3D POINT CLOUD CLASSIFICATION WITH GANS: ACGAN AND VACWGAN-GP

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ONUR ERGÜN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

FEBRUARY 2022

Approval of the thesis:

3D POINT CLOUD CLASSIFICATION WITH GANS: ACGAN AND VACWGAN-GP

submitted by **ONUR ERGÜN** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Halit Oğuztüzün
Head of Department, **Computer Engineering** _____

Assoc. Prof. Dr. Yusuf Sahillioğlu
Supervisor, **Computer Engineering, METU** _____

Examining Committee Members:

Assoc. Prof. Dr. Sinan Kalkan
Computer Engineering, METU _____

Assoc. Prof. Dr. Yusuf Sahillioğlu
Computer Engineering, METU _____

Assist. Prof. Dr. Hamdi Dibeklioğlu
Computer Engineering, Bilkent University _____

Date: 11.02.2022

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Onur Ergün

Signature :

ABSTRACT

3D POINT CLOUD CLASSIFICATION WITH GANS: ACGAN AND VACWGAN-GP

Ergün, Onur
M.S., Department of Computer Engineering
Supervisor: Assoc. Prof. Dr. Yusuf Sahillioğlu

February 2022, 76 pages

With the developing technology and the power of sensors, 3D data has started to be used in almost every field. Point clouds detected with LIDAR sensors or obtained by sampling 3D meshes have begun to come to the fore in many areas from autonomous driving to data visualization, from generating new data and mesh to classifying detected 3D objects. Machine learning and deep learning techniques are widely used to make sense of this produced data and to implement various applications. In this work, we propose networks to predict the class to which the 3D point cloud belongs, with Auxiliary Classifier Generative Adversarial Network and Versatile Auxiliary Conditional Wasserstein Generative Adversarial Network with Gradient Penalty, which are kind of GANs working with class labeled data. Unlike other classifiers; we are able to enlarge the limited data set with the data produced by taking advantage of the power of generative models, thus we aim to increase the success of the model by training it with more data. As suggested by the ACGAN models, the Discriminator is trained with synthetic data generated by the Generator with using the class label, in addition to the real dataset, ensures that data can be classified while separating real and fake data. Thus, as the training evolves, the Generator is trained to produce more

realistic data; which forces Discriminator to classify better. Wasserstein GAN with GP demonstrates similar abilities with a better training by replacing its Discriminator with Critic and modifying its loss function. In this work, we focus on merging Wassterstein GAN-GP with conditional GAN in order to improve the classifier's performance. With this study, the proposed models were tested on 3D datasets and the results were compared with other studies.

Keywords: Machine Learning, Deep Learning, Generative Models, GAN, CGAN, ACGAN, Wasserstein GAN, Wasserstein GAN-GP, Classification, Shape Retrieval

ÖZ

3B NOKTA BULUTUNUN ÇEKİŞMELİ SİNİR AĞLARI İLE SINIFLANDIRILMASI: ACGAN VE VACWGAN-GP

Ergün, Onur
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi: Doç. Dr. Yusuf Sahillioğlu

Şubat 2022 , 76 sayfa

Gelişen teknoloji ve sensörlerin gücüyle birlikte, 3B veriler neredeyse her alanda kullanılmaya başlanmıştır. LIDAR sensörleriyle tespit edilmiş veya sentetik 3 boyutlu ağların örneklenmesiyle elde edilmiş nokta bulutları; otonom sürüşten veri görselleştirmeye, yeni veri ve 3 boyutlu ağ üretmekten tespit edilen 3 boyutlu nesnelerin sınıflandırılmasına kadar bir çok alanda ön plana çıkmaya başlamıştır. Üretilen bu verilerin anlamlandırılmasında ve çeşitli uygulamalarının gerçeklenmesinde makine öğrenmesi ve derin öğrenme teknikleri sıkça kullanılmaktadır. Bu çalışmada, biz, etiketlenmiş veri ile çalışan bir çeşit Çekişmeli Sinir Ağları olan Sınıflandırıcı Eklenmiş Çekişmeli Sinir Ağları ile 3B nokta bulutunun ait olduğu sınıfı tahmin etmesi için bir ağ öneriyoruz. Diğer sınıflandırıcılardan farklı olarak; üretici modellerin gücünden yararlanarak üretilen veriler ile sınırlı veri setini büyütebiliyor ve modeli daha fazla veri ile eğiterek başarısının arttırılmasını hedefliyoruz. Sınıflandırıcı Eklenmiş Çekişmeli Sinir Ağları modellerinin önerdiği şekilde, gerçek verisetine ek olarak Üretici tarafından sınıf etiketi kullanılarak üretilen sentetik veriler ile eğitilen Ayrıştırıcı sayesinde; gerçek ve sahte veriyi ayırırken aynı zamanda verinin sınıflandırılabilme-

sini de sağlıyoruz. Böylece, eğitim ilerlerken Üretici daha gerçekçi veriler üretmek için eğitilirken; Ayrıştırıcıyı da daha iyi sınıflandırmaya zorlamaktadır. Gradyan Cezalandırmalı Wassertein Çekişmeli sinir ağları ise benzer yetenekleri, Ayrıştırıcıyı Eleştirmen ile değiştirip; kayıp fonksiyonunu değiştirerek daha iyi bir eğitim ile göstermektedir. Bu çalışmada, biz Gradyan Cezalandırmalı Wasserstein Çekişmeli sinir ağlarını koşullu Çekişmeli Sinir Ağları ile birleştirerek veri sınıflandırma performansını iyileştirmeye odaklanıyoruz. Yapılan bu çalışma ile önerilen modeller 3 boyutlu verisetleri üzerinde denenmiş ve sonuçları diğer çalışmalar ile karşılaştırılmıştır.

Anahtar Kelimeler: Makine Öğrenmesi, Derin Öğrenme, Üretici Modeller, Çekişmeli Sinir Ağları, Koşullu Çekişmeli Sinir Ağları, Sınıflandırıcı Eklenmiş Çekişmeli Sinir Ağları, Wasserstein Çekişmeli Sinir Ağları, Gradyan Cezalandırmalı Wasserstein Çekişmeli Sinir Ağları, Sınıflandırma, Şekil Yeniden Alma

To my father, may his memory forever be

To my family and beloved wife

ACKNOWLEDGMENTS

First of all, I am grateful to my adviser Assoc. Prof. Dr. Yusuf Sahillioğlu for his understanding, guidance and help. I would not have been able to bring this thesis to this level without his suggestions, opinions and collaborations. Working with him during this process was a source of pleasure and success for me.

I would also like to convey my other gratitude to my family, who supported me in every way in this process as well as in my whole life. Feeling that they are with me at every moment gives me extra strength and a desire to do better.

I would like to express my deepest gratitude to my dear wife. It would have been impossible for me to do all those without her who has always supported and motivated me throughout this entire process. She was by my side through all the difficulties I faced and each time she managed to bring me out of despair. Likewise, I shared every success I had with her and we rejoiced together. Most importantly, I know that she will always support me and will always be by my side. Being with her and feeling her endless love is priceless.

Another thanks to my dear friend Ersel Er. It was enjoyable and very valuable to overcome the difficulties in the educational process with him.

I would also like to thank my whole colleagues, especially Hasan Ali Duran, and my team leader Cemal Samur. I am grateful to them for supporting me with their ideas and understanding and at the same time tolerating me in difficult times.

And finally, I would like to thank all my friends and relatives who have been with me on this path. Their support is very important and valuable to me.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

2D	2 Dimensional
3D	3 Dimensional
GAN	Generative Adversarial Network
CGAN	Conditional Generative Adversarial Network
DCGAN	Deep Convolutional Generative Adversarial Network
ACGAN	Auxiliary Classifier Generative Adversarial Network
VACGAN	Versatile Auxiliary Classifier Generative Adversarial Network
WGAN	Wasserstein Generative Adversarial Network
WGAN-GP	Wasserstein Generative Adversarial Network with Gradient Penalty
GAAL	Generative Adversarial Active Learning
WGAAL	Wasserstein Generative Adversarial Active Learning
WGAAL-GP	Wasserstein Generative Adversarial Active Learning with Gradient Penalty
PCA	Principal Component Analysis
SVM	Support Vector Machine
OCSVM	One Class Support Vector Machine
KNN	K Nearest Neighbour
SOGAAL	Single Objective Generative Adversarial Active Learning
MOGAAL	Multiple Objective Generative Adversarial Active Learning
VACWGAN-GP	Versatile Auxiliary Classifier with Wasserstein Generative Adversarial Network with Gradient Penalty
G	Generator
D	Discriminator

C	Critic
GP	Gradient Penalty
AI	Artificial Intelligence
ML	Machine Learning
CD	Chamfer Distance
ADV	Adversarial
CNN	Convolutional Neural Network
DBN	Deep Belief Network
RBM	Restricted Boltzmann Machine
ANN	Artificial Neural Network
LIDAR	Light Detection and Ranging
AR	Augmented Reality
PRC	Precision Recall Curve
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
ACC	Accuracy
OVO	One vs One
OvR	One vs Rest

CHAPTER 1

INTRODUCTION

1.1 Motivation and Problem Definition

In simple terms, classification is the task of deciding to label of the given sample with some techniques that learns from other known samples. Classes can differ from the dataset working on, the objective of classification and the nature of problem and also named as targets, labels or categories. Furthermore, some problems are defined as the detection of samples that belong to one class or another, called as binary classification, while others can be expressed as multi class and multi label classification. Multi class classification is a process that trying to categorize every sample which can belong to exactly one of the target classes. Unlike binary classification, the target class space has more labels than two. In multi label classification, one sample can have more than one label. For example, detection of mail whether spam or not is a binary classification problem, distinguishing animals at 2D images from each other is multi class classification problem. Identifying the categories of a film is a good example of a multi-label classification problem.

As can be seen from the examples, classification is general, suitable and useful for

use in many areas. With the advancement of technology, it is possible to encounter classification problems in almost every field. As the amount of data produced increases, the problems are shaped accordingly. With the increasing processing power of computers, the way of approaching these classification problems is also changing and developing. Solutions starting from simple machine learning methods are evolving into very complex artificial neural networks and even deep learning architectures in order to increase classification accuracy.

These changes have an impact on the data that can be handled. The success of categorizing 2D images has paved the way for 3D object recognition and classification. Many fields of study have emerged and have become challenging problems, such as detecting objects in a room scanned with LIDAR sensors, detecting and classifying other vehicles, pedestrians, living things, and other objects in autonomous driving, detecting anomalies in data obtained from MRI devices and so on.[23][24][25][26][27]

In this study, we appeal to the generative models for the classification task of 3D objects using 3D point cloud data. This data can be detected directly with sensors, or it can be obtained by converting synthetically created 3D meshes into point clouds by sampling. In order to increase the classification success, we propose using the data generated by the generator close to the real data set distribution in classifier training. For that purpose, we propose both a modified ACGAN for the 3D point cloud and a combined version of a Wasserstein GAN-GP method with a classifier. We adapt the gradient penalty part to work with 3D point cloud data. As a result, we present methods for labeling 3D shapes without relying on the surface or connection information that characterizes 3D objects.

