

A comparative analysis of GANS for generating synthetic network traffic datasets



By

Madeeha Minhas

Fall-2020-MSCS 328050 SEECS

Supervisor

Dr. Syed Taha Ali

Department of Electrical Engineering

A thesis submitted in partial fulfillment of the requirements for the degree of Masters of
Science in Computer Science (MS CS)

In

School of Electrical Engineering & Computer Science (SEECS) ,

National University of Sciences and Technology (NUST),

Islamabad, Pakistan.

(August 2022)

THESIS ACCEPTANCE CERTIFICATE

Certified that final copy of MS/MPhil thesis entitled "A comparative analysis of GANS for generating synthetic network traffic datasets" written by Madeeha Minhas, (Registration No 00000328050), of SEecs has been vetted by the undersigned, found complete in all respects as per NUST Statutes/Regulations, is free of plagiarism, errors and mistakes and is accepted as partial fulfillment for award of MS/M Phil degree. It is further certified that necessary amendments as pointed out by GEC members of the scholar have also been incorporated in the said thesis.

Signature: _____  _____

Name of Advisor: Dr. Syed Taha Ali _____

Date: 23-Jul-2022 _____

HoD/Associate Dean: _____

Date: _____

Signature (Dean/Principal): _____

Date: _____

Approval

It is certified that the contents and form of the thesis entitled "A comparative analysis of GANS for generating synthetic network traffic datasets" submitted by Madeeha Minhas have been found satisfactory for the requirement of the degree

Advisor : Dr. Syed Taha Ali

Signature:  _____

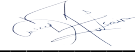
Date: 23-Jul-2022

Committee Member 1:Arshad Nazir

Signature:  _____

22-Jul-2022

Committee Member 2:Dr. Wajahat Hussain

Signature:  _____

Date: 22-Jul-2022

Signature: _____

Date: _____


Dedication

I dedicate this thesis to my beloved parents. Without their endless love and support, the completion of this research would not have been possible.

Certificate of Originality

I hereby declare that this submission titled "A comparative analysis of GANS for generating synthetic network traffic datasets" is my own work. To the best of my knowledge it contains no materials previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any degree or diploma at NUST SEECS or at any other educational institute, except where due acknowledgement has been made in the thesis. Any contribution made to the research by others, with whom I have worked at NUST SEECS or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except for the assistance from others in the project's design and conception or in style, presentation and linguistics, which has been acknowledged. I also verified the originality of contents through plagiarism software.

Student Name: Madeeha Minhas

Student Signature: 

Acknowledgments

Firstly, I would like to thank my supervisor Dr. Syed Taha Ali, for his constant support, guidance, and motivation. His guidance helped me throughout my research and thesis. I could not have thought having a better supervisor and advisor for my MS research.

Besides my supervisor, I would like to express my gratitude to the rest of my thesis committee: Dr Wajahat Hussain and Mr Arshad Nazir, for their motivation and insightful feedback, that made this work possible.

A special thanks to my family and friends for their continuous support and understanding throughout my research.

Finally, I would like to extend my utmost gratitude to Allah Almighty, for letting me go through all the difficulties of my life.

Madeeha Minhas

Contents

1	Introduction and Motivation	1
1.1	Motivation	2
1.2	Problem Statement	2
1.3	Contributions and Research Goals	3
1.4	Scope	3
1.5	Thesis Structure & Outline	4
2	Background	6
2.1	ML-based traffic classification	6
2.2	Generative Adversarial Networks	7
2.2.1	Wasserstein GAN with gradient penalty	7
2.2.2	Tabular GAN	8
2.2.3	CycleGAN	9
3	Literature Review	11
3.1	Traffic classification using ML models	11
3.2	GAN for images	13
3.2.1	GANS for medical images	13
3.2.2	GANS for image-to-image translation	15
3.2.3	GANS for face Aging	16
3.2.4	GANS for super-resolution image generation	18

CONTENTS

3.2.5	GANS for photograph editing	20
3.3	GANS for Text	22
3.4	GANS for Audio	24
3.5	GANS for Video	25
3.6	GANS for Network Data	27
4	Proposed Methodology	30
4.1	Generative Adversarial Networks	30
4.1.1	CycleGAN	31
4.1.2	Wasserstein GAN	31
4.1.3	Tabular GAN	31
4.2	ML-based traffic classification	32
4.2.1	Decision Trees	32
4.2.2	Random Forest	33
4.2.3	XGBoost	33
4.3	Pre-processing of data for traffic classification	33
4.4	Training of ML models	34
5	Implementation	35
5.1	Datasets Collection	35
5.1.1	Covertcast-YouTube dataset	35
5.1.2	VPN-NonVPN dataset	35
5.1.3	Tor-NonTor dataset	36
5.1.4	Doh-NonDoh dataset	36
5.2	Generative Adversarial Networks	37
5.3	ML models	38
6	Results	40
6.1	Evaluation Metrics	40

CONTENTS

6.1.1	Accuracy:	40
6.1.2	AUC Score:	40
6.1.3	F1 Score	40
6.2	Evaluation of GANS	41
6.2.1	Accuracies & AUC Scores	41
6.2.2	F1-Scores	42
7	Discussion	45
7.1	Generative Adversarial Networks	45
7.1.1	Vanilla GAN & Cycle GAN	45
7.1.2	Tabular GAN	45
7.1.3	Wasserstein GAN with gradient penalty	46
7.2	Machine Learning classifier	46
8	Conclusion & Future Work	47
8.1	Future Work	47
A	GANS architectures	

List of Tables

5.1	Statistics of Datasets. The term samples refer to network traffic streams.	36
5.2	GAN hyperparameters.	37
5.3	GAN training time.	38
6.1	CovertCast-YouTube dataset Results	42
6.2	VPN-NonVPN dataset Results	42
6.3	Tor-NonTor dataset Results	42
6.4	Doh-NonDoh dataset Results	42

List of Figures

5.1	ML classifiers	38
6.1	Evaluation process	41
6.2	CovertCast-YouTube dataset F1 scores	43
6.3	VPN-NonVPN dataset F1 scores	43
6.4	Tor-NonTor dataset F1 scores	44
6.5	Doh-NonDoh dataset F1 scores	44
A.1	Vanilla GAN generator	
A.2	Vanilla GAN discriminator	
A.3	Cycle GAN generator	
A.4	Cycle GAN discriminator	
A.5	Tabular GAN generator	
A.6	Tabular GAN discriminator	
A.7	Wasserstein GAN architecture	

Abstract

Modern Machine learning and deep learning research have shown massive advancements in the area of generative models. The most popular framework of generative models is known as Generative Adversarial Networks or GANS. Although the primary research on GANS was based on computer vision problems, recent literature has shown a diverse range of GANS applications including but not limited to text, audio, video etc. Currently, GANS is attracting the attention of networking experts to achieve certain security goals. Till date, many different variants of GANS have been proposed, making it difficult to choose the right variant of GANS, especially for networks as this area is not studied deeply. This thesis aims to provide an in-depth analysis of CycleGAN, Tabular GAN and Wasserstein GAN on different networking datasets to judge their ability to generate synthetic network data. The different variants are evaluated on 3 Machine Learning classifiers (i.e., Decision Trees, Random Forest, and XGBoost) by comparing their results with original datasets. These results and analysis will help network researchers and professionals to decide the best variant of GANs for their data. Furthermore, our work will help the network analysts by providing synthetic labelled data since data collection and labelling is also a challenge in networking.

CHAPTER 1

Introduction and Motivation

Machine learning is gaining popularity and attention in almost every walk of life from health-care to marketing, from banking to self-driving vehicles. Many companies and businesses have started relying on Machine Learning to make real-life decisions, making life impossible without incorporating Artificial Intelligence. This increased popularity and application has forced researchers to find an efficient method to apply these techniques in real-life scenarios. Due to the adverse need for Machine learning with every passing day, very active research is being carried out in different areas like computer vision, Natural Language processing, information retrieval, object detection, sentiment analysis and recommendation systems etc.

Over the past few years, another stream of Machine learning known as generative models has evolved and gained tremendous popularity in no time. Generative Adversarial networks (GANs) is a class of generative models responsible for real and synthetic data generation. The initial focus of generative models was on images but the advancement in research has found a number of applications of GANS like text, audio etc. Till date, Gans is used for solving a number of problems, e.g., image-to-image translation, text-to-image translation, face aging, 3-dimensional object generation, photo blending, audio generation, text generation, photograph editing, video prediction, privacy-preserving, drug discovery etc. Modern researchers are now so convinced of the power of generative models that they started using them for crafting statistical features of network traffic to make it indistinguishable from the original traffic. This feature of GANs makes it so popular because network analysts are now using machine learning classifiers to detect the type of traffic making it impossible to fool modern state-of-the-art censors, without GAN.

Apart from this, many network specialists are interested in generating synthetic and real network

datasets to preserve the privacy of original datasets while providing open access to synthetic datasets to the public. Also, collection and labelling of large amounts of network encrypted traffic is also a challenge because a large amount of network data is required for satisfactory performance of traffic classification using ML models. All of these goals can only be achieved by generative models specifically Generative Adversarial Networks.

With the increase in popularity of Generative Adversarial Networks among security researchers and professionals, the problem of choosing the right variant of generative models for a specific type of traffic is also raised, due to the availability of many variants of GANS. To solve this issue, this thesis provides a comparative analysis of some popular variants of GANS by evaluating different network traffic types to check which variant is best suited among many network traffic datasets.

1.1 Motivation

The emerging need for generative models and their increased application in many fields had motivated researchers to provide new variants of GANs to fulfil different requirements. This results in a large variety of GANs serving different purposes. Choosing the right type of Generative Adversarial Networks for the right type of network traffic can be troublesome for network researchers because trying every single variant can be time-consuming. This is also because of the fact that research of GANS in networking has not evolved much just like it did in computer vision. The biggest motivation of this thesis is to provide a detailed analysis of different variants of Gans with different types of network traffic. This will not only help researchers to find the best variant of GANS but it will also help network analysts to decide upon the best Machine learning classifier depending on the type of network dataset. Moreover, it will enable researchers to find a set of the best hyperparameters that fit a specific variant of GANS or a Machine learning classifier.

1.2 Problem Statement

Generative Adversarial Networks are getting popular in the field of networks and security due to a number of reasons. Firstly, most of network specialist uses ML classifier on statistical features of network traffic which is difficult to evade and can efficiently be done using GANS. Secondly, many businesses and companies do not release open access network datasets due to privacy

concerns, and this is where GAN can be used to generate synthetic data without harming the privacy of users and organizations. Thirdly, the collection and labelling of large network data especially encrypted traffic data limit traffic classification performance so GANS can solve this problem. Due to the increasing demand for GAN in network data and the number of available variants for GANS, selecting the right type of GANS is a challenging problem. Keeping in mind the increasing scope of Machine learning in networking, there is an immense need for detailed analysis that helps researchers decide on the best type of technique depending on the specific requirements.

1.3 Contributions and Research Goals

The goal of this thesis is to analyze the right choice of GANs for generating synthetic datasets based on the type of network traffic data. The thesis has following contributions:

- Provide a clear comparison among some popular variants of GAN namely, Cycle GAN, Tabular GAN and Wasserstein GAN (with gradient penalty) along with the original variant of GAN (vanilla GAN).
- Comparison of different Machine learning classifiers on different networking datasets.
- A detailed analysis that highlights the suitable type of GAN for generation of synthetic data depending on the type of dataset.

1.4 Scope

Just like other areas including computer vision and/or NLP, there is a wide range of applications that require GAN. With a security perspective, GAN can be used to modify statistical features of network traffic to make it look like normal traffic to avoid detection by network analysis tools. This feature of GANs makes it popular in security applications like censorship circumvention, malware and intrusion detection etc. Apart from these, many network and security organizations find it difficult to publish their datasets to maintain the privacy of business or user secrets. Since no one can deny the need for real and synthetic datasets due to the evolution of Machine learning, hence GANs can be used to generate synthetic datasets that can be openly accessed by the public.

1.5 Thesis Structure & Outline

The thesis has been divided into different sections, each of which highlights some important information covering different aspects of the research topic. This thesis is divided into seven chapters that describe the full roadmap of research work done so far. A brief summary of each of the thesis chapters is as follows:

- **Introduction:**

A brief depiction of the research topic, its scope and research goals, research contributions and why it is important.

- **Background:**

Brief Summary of the concepts necessary to understand the idea of thesis and research. Mainly, this chapter will cover different types of Generative Adversarial Networks along with their applications.

- **Literature Review:**

A summary of similar work done by other researchers in a similar area or any research that can cover the problem similar to the current research problem. It will cover research articles related to Generative Adversarial Networks applied in different areas including networks.

- **Proposed Methodology:**

A detailed description of the methodology to be implemented to solve the presented research problem. This includes all the tools, techniques, and algorithms to be used for the implementation step.

- **Implementation:**

It also includes all details related to research implementation including algorithms, tools and technology used, hardware/software requirements etc.

