Used

MNIST is a commonly used dataset in the field of machine learning and computer vision. It consists of 70,000 grayscale images of handwritten digits from 0 to 9, each of size 28x28 pixels. The dataset is often used for image recognition tasks, particularly for training and evaluating models that can identify handwritten digits. MNIST has been widely used as a benchmark dataset for evaluating the performance of machine learning algorithms and is considered a standard dataset in the field.

3.4.3 – Experimental Setup

The experiment setup for this project involved utilizing the MNIST dataset to train and evaluate various GAN models. We used TensorFlow and Keras libraries for creating and training the models, and Google Colab as our platform for cloud-based computation. Our goal was to compare the performance of several different GAN variants, including DCGAN, WGAN, and CGAN, in terms of image generation quality and training efficiency.

Our experiment involved training each GAN variant with a fixed number of epochs and evaluating the generated images using various metrics such as Inception Score and Fréchet Inception Distance. We also experimented with different hyperparameters and loss functions to optimize the performance of the models.

Chapter 4

Implementation

4.1 – Model Architectures

4.1.1 - Deep Convolutional GAN (DCGAN)

The Deep Convolutional GAN (DCGAN) was proposed in 2015 as an improvement to the original GAN architecture proposed in 2014. DCGAN uses convolutional layers instead of fully connected layers, which have been traditionally used for computer vision tasks. Convolutional layers help extract main features from images and matrices of data while preserving the correlation between adjacent pixels. DCGAN also introduces other changes, such as replacing pooling layers with strided convolutions, using batch normalization layers, and using different activation functions for the hidden and output layers. These changes improve the stability and performance of GANs. The DCGAN paper also presents a technique to visualize the filters learned by the models, which aids in the understanding of GANs' learning methods. The DCGAN architecture has become a standard in GANs design and training, and its innovations are applied in most of the following GAN models.

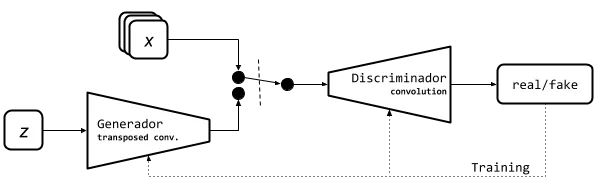


Figure 4.1: DCGAN Architecture

Loss Function:



Train Summary:

* Rescale the MNIST images between -1 and 1.
* Use transposed convolution in Generator with ReLU activation and BatchNormalisation. Tanh is the last activation.
* The Discriminator includes CNN and LeakyRELU action with last being Sigmoid.
* We use Binary Crossentropy Loss Function and Adam with lr = 0.0002 and beta\_1 = 0.5.

4.1.2 - Conditional GAN (CGAN)

In 2014, the CGAN model was introduced, which adds a new latent class label c to the latent space. This label helps in classifying the processed data into different classes, leading to the generation of synthesized data based on the input label's class. This approach is useful in problems that require data generation for various classes and has shown to prevent mode collapse effectively. However, its application to some problems is complicated due to the requirement of labeled datasets for training. Despite its simplicity, the CGAN model has had a significant impact on GAN models, and many variations have been developed since its introduction.



Figure 4.2: CGAN Architecture

Loss Function:

Cost function CGANs by fernanda rodríguez.

Train Summary:

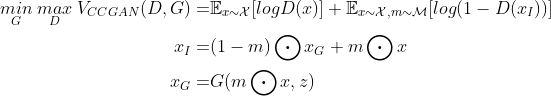
* Rescale the MNIST images between -1 and 1.
* Use transposed convolution in Generator with ReLU activation and BatchNormalisation. Tanh is the last activation.
* The Discriminator includes CNN and LeakyRELU action with last being Sigmoid.
* We use Binary Crossentropy Loss Function and Adam with lr = 0.0002 and beta\_1 = 0.5.

4.1.3 – Context-Conditional GAN (CCGAN)

Context-Conditional Generative Adversarial Networks (CC-GANs) are a type of conditional GANs that focus on a different task compared to traditional GANs. CC-GANs aim to determine if a specific part of an image is real or fake by utilizing the surrounding context. To achieve this, the Generator 𝐺 is trained to fill in a missing image patch while being conditioned on the surrounding pixels. The Generator receives an image with a randomly masked out patch as input and generates an entire image as output. The completed image is then passed into the Discriminator 𝐷 to determine whether the masked-out patch is real or fake. CC-GANs have a unique ability to fill in missing image patches and detect fake patches within a specific context. 