1.2 Proposed Methods and Models

For many years, machine learning methods and approaches have been widely employed in classification problems. The success of machine learning models using supervised learning in classification has increased considerably with the developed models. Many methods have been proposed, from Decision Tree, which is one of the most basic methods, to deep convolution networks and have proven themselves in

various applications. These machine learning algorithms train themselves using the data they encounter during the training phase and gain the ability to recognize and understand the underlying mean of data that has similar characteristics as the training progresses. After sufficient training, it models the data and defines a unique function in order to obtain the desired output from the given data.

In 3D point cloud classification problem, many machine learning approaches proposed. Many researchers utilize CNN models because of the nature of the problem and the resemblance to 2D image classification challenges. Unlike 2D images, the points are unordered in the 3D point cloud. It also varies from 3D meshes in that it is devoid of structural information. While there is inter-vertex connection information that defines surfaces in 3D meshes, point cloud is defined as a set of scattered points in space. Some methods employ three-dimensional voxels as a solution, which are analogous to pixels in a two-dimensional image.

In this study, we employ point cloud data to classify objects in 3D space. As a classification tool, we propose Generative Adversarial Networks (GAN) which is the most popular and powerful reconstruction-based deep learning method of recent times[5]. Basically, GAN is a combination of two networks trying to beat each other and essentially both models competing in a zero-sum-game and finally both are training together. In this game, while the Generator tries to trick the Discriminator by generating data as close to the real data set as possible; Discriminator also tries to distinguish the data produced by the Generator from the real data. Furthermore, GANs address unsupervised generative problems as supervised aspects with this network combination approach. Besides, it is very powerful tool to data augmentation with its Generator model. As the power of GAN networks has been realized, several other forms of GANs have been proposed in a row to be used in many fields and to serve different purposes. While vanilla GANs are firstly introduced with multi-layer perceptron networks, Deep Convolution GANs (DCGAN) comprise of Convolutional Networks, Conditional GANs (CGAN) take into consideration condition information to generate better results. Another variation of GANs is Auxiliary Classifier GAN that produces labels in addition to real/fake results. We offer to use the Discriminator of the ACGAN solution, which we have adapted according to the 3D Point cloud, as a classifier in the first proposed method. By developing this compact solution with

another successful GAN model, Wasserstein GAN with GP, we strengthen the generator side and force the classifier we added in parallel to get better results and we offer this approach as our second method.

1.3 Contributions and Novelties

An ACGAN network is presented as a unique approach to 3D object classification problem which allows classification of data on the point cloud using only the 3D position information of the points without the use of additional features.

The Discriminator of this network is utilized as a classifier and the dataset has been augmented owing to the data generated by the Generator during training, resulting in enhanced generalization of the model.

The suggested method allows the generator to be multiplied up to the number of target classes theoretically, thus overcoming the possible mode collapse problem during training and enabling each Generator to be used as a data generator for the specific classes after training.

We benefit from the generative power of Wasserstein GAN-GP by combining it with a separate classifier in our second approach. WGAN-GP, which we have customized the gradient penalty part for the 3D point cloud, learns the distribution of the real data set and undertakes the task of data augmentation alongside the classifier in this method.

As a summary our contributions can be summed up as follows:

- Compared to other studies, ACGAN is used in classification of 3D point clouds with only 3D coordinate information of points that compose the 3D shapes.
- With multiple generator support, avoiding possible mode collapse problem which might occur due to the nature of the algorithm of GANs
- Inspired by the spirit of the VACGAN approach, using the generative model of Wasserstein GAN-GP which placed next to the separated classifier as data augmenter

1.4 The Outline of the Thesis

This thesis is divided into six chapters. In the Chapter 1, the definition of the problem and the motivation is covered and the purposed method are presented along with a brief overview of the model. Contributions are also given and the thesis outline is stated. The background information needed to comprehend the proposed approaches is provided in Chapter 2. Previous studies to classify 3D shapes are detailed in the Chapter 3 and a literature search is also presented. In Chapter 4, the suggested approaches to address the problem, as well as the methods of application and how the proposed solution works are detailed. Chapter 5 describes the data sets with their attributes and explains the experiments we conducted on those datasets using the proposed methods, as well as the results, and compares them to previous studies. Finally, in Chapter 6, the work conducted is summarized, concluded and suggestions for future works are given.

CHAPTER 2

BACKGROUND

2.1 Point Cloud

In a spatial reference system, a point cloud is a collection of data points. In practice, the phrase refers to 3D points that reflect the shape, size, and placement of real-world objects in three dimensions. Every point in a point cloud data collection has at least three attributes: X, Y, and Z. RGB for color or intensity of the laser reflection are two more frequent characteristics. The object's surface's reflective qualities are represented by the intensity value. These attributes can be utilized in the categorization process to offer information on the qualities of the scanned item.[1]

Point clouds are commonly used to model physical things in three dimensions, with points taken from the object's exterior surface forming the envelope. This form of point cloud is used in computer vision, computer graphics, and computational geometry, and is often the raw output of 3D sensors or made synthetically by computer algorithms. With the fast development of reality capture devices and sensing technologies like 3D laser scanners and LIDAR (light detection and ranging), 3D point clouds have become widespread and cheaply obtainable, spawning a slew of groundbreaking 3D environment-based applications. Nonetheless, seeing and interacting with the surrounding environment based on the data obtained is a common job shared by most of these applications. For instance, in robot manipulation, which has a wide range of applications including autonomous driving, autonomy, and robotic surgery, the robot must be aware of and comprehend the surrounding environment in order to complete its motion planning. The virtual effect in augmented reality (AR) is created by recognizing features in the surroundings, and the appropriate displacement of this

effect is heavily dependent on perfect 3D registration of virtual and actual objects. As a result, developing algorithms that can consume 3D point cloud data directly and generate semantic characteristics that may be employed in the perception job is critical.[2]

Different data formats appropriate for 3D object representation exist, which may be geometric form based on their data structure (irregular) and categorized into rasterized form (regular grids).

Rasterized Form

The rasterized form includes a structured underlying grid that allows mathematical operations to be done without too much effort, including multi-view pictures, depth maps, and volumetric representation (voxel girds). Multi-view pictures, for example, project 3D situations onto 2D image planes from various angles and use the collection of those 2D images as a representation of the original 3D scenarios. Following that, standard image computing techniques (2D matrix) become instantaneously available. The volumetric representation typically constructs things using regularly spaced 3D cubes, where values can be supplied to the cubes to produce voxels, which are comparable to pixels in 2D. As a result, a three-dimensional grid will be formed, and computational tools will be easy to use once more.

This computationally advantageous aspect has led to the widespread use of rasterized form representation in hundreds of simulation and visualization applications. However, much as a coin has two sides, this representation has significant drawbacks: In multi-view projection, depth information is lost (projection loss), making a sphere indistinguishable from an ellipsoid in the simplest instance. Furthermore, if the viewpoints remain the same, the projection of a rotated object may alter dramatically (not rotation invariant). Because the approach must quantify the space into 3D grids, the volumetric representation suffers from quantization loss. Last but not least, as the grid resolution grows, the complexity increases at least cubically, resulting in finer structures being overlooked in practice.

Geometric Form

Unlike the rasterized form, which always has a well-ordered grid, the geometric

form's data structure frequently has an irregular feature. The most common representations are simplicial meshes and point clouds, among others. The polygon mesh is a set of convex polygons, such as triangles and quadrilaterals, that is widely used to determine an object's profile in geometric modeling or as a computational mesh in scientific computing. As a result, rendering and modeling with significant hardware support prefer the polygon mesh. Dealing with an uneven mesh, on the other hand, is never easy. For starters, polygon mesh is man-made, and constructing it can be difficult because to the wide variety of polygons and sizes available. Furthermore, editing polygon mesh is expensive and unintuitive, limiting the versatility with which appropriate algorithms may be designed.

Point clouds, on either side, are significantly easier since they represent the raw output data of multiple 3D sensors; no quantization, projection, or mesh construction is required, and the information is preserved. However, because only point coordinates are provided, any processing method must implicitly infer the link between points in order to form the actual object, and the connection might differ even within the same set of points.

2.2 Shape Retrieval

The quantity of 3D models generated is rising day by day as the frequency and application areas of 3D models grow. This necessitates querying the continually developing and expanding 3D model sources. In summary, shape retrieval is the process of creating a search engine for 3D models stored in various forms in repositories and gathering comparable and suitable models to the model queried by these search engines. There are three fundamental approaches for retrieving shapes. The first is to do a search using a text-based query. It is trivial to make a text-based query on 3D models tagged with annotations. However, relevant findings may not be produced in many circumstances[3]. It also necessitates labeling all of the repository's data. Labeling may be both limiting and ambiguous. Latter, content-based search is described as gathering shapes with similar properties with using features defined on 3D models and a wide variety of features can be used as shape descriptors. Furthermore, 2D techniques being applied in 3D shape retrieval challenges. Using 2D simple sketches,

3D models can be mapped and thus used in shape retrieval applications [4].The last one is defined as example-based approaches. This method brings the closest shapes to the questioned shape by querying a given sample shape. This approach determines and evaluates the similarities between two shapes by measuring them using a predetermined procedure. As with the content-based technique, it's commonly calculated utilizing a global feature that may describe 3D models. Alternatively, it may be preferable to search for a feature on the sample in the query that is able to distinguish it from other sorts of samples in the 3D model repository. In this scenario, classification methods such as multi-class classification utilizing machine learning and deep learning techniques as well as approaches recommended in other domains can be used to retrieve related shapes.

2.3 Classification

The process of assigning an object class label to a group of points is known as object categorization. There are two challenges: determining which points belong to the same item and determining the entity's class. Both difficulties can be approached in two ways.[2] The first method involves segmenting the point cloud into clusters of points that are most likely to be associated with the same item. The clusters may then be categorized to identify the object's class after segmentation (object classification). The second method provides a class label to each individual point (point-wise classification or semantic segmentation) before grouping close points with the same label. Individual point classification is also difficult. Because data from a variety of sources is used to classify things, classification algorithms perform effectively.

Since data from points across the whole object cluster is used, classification techniques function effectively for objects. Data from a single point can be utilized for segmentation or point-wise classification, but this is generally insufficient for good prediction. Point-by-point categorization might be improved with more information from nearby points. The concept of a neighborhood, as well as the procedure for effectively selecting neighboring points, are not simple.

2.4 Machine Learning

Machine learning is a branch of artificial intelligence concerned with the creation, analysis, and development of algorithms and techniques that enable machines to learn from data. It develops software that can generalize behavior based on patterns or classifications. It has something to do with statistics, but it also has something to do with model building approaches and statistical learning. Natural language processing, search algorithms, medical diagnosis, bioinformatics, fraud detection and classification are some of the domains where this sort of learning has been used.