- **Results:**

A detailed discussion of the results from implemented methodology explained above usually in the form of charts, tables and figures where applicable, and a possible discussion of reasons for these results.

- **Discussion**

This chapter explains the final results of the research implementation and answers the main research question in detail. It also provides an in-depth analysis of results and highlights the limitations of the research. These limitations can pave the way for future research.

- **Conclusion & Future Work:**

A brief summarized overview of thesis along with the possibilities to improve the presented research work.

Background

2.1 ML-based traffic classification

Machine learning is a subclass of Artificial Intelligence that allows for predicting outcomes based on the data provided. Nowadays Machine learning models can be trained on any type of data (i.e., images, text, audio, video etc.) and are able to make efficient predictions without explicitly programming the rules for predictions. This unique feature has made Machine learning significantly popular and hence it found applications in every area including but not limited to, medical science, astronomy, weather, economy etc. Machine learning can be classified into four categories i.e., supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning.

- Supervised learning involves the training of models using annotated data, so that it learns the relation among various attributes and labels, and makes accurate predictions on unseen data with no labels.
- Unsupervised learning does not require labelling of data which sometimes makes it a reasonable choice. It learns the patterns and relationships between given input data. The idea of unsupervised learning is inspired by the human brain, having the capability to learn new things.
- Semi-supervised learning is a technique that takes labelled data, in small quantities and unlabeled data in large quantities. The idea of semi-supervised learning gives the advantage of both supervised and unsupervised learning and solves the problems of collecting large, annotated datasets.

- Reinforcement learning is a technique in which an agent learns on the basis of performance rewards. An agent takes an action and is rewarded for desired actions and penalized for undesired actions which lay the foundation of learning for reinforcement learning agents.

The type of ML category to be used depends on the amount of labelled data present and the problem to be solved. Modern research attempts to apply Machine learning in cybersecurity with a special focus on the classification of network traffic. As expected, Machine learning gave promising results in classifying traffic and is currently used by cybersecurity specialists to analyze traffic for a variety of tasks like intrusion detection, malware detection etc.

2.2 Generative Adversarial Networks

: Generative Adversarial Networks are generative models responsible for generating new samples by learning a distribution from training data. GANs is a neural network architecture that consists of two neural networks i.e., generator and discriminator. The generative network is trained to generate samples much like input training data. The discriminator is trained on both training data and the data generated by the generative model. The purpose of the generator is to generate samples that cannot be distinguished from actual training data by the discriminator whereas the discriminator has the role to differentiate between actual and GAN-generated samples. This idea of GANS presents it as a zero-sum game in the context of game theory. Most of the GAN-related research has been done in the area of computer vision with image datasets however modern researchers are trying to expand its application to other areas as well. One of such attempts is to apply GANs to generate network traffic and modifies traffic statistics so that it looks like regular traffic statistics and can avoid detection by machine learning classifiers.

This section will discuss some popular variants of Generative Adversarial Networks.

2.2.1 Wasserstein GAN with gradient penalty

Although GANS proved to be successful in generating realistic and appealing samples, it is difficult to train which leads to problems like in-stable training and mode collapse. Mode collapse refers to the problem where the generator keeps generating one or few types of examples from a diverse set of training data, which is harmful to the generalization properties of GAN. These problems are solved by using a new variant of GAN called Wasserstein GAN. The reason for sta-

ble training of Wasserstein GAN is the use of a loss function called Wasserstein distance $W(p,q)$ which in simple terms is defined as the cost of transmitting mass to convert the distribution p into q . The loss function used by primary GAN reduces divergence, which is not continuous with the parameters of GANS, resulting in unstable and difficult training while Wasserstein loss is continuous and differentiable everywhere. Previous GAN uses discriminator to classify between real and fake samples whereas Wasserstein GAN uses realness and fakeness scores of generated samples. This is because GAN should try to reduce the distance between the distribution of training data and the distribution of generated samples. Despite the training stability achieved by Wasserstein GAN, it can still produce poor examples or face convergence failure. This is because of using weight clipping in Wasserstein GAN, which enforces the discriminator to be in the range of Lipschitz function, which may lead to vanishing or exploding gradients. To prevent this issue, a new method of weight clipping is proposed, that involves the penalization of the gradient norm of the discriminator (or critic) in accordance with its input. This constraint causes the gradient norm of the critic model to be at most 1. This unique method of weight clipping is known as the gradient penalty.

2.2.2 Tabular GAN

Generative Adversarial networks proved their remarkable results in generating images, however generating tabular data like educational or medical records, is difficult, due to the presence of different types of data like time-series, discrete, categorical etc. These problems of tabular data cannot be addressed by conventional GAN architectures used for images. To generate tabular data containing mixed types, the idea of tabular GAN is used. It contains LSTM (long short-term memory) with attention layers to produce data column-wise. Due to the use of attention layers, it is able to capture correlation column-wise by calculating mutual information.

In TGAN, the discriminator tries to check if the samples are from real distribution or not and the generator produces synthetic data in order to fool the discriminator. TGAN uses Multi-layer Perceptron (MLP) for the discriminator and LSTM layers for the generator. To make the model work more efficiently, KL divergence is optimized for discrete features and cluster vector for continuous features by including these in the loss function. Including KL divergence also makes GAN stable. Another advantage of TGAN is that it has the ability to scale up for large enough datasets which means its generalization properties and performance both increase with the increase of training data.

2.2.3 CycleGAN

The image-to-image translation is the transformation of an image into another form like transforming an image showing daytime into a night image.

It requires very large data along with paired examples to train a classifier for image-to-image translation. CycleGAN is an approach that performs direct image translation without using a set of paired examples. Paired datasets are difficult and costly to prepare, and, in some cases, such datasets do not even exist. So, there is a need for a technique that performs image translation without paired data, by using images from different domains only. Training of CycleGAN models is unsupervised and hence requires image data of source and target domains that are purely unrelated to each other, which is also known as the unpaired image to image translation.

CycleGAN is an extended version of GAN, which involves the training of four models, two generators and two discriminators, simultaneously. One generative network converts images from one domain to another domain, while the other one transforms the images from the second domain to the first domain, hence reversing the task performed by the first generator. Discriminators decide the validity and realness of generated images and provide feedback to generators accordingly.

This modification might generate synthetic images but is insufficient to perform image-to-image translations. Hence, CycleGAN uses the idea of cycle consistency which states that training images are fed to the first generator, whose output is fed to the second generator as input so that the generated samples of the second generator should resemble the original training data. This idea of cycle consistency is inspired by machine translation.

The cycle consistency is implemented in CycleGAN by using an additional loss function that calculates the difference between the output from the second generator and original training data, usually by L1 norm or total absolute difference in pixels.

CycleGAN has found many applications in computer vision. Some of them are listed below:

- Style Transfer, that is the conversion from one artistic style of photographs or portraits, to another.
- Object Transformation, that involves transforming one object in an image to some other object, such as the transformation of horses in an image into zebras.
- Season transfer, as its name implies, is the conversion of one season in an image to some

CHAPTER 2: BACKGROUND

other season like winters to summers or vice versa.

- Image enhancement is the improvement of the input image in some way.

Literature Review

3.1 Traffic classification using ML models

- Multimedia protocol tunnelling is a technique to create covert channels to pass network data into the stream of some well-known multimedia applications like YouTube or Skype. The most important feature of protocol tunnelling is unobservability which means that the regular streams are difficult to differentiate from covert channel streams. However, it is highly important to assess the efficiency of these systems which otherwise can lead to serious security concerns. To provide a common mechanism to evaluate these systems, a number of anomaly detection techniques including supervised, unsupervised and semi-supervised have been investigated. The results showed that some of these systems are flawed and can be detected by decision trees-based ML techniques. Hence, these techniques were found useful for detecting covert channels with low false-positive rates. In case the adversary is deprived of labelled network data, semi-supervised and unsupervised techniques have also been analyzed. The findings showed that unsupervised approaches are of no use for detecting covert channels while semi-supervised techniques result in a significant number of false positives which can be improved by tuning parameters or increasing the training data.
- Network analysis and management depend on the accurate categorization of network traffic produced from different protocols and applications. To do this, three supervised ML models, Multilayer perceptron, decision trees and Bayesian Networks are used to perform the classification of six different variants of network traffic like peer-to-peer and Akamai network traffic. The relation between traffic classification results and the content of train-

ing data s analyzed by a series of experiments which demonstrate the fact that models like Decision trees and the Bayesian model can perform classification at high speed and give good results in the case of network applications which modify its source ports dynamically.

- Traffic classification is crucial for monitoring and managing network applications. Prior work based on flow header fields or protocol decoding methods of the application layer are complicated and ineffective for encrypted traffic or peer-to-peer traffic flows. This work aims at classifying flows using Machine learning (ML) classifiers. The ML algorithm used for flow classification is the extreme gradient boosting (XGBoost) algorithm, which was never used for traffic classification in prior works. Evaluation results on a network dataset having real traffic flows show 99.5 % classification accuracy. Moreover, XGBoost outperformed in terms of accuracy compared to other state-of-the-art ML models.
- Due to excessive use of the internet around the world, network experts face a massive growth in internet traffic. There are some approaches, like software-defined networks (SDN) that can support a centralized control method for managing and controlling network traffic, but the data collected by SDN is in very large amount, making it difficult to manage. Machine learning has been proposed recently for processing such data. This work reviews the prior suggestions of using ML for measuring and classifying network traffic in SDN scenarios. This work has a special focus on deep learning techniques for traffic classification which is one of the most ignored cases in prior literature.
- Internet traffic prediction lays the foundation for many tasks like network analysis and security. Due to its importance, it has been researched for a very long period with existing available machine learning and deep learning techniques. The emergence of powerful encryption techniques and protocols has raised new problems. One such problem is the annotation of large, encrypted traffic datasets for training machine learning and deep learning models because existing approaches depend on deep packet inspection (DPI) which gives poor results on encrypted traffic. This work presents a semi-supervised technique that uses Deep Convolutional GAN (DCGAN). It works by generating samples from DCGAN and using them along with remaining unlabeled data to enhance the results of the classifier which is trained on some annotated samples. This solves the problem of the collection and annotation of large datasets. The proposed approach is evaluated on a self-collected dataset which is publicly available as the ISCX VPN-NonVPN dataset. The

results show the accuracy of 89 % and 78 % with very few annotated samples.

3.2 GAN for images

3.2.1 GANS for medical images

- Medical science is a very sensitive field as it deals with human lives on a daily basis. Due to this, practical application of Machine learning in this area is very rare and researchers are trying their best to get the best possible outcome that can match human analysis results to mitigate the human effort in this field. GANS can be very useful particularly in the augmentation of labelled datasets to help ML models make the best possible predictions. The problem was solved by the introduction of MedGAN which performs end-to-end translation of medical images. This is done by combining the adversarial architecture with non-adversarial loss. A discriminator model is used to find differences between generated images and required modalities. Loss functions are used for style transfer in order to mimic the textures of target images. A special type of generator network called CasNet is used to improve the sharpness of generated images using a pair of encoder-decoder models. The proposed GAN is applied to PET CT translation, MR correction and PET denoising, and is evaluated by medical professionals demonstrating exceptional performance ever done by other translation techniques.
- To generate real and synthetic labelled datasets for medical images, many researchers have proposed GANS noise to image and image to image GANS which improved the classification results of ML models. However, none of this research has combined both noise-to-image and image-to-image techniques to get further improvement in results. The proposed GANS uses two-step data augmentation of GAN that are capable of generating and improving MR images with or without tumors. Progressive growing noise to image GAN is used for generating multi-stage realistic magnetic resonance images. Multi-modal unsupervised image translation model combines GAN, VAE or SimGAN that is used to improve the texture or shape of PGGAN-generated images to mimic original ones. The generated images are evaluated on CNN-based tumor classifiers which shows that two-step proposed GANS outperforms the results of existing single-step-based GANS in detecting tumors.
- A lot of work has been done on generating synthetic medical images that resemble origi-

nal ones using GAN, but many of them lack the generalization properties of GANS hence failing to provide new images generated from a distribution similar to input data. To overcome this problem, a novel GAN was introduced resembling the approach of deep convolutional GAN and Wasserstein GAN to generate realistic multi-sequence brain Magnetic Resonance images crucial in the medical field for increasing reliability of diagnosis both in computer-based diagnosis and physician training. Moreover, the proposed GAN architecture solves the problem of low contrast images, and intra-sequence mutability and is evaluated by expert physicians through a visual Turing test. The evaluation demonstrated the validation of proposed GANs by medical professionals.