Figure 4.3: CCGAN Architecture

Loss Function:



Train Summary:

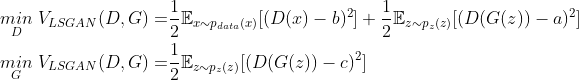
* Rescale the MNIST images between -1 and 1. Resize 32x32
* Generator is a U-net network. The input is the normal distribution z and the last activation is tanh.
* Discriminator is a CNN and have LeakyReLU activation. The last activation is a softmax.
* We use Categorical CrossEntropy and MSE Discriminator loss and the adversarial loss is MSE.
* We use Adam as optimiser with lr = 0.0002 and beta\_1 = 0.5

4.1.4 – Least Square GAN (LSGAN)

Least Squares Generative Adversarial Networks (LSGANs) is a type of generative adversarial network (GAN) that uses the least squares loss function instead of the binary cross-entropy loss function commonly used in traditional GANs. LSGANs aim to improve the stability and quality of generated samples by adopting a different approach to the loss function used in the Discriminator. 

Figure 4.4: LSGAN Architecture

Loss Function:



Train Summary:

* Rescale the MNIST images between -1 and 1.
* Generator is a simple fully connected neural network with Leaky ReLU activation and BatchNormalisation. The last activation is tanh.
* Discriminator is a simple fully connected neural network with Leaky ReLU activation. The last activation is a sigmoid.
* We use MSE as loss and Adam as optimiser with lr=0.0002 and beta\_1=0.5.

4.1.5 – Wasserstein GAN (WGAN)

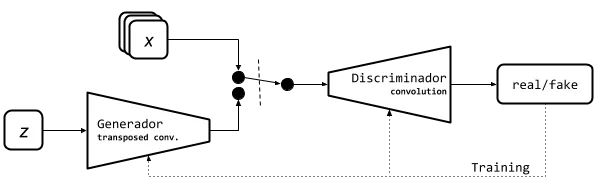
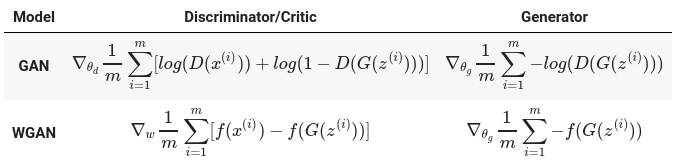
The WGAN architecture is based on the Earth Mover distance, also known as the Wasserstein-1 distance, which measures the cost of transforming one distribution into another. In the case of GANs, the Wasserstein distance measures the difference between real and synthesized data distributions. To apply this new objective function, changes are made to the GAN architecture. The discriminator becomes the "critic," which measures the realness of an image. Weight clipping is used to keep the parameters in a compact space. WGAN has better convergence, stability, and mode collapse avoidance than traditional GANs, especially in low-dimensional manifold distributions. Additionally, the WGAN loss function correlates with the quality of synthesized samples and converges to a minimum. 

Figure 4.5: WGAN Architecture

Loss Function:



Train Summary:

* Rescale the MNIST images between -1 and 1.
* Generator uses transposed convolution with ReLU activation and BatchNormalisation. The last activation is tanh.
* Discriminator is a CNN with Leaky ReLU activation. There is no last activation function.
* We use Wasserstein loss as loss and RMSprop as optimiser with lr=0.00005.