Any intelligent system must have the ability to learn, which means that it must be able to develop over time.

The programs employed are learning systems capable of accumulating high-level information and problem-solving strategies through the use of examples, similar to how the human mind does it.

2.5 Deep Learning

Deep learning is a data-driven research paradigm that has exploded in significantly in recent years, achieving considerable success in a variety of artificial intelligence subfields. Deep learning is, at its core, a branch of machine learning that refers to a set of issues and strategies for solving them.

Artificial intelligence is defined as systems or machines that perform various tasks similar to human intelligence and constantly improve themselves. Since there are systems that can learn from artificial intelligence mistakes that emerged in the 1950s, the system is constantly improving. Machine learning, on the other hand, emerged in the 1980s and is to process a given dataset and make predictions or classify. There are two types of learning in machine learning algorithms: supervised and unsupervised learning.

Supervised Learning involves using labeled datasets with inputs and expected outputs. When training an AI using supervised learning, an input is given to it and the expected

output is said. If the output produced by AI is incorrect, it readjusts its calculations. This process is repeated over the dataset until the AI minimizes the error rate. An example of supervised learning is weather-defining Artificial Intelligence. Learns to predict the weather using historical data. These training data include inputs (pressure, humidity, wind speed) and outputs (temperature).

Unsupervised Learning is the task of machine learning that uses datasets with no specific structure. If it trains an AI using unsupervised learning, it allows AI to logically classify data. An example of unsupervised learning is artificial intelligence that makes predictions for an e-commerce website. Because here it is not learned using a labeled input and output dataset. Instead, it will create its own classification using input data. It will tell you which type of users can buy more different items.

Deep learning became popular in the field of computer vision, particularly for image classification tasks, and convolutional neural networks (CNN) became a household word.

2.5.1 Artificial Neural Network

Artificial neural networks (ANNs) are computer systems developed with the aim of automatically performing the abilities of the human brain, such as deriving new information, creating and discovering new information through learning, without any assistance.

Artificial neural networks have emerged as a result of mathematical modeling of the learning process by taking the human brain as an example. It mimics the structure of biological neural networks in the brain and their ability to learn, remember and generalize. Learning process in artificial neural networks is carried out using examples. During learning, input and output information is given and rules are set.

Artificial Neural Networks consist of many cells and these cells work simultaneously to perform complex tasks. They have the ability to learn and can learn with different learning algorithms. They can produce results (information) for unseen outputs. There is unsupervised learning. They can make pattern recognition, classification and complete the missing patterns. They have fault tolerance. They can work with

incomplete or unclear information. In faulty conditions, they show graceful degradation. They can work in parallel and process real-time information. Artificial neural networks are mainly used in areas such as diagnosis, classification, prediction, control, data association, data filtering and interpretation. It is necessary to compare the properties of the networks with the properties of the problems in order to determine which mesh is more suitable for which problem.

Unlike other algorithms, neural networks cannot be programmed directly with deep learning. Rather, just like a child's developing brain, they need to learn information. Learning strategies are implemented in three ways: Supervised learning: This learning strategy is the simplest as the computer has a dataset that it goes through and the algorithm is modified until it processes the dataset to get the desired result. Unsupervised learning: This strategy is used when there is no dataset available to learn. The neural network analyzes the dataset and tells the neural network how far the target is. The neural network is then adjusted to increase the accuracy of the algorithm. Reinforced learning: In this algorithm, the neural network is augmented for positive results and the probability of negative results is low.

2.6 GAN

GAN is made up of two independent networks: a Generator and a Discriminator, as presented by Goodfellow et al.[5]. This generator generates fictitious data samples in an attempt to deceive the discriminator. The Discriminator, on the other hand, aims to distinguish between real and fake samples. The discriminator and the generator are both multilayer perceptrons that compete in the training phase. The phases are repeated several times, and the generator and discriminator improve in their respective responsibilities with each iteration. The Generator makes synthetic samples from random noise (chosen from a latent space), whereas the Discriminator is a binary classifier that determines if the input sample is real (output a scalar value 1) or fake (output a scalar value 0). The Discriminator strives to be the finest in its field. When the Discriminator is fed with a synthetic sample (produced by the Generator), it wants to identify it as false, but the Generator wants to create samples that the Discriminator recognizes as real. The Generator is, in a sense, aiming to trick the Discriminator. In

this structure, the Generator and the Discriminator are hostile.

GANs are min-max systems in which two algorithms compete: the generator produces data, while the discriminator distinguishes between false and genuine data. The discriminator's goal is to decrease the discriminator error, whereas the generator's goal is to maximize it. This is an iterative procedure that finishes when the discriminator error, or baseline error in bi-classification, reaches equilibrium.

Generator evolves into a model that attempts to create data that is as close to genuine data as possible. It does it by using a random latent vector basis to generate data that is the same size as the original data and has the same values. During the training process, it tries to learn the distribution of actual data, and as a consequence of proper training in optimal conditions, a model is developed that can produce data suited for the distribution of real data.

The Discriminator, on the other hand, is assigned to a separate task. The Discriminator accepts a piece of data in the dimensions of actual data as input which consists both the genuine training data and the fake data generated by the Generator. The Discriminator's job is to figure out if the incoming input is legitimate training data or fake data generated by a Generator.

2.6.1 Conditional GAN

Conditional GAN is a generative adversarial network introduced in 2014 by University of Montreal PhD student Mehdi Mirza and Flickr AI architect Simon Osindero. It is a generative adversarial network in which the Generator and Discriminator are conditioned during training using some additional information. [22] In principle, this auxiliary data may be anything, including a class designation, a series of tags, or even a written explanation.

The Generator learns to generate realistic instances for each label in the training dataset, while the Discriminator learns to discriminate false example-label pairs from actual example-label pairs during CGAN training. The Discriminator in a CGAN does not learn to determine which class is which, like the Semi-Supervised GAN from the previous chapter, whose Discriminator learns to separate real instances from

fake ones.

2.6.2 Auxiliary Classifier GAN

The fundamental GAN model's training has been enhanced in AC-GAN. Generator generates fake samples using random points from a latent space as input. Discriminator distinguishes between actual (dataset) and false (generated) pictures and predicts the class label. Instead of one parameter, the generator is given two. It takes as input random points from the latent space and a class label, then attempts to build a picture for that class. The addition of the class label as an input makes the picture production and classification processes reliant on it, thus the term. The training process becomes more reliable with this Generator model, and it can be used to create pictures of a given kind using the class label.

2.6.3 Mode Collapse

The generator may collapse to a configuration where it always generates the same outputs throughout training. This is known as Mode Collapse and is a typical failure situation for GANs. Despite the fact that the generator may be able to fool the discriminator, it is unable to learn to represent the complicated real-world data distribution and becomes trapped in a narrow space with very little variability.

2.6.4 Wasserstein GAN

The Wasserstein Generative Adversarial Network, or Wasserstein GAN, is a generative adversarial network extension that enhances model stability and gives a loss function that corresponds with produced picture quality. It produces more stable training with less indications of mode collapse than original GANs, as well as informative curves for debugging and finding hyperparameters. In Wasserstein, the distance between two probability distributions is measured in distance. It's also known as the Earth Mover's distance, or EM distance, since it may be understood as the least amount of energy required to move and change a mound of dirt in the shape

of one probability distribution into the shape of another. The cost is calculated by multiplying the amount of dirt transported by the distance traveled.

2.6.5 WGAN-GP

While the original Wasserstein GAN increases training stability, it still produces bad samples or fails to converge in some circumstances. The fundamental problem with WGAN is the weight clipping approach used to impose Lipschitz continuity on the critic. To guarantee Lipschitz continuity, WGAN-GP substitutes weight clipping with a restriction on the critic's gradient norm. This enables for more stable network training than WGAN and only requires little hyper-parameter adjustment. WGAN-GP is extension of Wasserstein GAN.

CHAPTER 3

LITERATURE SEARCH

Sensors such as Kinect and LIDAR have recently made it feasible to quickly and easily create 3D models of interior and outdoor settings. As a result, the computer vision and robotics groups have placed a premium on retrieving useful information from 3D models. Segmentation and classification difficulties have been addressed as an active subject to achieve this. Segmentation is the process of grouping points based on their properties, whereas classification is the process of assigning points to classes. For these challenges, a variety of approaches have been presented, and Grilli et al. [6] examine some of the more common algorithms. Edge-based [18], region growth [18], model fitting [7] [8], hybrid approach [9], and machine learning are five separate types.

The first method involves segmenting the point cloud into clusters of points that are most likely to be associated with the same item. The clusters may then be categorized to identify the object's class after segmentation (object classification). Both segmentation and object categorization are difficult with this method. On data sets with fully divided objects. This demonstrates that segmentation is the most difficult aspect of this method.[6]

In traditional machine learning, descriptors that are relevant to the 3D model's properties are first selected, and then the attributes are acquired. The point cloud is split into relevant pieces based on these characteristics. This method's drawback is that it is heavily reliant on descriptors and is unsuitable for complicated data [10]. Furthermore, because 3D descriptors are very high dimensional, it tends to over-fit [11].

Qi et al.[14], indicate in their study that manually designed feature development or

selection is difficult and application-dependent. Learning techniques based on such characteristics are confined to the capacities of those features to represent the original data, not to the original data itself. This imposes an unnecessarily restrictive constraint on learning algorithms.

Socher et al. [17] suggested one of the earliest uses of CNN in the construction of 3D form descriptors. A 3D object classifier is created in this application utilizing a mix of CNN and recurrent neural networks. In further detail, RGB-D pictures (representing 3D data) are first fed into a CNN, which extracts low-level characteristics like edges. The features are then sent into a recurrent neural network, which is trained to learn combinational higher level features and their relationships. This is done by translating the input into a lower-dimensional space, which produces the shape descriptors that are utilized for object categorization. In contrast to other approaches such as support vector machine and random forests, they found that their quick combinatorial model performed better.

Wu et al. [19] presented another use of DBNs on 3D data. They turned the unstructured input data into a volumetric representation to be utilized as input in their technique, rather than utilizing a view-based approach (approaches that employ 2D/2.5D pictures as input). A convolutional DBN is used in this innovative application to represent a 3D model as a probability distribution of binary variables on a 3D voxel grid. Their method, 3DShapeNets, gets a 2.5D depth picture as input and constructs a volumetric structure in which the shape distribution is learnt. It is the first use of deep learning on 3D data. They express the shape in the form of a 3D matrix with a size of 30x30x30. Each member of this matrix is set to 1 or 0 depending on whether the relevant voxel was within or outside of the mesh. Following that, many tiers of filters are convolved with the input, resulting in the computation of a 1D vector. The weight sharing characteristic of convolution is used in their suggested design to limit the number of parameters to be learnt; however, this structure does not include pooling levels to prevent excessive uncertainty in form reconstruction. Finally, the 1D vector is delivered to the top layer of a learning architecture, which is created as an RBM with 4000 hidden units and includes a class label. This layer made an associative memory that learns the distribution of the binary values mentioned above. Such a distribution is used as a global descriptor for the input shape. The proposed descriptor

was evaluated in classification and retrieval applications.