- The development of an AI-based disease diagnosis system requires large annotated image datasets. Medical experts are needed to annotate these large datasets which require time, effort, and money. Datasets available online are poor in size and quality. Moreover, generating data from real-life diagnoses requires patient consent and involves privacy problems. The problem is particularly critical in terms of diseases like Alzheimer's disease (AD) where a patient cannot be cured after a certain stage hence it is important to detect the patient's AD stage to initiate timely treatment. Hence data augmentation techniques are required to solve the problem. Some of the data augmentation techniques like flip, scale, rotation, and translation are not efficient for computer vision applications and cannot be used for solving critical problems like AD. Therefore, a novel GAN architecture has been proposed to generate realistic tomography images for three important stages of AD namely, normal control, mild cognitive impairment, and AD.
- Machine Learning models are trained to make predictions based on the distribution of input data. However, these models make predictions with the same confidence, although these are not trustworthy. The practical application of ML models in pathology requires an approach to identifying anomalies to prevent incorrect predictions of outlier examples. An unsupervised technique based on Generative Adversarial Networks is proposed for outlier detection in histopathology datasets. The proposed approach showed significant results on histopathology data compared to other GAN-based approaches used for medical imaging. The results of the proposed approach demonstrated more complex pathological imagery as compared to other existing methods. These results achieved required an advanced and complex GANS architecture along with a suitable evaluation metric for detecting outliers and anomalies to test the quality of generated images.

3.2.2 GANS for image-to-image translation

- Even though research on image-to-image translations achieved good results but it still faces the problem of poor performance in cases where image translation needs large changes in shapes. This problem is known as a high-resolution bottleneck. To solve this problem, a novel approach is known as deep image-to-image translations (Deep I2I). The model is learned by using hierarchical features, shallow layers contain structural information while deep layers contain complex contextual information. The transfer learning method (where information is gained from pre-trained models) is used for training I2I models on small datasets. The proposed GAN uses pre-trained GAN discriminators to initiate I2I discriminators and pretrained generators to initiate I2I generators. Implementing transfer learning leads to alignment issues between the generator and encoder, which are solved by using adapter networks. Deep I2I is evaluated on multi-class datasets and the results showed that transfer learning significantly enhances the performance of deep I2I results, particularly for small datasets.
- Conditional GANS for image-to-image translations have made remarkable progress. However, it needs thousands or millions of annotated image pairs to train a conditional GAN. Moreover, human annotation involves time, effort, and money. Taking inspiration from dual learning, a novel dual-GAN is proposed. In the proposed architecture, the first GAN is responsible for translating images from one domain to another, whereas the dual GAN is trained to reverse this task. This architecture allows the translation of images from one domain to another along with reconstruction. Therefore, the loss that measures for reconstruction discrepancy from one domain to another, is also capable of training the translators. Dual-GAN is evaluated on several image translation problems, using unlabeled image data and results show the significant performance improvement of dual GAN compared to normal GAN. Even dual-GAN can achieve somewhat better results than conditional GAN with labelled images.
- Pixel2style2pixel is a novel approach that uses an encoder model to generate a set of style vectors, which is passed to pretrained style GAN hence generating an enlarged latent space. The proposed encoder is capable of doing direct image-to-image translations by considering it as an encoding task from input to a latent domain. The proposed approach is different from existing StyleGAN methods in the sense that it is capable of handling various problems even if input data is not presented in the domain of StyleGAN. The pro-

posed study showed that using StyleGAN for image translation eases the overall training procedure because the presence of an adversary is not required, can provide better support for handling problems without the need for pixel-to-pixel correspondence, and provides support for a multimodal generation.

- Information hiding or data hiding is a method of information security and is classified into two categories, i.e. reversible and irreversible data hiding based on whether the cover image can be reversed or not. Deep Convolutional Generative Adversarial Networks were used for image steganography which converts secret images into noise vectors. It involves the conversion of secret information into noise. The proposed framework learns this mapping and recovers images using CycleGAN. It uses the idea of secret data mapping to generate cover images from secret data. Then the cover image is converted into a target encrypted image using CycleGAN. Moreover, an extractor is proposed that can extract secret information keeping the entire data hiding process reversible. The evaluation results demonstrated the feasibility of this approach.
- GANS proved to be successful in image-to-image translation where it can map images from one domain to another domain. However, these approaches are only able to transfer low-level data and are not capable of passing complex semantic information of images, which makes GANS generate low-quality images. To solve this problem, Attention guided Generative Adversarial Networks are proposed which can handle semantic problems. The AGGAN generators have default attention properties to create an attention mask which is fused with input data to get images with good quality. It also contains an attention-based discriminator which handles attended areas only. Evaluation results showed that AGGAN produces sharp high-quality images compared to existing GAN architectures.

3.2.3 GANS for face Aging

- Face aging refers to predicting the facial images of an individual in different phases of his life. A lot of work has been done in this area but still, it poses some challenges that have not been addressed previously. Firstly, the majority of research done so far needs sequential input, which is not feasible in real-life cases. Secondly, providing face-aging images while maintaining the personality is another challenge. To solve these problems dual conditional GANS is proposed which is trained from many unlabeled images of

people of different ages. In the proposed architecture, the primary GANS converts a facial image into another age depending upon the aging condition, whereas other GAN reverses this task. Therefore, the reconstruction loss can conserve the personality while the discriminator can grasp the transition changes (like patterns of shape or texture in different age groups) of generated images and can train the generation of facial images based on different age groups. The evaluation results showed demonstrated performance of proposed GANS compared to other existing techniques.

- Despite tremendous work done in producing facial images of different ages while maintaining facial features, it still lacks the quality of producing high-quality images. Existing work failed to provide high-resolution images that can capture minute details of aging like wrinkles, fine lines, pigmentation etc. Moreover, aging, in reality, is based on a number of factors like ethnicity, cultural background or lifestyle habits etc., making it a subjective area. Hence, these factors require a more fine-grained method for generating facial images of different ages. To solve these problems, AgingMapGAN is introduced which showed that using high-resolution datasets can produce remarkable results. The proposed study uses weak spatial supervision to control minute changes in aging procedure on different parts of the facial image. Evaluation results showed its remarkable performance in terms of control and quality while keeping low computational costs.
- Existing literature on face aging did not study some important factors like aging accuracy and/or durability of a person's identity. To tackle these issues, a novel Generative Adversarial network is proposed. It uses Convolutional Neural Network (CNN) for the conversion of facial images into specified age(s), and separately learns the changes in facial features over different ages, to ensure that generated images present the desired aging features while maintaining the same identity details. Moreover, to generate more realistic facial images, high-level features of generated images are approximated by a pyramidal discriminator which can model aging features in a fine-grained way. The proposed technique can be applied to a variety of faces along with makeup, pose etc. and still can achieve excellent aging results.
- GANS can be used to generate faces and hence the idea of face aging using GANS is getting popular. The existing literature on face aging using GANS requires lots of pre-processing of image datasets, which requires great time and effort and is quite tedious. It also causes a lot of computational loads making the idea of face aging impractical. To

address this issue, a novel face aging GAN is proposed, whose architecture is based on IcGAN and does not require any pre-processing. This is done by mapping facial images into age and personality vectors using encoders. Unlike other methods, this approach tries to preserve both aging and personality features. Moreover, it uses a reconstruction loss that can retain personality features while preserving other details like hairstyle, pose etc. of input facial image. An aging vector optimization approach is also introduced along with a parameter λ which is adjusted between age and texture features.

- The use of GANS for face aging is gaining attention because of its capability to produce realistic synthetic images through the use of adversarial techniques based on some efficient models like Convolutional Neural Networks (CNN). However, existing literature on GANS represent its models as black box models since the way CNN learn to generate from the distribution of input data is still unknown. Hence, this study proposed a method of explaining GANS by using a qualitative and quantitative analysis of the internal architecture of the model. This is done by analyzing the common filters present in two CNN models. The results showed that GANS used for face aging partly shares its parameters with GANS for diverse applications and the aging process can be trained by general purpose data along with fine-tuning. Experiments with public datasets validated the proposed approach.

3.2.4 GANS for super-resolution image generation

- Image resolution is crucial as images containing fine details are always preferable. GANS is proved to be very useful for generating high-resolution quality images from low-resolution images. It has various applications ranging from medical to surveillance and security. This study proposed the use of Generative Adversarial Networks to generate high-resolution facial images from the low-resolution face image. Despite the fact that GANS is useful for many applications, it is difficult to train. The proposed method uses Wasserstein GANS for generating super-resolution facial images as it provides stable training with less fine tuning and reduces the chances of mode collapse. This study also compared the Wasserstein distance with other objectives on various GAN architectures for generating super-resolution face images. The results showed that WGAN with gradient penalty is easy to converge and avoids mode collapse. Also, it showed that Wasserstein distance is a good metric to improve training.

- Prior work on generating super-resolution face images was not able to provide good results when applied to real-life scenarios. To solve this issue, a novel technique of generating super-resolution is introduced which involves a two-step process. Firstly, GANS is trained to produce low-quality images from high-resolution ones. For this, GANs learn to downgrade high-quality images with unpaired low and high-resolution images. Then in the second step, GANS is trained to upgrade low-quality images into high-resolution ones using paired low and high-quality images. The results showed that this approach outperforms other techniques on real-world images. This approach is applied to high-resolution facial images where results were quite promising, but it can be applied to other categories as well.
- High-quality MRI images can give minute details but the process of acquiring data requires a very long time frame for scanning. To solve this problem, fused attentive GANS is proposed which is able to produce high-resolution magnetic resonance images from low-quality images and reduce the scanning time. In fused attentive GAN, a local fusion module containing three pass models with different convolution filters is used for feature extraction. The global fusion containing self-attention block, channel attention block and fusion operation block is used to improve important features of MR images. Spectral normalization is used to keep the discriminator model stable. The network is trained using 40 sets of MR images and is evaluated using 10 sets of images. The results showed that fused attentive GAN outperforms other reconstruction approaches,
- Single image super-resolution is a very active area of research these days. This is difficult because a single image with low resolution can be generated by many high-resolution images. Although a lot of work has been done in this area, it still poses one challenge, regaining photorealistic effects with natural textures along with pleasant and desired artifacts. To solve this issue, GMGAN is introduced. To generate images with desired artifacts, a novel loss is introduced by combining the image quality assessment. This loss metric is known as gradient magnitude similarity deviation. To overcome GANS instability and mode collapse problems, WGAN with gradient penalty is used. Experiments showed that GMGAN along with quality loss and WGAN with gradient penalty can generate images with appealing textures. Moreover, using large datasets of high-resolution training images can also improve the results.
- Single image super-resolution is very crucial in computer vision applications. Prior work

shows that GANS were quite successful in single image super-resolution problems. But SISR is still a problem for medical images due to a number of reasons. First, medical images have a low signal-by-noise ratio. Secondly, pretrained GAN models on other images may produce inefficient patterns of medical images which could alter the diagnosis results. Thirdly, vanilla Gan may suffer from mode collapse producing poor results of SISR. To solve these problems, a novel method is proposed to use GAN for achieving promising SISR results for brain tumor MR images. The proposed study compared different variants of GANS including WGAN, WGAN with gradient penalty and the proposed multi-scale (MSGAN). The results proved that MSGAN has better performance for achieving both efficiency and quality.

3.2.5 GANS for photograph editing

- Due to improvements in cameras and technology, high-resolution captured images made photo editing a great challenge because high-resolution images are difficult to edit which cannot be done by photo editing tools like Photoshop. Due to the increasing need for high-resolution image editing tasks like blending, Gaussian Poisson GAN is proposed to solve the problem of high-resolution image blending, when combined images are given. To do this, the gradient data is gained by the application of gradient filters. To produce information related to colors, Blending GAN is proposed to map combined and blended images. The proposed approach is capable of generating high-resolution images with some bleedings and undesired artifacts. Results showed that the proposed approach achieves good performance in the Transient Attributes dataset.
- Memorability is a crucial feature of visuals like images, especially for advertisements or educational purposes. Although a study has been done on image memorability still there is no improvement in image memorability, this work studies two approaches to editing images i.e., GANS and image processing and demonstrates its impact on memorability. Since the visual characteristics affecting memorability are still unknown, it is not possible to control it. For this, GAN is used to learn it using labelled image data and then it is used for conditional generation of synthetic images. The aim of this study is to use statistics and recent research advancements to predict memorability. This study will be beneficial for designers, advertisers, photographers etc.
- The need and popularity of photo editing cannot be ignored. Generative Adversarial Net-

works were quite successful in generating realistic and synthetic faces in face aging, attribute changing, pose modifying etc. Despite the success of GANS in generating realistic images, it still lacks the generation of quality images that include fine-grained details in skin, background, hair etc., making the resulted generations unrealistic. To address these issues, a novel framework using Conditional GANS is trained by face masks to generate faces. First, it learns embeddings for the features of every individual part of the face like hair, mouth, eye etc. to generate better images for image translation and editing. Due to the proposed masking component, the proposed approach can be used in a number of applications including manipulation, face synthesis or face swap etc. It can also increase the parsing faces performance to be used for data augmentation.