4.2 – Observations (Loss per Epoch for first 100 epochs)

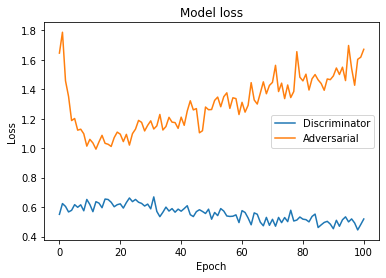


Figure 4.6: Original GAN on MNIST

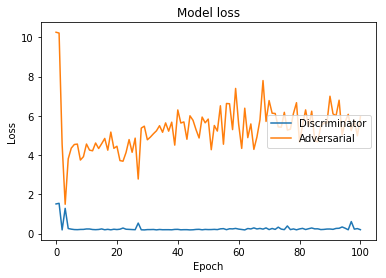


Figure 4.7: DCGAN on MNIST

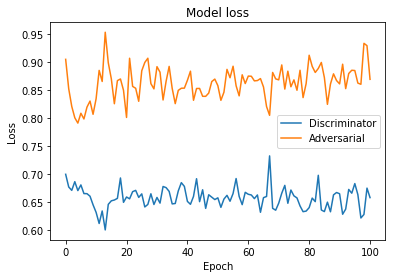


Figure 4.8: CGAN on MNIST

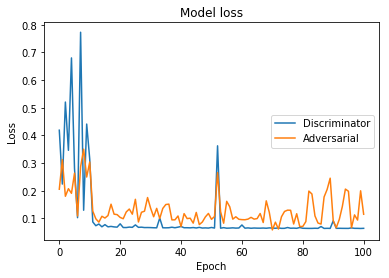


Figure 4.9: CCGAN on MNIST

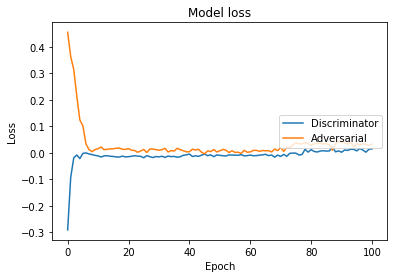


Figure 4.10: WGAN on MNIST

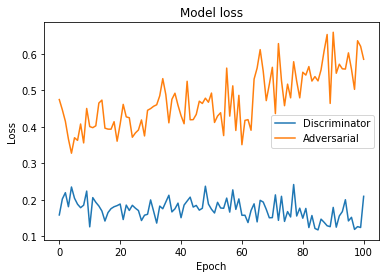


Figure 4.11: LSGAN on MNIST

4.3 – Result Analysis

Generative Adversarial Networks - GANs

|  |  |
| --- | --- |
| Epoch 00 | Epoch 100 |
| GAN with MNIST | GAN with MNIST |

Deep Convolutional Generative Adversarial Networks - DCGANs

|  |  |
| --- | --- |
| Epoch 00 | Epoch 100 |
| GAN with MNIST | GAN with MNIST |

Conditional Generative Adversarial Nets - CGANs

|  |  |
| --- | --- |
| Epoch 00 | Epoch 100 |
| CGAN with MNIST |  |

Context-Conditional Generative Adversarial Networks - CCGANs

|  |  |
| --- | --- |
| Epoch 00 | Epoch 100 |
| CGAN with MNIST | WGAN with MNIST |

Wasserstein Generative Adversarial Networks - WGANs

|  |  |
| --- | --- |
| Epoch 00 | Epoch 100 |
| WGAN with MNIST | WGAN with MNIST |

Least Squares General Adversarial Networks - LSGANs

|  |  |
| --- | --- |
| Epoch 00 | Epoch 100 |
| LSGAN with MNIST | LSGAN with MNIST |

4.4 – Quality Assurance

Our project has successfully met the necessary quality standards by adhering to industry best practices and guidelines. We have utilized appropriate methodologies, techniques, and tools to maintain quality throughout the project development lifecycle. Regular testing and validation have been conducted to verify that the system complies with the required specifications and delivers the anticipated results. Our team has also followed quality assurance processes to identify and address any defects or issues in the system. Additionally, we have kept the project documentation up-to-date and precise, reflecting the system's design and functionality. Proper communication and collaboration with stakeholders have been maintained to ensure on-time delivery and satisfaction with the final product.

Chapter 5

Standards Adopted

5.1 – Design Standards

The project's design adhered to the principles of modularity and extensibility. The system was structured in a layered architecture, with each layer fulfilling a specific function. The layers were devised with flexibility in mind, making it straightforward to modify or enhance them in the future.

5.2 – Coding Standards

The code was well-documented and follow standard coding practices. The code was written in a way that is easy to read and understand. The code followed a consistent coding style.