Han et al. [20] introduced a local shape descriptor termed circle convolutional RBM, which combined the benefits of the DBN in unsupervised learning and the advantages of convolutional networks to propose a circle convolutional RBM. Bu et al. [21] reported work that was innovative in terms of being applied to 3D meshes, but it was hampered by the need to compute local features and use them as input to the learning method. Han et al., on the other hand, suggest using 3D models directly as input for a DBN method. Because of the uneven architecture of 3D models, applying convolution to them is more difficult than applying it to 2D photos. Han et al. proposed a circular convolution to solve the problem, in which a sector window is placed on a vertex and is rotated around the vertex's normal vector by a stride angle in a given direction to compute the convolution. The local area is projected onto the tangent plane perpendicular to the normal vector of the centre vertex while the convolution is being calculated.

Su et al. [11] used a two-layer CNN system to create a 3D form descriptor from 2D pictures in a state-of-the-art technique. Multiple 2D pictures produced from a 3D model are used to construct the suggested descriptor [11]. A CNN hierarchy is used to construct a compact 3D form descriptor from a series of 2D images. Each CNN at the first level collects information from a single view input picture to produce a view descriptor. The higher-level CNN then learns how to combine all of this data into a single shape description. Rather than just averaging or concatenating the numerous descriptors, this research use a CNN architecture to merge them. To train their system for classification and retrieval applications, the researchers employed several 2D views of a single 3D model. When compared to ShapeNets [19], their suggested system performs better in retrieval applications. In addition, evaluating the system for classification applications using a single view resulted in a ShapeNets improvement.

VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection[12] described one method for classifying objects in point clouds. The researchers used point clouds to build voxels, which were then run through an ANN. A LIDAR is a faster means of obtaining a point cloud. The study that conducted by Caltagirone et. al. [13] illustrates how CNN may be used to directly identify a road using LIDAR data as in-

put. The approach utilized here, however, does not contain any detection of moving things in a traffic environment, such as vehicles and people, but it does demonstrate that LIDAR data may be used in a CNN.

Hybrid systems, such as Multi-view Convolutional Neural Networks for 3D Shape Recognition[11], have also been proposed. The goal of that article was to transform 3D point clouds into 2D pictures and train a neural network to recognize things in those images. One disadvantage of this strategy is that it eliminates the benefits of using 3D data in the first place. This includes tasks like point categorization, shape completion, and scene comprehension [14], [15], all of which can help an automobile comprehend its environment. Object identification directly from point clouds has been the subject of a few articles. In PointNet [14], researchers trained a neural network to recognize an item based on the raw data points gathered.

Anguelov [16] proposed that a 3D point cloud segmentation method should have three key characteristics. To begin, the algorithm should be able to take use of a variety of qualitatively distinct types of characteristics, such as the differences between trees and vehicles. When the number of characteristics increases, the segmentation algorithm should be able to automatically learn how to trade them off. Second, depending on the information of their neighbors, a segmentation algorithm should be able to deduce the label of points in poorly sampled regions. Third, the segmentation method should be tailored to the 3D scanner in use, because various laser scanners create qualitatively distinct point cloud data, and even within the same scene, they may have different qualities.

Traditional convolutional neural networks (CNNs) are extended to accommodate data that is supported on a graph by graph convolutional neural networks (Graph-CNNs)[33]. The fact that the support set does not always have a natural ordering and that the graph topology is not always regular. These are two major obstacles when working with data on graphs. As a result, Graph-CNNs offer a lot of promise for dealing with 3D point cloud data produced by sampling a manifold. In this study, Zhang et. al. provide PointGCN, a GraphCNN for categorizing 3D point cloud data. A graph is formed using k-nearest neighbors from a point cloud and each edge is weighted using a Gaussian kernel in PointGCN. Chebyshev polynomials in the graph spectral domain

are used to create convolutional filters. To capture global and local properties of the point cloud, global pooling and multi resolution pooling are utilized.

The goal of most of these systems is to categorize aerial point clouds into meaningful semantic classes, such as ground level objects, vegetation, building facades, and building roofs, since they[34] are based on point clouds with correct semantic classes. They used three deep learning algorithms and one machine learning algorithm to test and evaluate various machine learning methods for classification in this study. Several hand-crafted geometric characteristics are employed in the experiments, depending on the dataset, and these geometric features are also used for deep learning, which is unusual.

Anguelov [35] propose an octree grouping-based network structure for PointNet++, named OctreeGrouping-PointNet++ (OG-PointNet++). It calculates the point density by creating an imbalanced octree for the point cloud and grouping points based on the density. These point groups are allocated to different layers based on their density, then PointNet++ extracts the local characteristic of each group. The last abstract layer yields the global feature, which is utilized for classification and segmentation. Experiments demonstrate that it performs well in a variety of 3D tasks, including object categorization and semantic segmentation.

Anguelov [37] offer PointAugment, a new auto-augmentation technology that optimizes and augments point cloud samples automatically to enrich data variety when training a classification network. PointAugment is a sample-aware auto-augmentation approach for 2D pictures that uses an adversarial learning strategy to jointly optimize an augmentor network and a classifier network, allowing the augmentor to learn to create enhanced samples that best suit the classifier. They also build loss functions to adopt the augmented samples based on the classifier's learning progress and create a learnable point augmentation function with a shape-wise transformation and a point-wise displacement. Extensive testing has also confirmed PointAugment's ability to increase the performance of diverse networks in shape categorization and retrieval.

PointNet[14] takes input samples from a point cloud and outputs class labels for each point in the input for point-wise classification (or semantic segmentation). The PointNet study also presents analogous designs for Object classification and Object-part

segmentation, in addition to point-wise classification.

A 2D matrix with x number of points is used as the input for PointNet. The point cloud data was divided into grid cells of 1 meter by 1 meter. A training sample is a set of points in a grid cell. The number of points in a training sample is regulated to keep the input dimensions consistent. The number of points is a free choice employ 4096 points per 1 meter by 1 meter sample for the indoor data collection S3DIS.

Qi et. al. improve the PointNet approach with PointNet++ [44] taking into consideration local features. They proposed a hierarchical structure that employes PointNet recursively on nested parts of the input.

In VoxNet[42], Maturana et. al. proposed a method integrating volumetric Occupancy Grid representation with a supervised 3D Convolutional Neural Network (3D CNN) in order to cope with large amounts of point cloud data. They trained their proposed network on LIDAR point-cloud data, RGB-D and modelnet datasets. Thus, they support different sources in the manner of 3D data production.

Shi et. al. [43], converts 3D shapes into a panoramic view with respect to their principal axes in DeepPano in order to solve the 3D Shape Recognition problem. Thus, suggested deep CNN becomes invariant to the rotation around the principle axis for 3D shape queried. They examined their methods on the Modelnet10 and Modelnet40 datasets.

Kumawat et al. propose a new block as an alternative to 3D Convolutional networks in their LP-3DCNN work [45] in order to increase the learning capacity of CNN networks and to overcome the problems caused by CNN networks such as large model size, computational and space complexity etc. This block aims to extract feature maps by evaluating 3D local neighborhoods. While testing their proposed method on volumetric data, they use voxels in their networks and conducted experiments with their method on classification task of modelnet10 and modelnet40 datasets.

Khan et. al. developed a pipeline for generative model in order to generate 3D Shapes and that pipeline consists of sequential transformations from coarse to fine [46]. Proposed method employs primitive parts as features and improves itself by adding fine scale details. As a way of demonstrating the learning power of their proposed gener-

ative model, they chose to run experiments using it as a feature in the classification task.

In order to solve classification of point clouds problem, Sheng Xu et. al. proposed new augmentation CNN at their study [47]. The suggested method inserts an augmentation layer before the sampling and convolutional layers. They aims to capture local information such as edges and contours from augmented input data before they are sampled. Proposed method projects provided point clouds into 2D space at different views in the augmentation phase and then creates input data as a fixed size square images which are also normalized and smoothed.

Wang et. al. proposed a neural network module for point cloud classification and segmentation in their work [48] named Edge Conv. It is possible to use this differentiable module in existing models. In that study, where they aim to benefit from the local neighborhood, global features also be able to be learned and some semantic features can be revealed. Even the spirit of the PointNet lives in the proposed study, Wang et. al. focusing on the construction of local neighborhood graph and applying operations on the edges of those neighbor pairs. Their graph is updated after each layer unlike graph CNNs. They evaluated their model in classification, segmentation and semantic segmentation tasks.

Huang et. al. proposed Spatio-temporal Self-Supervised Representation method that be able to learn scene understanding for 3D Scenes from unlabeled 3D point clouds in [49]. They work with two 3D point cloud frames as a sequence input and applies spatial data augmentation in order to reveal invariant representation. Their empirical works are on different datasets and different tasks which one of them is 3D Shape classification. They employed modelnet40 dataset for their experiments and worked with point cloud on proposed self-supervised model.

Song et. al. proposed method for 3D point cloud classification without need of preprocessing in their work [50]. They employed multiple 2D convolutional layers for coordinates of points and applies max pooling layer like PointNet in order to build global features for shape. Proposed Pointwise CNN was subject to classification task on Modelnet10 dataset during their experiments.

CHAPTER 4

METHODOLOGY

In order to better express the proposed methodology, we must first describe how GAN models may be utilized as classifiers or tools for classification tasks. Vanilla GAN, as it is known, is a machine learning method that enables the development of both models by taking advantage of the conflict between two different networks, which allows the data generated by the Generator to be classified as real or fake by the Discriminator. While Discriminator specializes in detecting whether the incoming data is real or fake, it provides better generation of fake data produced by Generator. Thus, the Generator is refining itself to provide more realistic data after failing to trick the evolving Discriminator. As a result of the training, we have a Generator that capable of producing data that is closer to the real data set, and a Discriminator that learns to distinguish between real and fake data. Discriminator, Generator, and GAN combined loss equations are defined in Equation 4.1, respectively.

$$\begin{aligned} L_{(D)} &= \max[\log(D(x)) + \log(1 - D(G(z))] \\ L_{(G)} &= \min[\log(D(x)) + \log(1 - D(G(z))] \\ L_{(C)} &= \min_{G}\max_{D}[\log(D(x)) + \log(1 - D(G(z))] \end{aligned} \tag{4.1}$$

In fact, this Discriminator we obtained functions as a binary classifier by nature. In other words, there is already a classification process on the basis of GAN models. This design-based classification capability of GAN networks inspires the development of multi-class classification versions of this method. In Conditional GAN, in addition to Vanilla GAN, the Generator is required to generate condition-specific data with a given condition set. Thus, the discriminator processes the fake data generated under specified conditions together with the condition provided. The Auxiliary Clas-

sifier GAN, on the other hand, extends and modifies this approach by using class label information as a condition on both fake and real data produced by the Generator and allowing the Discriminator to predict which class the sample belongs to in addition to determining the sample's realness or fakeness. In ACGAN, the Discriminator needs to make classification in addition to real/fake discrimination, and these two evaluations contribute to the Discriminator loss, which is tried to be minimized. This makes the problem more challenging than vanilla GAN. Equation 4.2 shows how the Discriminator loss function of ACGAN differs from the Vanilla GAN loss by splitting it into adversarial and classification losses.

$$\begin{aligned} L_{adv} &= E_x[\log D(x)] + E_{\hat{x}}[\log(1 - D(\hat{x}))] \\ L_{ac} &= E_x[\log p(c|x))] + E_{\hat{x}}[\log(1 - p(c|\hat{x}))] \\ L_D &= \frac{W_{adv} * L_{adv} + W_{ac} * L_{ac}}{W_{adv} + W_{ac}} \end{aligned} \tag{4.2}$$

In addition, as the number of classes increases, it becomes more difficult for the Generator to produce data close to the real data. M. Kang et. al. is also defining this problem in their research and expressing early collapse problem [31]. Furthermore, Cho et. al. proposes Conditional Activation GAN (CAGAN) method which has multiple GANs with shared hidden layers and their integration considered as a single GAN [32].