- Generative Adversarial Networks are getting popular for generating synthetic images as well as image editing. Despite the success of GANS for image editing, due to the high computational expenses of generators, it takes some time to observe the results of a single modification on edge devices, thus preventing a good user experience. To provide an interactive user experience, AnyCost GAN is proposed. Anycost GAN is trained for fast image generation by elastic channels and resolutions. Using subsets of generator produce outputs that resemble generations of the full generator, making it a good option for a fast preview. Anycost generator can be tested on many configurations, by using discriminator conditioned by generator, multi-resolution and adaptive channel training, hence achieving better results than other models. Moreover, novel techniques like optimizing latent code and training encoders can help maintain consistency among sub-generators during image editing. Anycost can be used in a number of cost budget scenarios and can be well suited for a number of latency or hardware requirements.
- Despite current advancements of Generative Adversarial Networks in generating synthetic images, it still lacks information about how GANS can convert a latent code from some random distribution into realistic image generations. This work proposed InterFaceGan for face editing by explaining the latent semantics of image that is learned by GANS. The process of encoding latent semantics in GAN latent space for facial image generation is also analyzed. The finding of this study shows that the latent code of a trained GAN learns a representation from linear transformation. The proposed work studied the various semantic disentanglements and tried to separate various semantic entanglements resulting in more control over various facial features. Other than interpreting age, gender,

and eyeglasses, the proposed approach can change the pose or even can set the artifacts mistakenly generated by GANs. It is also applied for manipulating images with the help of a combination of GAN inversion techniques or encoder models.

3.3 GANS for Text

- Language models or seq2seq models are considered a good choice for text generation. These text generation models work by sampling words sequentially, where each word depends on its previous one and is widely used for several summarization and translation problems. These techniques are evaluated in terms of perplexity which is actually not a good fit to test the quality of the text. These methods are a good way to boost perplexity but often result in poor quality because text generation needs conditioning on word sequences that may not be used during training. To enhance the quality of samples, the idea of Generative Adversarial Networks is proposed as GAN is able to generate samples of high quality and is quite successful in generating images. The design of GAN enables it to generate differentiable output and hence generation of discrete values is a challenge. Moreover, this study shows that perplexity is not the only metric that can be used independently to evaluate the quality of the generated text. This work proposed a conditional GAN for text generation that produces words based on their context. The proposed work showed that GAN-generated text was more realistic compared to other existing language models.
- Current text generation models work by training the model through the text from left to right and predicting the next word based on maximizing the probability of history (previous words). This process is known as Language modelling. The problem with this method is that it suffers from exposure bias i.e., the model predicts the next word based on the previous history of ground truths during training whereas at test time the tokens are predicted based on previously predicted tokens resulting in training testing discrepancy. This problem of exposure bias can be solved by using GAN for text generation. Currently, there is no evaluation metric to test the quality of the generated text. This work proposed a novel evaluation metric that evaluates generated text by probability distribution-based LM metric. The currently available GAN for text generation is evaluated on the proposed metric that shows that the currently used GAN for text is worse compared to language models. This study showed a connection between LM and GAN and will be useful for

future research of GAN for text generation purposes.

- GAN showed great progress in generating realistic images. But generating text is still a challenge for the current architectures of GAN. To produce realistic text, RelGAN is proposed that contains three main modules, a generator based on relational memory to model long-term dependency, Gumbel SoftMax relaxation to train GAN on discrete values, and multiple embeddings in discriminators to give more information about generator updates. Evaluation results showed that RelGAN performs well compared to other models in terms of both quality and diversity and also showed that each module of RelGAN contributes to boosting its performance. RelGAN is different from other text GAN architectures in the sense that it is able to balance between quality and diversity of generated samples. The proposed work is the first attempt to apply Gumbel SoftMax relaxation on GAN for text generation.
- GAN gained popularity in many applications, especially computer vision. But applying GAN in Natural Language processing is still a challenge because text sequences are discrete values and hence gradients cannot be passed from the discriminator model to generators. Recent work use reinforcement learning to pass gradients to generators, but this results in inefficient training. To produce realistic text using GAN without reinforcement learning, the use of an autoencoder to get a lower dimensional text representation is proposed. GAN is then used to generate its representations from this space which are then decoded into sentences. The proposed system is evaluated on both human and BLEU scores which showed that the proposed model is able to generate realistic text.
- Text generation has many applications in the NLP domain such as machine translation or summarization. Inspired by the success of GAN in computer vision, GAN is now attracting the attention of NLP researchers. NLP is a great challenge because text sequences contain discrete values. Hence, a novel technique of using knowledge distillation to make GANS capable of generating realistic text is proposed. The proposed study showed that autoencoders can be used to learn a continuous representation of text, that is a representation which set non-zero probabilities for multiple words. This representation is distilled to train GAN to produce synthetic representations. This approach is evaluated using BLEU and JSD scores on a number of datasets that showed the good performance of the proposed method compared to other GAN architectures used for text generation.

3.4 GANS for Audio

- GANS have shown remarkable progress with images but very little work has been done to generate synthetic audio samples. This work proposed WaveGAN for audio generation based on raw audio data. WaveGAN generates chunks of raw audio with GAN and is inspired by DCGAN architecture, which serves as a pathway to apply other image-based architectures to audio data. Experiments showed that WaveGAN can generate clear sounds even from a set of small vocabulary speech and can generate audios from other domains like birds sounds, piano or drum etc. WaveGAN is evaluated on pre-trained model scores and human assessment. Results showed that WaveGAN performed better than other naive GANs in terms of audio quality and diversity of speakers.
- Efficient audio synthesis is a challenging problem because it requires scaling with more than five magnitude orders. Models like WaveNet, solve this problem of scaling but it lacks to provide a global structure which is equally important in high-quality audio synthesis. This problem can be solved by using GAN for audio synthesis, but current GAN architectures failed to generate locally coherent sounds. To solve this problem, a novel approach is proposed to generate coherent audio by producing log magnitude phases and spectrograms with GAN. This approach is evaluated on NSynth dataset which shows that it is able to outperform WaveNet based on human and automated assessments and can produce audio with many orders faster magnitude than other autoregressive models.
- Synthesizing audio samples from visual content is a challenge that is not fully addressed yet. This study proposes a novel technique which considers audio synthesis as a regression task, which enables to use neural audio synthesis method named V2RA. Moreover, V2RA architecture can be trained without excessive inputs, hence making it scalable and reusable for other applications as well. Compared to other visual to audio conversion approaches, V2RA is solved by using GAN. Moreover, the proposed architecture can predict synchronized sound signals and generate real-time audio samples. This approach is evaluated on two different performance scores. Evaluation results show that V2RA can produce high-quality audio samples and can be used for many audio applications like sound dubbing and audio design.
- GANS has gained remarkable results in audio synthesis in recent years. However, audio synthesis with meaningful semantic controls is still an open area of research. Apparently,

this can be achieved by controlling Generative Adversarial Networks by using metadata in audio data. Yet many audio datasets lack proper labelling and annotation, particularly musical datasets. Hence, an automatic annotation system is needed to properly generate the desired annotations. The soft labels of these annotation systems contain information about properties of corresponding audios that can be used for knowledge distillation from teacher to student model. In this work, knowledge is distilled from an audio annotation system to an audio generator known as DarkGAN. Evaluation results showed that DarkGAN can produce quality audio samples and can maintain attribute control.

- GANS has shown remarkable results in generating representations of unlabeled data in the presence of small labelled data. This approach can be used to learn audio representations. Most of the research has been done on images however, some work has been done on using GAN for speech synthesis conditioned on texts, which limits its use for audio or speech in the absence of transcripts. This problem is solved by using Guided GAN (GGAN), which is able to learn impressive representations and can synthesize high-quality audio samples, using only all, labelled audio data. Evaluation results on speech and non-speech datasets showed that GGAN can learn powerful representations compared to other models, by using 5 % labelled data only.

3.5 GANS for Video

- Video content can be divided into content which shows the objects present in videos and motion that shows the gestures of those objects. Inspired by this concept, Motion and Content decomposed GAN (MoCoGAN) is proposed. It works by converting a set of vectors into a set of video frames. Every vector contains content and motion modules. When the content module is fixed, the motion module is considered a stochastic action. In order to learn motion and content division using unsupervised approaches, a novel adversarial learning technique that uses separate discriminators for both images and videos is proposed. Evaluation results on many challenging datasets along with a comparison to other approaches demonstrate the potential of MoCoGAN.
- The idea of generating realistic videos using two video frames has many applications like movie production or video compression, forensic analysis etc. However, generating videos is a challenge due to the wide search space of video frames. Prior works on video

generation or prediction, produce low-quality videos with blurred motion. To solve these problems, a novel approach to generating realistic videos using GAN is proposed. The proposed architecture contains two GAN models concatenated with each other, one for capturing movements and motion, and the second to produce details of video frames. The loss function designed includes adversarial loss, normalized correlation difference for details of video frames, and gradient loss for motion capturing. Experimental results on three datasets show that this approach produces high-quality videos compared to other prior methods which produce blurry and noisy videos.

- Video generation from text using GAN is a quite challenging problem. This problem is solved by using a conditional GAN to gain static and dynamic data from text. This is done by using a combined architecture including Variation Autoencoder and GAN. The static attributes are used to draw text-based color and layout form whereas dynamic attributes are used to convert text to image kernel. To get large datasets to train models, an approach for generating text-to-video matching corpus from freely available videos is proposed. Evaluation results show that this method can generate short realistic videos, that are perfectly in accordance with the input text. It also outperforms other techniques that use text-to-image procedures to generate videos. Evaluation is done visually as well as by using modified inception scores.
- GAN proved successful for generating realistic images but still faces challenges in producing content having some constraints like gaming levels. Generating game levels that have a pleasant user interface and are also playable. Moreover, due to limited training data, generating distinctive levels with existing GAN architectures is a challenge. To tackle these problems, a novel GAN called CESAGAN, along with a novel training is proposed. CesaGAN is a variation of self-attention GAN that includes a feature vector as input for the training of generator and discriminator, which enables the model to capture non-local dependency. Moreover, to shrink game levels for GAN training, bootstrapping method is proposed which adds playable game levels to the training dataset. Evaluation results show that this approach can produce a greater number of playable levels and also produces some identical levels compared to other existing approaches.
- A video is usually composed of content (appearance) and motion. Appearance refers to the information that remains static over time like the objects present in video and motion is the dynamic gestures that vary over time. This work proposes a self-supervised tech-

nique for video generation using GAN, to get consistency in content and coherence in motion, for generating quality videos. The idea of dual discriminators is used to separate the learning of image and video, to solve the problems of content contrast and temporal structure learning. These tasks allow discriminators to get appearance representations and temporal structure, and to make the generator produce videos with content consistency and normal motion flow. Evaluation results show that the proposed approach performs better compared to other existing GANs for video generation.

3.6 GANS for Network Data

- Even though modern malware is not easily detected due to unknown properties, some latest machine learning techniques are proved to be highly effective in detecting this malware. Due to these ML-based detection mechanisms, a new technique was needed that can adapt to network traffic in order to bypass these detection methods. To solve this problem, GAN is used to adapt malware traffic so that it can mimic regular (non-malware) traffic to avoid detection. This idea is implemented by training a GAN to mimic Facebook chat traffic and provide feedback to malware about how to modify its traffic to prevent blockage. Real Facebook chat traffic features are passed to train a GAN for the specified number of epochs. The GAN-generated parameters are passed to malware which modifies its traffic in accordance with provided parameters. The malware never stops in the process, it just modifies itself. In this ongoing process, the malware evaluates if it is detectable and uses this data as feedback for further training of GANS. Stratosphere IPS system (based on ML techniques) is used to detect or block the traffic.
- Modern machine learning-based traffic classification techniques proved to be successful in differentiating meek traffic from regular HTTP traffic by using information like packet size and timings. Generative Adversarial Network (GANS) was used to obfuscate meek traffic in such a way that it looks like HTTP traffic to avoid detection by classifiers. This was done by devising a mechanism that generates a large number of reproducible HTTP and meek packet streams. The statistical features containing side-channel features like payload size and packet inter-arrival timings are extracted from these traffic streams. After getting the required statistical signatures, STARGAN was trained to modify meek traffic signatures so that they cannot be differentiated from HTTP traffic. The GAN results are evaluated using state-of-the-art machine learning classifiers. The evaluation results

showed that GANS greatly reduced the efficiency of classifiers as the false positive rate of these ML classifiers was increased from 0.183 to 0.834 and the average area under the PR (precision-recall) curve reduced from 0.990 to 0.414.