* The code was written as short as possible with industry-standard refactoring.
* Usage of appropriate naming conventions was followed.
* Code was properly indented with 4 spaces.
* A single function has carried out a single specific task

5.3 – Testing Standards

The system was tested thoroughly before deployment. The system is tested on a large dataset and should be able to handle different types of input images. The system should also be tested for different image processing tasks such as image super-resolution, style transfer, and image synthesis.

Chapter 6

Conclusion and Future Scope

6.1 – Conclusion

This report provides an overview of recent advancements in GANs, from their fundamental principles to the latest innovative architectures. The report categorizes the various problems that GANs can encounter and explains the most commonly used evaluation metrics. The report proposes a taxonomy for GAN variants, dividing them into two groups based on whether they focus on architecture optimization or objective function optimization. It is emphasized that these groups are not mutually exclusive and that each variant builds upon the progress of previous research. The report also summarizes the various applications of GANs in recent years, which have been influenced by the development of the field and its impact on society and industry. Finally, a comparison of the performance of different GAN architectures is presented to provide a quantitative view of their evolution.

6.2 – Future Scope

The future scope of GANs is vast and promising. One potential area of research is the development of more efficient and effective training algorithms for GANs. This could involve exploring different optimization techniques, regularization methods, or training strategies to improve the stability and convergence of GANs.

Another area of research is the application of GANs in new domains, such as natural language processing, where GANs have already shown promising results. The development of GANs for multi-modal data, such as video or audio, is also an active area of research.

Additionally, there is a growing interest in developing GANs that can operate in more complex and dynamic environments, such as real-time video processing or robotic control. These applications require GANs that can adapt quickly to changing environments and generate high-quality outputs in real-time.

Overall, the future of GANs is exciting, and the potential applications of this technology are vast. Continued research and development in this field are likely to lead to significant advancements in machine learning, computer vision, and other areas of artificial intelligence.

References

[1] Generative Adversarial Nets. Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio.

[2] UNSUPERVISED REPRESENTATION LEARNING WITH DEEP CONVOLUTIONAL GENERATIVE ADVERSARIAL NETWORKS. Alec Radford & Luke Metz, Soumith Chintala.

[3] Conditional Generative Adversarial Nets. Mehdi Mirza, Simon Osindero.

[4] Wasserstein GAN. Martin Arjovsky, Soumith Chintala, and L´eon Bottou

[5] Least Squares Generative Adversarial Networks. Xudong Mao, Qing Li, Haoran Xie, Raymond Y.K. Lau, Zhen Wang, and Stephen Paul Smolley

[6] COMPARISON OF GENERATIVE ADVERSARIAL NETWORKS IN MEDICAL IMAGING APPLICATIONS. MR to CT image synthesis. Christoffer Edlund.

[7] Continuous Conditional Generative Adversarial Networks: Novel Empirical Losses and Label Input Mechanisms. Xin Ding, Yongwei Wang, Zuheng Xu, William J. Welch, and Z. Jane Wang.

[8] A survey on GANs for computer vision: Recent research, analysis and taxonomy. Guillermo Iglesiasa, Edgar Talaveraa, Alberto D´ıaz-Alvarez.

[9] From GAN to WGAN. Lilian Weng.

[10] Conditional Image Synthesis with Auxiliary Classifier GANs. Augustus Odena, Christopher Olah, Jonathon Shlens.

**SAMPLE INDIVIDUAL CONTRIBUTION REPORT:**

**ON A COMPARATIVE ANALYSIS OF VARIOUS GAN MODELS**

Sandeep Kumar Swain

20051025

**Abstract:** The aim of the project is to provide a comprehensive overview of Generative Adversarial Networks (GANs), including their latest architectures, validation metrics, and applications. The project's objective is to evaluate the effectiveness of different GAN variants, identify the best application areas, and summarize the evolution and performance of GANs. The project seeks to provide guidance for future researchers in the field and serve as a reference for those seeking to understand GANs' capabilities and limitations.

**Individual contribution and findings:**

In this project, I delved into the mathematical foundations of Generative Adversarial Networks (GANs) and their associated loss functions. Specifically, I studied the contours of the loss functions used in GANs, including the generator and discriminator loss, and how they affect the training process. I also explored various optimization techniques used to train GANs including change in optimization algorithms and the GAN architectures.