In ACGAN method, as specified in Equation 4.2, both adversarial and classification loss affect the loss of the Discriminator together with certain weights and determining these weights can be challenging. Bazrafkan et. al. propose the Versatile Auxiliary Classifier GAN[36] to remove the classification operation from the objective of the Discriminator and this method performs labeling with a parallel network. Figure 4.1 illustrates Vanilla GAN, CGAN, ACGAN and VACGAN, respectively.

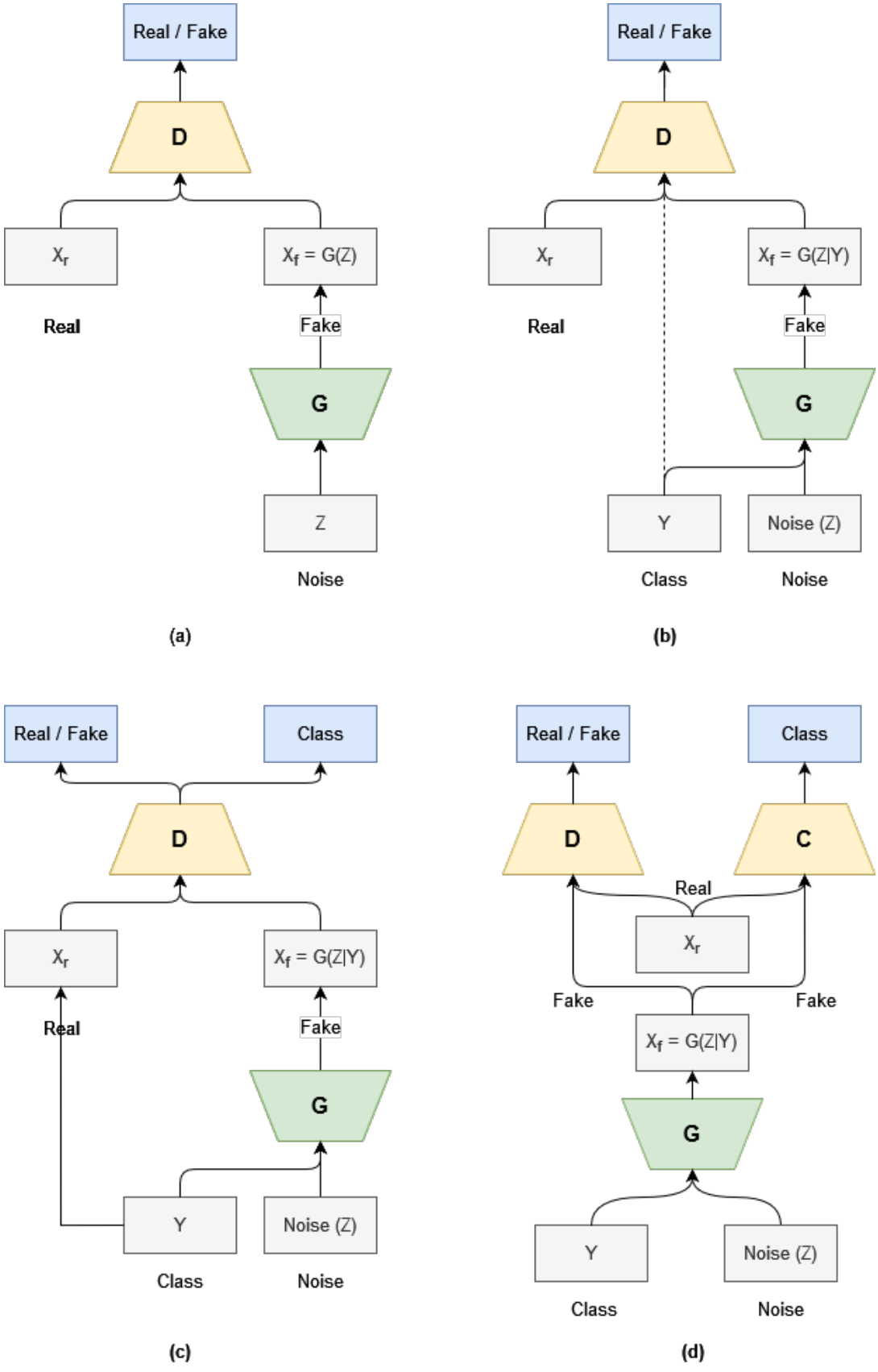

Figure 4.1: Network structures of known GAN models

(a): Vanilla GAN, (b): CGAN, (c): ACGAN, (d): VACGAN

The frustration associated with training GAN models is a well-known issue and is always a hot topic. Despite the differences in the selected network structure and GAN type, it is possible to suffer from the mode collapse problem. TThe primer and the most important cause of this issue lies in the design of the GAN. In the min-max game played, either it takes too long for two different networks to try to fool each other until they reach the Nash equilibrium level or they cannot reach that point at all. In this circumstance, the adversarial loss defined for the model and as well as the algorithms based on gradient descent that attempt to reduce that loss play a significant role. Although utilizing Deep Convolutional architectures instead of MLP layers in the generator and discriminator networks improves GAN performance, it is insufficient on its own. This type of GAN is commonly referred to as DCGAN and was first described by Radford et al. [38]

It is also challenging to check the output success of the Generator, which is a generative model and therefore performs unsupervised learning, and although it is customized according to the problem, visual quality or classification accuracy is generally preferred. Even though the outcomes appear to be acceptable, outputs that do not match the data's overall structure and distribution may be observed and vice versa.

In order to overcome the GAN problems mentioned above, especially to avoid the mode collapse problem, Arjovsky et. al. proposed a powerful approach named Wassertein GAN[37]. For a more stable and robust training, they define a distance measure called Wassterstein or Earth's Movers Distance. Introduced this loss is addressing to solve the mode collapse problem. Instead of a binary output like real / fake, they replace the discriminator with a network that provides a "realness/fakeness score" and rename it as critic. Since this network becomes a critic rather than a discriminator. The distance between the distribution of samples in a real dataset and the distribution of samples generated by the generator is measured by proposed distance. Earth Mover Distance, which is differentiable and continuous, ensures that the critic can be trained for a longer time, while discriminator converges rapidly, thus a reliable gradient can be obtained during training[37]. It advises clipping the "weights of the critic model" with each mini-batch update, in addition to using Wasserstein loss during training. It is also proposed that the critic should be trained more than the generator, suggested 5 times, and employing the RMSprop gradient descent optimizer

with a low learning rate is motivated. While Figure 4.2 shows the general structure of WGAN, Equation 4.3 defines the loss for generator and critic respectively. The cost function of WGAN is defined in Equation 4.4 where f is 1-Lipschitz function.

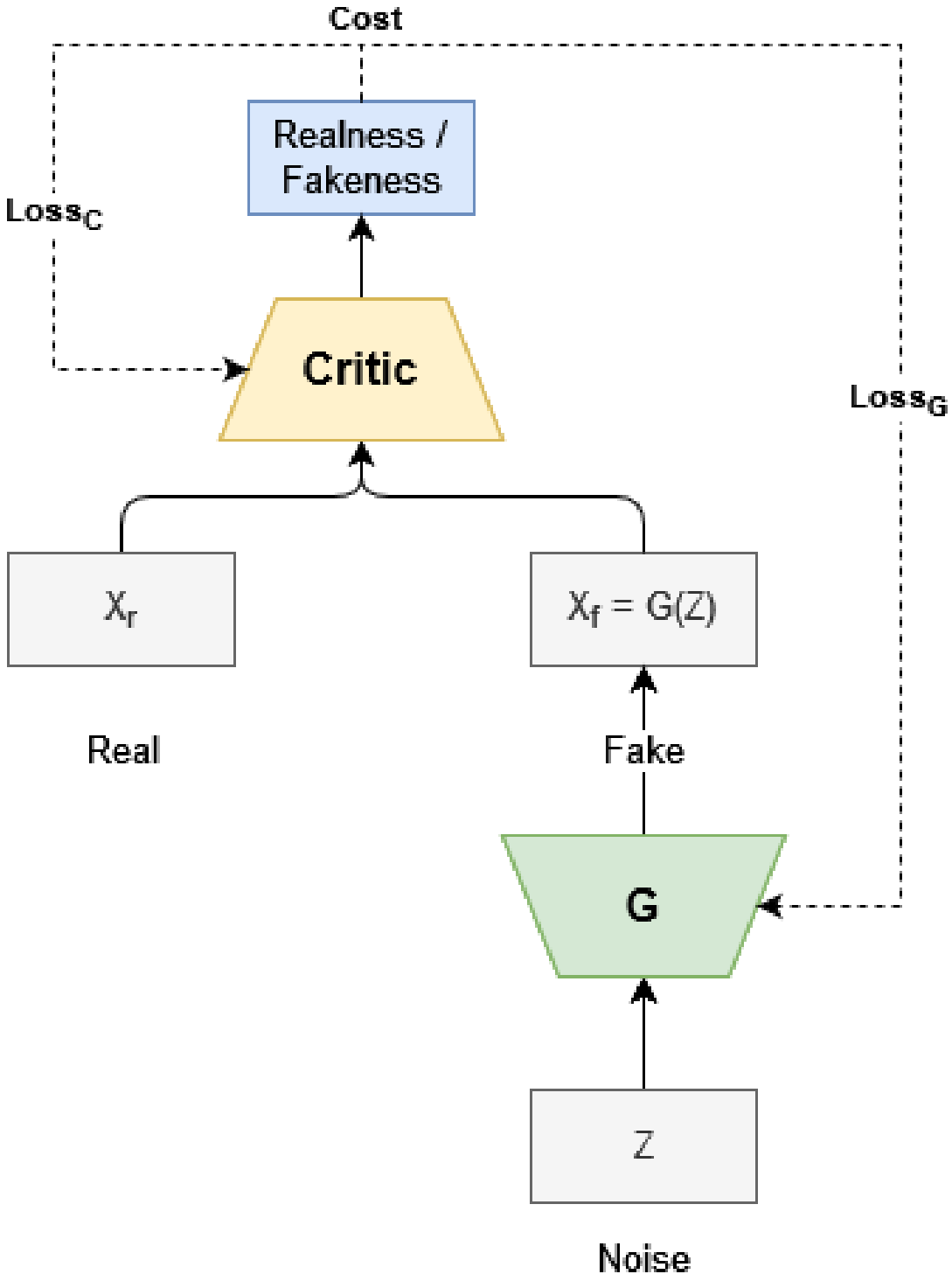

Figure 4.2: WGAN Network architecture

$$
\begin{aligned}
Loss_G &= \nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} [\log D(x^{(i)}) + \log (D(G(z^{(i)})] \\
Loss_C &= \nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log (D(G(z^{(i)})
\end{aligned}
\tag{4.3}
$$