- Limited data sharing has been a major concern for the research and development of the networking community. This is because of fact that stakeholders are reluctant to share data due to users' or businesses' privacy concerns. Hence GANS was evaluated on the ability to generate synthetic data with high fidelity and minimum human expertise. To implement this idea, an important category of networking data was used i.e., time-series measurements. The major challenges of using this approach were fidelity and privacy. To overcome fidelity challenges, DG separates metadata generations from time series and introduced an auxiliary discriminator for metadata. Mode collapse was tackled by generating minimum and maximum limits along with a normalized time series which can be rescaled to the normal range. By resolving fidelity issues, DG achieved about 43 % improved fidelity as compared to other baseline approaches. However, the authors of this work identified key privacy problems and proposed some suggestions for the improvement of privacy properties of generative adversarial networks.
- Generating realistic network traffic is a great challenge for network and cyber security professionals due to a number of reasons like insufficient or incorrect labelling, availability of obfuscated or outdated network datasets only for privacy preservation purposes, etc. Therefore, a cost-effective method for realistic traffic generation was needed. Inspired by CNN-based GANS used for image generation, PAC-GAN was introduced to generate realistic traffic for a diverse set of network traffic variants. The variants range from DNS to HTTP and ICMP traffic. PAC-GAN generates realistic packets by encoding them into Convolutional Neural Networks (CNN). The purpose of using the CNN model was the conversion and mapping of IP traffic into images containing the representation of generated traffic in the form of matrices. The final results showed that generated packets can be passed to the Internet getting efficient responses.
- The generation of network traffic using GANS was a great challenge. Though some attempts of generating network traffic were successful, they still pose some problems during analyzing the generated traffic. For example, the packets cannot be analyzed using Wireshark or other traffic analysis tools. To solve this problem, a new variant of GANS was introduced, known as PCAP-GAN to generate synthetic traffic datasets that can be

analyzed easily throughout the generation process. The proposed GAN architecture comprises an encoder, generator, and decoder. Firstly, the network traffic is divided into four categories, using an encoder. Then, synthetic data is generated by the data generated using each category as input, and finally, the data generated by the generator is combined in the form of realistic and synthetic network datasets. The generated data is evaluated to judge the similarity between original and generated datasets, using intrusion detection algorithms. PCAP-GAN showed increased performance hence validating the generating capability of the proposed model.

Proposed Methodology

This chapter presents a detailed overview of our proposed methodology. We implemented four variants of Generative Adversarial Networks including vanilla GAN, cycle GAN, Wasserstein GAN with gradient penalty and Tabular GAN, on four different network datasets to provide a detailed analysis for the right choice of GAN as well as ML classifier for the right type of network datasets, with a special focus on Machine Learning challenges like imbalanced data, mode collapse etc. We used three machine learning classifiers to test the quality of data generated by the above-mentioned variants of Generative Adversarial Networks.

4.1 Generative Adversarial Networks

Both the generator and discriminator use the Fully connected layers model, which is the simplest possible architecture in which every neuron in a layer gets its input from all neurons of the previous layer. Leaky-Relu is used as an activation between layers which is good for dying Relu problem. Adam optimizer is chosen with a learning rate of 0.0002 by following the standard in literature and after doing a lot of hyper-parameter tuning. Another important hyper-parameter in GANS is the latent space dimension which greatly affects the generation results of GAN as the generative model selects a point from latent space to generate a new sample. We observed that latent space having dimensions power of 2 or 4 is particularly important in generating samples similar to input data. A batch size of 1024 is selected which should also be a power of 2. Our GANS was able to balance the losses of generator and discriminator though it cannot be taken as a standalone metric for judging the quality of generated samples.

4.1.1 CycleGAN

CycleGAN has shown remarkable results in image-to-image translation. The design of CycleGAN enables it to convert data from one domain to another. This idea is very useful in networks and security where one has to mutate traffic statistics to look like normal traffic. While this idea can be implemented using other versions of GANS since we can provide normal traffic features, however in an end-to-end application where only traffic to be mutated is given as input, CycleGAN is a perfect fit to mutate it to be similar to regular traffic. Just like vanilla GAN, our CycleGAN model also uses fully connected layers in both generator and discriminator. However, the loss functions used for cyclegan are mean square error and mean absolute error based on assigned weights, as this is the standard practice for cyclegan studied in prior literature. To add the effect of regularization and prevent overfitting, gaussian noise and dropout layer are also added to the discriminator model.

4.1.2 Wasserstein GAN

The selection of Wasserstein GAN is made on the fact that vanilla GAN might suffer from mode collapse or training instability issues. We want to check if Wasserstein GAN properties to support better training can also be useful for synthesizing better traffic statistics. Wasserstein GAN also uses fully connected layers in the generator and discriminator. Gumbel SoftMax is used as the last layer activation. Gumbel-SoftMax is used for gaining differential approximation for generating discrete values, so it is suitable when GAN has to produce data with discrete values. As mentioned earlier, Wasserstein GAN uses a special loss function known as Wasserstein distance. The idea of gradient penalty is also used to prevent vanishing/exploding gradients problems.

4.1.3 Tabular GAN

Tabular GAN is used for synthesizing tabular data of various types. Since network features data is also a form of tabular data, with discrete and continuous values, Tabular GAN could be a perfect fit for network data. Moreover, the literature on tabular GAN used very large datasets which gave promising results, however, the main goal of using TGAN is to assess its ability to generate synthetic data on small datasets as well as on network data, which is not done in prior work. TGAN uses fully connected layers for discriminator and LSTM layers with

attention for generators. The intuition behind using LSTM with attention layers is to help the model to memorize large data sequences. TGAN has the property to preserve correlation among different features of data, so the attention layer is used as it is particularly useful for learning the correlation. KL divergence and cluster vectors are also included in loss functions for more stable training.

4.2 ML-based traffic classification

Machine learning proved to be successful in many applications, it can be used for traffic classification as well. However, the choice of using some ML classifier to evaluate our synthetic datasets, was a challenge, due to the availability of many ML models. After going through a lot of prior research on traffic classification, we found that Decision tree-based classifiers are best suited for traffic classification, and that deep learning models and semi-supervised or unsupervised models are comparatively less efficient than these models. After careful study and observation, we selected the following classifiers:

4.2.1 Decision Trees

Decision trees are supervised learning algorithm that designs a model in a tree-like structure. In this tree, every node represents a decision or a leaf node, where the decision shows a branch, and the leaf node shows the class label. Tree branches are split by attributes or features of training data. The feature splitting the branch is selected by a metric known as information gain which refers to the expected decrease in entropy by splitting feature. In simple words, the attribute showing the lowest entropy or highest information gain is selected as the splitting attribute. In decision trees, a decision to be made depends on the sample attributes.

This model is particularly useful for demonstrating the feature importance of the classifier, where the attributes closer to the root node have more importance as compared to attributes below it. Even with the simple tree structure of the model, it can lead to complex models with poor generalization properties, or it can lead to unstable models because of many correlated features in training data. These issues can be solved by using decision tree-based ensemble models.

4.2.2 Random Forest

Random Forest is another supervised learning algorithm that uses an ensemble learning approach, in which a class label prediction is done on the basis of the majority vote of various decision tree models. Random Forest uses the bootstrap aggression technique to avoid overfitting, in which every decision tree model uses a random sample from training data, for training. It also uses the feature bagging approach, in which it builds each tree by choosing random features from the available attribute set.

To assess the feature importance of an attribute, average its information gain for all tree models present in the ensemble. Random forest is a good choice for making predictions as it gives more importance to hyperparameters to get an optimized model.

4.2.3 XGBoost

XGBoost is another approach that uses an ensemble of decision trees, for which it uses a method named gradient tree boosting. It initiates with building a simple decision tree known as a weak learner. With every passing step, XGBoost builds a new tree, thus optimizing the performance of predictions by decision trees compared to previous steps. XGBoost takes advantage of formalizing a regularized model in order to prevent overfitting.

The feature importance can be assessed in a similar way as it is done in Random Forest. Another advantage of XGBoost is that it can efficiently handle missing values of data. It is also good for reduced-cost tasks since it focuses on functional space, hence reducing the costs of the model.

4.3 Pre-processing of data for traffic classification

Pre-processing of data is very important for supervised learning problems; Our Pre-processing includes the following steps:

- Removal of null or infinity values, since ML models do not give accurate results with data containing Nan or infinity.
- Removal of unnecessary columns, that are not useful but only increase training time of ML models.
- Class labels were initially available in string or byte format, which was encoded to be

passed to the model.

- Randomly shuffles the dataset and split data into train and test sets to avoid overfitting problems.

4.4 Training of ML models

We trained machine learning models on four different datasets i.e., CovertCast/YouTube, Tor/NON-Tor, VPN/NON-VPN, and doh (DNS over HTTPS)/NON-doh datasets, by taking each of these datasets in the training set and the corresponding GAN-generated datasets in the test set, to judge the performance of our GANS variants.

Implementation

This chapter provides a detailed overview of different implementation steps performed to answer our research question. This chapter is divided into three sections, Datasets section which provides all the steps related to data collection and annotation, Generative Adversarial Networks which explains the in-depth implementation of different variants of GANS, and the Machine Learning models section which describes the implementation process of different ML classifiers.

5.1 Datasets Collection

We have selected four different network datasets that cover a diverse range of network traffic. These datasets along with their brief description are as follows:

5.1.1 Covertcast-YouTube dataset

CovertCast is a censorship circumvention tool that tunnels web content including news websites, by encoding the content into a series of images and then passing the images to users through live streaming services like YouTube. We extracted the features of YouTube and CovertCast traffic and used manual annotation techniques to compile the final dataset.

5.1.2 VPN-NonVPN dataset

Virtual Private Network (VPN) builds an encrypted and private connection between users' devices and the Internet, hence making a private channel for communication on public networks.

This data is a collection of real-world traffic in ISCX [10] while ensuring the quality of data in size and diversity. For this, a large number of regular and VPN traffic captures are collected, from different applications. This dataset is useful for network analysts for traffic classification of VPN and NON-VPN traffic. We used extracted features of ISCXVPN2016 as our VPN dataset.

5.1.3 Tor-NonTor dataset

Like VPN, Tor is free software that allows its users to communicate anonymously. This data is also a collection of traffic from ICSX, [25] which ensures diversity of data. The data contains tor and regular traffic captures from different applications. We used extracted features of ISCXTor2016 as our Tor dataset.

5.1.4 Doh-NonDoh dataset

Domain Name System is an early protocol which has been highly vulnerable to attacks and hence was a concern for security research. To strengthen this protocol, IETF proposes DNS over HTTP (Doh) which is a protocol that improves privacy concerns and prevents different malicious attacks through encryption and tunnelling DNS queries to protect them against external intervention. The dataset was collected using Dohmeter to capture doh and nondoh traffic and is made publicly available. We used extracted features of the CIRA-CIC-DoHBrw-2020 dataset as our Doh dataset. [31]

The details of the above-mentioned datasets are listed in Table 5.1.

Dataset	No. of Samples	Positive Samples	Negative Samples
YouTube-CovertCast	400	200	200
VPN-NonVPN	10782	5631	5151
Tor-NonTor	67834	8044	59790
Doh-NonDoh	539355	167518	371837

Table 5.1: Statistics of Datasets. The term samples refer to network traffic streams.

5.2 Generative Adversarial Networks

We selected the most popular and common variants of Generative Adversarial Networks with a special focus on unique variable characteristics of network data and the application and usefulness of different GANS on network data that can be helpful for future network and security research and development.

The selected variants of GAN are listed below:

Vanilla GAN: Binary cross-entropy is used as a loss function with a batch size of 1024 and the model is trained for 5000 epochs. The latent space dimension is set to 100.

Cycle GAN: CycleGAN is trained by using Dropout and Gaussian Noise of 0.6 in discriminator to avoid overfitting. Mean square error and mean absolute error is used as loss functions. A batch size of 100 is used as a standard choice of cycleGAN

Tabular GAN: Tabular GAN is trained for 5 epochs with 300 training steps per epoch. Though the training steps were set to 10000 by the authors of Tabular GAN, it increased the training time making the training process difficult. L2 norm of 0.00001 and batch size of 200 are used in Tabular GAN.

Wasserstein GAN: Batch size of 500 is used to train WGAN. The latent space dimension is set to 128 and the model is trained for 300 epochs.

Table 5.2 shows the hyperparameters used for the above-mentioned variants of GANS.

GAN	Optimizer	Learning rate	Activation
Vanilla GAN	Adam	0.0002	Leaky-relu/sigmoid/tanh
Cycle GAN	Adam	0.0002	Leaky-relu/sigmoid/tanh
Tabular GAN	Adam/Adadelta	0.0002, 0.0005, 0.001	Leaky-relu/softmax
WGAN-GP	Adam	1e-4	relu/ GumbelSoftmax

Table 5.2: GAN hyperparameters.

The detailed architectures of different variants of GANS are provided in Appendix.

Training time is also an important factor that is crucial while selecting a GAN for training. Training time taken by different GANS is listed in Table 5.3.

GANS	Training Time
Vanilla GAN	12m 7s
CycleGAN	19m 23s
TabularGAN	40s to 2 hours
WGAN-GP	26m 46s

Table 5.3: GAN training time.

5.3 ML models

Python 3.7 is used as a programming language for the implementation purpose. All the ML classifiers are implemented using the Sklearn library. Sklearn is an open-source ML library that allows users to perform various Machine learning operations including classification, regression and clustering, using Python. Matplotlib is used to create graphs and visualizations. Matplotlib is a free python library that creates appealing and user-friendly visualizations that are easily understandable by users.