**Individual contribution to project report preparation:**

I contributed in the “basic concepts” chapter of the report and the mathematical equations involved in the report.

**Individual contribution for project presentation and demonstration:**

I provided the slides containing the mathematics and their justifications.

Full Signature of Supervisor:                                 Full signature of the student:

**SAMPLE INDIVIDUAL CONTRIBUTION REPORT:**

**ON A COMPARATIVE ANALYSIS OF VARIOUS GAN MODELS**

Shivam Mishra

20051028

**Abstract:** The aim of the project is to provide a comprehensive overview of Generative Adversarial Networks (GANs), including their latest architectures, validation metrics, and applications. The project's objective is to evaluate the effectiveness of different GAN variants, identify the best application areas, and summarize the evolution and performance of GANs. The project seeks to provide guidance for future researchers in the field and serve as a reference for those seeking to understand GANs' capabilities and limitations.

**Individual contribution and findings:**

In this project, I contributed to the coding part by implementing various GAN models using TensorFlow or Keras. To ensure that the models were properly implemented and trained, I carefully followed the model architecture and loss function specifications provided by the project team. I also fine-tuned the hyperparameters to optimize the training process and improve the performance of the models.

**Individual contribution to project report preparation:**

I contributed in the “implementation” chapter of the report and the diagrams related to the architecture along with the Training Summary.

**Individual contribution for project presentation and demonstration:**

I provided the slides containing the architecture diagrams and their layers.

Full Signature of Supervisor:                                 Full signature of the student:

**SAMPLE INDIVIDUAL CONTRIBUTION REPORT:**

**ON A COMPARATIVE ANALYSIS OF VARIOUS GAN MODELS**

Shivansh Maheswari

20051029

**Abstract:** The aim of the project is to provide a comprehensive overview of Generative Adversarial Networks (GANs), including their latest architectures, validation metrics, and applications. The project's objective is to evaluate the effectiveness of different GAN variants, identify the best application areas, and summarize the evolution and performance of GANs. The project seeks to provide guidance for future researchers in the field and serve as a reference for those seeking to understand GANs' capabilities and limitations.

**Individual contribution and findings:**

In this project, I implemented and trained six different Generative Adversarial Network (GAN) models on the MNIST dataset. These models included vanilla GAN, Deep Convolutional GAN (DCGAN), Conditional GAN (CGAN), Context-Condition GAN(CCGAN), Wasserstein GAN (WGAN), Least Squares GAN (LSGAN). To ensure that the models were properly trained and performed well, I chose appropriate hyperparameters for each model.

**Individual contribution to project report preparation:**

I contributed in the “observations and result analysis” chapter of the report. The outputs and the results involved in the report.

**Individual contribution for project presentation and demonstration:**

I provided the slides containing the outputs and the final results.

Full Signature of Supervisor:                                 Full signature of the student:

**SAMPLE INDIVIDUAL CONTRIBUTION REPORT:**

**ON A COMPARATIVE ANALYSIS OF VARIOUS GAN MODELS**

Pranshu Priyaranjan

20051018

**Abstract:** The aim of the project is to provide a comprehensive overview of Generative Adversarial Networks (GANs), including their latest architectures, validation metrics, and applications. The project's objective is to evaluate the effectiveness of different GAN variants, identify the best application areas, and summarize the evolution and performance of GANs. The project seeks to provide guidance for future researchers in the field and serve as a reference for those seeking to understand GANs' capabilities and limitations.

**Individual contribution and findings:**

In this project, my contributions included documentation of the project cycle, creation of a presentation summarizing the results, and evaluation of the models' performance. Through my efforts, I helped ensure that the project was well-documented and the models were performing effectively, which was critical to the success of the project.

**Individual contribution to project report preparation:**

I contributed in the “overall theory” part of the report and its re-valuation. The aim was to make the documents as concise as possible for the new readers.

**Individual contribution for project presentation and demonstration:**

The overall slides were prepared by me taking the various inputs from the team members.

Full Signature of Supervisor:                                 Full signature of the student:

**TURNITIN PLAGIARISM REPORT**