$$
\min_{G} \max_{||f||_L \leq 1} E[f(x)] - E[f(\hat{x})] \tag{4.4}
$$

Even better generative performance and stable training for GANs are established with WGAN, during the convergence, several unstable circumstances, such as exploding and vanishing gradients, may occur. The fundamental problem with WGAN is the weight clipping approach used to impose Lipschitz continuity on the critic. To guarantee Lipschitz continuity, WGAN-GP method [39] is proposed by Gulrajani et. al. which substitutes weight clipping with a restriction on the critic's gradient norm. While WGAN with weight clipping approaches leading to learning simple functions, Gradient Penalty constraint that the gradients of the critic's output with respect to the inputs to have unit norm. Thus, it offers a more stable training. Equation 4.5 defines WGAN-GP cost with P_x and $P_{\hat{x}}$ which are distribution of real samples and distribution of generated samples respectively. λ is coefficient which used to weighting the penalty.

$$
L = \underset{\hat{x} \sim P_g}{E} [f(\hat{x})] - \underbrace{\underset{x \sim P_r}{E} [f(x)]}_{\text{critic loss}} + \lambda \underbrace{\underset{\hat{x} \sim P_{\hat{x}}}{E} [(||\nabla_{\hat{x}} f(\hat{x})||_2 - 1)^2]}_{\text{gradient penalty}} \tag{4.5}
$$

All these mentioned methods are recommended for working with 2D images, and as a result, they are aimed at producing more reasonable new images. During the training phase, the deep convolutional networks utilized in these models require a large number of samples. Fortunately, there are numerous 2D image sets produced today that are suitable for use in training. 3D mesh sets are more limited than 2D images, despite the fact that their quantity is growing by the day. Because 3D convolution layers are heavily employed, the problem to be addressed gets more complicated and the computing cost increases as the new dimension is added to the input. In addition, while 2D images are expressed and stored in pixels, they can also be easily processed over pixels. In contrast, 3D meshes are held and rendered in a variety of formats, in-

cluding the interconnections of points in 3D space, thus defining surfaces. Two points close to each other in 3D space may not contain a connection, whereas pixels that are adjacent to each other in 2D images may construct a big meaning. For this purpose, similar to pixels in 2D images, 3D meshes can be expressed with voxels for ease of data manipulation. 3D meshes can be expressed in low-resolution and sharp-edged forms using the voxel information defined at the corners of the 3D space by dividing it into grids. However, this is a form that differs significantly from the actual data produced and is generally used to facilitate data processing. In this study, we propose methods for working with point cloud datasets that may be readily extracted from real-world data and even generated directly from various sensors, such as LIDAR, that can currently fit in our pockets. With omitting the connection information in 3D meshes, we define 3D meshes as $nx3$ data in a more lightweight way and use the x, y, and z coordinates of the points as features where n is the sampling size chosen for the express 3D mesh. Throughout this study, triangle point picking method was applied for sampling from 3D mesh models and experiments were carried out with different sampling sizes. [1]

In this work, inspired by the methods described above, we propose two different approaches for classification on the 3D Point cloud and compare our results with the well-known study, PointNet[14]. The first method we propose includes an adapted version of the ACGAN model designed for 2D images for point cloud classification. We rearrange the network layers of the Discriminator and Generator models to dealing with our problem. In the network structure that we have keep up to be faithful to the original ACGAN topology in Figure 4.1.c, we leave the W_{adv} and W_{ac} weights in Equation 4.2 equal to each other. Latter, we propose another method to perform the classification task which we are based on the VACGAN method, but here we replace the vanilla GAN part with Wasserstein GAN with GP and named as VACWGAN. Our network layer structure is being updated once again so that our technique can work properly with the 3D Point cloud. The layers of classifier parts of both networks we proposed have a similar approach with the PointNet study with some hyperparameter changes. The architectural structure of the two proposed methods is given in Figure 4.3, respectively.

[1] https://mathworld.wolfram.com/TrianglePointPicking.html

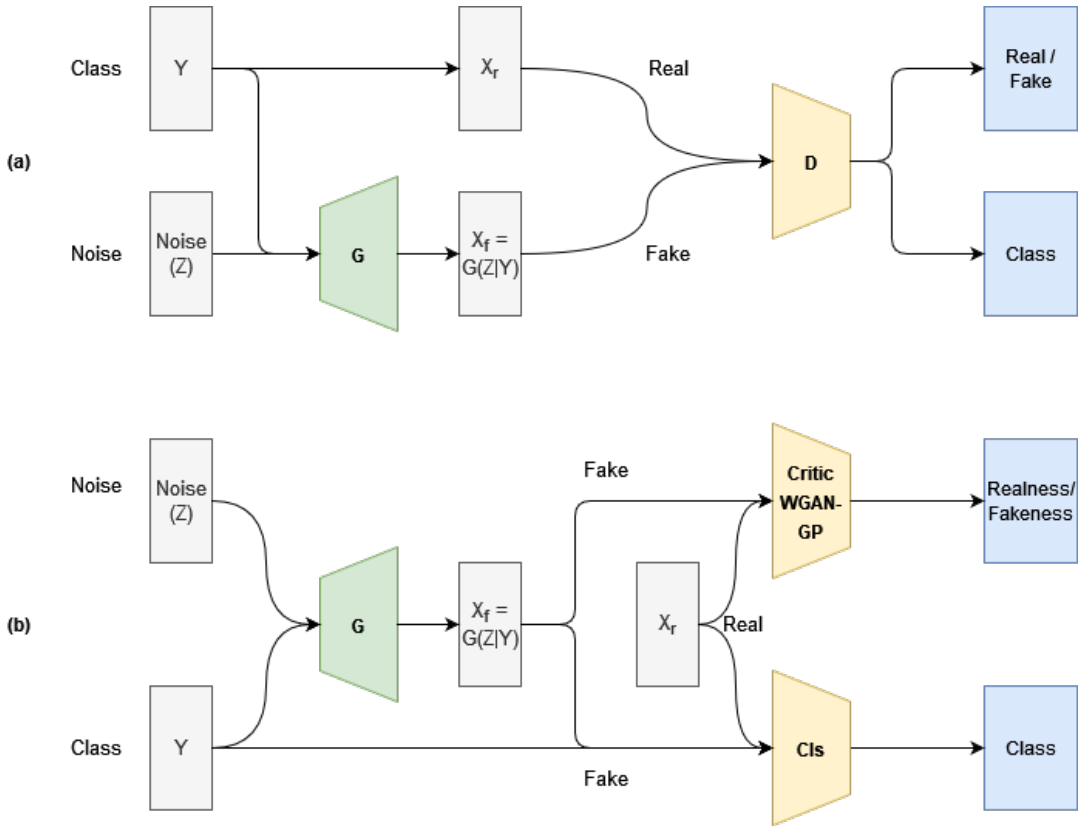

Figure 4.3: Proposed Models: (a) Modified ACGAN, (b) VACWGAN

In GAN models, the Z input is collected from the latent space to feed the Generator and has no real meaning on its own. Using this input, the Generator tries to generate meaningful data as close to real data as possible. If the generated data can be produced well enough, means representing the real dataset well, it undertakes the task of data augmentation. If the relationship between the data is acceptable enough, training through viewing more data produces higher outcomes in terms of classification success. Therefore, the classifier will generalize the data better and give more outstanding results. We propose to apply generative models for data augmentation to increase our classification success based on this claim. For this purpose, in this study, we compare the results of the two methods we proposed with a bare classifier reflects PointNet classification spirit. We choose modified ACGAN to accomplish data augmentation and classification together and improved its classification performance with enhancing VACWGAN with our classifier separately.

Similar to other GAN models, it is possible to encounter mode collapse problem in

ACGAN models. If Discriminator can no longer update itself and gets stuck on local minima, for example, the Generator can trick this Discriminator with generating a small dataset as an output. Generator restricts itself to generate the data set similar with less variation. Different methods are offered to circumvent this situation and Liu et. al. suggest using more than one generator in their study[2]. A possible architectural structure would be as in Figure 4.4. However, increasing the number of generators and training them in certain conditions, in this case conditions are class labels, is not always a feasible, scalable and applicable solution especially when the number of classes increases. We integrate Wassterstein GAN-GP with VACGAN to avoid a potential mode collapse problem and provide more stable training.

Besides the dataset where the original GAN solutions run, in the 3D point cloud, unlike the features on limited grids in 2D images, the set of points scattered in space is unordered and the sample points can be organized in any combination. Therefore, the solution space's resolution is substantially higher. In 2D pictures, for example, when the 28x28x1 picture space is considered, the space of the function learned with the convolutional layers is again stuck in a 28x28x1 space with respect to this neighborhood relationship. In the 3D Point cloud, although the x, y and z coordinates of the generated data are compressed between 0 and 1, the range of values is relatively vast, and alternative orderings are conceivable. This makes the problem challenging. This concern makes difficult to measuring performance visually and in certain situations, this measurement is worthless in relation to the problem at hand. While we are actually doing data augmentation with WGAN, we naturally aim to reproduce the data that is close to the original data set, our main goal is to expand the data set for the classifier with generated fake data that can best reflect the distribution of the original data set. In this regard, WGAN with GP is ideal for this task. It promises to produce the closest data set distributionally to the data set given as input with the help of defined gradient penalty. However, in the original work, this gradient penalty is described for 2D images. In this study, we modify the distance measure defined in WGAN-GP with Chamfer Distance to fit measuring the distance between 3D point clouds. Thus, for the critic, we update the gradient penalty part in the loss objective defined in Equation 4.5 to use Chamfer Distance. Chamfer Distance metric is continuous and be expressed as in Equation 4.6, where S_1 and S_2 are point sets defined in

$\mathbb{R}^3$.

$$d_{CD}(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} ||x - y||_2^2 + \sum_{y \in S_2} \min_{x \in S_1} ||x - y||_2^2 \tag{4.6}$$

The method we propose is similar in spirit to the Multiple Objective Generative Adversarial Active Learning (MO-GAAL)[40] method proposed by Yezheng Liu et al. While their work aims to use Generative models to better define the boundaries of normal data in outlier detection, in our method, the generative model learns the distribution of the data set and produces data close to the real data, thus allowing the classifier to generalize more effectively.