A single python file is used for implementing all the classifiers as shown below:

```

if __name__ == "__main__":
    data_folder = ''

    cfgs = [
        [train_features,
         train_label,
         test_features,
         test_label]]

    classifiers = [
        [RandomForestClassifier(n_estimators=100, max_features=None), "RandomForest"],
        [DecisionTreeClassifier(), "Decision Tree"],
        [XGBClassifier(), "XGBoost"]
    ]

    if not os.path.exists('xgBoost'):
        os.makedirs('xgBoost')

    for cfg in cfgs:
        for classifier in classifiers:
            print ("Running classifiers " )
            runClassification_CV(data_folder, cfg, classifier)

```

Figure 5.1: ML classifiers

CHAPTER 5: IMPLEMENTATION

As evident in the above figure, All the classifiers along with parameters are put into a single array and are looped one after the other to get results with a single run. This is the power of Sklearn that can save the time and memory of its users compared to other ML libraries.

Another important factor that greatly impacts the performance of machine learning classifiers, is the splitting of data into train and test sets. After carefully going through prior research works, we used half of the available data for training and the remaining in test sets.

CHAPTER 6

Results

This chapter provides a detailed analysis of the results of our implementation methodology.

6.1 Evaluation Metrics

Evaluation metrics refer to the criteria on which the implemented technique is judged or tested. In Machine Learning, the choice of correct evaluation metric is very important because the ML metrics are dependent on a number of factors including the type of dataset used etc. We selected the following metrics to evaluate our results due to the diversity of datasets used in our research:

6.1.1 Accuracy:

Accuracy is one of the common metrics for evaluating ML models. It refers to the correct number of predictions made by a classifier.

6.1.2 AUC Score:

Unlike accuracy, the AUC score is a more flexible metric for testing ML classifiers. AUC is the area under the ROC curve, where the latter is a probability curve. AUC is a way to check how well a model is to differentiate between two classes.

6.1.3 F1 Score

F1 is a weighted average of two important metrics, precision and recall. Precision tells that out of total predictions made for positive class, how many are actually correctly predicted. Recall

refers to correct positive class predictions out of total positive examples available in data.

6.2 Evaluation of GANS

We selected three classifiers for the evaluation of our synthetic datasets. The evaluation method of synthetic data is illustrated in the figure below:

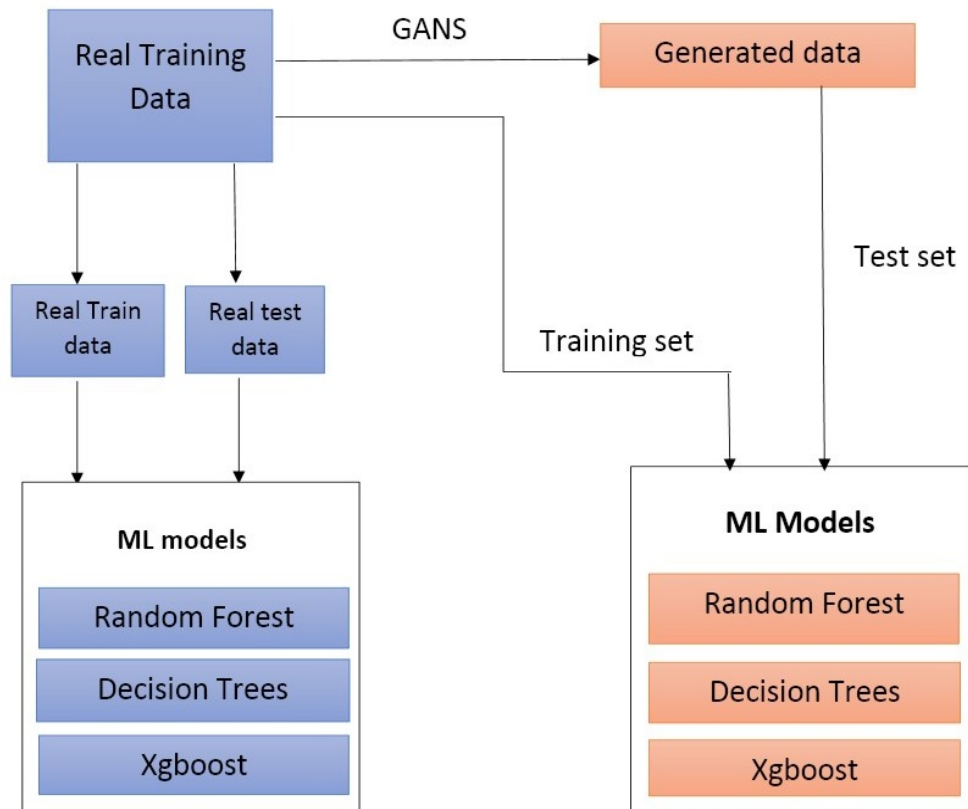


Figure 6.1: Evaluation process

6.2.1 Accuracies & AUC Scores

Accuracies and AUC scores of different variants of GANS are listed in the tables below:

-	Real data		Vanilla GAN		CycleGAN		TGAN		WGAN	
Classifier	Acc	AUC	Acc	AUC	Acc	AUC	Acc	AUC	Acc	AUC
RF	99	0.995	97.5	0.995	82	0.955	97	0.995	96.25	0.995
DT	98	0.983	97.5	0.973	81	0.851	47	0.686	96.25	0.962
XGBoost	99	0.995	97.5	0.973	96.5	0.959	84	0.846	96.25	0.995

Table 6.1: CovertCast-YouTube dataset Results

-	Real data		Vanilla GAN		CycleGAN		TGAN		WGAN	
Classifier	Acc	AUC	Acc	AUC	Acc	AUC	Acc	AUC	Acc	AUC
RF	89	0.951	48	0.435	39	0.571	58	0.736	77	0.8
DT	85	0.86	52	0.437	35	0.356	62	0.685	98	0.851
XGBoost	83	0.912	43	0.382	56	0.64	60	0.64	46	0.5

Table 6.2: VPN-NonVPN dataset Results

-	Real data		Vanilla GAN		CycleGAN		TGAN		WGAN	
Classifier	Acc	AUC	Acc	AUC	Acc	AUC	Acc	AUC	Acc	AUC
RF	99	0.993	83	0.647	89	0.616	94	0.928	86	0.29
DT	99	0.968	81	0.542	88	0.579	91	0.769	86	0.29
XGBoost	99	0.993	93	0.954	88	0.959	88	0.965	86	0.29

Table 6.3: Tor-NonTor dataset Results

-	Real data		Vanilla GAN		CycleGAN		TGAN		WGAN	
Classifier	Acc	AUC	Acc	AUC	Acc	AUC	Acc	AUC	Acc	AUC
RF	99	0.995	76	0.75	81	0.92	98	0.995	77	0.562
DT	99	0.995	76	0.75	71	0.61	97	0.889	77	0.562
XGBoost	99	0.995	76	0.75	71	0.612	97	0.889	77	0.562

Table 6.4: Doh-NonDoh dataset Results

6.2.2 F1-Scores

As some of our datasets are highly imbalanced, so we also compared our GANS against the F1-score for each dataset. Plots showing F1-scores for different datasets are as under:

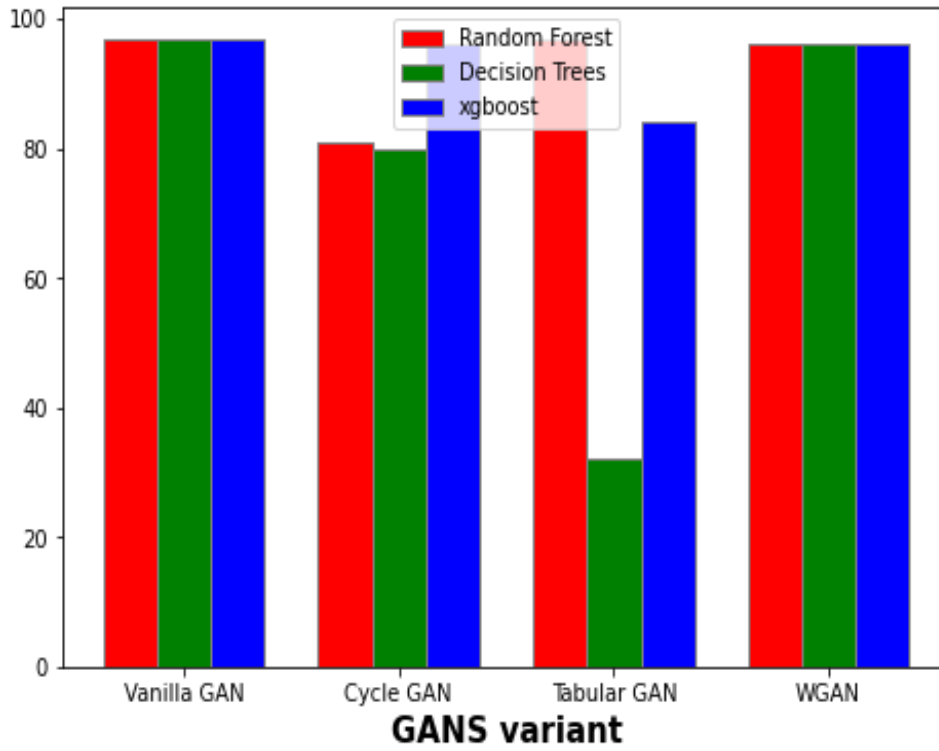


Figure 6.2: CovertCast-YouTube dataset F1 scores

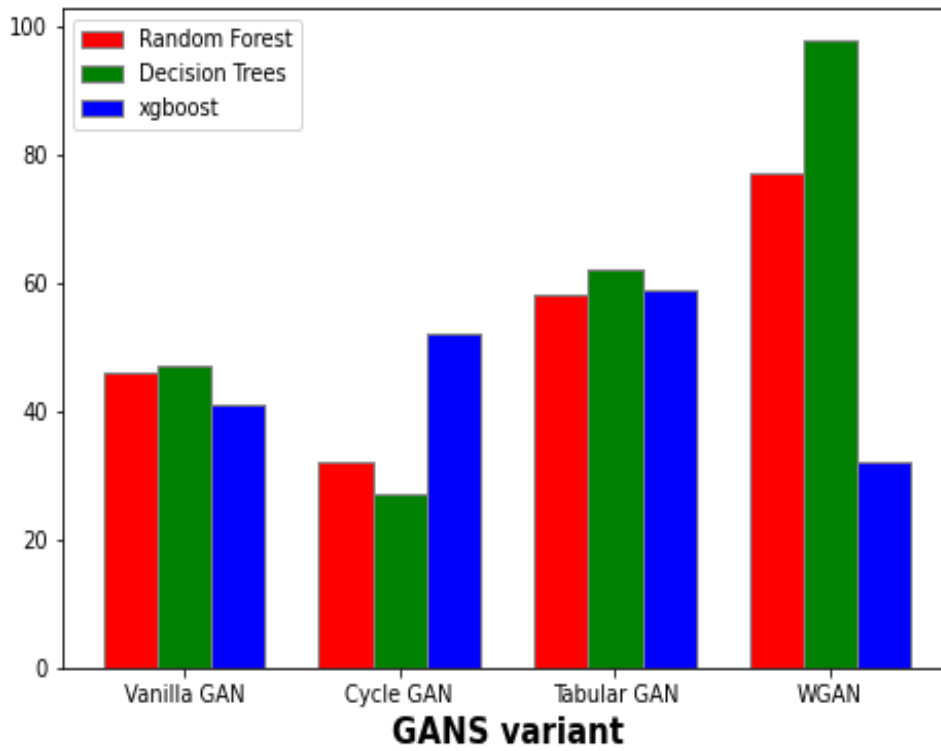


Figure 6.3: VPN-NonVPN dataset F1 scores

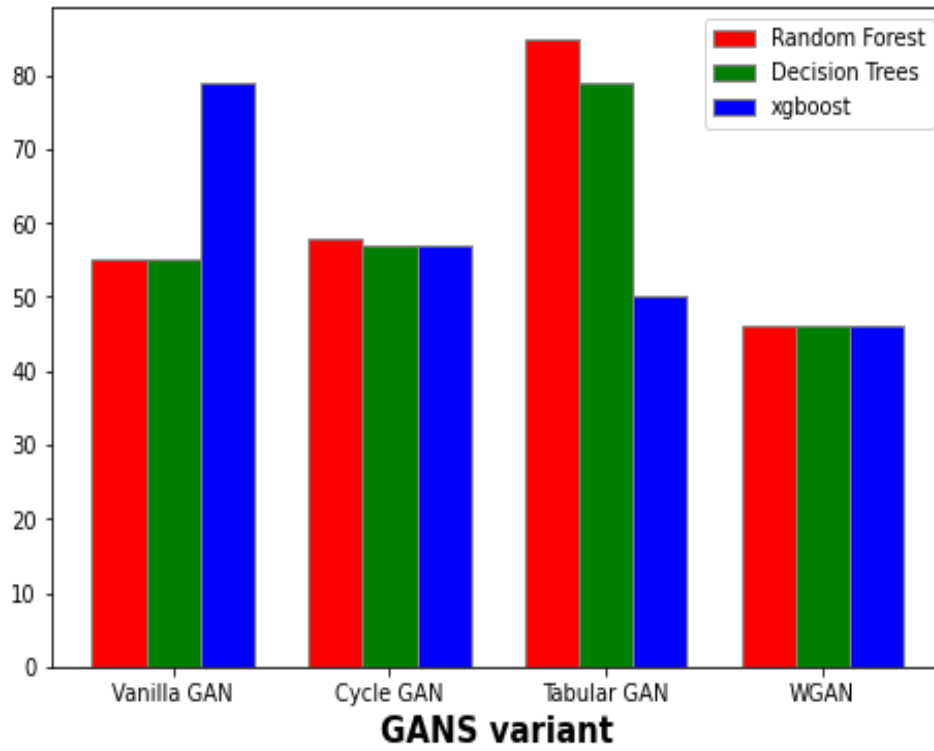


Figure 6.4: Tor-NonTor dataset F1 scores

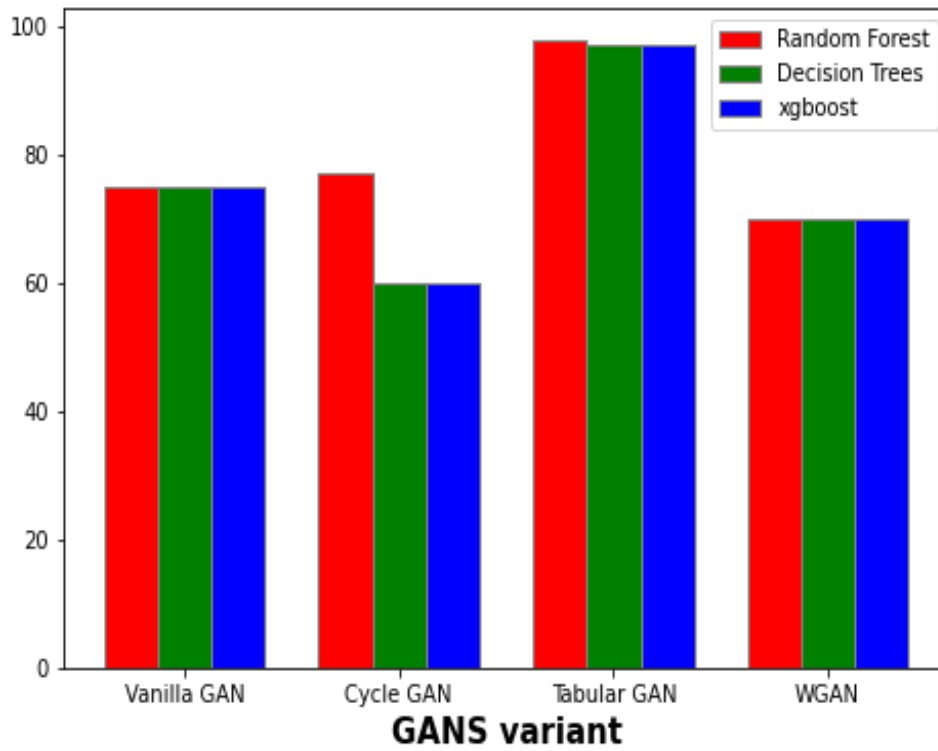


Figure 6.5: Doh-NonDoh dataset F1 scores

CHAPTER 7

Discussion

This chapter will analyze and discuss the reasons and logic for our results and will highlight the arising problems along with the possible solutions.

7.1 Generative Adversarial Networks

The results showed variable results for each variant of GAN in each dataset. In this section, we will discuss what these proposed variants of GANS can and cannot do.

7.1.1 Vanilla GAN & Cycle GAN

These two variants of GANS can handle the simplest possible datasets, i.e., balanced data. Moreover, these GANS are incapable of handling complex data that shows variations along features and samples. In other words, complicated data with hard-to-learn distribution is not meant for these types of GANS. However, these GANS are fast and can be trained on small and simple data in no time, so these can be used for data augmentation.

7.1.2 Tabular GAN

Tabular GAN is the variant which shows very promising results, especially for large datasets with imbalanced class problems. This GAN is capable of handling complicated network data because it uses LSTM and attention layers that can catch the correlation between different features. However, for very large datasets with a large number of features, training Tabular GAN is very time-consuming. Moreover, Tabular GAN gives very poor results for small datasets.

7.1.3 Wasserstein GAN with gradient penalty

This variant of GAN has the ability to produce generalized data and has better generation capabilities than vanilla GAN. However, it is unable to handle highly imbalanced data and achieve results similar to vanilla GAN for such datasets. Still, it is useful to learn complex patterns and correlations among various attributes.

7.2 Machine Learning classifier

We implemented three classifiers, Random Forest, Decision Trees, and XGBoost. This section will discuss some pros and limitations of these classifiers.

XGBoost does not give good results on sparse datasets. Although sparsity can be removed from many ML datasets like text data, networking data cannot be modified. For example, if the inter-arrival time of data is 0 in most packets, we cannot change it, otherwise, it will be inefficient from the networking perspective.

Random Forest and Decision trees, on the other hand, are equally efficient in handling such types of network data.

Conclusion & Future Work

Generative Adversarial Networks have shown massive advancements in computer vision and active research is in progress to use GANS in other fields due to its ability to generate synthetic data, especially in machine learning, where large amounts of data are required for machine learning models. Like other streams, the application of Machine learning in networks is also crucial for the detection and analysis of large network data. So, the application of GANS in networks is always needed to generate synthetic datasets.

This research provides an in-depth analysis by implementing different types of Generative Adversarial Networks on different types of network traffic datasets. Our final results and analysis show how every variant is suitable for specific network data which will not only help in applying GANS to network data to overcome various security challenges but also help with future research in networks, where acquiring large amounts of labelled data is challenging.

Our results showed that later and modified versions of GANS give more promising results. In networks, correlations are important, so GANS which are good at preserving correlations mostly produce realistic samples.

8.1 Future Work

This work can be extended in a number of ways. Firstly, more GANS variants need to be explored to further improve the performance. Since no special variant of GAN is not introduced so far, which can handle the problems specifically related to network data, a new variant of GAN can be introduced. Thirdly, many ML algorithms have been implemented in networks, but no one discussed the methods to pre-process network data without losing its originality, since pre-

CHAPTER 8: CONCLUSION & FUTURE WORK

processing is an important step in ML and can greatly affect the results of ML models. GANS implemented in this thesis can be evaluated using any other criteria or using different machine learning classifiers.

Bibliography

- [1] Karim Armanious et al. “MedGAN: Medical image translation using GANs”. In: *Computerized medical imaging and graphics 79* (2020), p. 101684.
- [2] Diogo Barradas, Nuno Santos, and Luis Rodrigues. “Effective detection of multimedia protocol tunneling using machine learning”. In: *27th USENIX Security Symposium (USENIX Security 18)*. 2018, pp. 169–185.
- [3] Adrian Bulat, Jing Yang, and Georgios Tzimiropoulos. “To learn image super-resolution, use a gan to learn how to do image degradation first”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 185–200.
- [4] Zhimin Chen and Yuguang Tong. “Face super-resolution through wasserstein gans”. In: *arXiv preprint arXiv:1705.02438* (2017).
- [5] Adriel Cheng. “PAC-GAN: Packet generation of network traffic using generative adversarial networks”. In: *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. IEEE. 2019, pp. 0728–0734.
- [6] Iyad Lahsen Cherif and Abdesselem Kortebi. “On using extreme gradient boosting (XGBoost) machine learning algorithm for home network traffic classification”. In: *2019 Wireless Days (WD)*. IEEE. 2019, pp. 1–6.
- [7] Julien Despois, Frédéric Flament, and Matthieu Perrot. “AgingMapGAN (AMGAN): High-resolution controllable face aging with spatially-aware conditional GANs”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 613–628.
- [8] Chris Donahue, Julian McAuley, and Miller Puckette. “Synthesizing audio with GANs”. In: (2018).
- [9] David Donahue and Anna Rumshisky. “Adversarial text generation without reinforcement learning”. In: *arXiv preprint arXiv:1810.06640* (2018).

BIBLIOGRAPHY

- [10] Gerard Draper-Gil et al. “Characterization of encrypted and vpn traffic using time-related”. In: *Proceedings of the 2nd international conference on information systems security and privacy (ICISSP)*. 2016, pp. 407–414.
- [11] Jesse Engel et al. “Gansynth: Adversarial neural audio synthesis”. In: *arXiv preprint arXiv:1902.08710* (2019).
- [12] William Fedus, Ian Goodfellow, and Andrew M Dai. “Maskgan: better text generation via filling in the_”. In: *arXiv preprint arXiv:1801.07736* (2018).
- [13] Angelo Genovese, Vincenzo Piuri, and Fabio Scotti. “Towards explainable face aging with generative adversarial networks”. In: *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2019, pp. 3806–3810.
- [14] Shuyang Gu et al. “Mask-guided portrait editing with conditional gans”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 3436–3445.
- [15] Ishaan Gulrajani et al. “Improved training of wasserstein gans”. In: *Advances in neural information processing systems* 30 (2017).
- [16] Md Haidar, Mehdi Rezagholizadeh, et al. “Textkd-gan: Text generation using knowledge distillation and generative adversarial networks”. In: *Canadian conference on artificial intelligence*. Springer. 2019, pp. 107–118.
- [17] Changhee Han et al. “Combining noise-to-image and image-to-image GANs: Brain MR image augmentation for tumor detection”. In: *Ieee Access* 7 (2019), pp. 156966–156977.
- [18] Changhee Han et al. “GAN-based synthetic brain MR image generation”. In: *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*. IEEE. 2018, pp. 734–738.
- [19] Kazi Nazmul Haque et al. “Guided generative adversarial neural network for representation learning and audio generation using fewer labelled audio data”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 29 (2021), pp. 2575–2590.
- [20] Sangeek Hyun, Jihwan Kim, and Jae-Pil Heo. “Self-supervised video gans: Learning for appearance consistency and motion coherency”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 10826–10835.

BIBLIOGRAPHY

- [21] Auwal Sani Iliyasu and Huifang Deng. “Semi-supervised encrypted traffic classification with deep convolutional generative adversarial networks”. In: *IEEE Access* 8 (2019), pp. 118–126.
- [22] Jyoti Islam and Yanqing Zhang. “GAN-based synthetic brain PET image generation”. In: *Brain informatics* 7.1 (2020), pp. 1–12.
- [23] Li Jia, Yonghong Song, and Yuanlin Zhang. “Face aging with improved invertible conditional GANs”. In: *2018 24th International Conference on Pattern Recognition (ICPR)*. IEEE. 2018, pp. 1396–1401.
- [24] Mingfeng Jiang et al. “FA-GAN: Fused attentive generative adversarial networks for MRI image super-resolution”. In: *Computerized Medical Imaging and Graphics* 92 (2021), p. 101969.
- [25] Arash Habibi Lashkari et al. “Characterization of tor traffic using time based features.” In: *ICISSp*. 2017, pp. 253–262.
- [26] Yitong Li et al. “Video generation from text”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 32. 1. 2018.
- [27] Ji Lin et al. “Anycost gans for interactive image synthesis and editing”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 14986–14996.
- [28] Zinan Lin et al. “Using GANs for sharing networked time series data: Challenges, initial promise, and open questions”. In: *Proceedings of the ACM Internet Measurement Conference*. 2020, pp. 464–483.
- [29] Shiguang Liu, Sijia Li, and Haonan Cheng. “Towards an end-to-end visual-to-raw-audio generation with gan”. In: *IEEE Transactions on Circuits and Systems for Video Technology* 32.3 (2021), pp. 1299–1312.
- [30] Aysşe Rumeysa Mohammed, Shady A Mohammed, and Shervin Shirmohammadi. “Machine learning and deep learning based traffic classification and prediction in software defined networking”. In: *2019 IEEE International Symposium on Measurements & Networking (M&N)*. IEEE. 2019, pp. 1–6.
- [31] Mohammadreza MontazeriShatoori et al. “Detection of doh tunnels using time-series classification of encrypted traffic”. In: *2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on*

BIBLIOGRAPHY

- Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCOM/CyberSciTech)*. IEEE. 2020, pp. 63–70.
- [32] Weili Nie, Nina Narodytska, and Ankit Patel. “Relgan: Relational generative adversarial networks for text generation”. In: *International conference on learning representations*. 2018.
- [33] Javier Nistal, Stefan Lattner, and Gael Richard. “DarkGAN: Exploiting Knowledge Distillation for Comprehensible Audio Synthesis with GANs”. In: *arXiv preprint arXiv:2108.01216* (2021).
- [34] Milda Pocevičiūtė, Gabriel Eilertsen, and Claes Lundström. “Unsupervised anomaly detection in digital pathology using GANs”. In: *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*. IEEE. 2021, pp. 1878–1882.
- [35] Elad Richardson et al. “Encoding in style: a stylegan encoder for image-to-image translation”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 2287–2296.
- [36] Maria Rigaki and Sebastian Garcia. “Bringing a gan to a knife-fight: Adapting malware communication to avoid detection”. In: *2018 IEEE Security and Privacy Workshops (SPW)*. IEEE. 2018, pp. 70–75.
- [37] Steven Sheffey and Ferrol Aderholdt. “Improving meek with adversarial techniques”. In: *9th USENIX Workshop on Free and Open Communications on the Internet (FOCI 19)*. 2019.
- [38] Yujun Shen et al. “Interpreting the latent space of gans for semantic face editing”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 9243–9252.
- [39] Oleksii Sidorov. “Changing the image memorability: From basic photo editing to gans”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2019, pp. 0–0.
- [40] Jingkuan Song et al. “Dual Conditional GANs for Face Aging and Rejuvenation.” In: *IJCAI*. 2018, pp. 899–905.
- [41] Murat Soysal and Ece Guran Schmidt. “Machine learning algorithms for accurate flow-based network traffic classification: Evaluation and comparison”. In: *Performance Evaluation* 67.6 (2010), pp. 451–467.

BIBLIOGRAPHY

- [42] Hao Tang et al. “Attention-guided generative adversarial networks for unsupervised image-to-image translation”. In: *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2019, pp. 1–8.
- [43] Guy Tevet et al. “Evaluating text gans as language models”. In: *arXiv preprint arXiv:1810.12686* (2018).
- [44] Ruben Rodriguez Torrado et al. “Bootstrapping conditional gans for video game level generation”. In: *2020 IEEE Conference on Games (CoG)*. IEEE. 2020, pp. 41–48.
- [45] Sergey Tulyakov et al. “Mocogan: Decomposing motion and content for video generation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 1526–1535.
- [46] Shiping Wen et al. “Generating realistic videos from keyframes with concatenated GANs”. In: *IEEE Transactions on Circuits and Systems for Video Technology* 29.8 (2018), pp. 2337–2348.
- [47] Huikai Wu et al. “Gp-gan: Towards realistic high-resolution image blending”. In: *Proceedings of the 27th ACM international conference on multimedia*. 2019, pp. 2487–2495.
- [48] Lei Xu and Kalyan Veeramachaneni. “Synthesizing tabular data using generative adversarial networks”. In: *arXiv preprint arXiv:1811.11264* (2018).
- [49] Hongyu Yang et al. “Learning face age progression: A pyramid architecture of gans”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 31–39.
- [50] Zili Yi et al. “Dualgan: Unsupervised dual learning for image-to-image translation”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2849–2857.
- [51] Lu Yu, Joost van de Weijer, et al. “Deepi2i: Enabling deep hierarchical image-to-image translation by transferring from gans”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 11803–11815.
- [52] Zhuo Zhang et al. “Generative reversible data hiding by image-to-image translation via GANs”. In: *Security and Communication Networks* 2019 (2019).

BIBLIOGRAPHY

- [53] Jin Zhu, Guang Yang, and Pietro Lio. “How can we make GAN perform better in single medical image super-resolution? A lesion focused multi-scale approach”. In: *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*. IEEE. 2019, pp. 1669–1673.
- [54] Jun-Yan Zhu et al. “Unpaired image-to-image translation using cycle-consistent adversarial networks”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2223–2232.
- [55] Xining Zhu et al. “GAN-based image super-resolution with a novel quality loss”. In: *Mathematical Problems in Engineering* 2020 (2020).

APPENDIX A

GANs architectures

This section contains code snippets of different GANs that we have implemented to provide a detailed overview of their architectures.

```
def build_generator(n_columns, latent_dim):
    model = Sequential()

    model.add(Dense(16, kernel_initializer = "he_uniform", input_dim=latent_dim))
    model.add(LeakyReLU(0.2))

    model.add(Dense(32, kernel_initializer = "he_uniform"))
    model.add(LeakyReLU(0.2))

    model.add(Dense(64, kernel_initializer = "he_uniform"))
    model.add(LeakyReLU(0.2))

    model.add(Dense(128, kernel_initializer = "he_uniform"))
    model.add(LeakyReLU(0.2))

    model.add(Dense(256, kernel_initializer = "he_uniform"))
    model.add(LeakyReLU(0.2))

    model.add(Dense(n_columns, activation = "tanh"))
    return model
```

Figure A.1: Vanilla GAN generator

```

def build_discriminator(inputs_n):
    model = Sequential()
    model.add(Dense(256, kernel_initializer = "he_uniform"))
    model.add(LeakyReLU(0.2))
    model.add(Dropout(0.25))
    model.add(Dense(128, kernel_initializer = "he_uniform"))
    model.add(LeakyReLU(0.2))
    model.add(Dropout(0.25))
    model.add(Dense(64, kernel_initializer = "he_uniform"))

    model.add(LeakyReLU(0.2))
    model.add(Dropout(0.25))
    model.add(Dense(32, kernel_initializer = "he_uniform"))
    model.add(LeakyReLU(0.2))

    model.add(Dropout(0.25))
    model.add(Dense(16, kernel_initializer = "he_uniform"))
    model.add(LeakyReLU(0.2))

    model.add(Dropout(0.25))
    model.add(Dense(1, activation = "sigmoid"))
    model.compile(loss = "binary_crossentropy", optimizer = optimizer, metrics = ["accuracy"])
    return model

```

Figure A.2: Vanilla GAN discriminator

```

def define_generator(data_shape):
    init = RandomNormal(stddev=0.02)
    in_shape = Input(shape=data_shape)
    g = Dense(512, kernel_initializer=init)(in_shape)
    g = LeakyReLU(alpha=0.2)(g)
    g = Dense(512, kernel_initializer=init)(g)
    g = LeakyReLU(alpha=0.2)(g)
    g = Dense(512, kernel_initializer=init)(g)
    g = LeakyReLU(alpha=0.2)(g)
    g = Dense(512, kernel_initializer=init)(g)
    g = LeakyReLU(alpha=0.2)(g)
    g = Dense(512, kernel_initializer=init)(g)
    g = LeakyReLU(alpha=0.2)(g)
    g = Dense(32, kernel_initializer=init)(g)
    out_shape = Activation('tanh')(g)
    model = Model(in_shape, out_shape)
    return model

```

Figure A.3: Cycle GAN generator

```

def define_discriminator(data_shape):
    init = RandomNormal(stddev=0.02)
    in_shape = Input(shape=data_shape)
    d = Dense(512, kernel_initializer=init)(in_shape)
    d = LeakyReLU(alpha=0.2)(d)
    d=GaussianNoise(0.6)(d)
    d=Dropout(0.6)(d)
    d = Dense(512, kernel_initializer=init)(d)
    d = LeakyReLU(alpha=0.2)(d)
    d=GaussianNoise(0.6)(d)
    d=Dropout(0.6)(d)
    d = Dense(512, kernel_initializer=init)(d)
    d = LeakyReLU(alpha=0.2)(d)
    d=GaussianNoise(0.6)(d)
    d=Dropout(0.6)(d)
    d = Dense(512, kernel_initializer=init)(d)
    d = LeakyReLU(alpha=0.2)(d)
    d=GaussianNoise(0.6)(d)
    d=Dropout(0.6)(d)
    d = Dense(512, kernel_initializer=init)(d)
    d = LeakyReLU(alpha=0.2)(d)
    d=GaussianNoise(0.6)(d)
    d=Dropout(0.6)(d)
    out = Dense(1, kernel_initializer=init,activation='sigmoid')(d)
    model = Model([in_shape, out])
    model.compile(loss='mse', optimizer=Adam(lr=0.0002, beta_1=0.5), loss_weights=[0.5])
    return model

```

Figure A.4: Cycle GAN discriminator

APPENDIX A: GANS ARCHITECTURES

```

def generator(self, z):

    with tf.variable_scope('LSTM'):
        cell = tf.nn.rnn_cell.LSTMCell(self.num_gen_rnn)

        state = cell.zero_state(self.batch_size, dtype='float32')
        attention = tf.zeros(
            shape=(self.batch_size, self.num_gen_rnn), dtype='float32')
        input = tf.get_variable(name='go', shape=(1, self.num_gen_feature)) # <GO>
        input = tf.tile(input, [self.batch_size, 1])
        input = tf.concat([input, z], axis=1)

        ptr = 0
        outputs = []
        states = []
        for col_id, col_info in enumerate(self.metadata['details']):
            if col_info['type'] == 'value':
                output, state = cell(tf.concat([input, attention], axis=1), state)
                states.append(state[1])

                gaussian_components = col_info['n']
                with tf.variable_scope("%02d" % ptr):
                    h = FullyConnected('FC', output, self.num_gen_feature, nl=tf.tanh)
                    outputs.append(FullyConnected('FC2', h, 1, nl=tf.tanh))
                    input = tf.concat([h, z], axis=1)
                    attw = tf.get_variable("attw", shape=(len(states), 1, 1))
                    attw = tf.nn.softmax(attw, axis=0)
                    attention = tf.reduce_sum(tf.stack(states, axis=0) * attw, axis=0)
                ptr += 1
                output, state = cell(tf.concat([input, attention], axis=1), state)
            states.append(state[1])
            with tf.variable_scope("%02d" % ptr):
                h = FullyConnected('FC', output, self.num_gen_feature, nl=tf.tanh)
                w = FullyConnected('FC2', h, gaussian_components, nl=tf.nn.softmax)
                outputs.append(w)
                input = FullyConnected('FC3', w, self.num_gen_feature, nl=tf.identity)
                input = tf.concat([input, z], axis=1)
                attw = tf.get_variable("attw", shape=(len(states), 1, 1))
                attw = tf.nn.softmax(attw, axis=0)
                attention = tf.reduce_sum(tf.stack(states, axis=0) * attw, axis=0)

            ptr += 1
        elif col_info['type'] == 'category':
            output, state = cell(tf.concat([input, attention], axis=1), state)
            states.append(state[1])
            with tf.variable_scope("%02d" % ptr):
                h = FullyConnected('FC', output, self.num_gen_feature, nl=tf.tanh)
                w = FullyConnected('FC2', h, col_info['n'], nl=tf.nn.softmax)
                outputs.append(w)
                one_hot = tf.one_hot(tf.argmax(w, axis=1), col_info['n'])
                input = FullyConnected(
                    'FC3', one_hot, self.num_gen_feature, nl=tf.identity)
                input = tf.concat([input, z], axis=1)
                attw = tf.get_variable("attw", shape=(len(states), 1, 1))
                attw = tf.nn.softmax(attw, axis=0)
                attention = tf.reduce_sum(tf.stack(states, axis=0) * attw, axis=0)
            ptr += 1
        else:
            raise ValueError(
                "self.metadata['details'][{}]['type'] must be either `category` or "
                "`values`. Instead it was {}".format(col_id, col_info['type'])
            )

    return outputs

```

Figure A.5: Tabular GAN generator

```

@auto_reuse_variable_scope
def discriminator(self, vecs):

    logits = tf.concat(vecs, axis=1)
    for i in range(self.num_dis_layers):
        with tf.variable_scope('dis_fc{}'.format(i)):
            if i == 0:
                logits = FullyConnected(
                    'fc', logits, self.num_dis_hidden, nl=tf.identity,
                    kernel_initializer=tf.truncated_normal_initializer(stddev=0.1)
                )
            else:
                logits = FullyConnected('fc', logits, self.num_dis_hidden, nl=tf.identity)

    logits = tf.concat([logits, self.batch_diversity(logits)], axis=1)
    logits = BatchNorm('bn', logits, center=True, scale=False)
    logits = Dropout(logits)
    logits = tf.nn.leaky_relu(logits)

    return FullyConnected('dis_fc_top', logits, 1, nl=tf.identity)

```

Figure A.6: Tabular GAN discriminator

```

class Generator(tf.keras.Model):
    def __init__(self, batch_size):
        self.batch_size = batch_size

    def build_model(self, input_shape, dim, data_dim, activation_info: Optional[NamedTuple] = None, tau: Optional[float] = None):
        input = Input(shape=input_shape, batch_size=self.batch_size)
        x = Dense(dim, activation='relu')(input)
        x = Dense(dim * 2, activation='relu')(x)
        x = Dense(dim * 4, activation='relu')(x)
        x = Dense(data_dim)(x)
        if activation_info:
            x = GumbelSoftmaxActivation(activation_info, tau=tau)(x)
        return Model(inputs=input, outputs=x)

class Critic(tf.keras.Model):
    def __init__(self, batch_size):
        self.batch_size = batch_size

    def build_model(self, input_shape, dim):
        input = Input(shape=input_shape, batch_size=self.batch_size)
        x = Dense(dim * 4, activation='relu')(input)
        x = Dropout(0.1)(x)
        x = Dense(dim * 2, activation='relu')(x)
        x = Dropout(0.1)(x)
        x = Dense(dim, activation='relu')(x)
        x = Dense(1)(x)
        return Model(inputs=input, outputs=x)

```

Figure A.7: Wasserstein GAN architecture