By design, the two networks in the GAN models take a long time to reach equilibrium and the training progresses slowly. In the original WGAN-GP approach, it is recommended to train the critic more than the generator. In the VACWGAN method we recommend, we train the critic five times more than the generator, as in the original WGAN-GP method. In addition, by training our classifier at a rate of 1/5 of the generator's steps, we prevent it from over-fitting during long GAN network training. We do not want fake data generated by the slow-paced GAN model to make the classifier unstable.

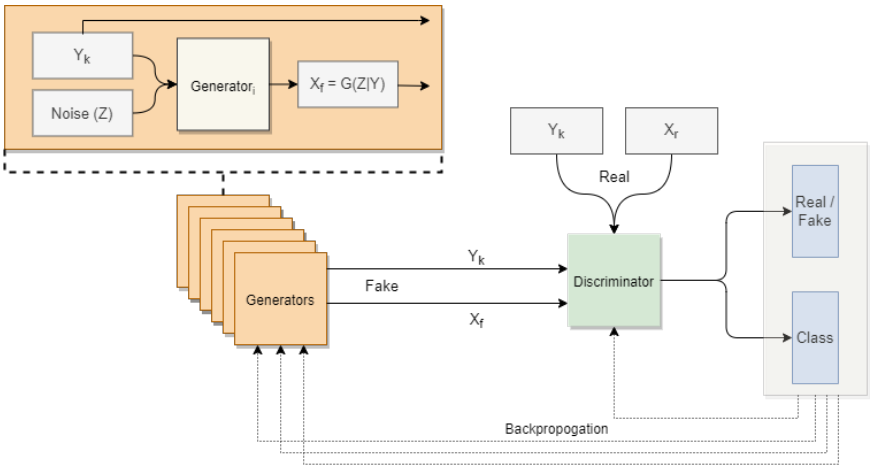

Figure 4.4: Possible Multiple Generator Structure for ACGAN Networks. By multiplexing Generative Networks, it is possible for generators to learn target label specific generation. Although it is not scalable, one generator can be dedicated for each label in the target label space, theoretically.

The structure of both Generator and Discriminator models we used during our experiments to test and evaluate our first method, modified ACGAN, are given in the Figure 4.5, respectively.

The layer structures of the models of the second method which we recommend, namely VACWGAN, are given in Figure 4.6. Note that, batch normalization layers are omitted in the critic since the correlation created between the samples in the same batch by those layers has a bad impact on gradient penalty effectiveness.

Sample point size is given as n and k is the number of labels for classes and latent vector dimension is expressed with L for both models. In the modified ACGAN model, Adam is employed as a gradient optimizer in both Discriminator and Generator, whereas in the suggested VACWGAN model, the RMSprop optimizer is selected for Generator and Critic. Adam optimizer is used by the classifier.

In order to increase the generator performance and indirectly the discriminator or the critic performance, we take this Z input from a Gaussian distribution space with mean 0 variance 1 as suggested in[30]. In addition, during the preprocessing step of training, we scale the input data between -1 and 1 such that each point vector relies in defined boundary cube without no distortion. As a result, we now have a model that is more stable and has converged early.

Lastly, we propose to use the classifiers we have obtained as a result of these two methods in shape retrieval applications. We offer an approach for data labeling for point cloud data detected directly by sensors or for an effective classification for dataset repositories currently maintained as 3D Mesh, without the need for voxelization or any additional data processing. We make it possible to use our proposed method for 3D mesh classification with sampling that can be done cost effectively on 3D meshes. Furthermore, we enable example-based shape retrieval from tagged data set repositories.

In the next section, we will describe our experiments with these proposed methods and discuss their results. We will evaluate and compare the training outcomes of the two different GAN-based models we propose. We will explain the model-based results of the classification task, which is our main goal. We will show whether a

sample point size or latent vector size change has an impact on model performance. We will share the results of the shape retrieval application, where VACWGAN classifier is employed, and we will present the images of the incorrect labeling made during the dataset labeling, together with their scores.

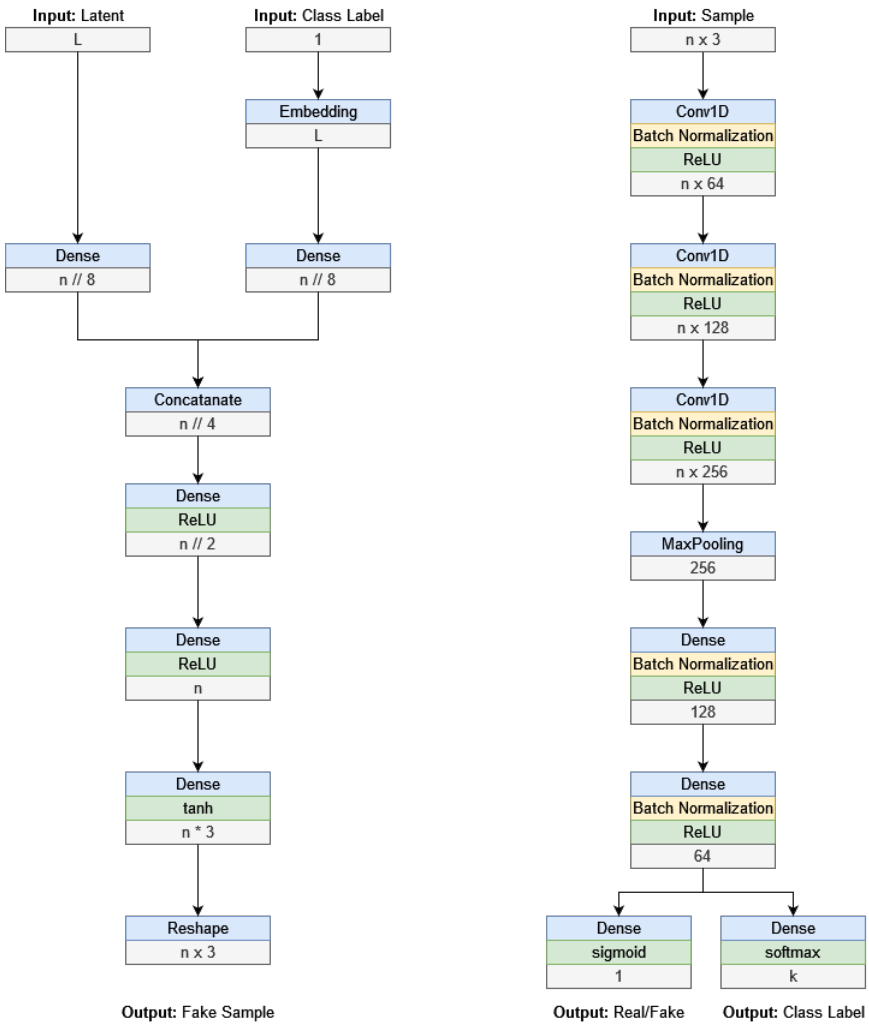

Figure 4.5: Proposed Modified ACGAN Structure with layers
Left: Generator, Right: Discriminator/Classifier
L: Latent vector dimension, n: Point size, k: Target Class size

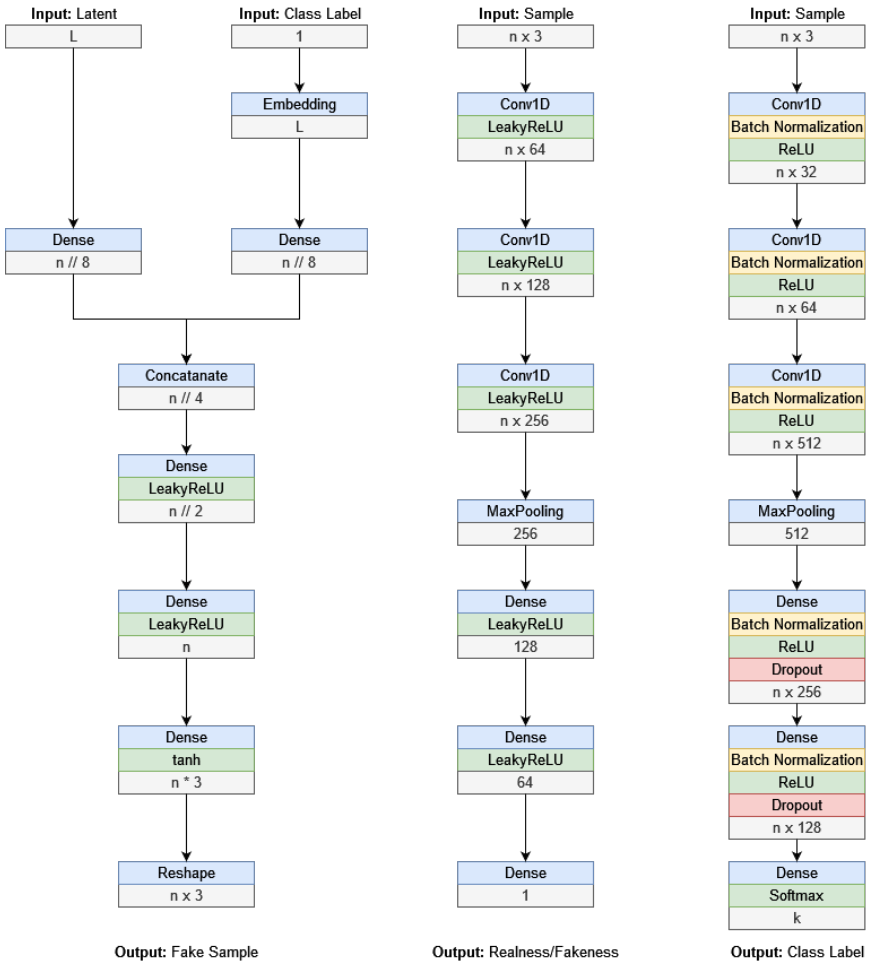

Figure 4.6: Proposed VACWGAN-GP Network Structure with layers
Left: Generator, Middle: Critic, Right: Classifier
L: Latent vector dimension, n: Point size, k: Target Class size

4.1 Evaluation of Proposed Methods

To summarize the development stages of the proposed methods, firstly, the bare network which consists of Conv1D layers was chosen as the classifier, which reflects the spirit of global feature extraction in PointNet. Furthermore, the power of generative models was used for the data augmentation method to increase the operability with smaller data sets and to provide a better generalization. Thus, the use of conditional GAN was recommended to provide conditional data generation and the ACGAN model, which combines the conditional GAN and classifier in the literature,

was preferred. This method which is originally designed to use in the classification in 2D images has been updated to accommodate 3D point clouds and layers have been reworked. With this relatively lightweight solution produced, classification success has been increased by utilizing the power of data augmentation. In order to overcome potential mode collapse problems, the use of Wasserstein GAN is recommended. In order to improve classification result the Classifier is separated from the GAN structure. The vanilla GAN in the VACGAN method in the literature was replaced with the Wasserstein GAN. Then, Gradient Penalty section was added to Wasserstein GAN in order to prevent problems such as Vanishing Gradients and increase the success of classification with better data augmentation. The Chamfer Distance metric has been integrated to the original Gradient Penalty section in order to can work with the 3D point cloud. As a result, the more compact modified ACGAN and the more promising VACWGAN-GP models are recommended for 3D point cloud classification. It was aimed to increase the success of classification with Data Augmentation by taking advantage of the power of generative models.

Figure 4.7 shows encountered mode collapse problem during developing modified ACGAN model. Whereas, Figure 4.8 and Figure 4.9 indicate unstable training encountered during Wasserstein GAN training without Gradient penalty.

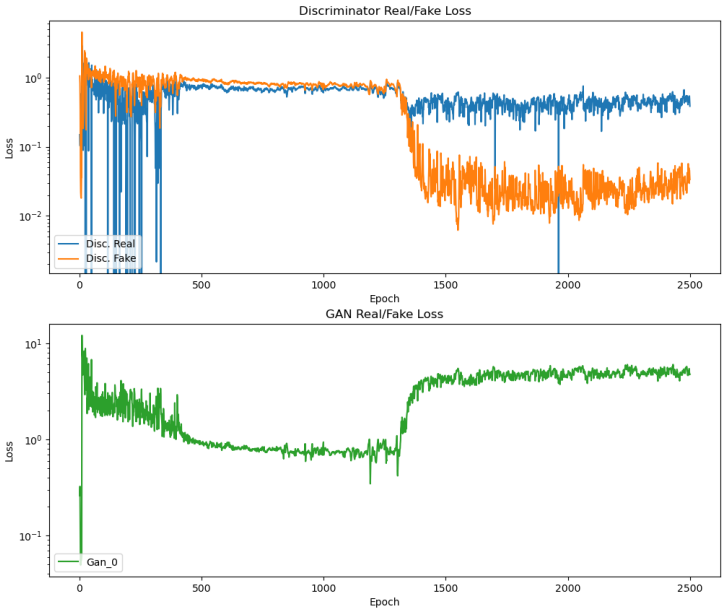

Figure 4.7: Mode collapse State

In GAN models, mode collapse problem is well-known and potential issue during the training phase. Training with optimal parameters is crucial, hard to manage and even sometimes it is impossible to get rid of. Figure shows mode collapse problem encountered at ACGAN

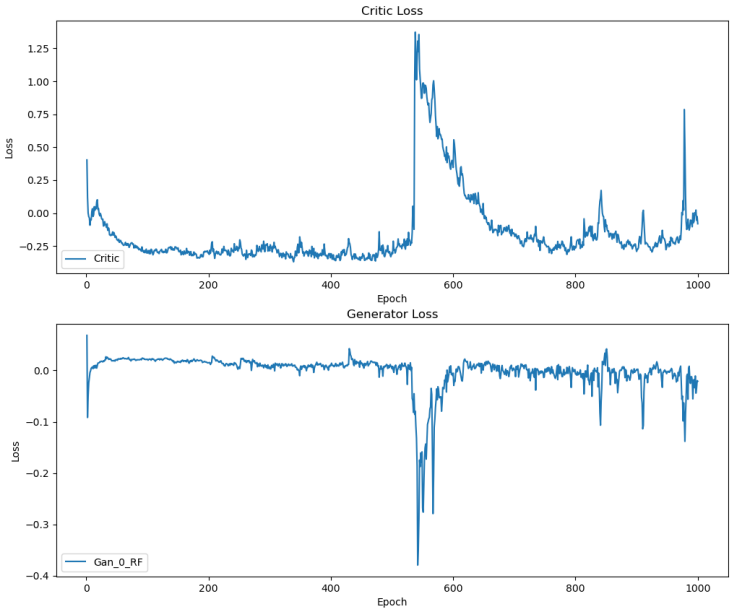

Figure 4.8: Unstable VACWGAN Training

Unstable training is potential problem in Wasserstein GAN models.

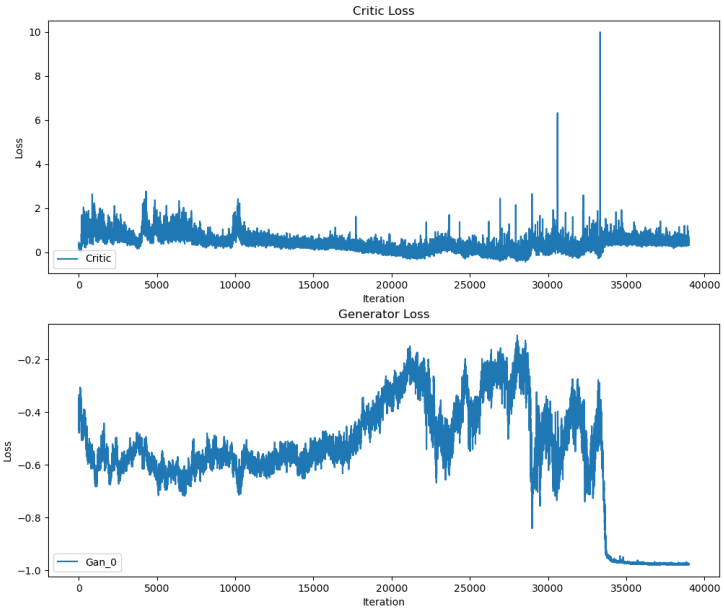

Figure 4.9: Unstable Wasserstein GAN training - 2

Loss for generator stuck at level -1.

CHAPTER 5

EXPERIMENTS

We tested the sufficiency of the proposed models in the 3D point cloud on Modelnet10 and Modelnet40 datasets. While developing our methods, we compared them with our base model as well as other well known approach PointNet. We compared the results of our test with other methods that work on the same dataset. Unless otherwise noted, all tests were carried out on the Modelnet10 dataset since the desired data class set is small and hence easy to explain and depict. Furthermore, the comparisons will be significant due to the widespread use of this dataset in other well-known studies. In addition, the point sample size n is choosen as 2048 per each mesh in the dataset during the experiments. Nvidia GeForce RTX 3090 24Gb graphics card was utilized to train neural networks and takes between 8 hours to a day depending on the epoch desired. This study was coded using Python programming language and Tensorflow and Keras v2.5 libraries were used for Neural Network training. The visualization and rending of 3D models was done with Blender.

5.1 Datasets

5.1.1 Modelnet10

The Modelnet10 dataset contains CAD models created from the most widely used object categories in the world. The samples were manually categorized and affected. In this data provided by Princeton University [1], CAD models' orientation are manually aligned. It is a subset of the Modelnet40 dataset and has 10 different object categories. In the dataset, the train and test set were separated and this split was em-

[1] https://modelnet.cs.princeton.edu/

ployed exactly as specified during the experiments. Table 5.1 provides a summary of the modelnet10 dataset.

Table 5.1: Modelnet10 Dataset Summary

Modelnet10	Bathtub	Bed	Chair	Desk	Dresser	Monitor	Night Stand	Sofa	Table	Toilet	Sum
Train	106	515	889	200	200	465	200	680	392	344	**3991**
Test	50	100	100	86	86	100	86	100	100	100	**908**
Total	**156**	**615**	**989**	**286**	**286**	**565**	**286**	**780**	**492**	**444**	**4899**

Figure 5.1 shows five distinct models chosen at random from the data set and which are rendered with Blender.

Figure 5.1: Randomly selected samples from Modelnet10 dataset for different categories: Bathtub, Bed, Chair, Table and Monitor. Renders are taken via Blender

5.1.2 Modelnet40

The summary of the Modelnet40 dataset, which is the expanded version of the Modelnet10 dataset with 30 different new categories, is given in Table 5.2 together with the Modelnet10 dataset. Similar to Modelnet10, this dataset is also divided into train and test, and this seperation was used during experiments without any changes.

5.1.3 Point Cloud Samples

Figure 5.2 shows five randomly selected examples of 3D point clouds created by sampling Modelnet10 data instances using the Triangle Point Picking method mentioned above. The classes to which the samples belong are: bathtub, bed, chair, sofa and toilet, respectively.

Table 5.2: Datasets Summary

Dataset	Modelnet10	Modelnet40
Categories	10	40
Train	3991	9843
Test	908	2468
Total	**4899**	**12311**

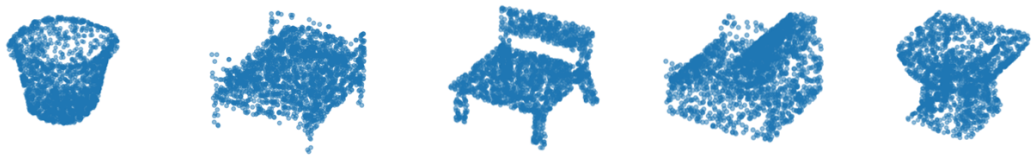

Figure 5.2: 3D Point Cloud Samples gathered from Modelnet10 dataset shapes. 5 meshes selected randomly from dataset and point size for each sample is 2048. Categories for point clouds are: batthub, bed, chair, sofa and toilet, respectively.

5.2 Results

In this section, we share the results of the proposed modified ACGAN and VACWGAN-GP methods during the experiments, respectively. These contain the suggested GAN models' learning-related loss charts. In addition, we provide the results that we use to assess categorization success. We also show examples from mislabeled samples by our proposed classifier while classifying the test set. Furthermore, we are discussing the impact of modifying the size of the latent vector or the number of sampled points on classification performance for both methods.

5.2.1 Modified ACGAN

As mentioned above, the first proposed method, modified ACGAN, consists of one generator and one discriminator network where the discriminator has also responsible from classification task. Figure 5.3 shows how both generator and discriminator

losses propogate through to iteration of training. As expected, spikes with large variance formed in the early stages of training reach equilibrium by shortening towards the end of training. This shows us that there is a healthy GAN training. As can be observed from the graph showing the change in classification loss during the epoch given in Figure 5.4, the loss value on fake data starts higher than the loss value in the real data and decreases more with a higher acceleration compared to the real data. Although the equlibrium period takes a little longer for fake data, it is seen that the equilibrium level has been reached for both types. This shows us that fake data converge to real data and generator increases classification success by producing data close to real data. The change in classification accuracy value on both the train set and the test set during the epochs of the training can be seen in Figure 5.5. The consistent improvement in accuracy and lack of sharp spikes in both the train and the test set during training support our statements above.

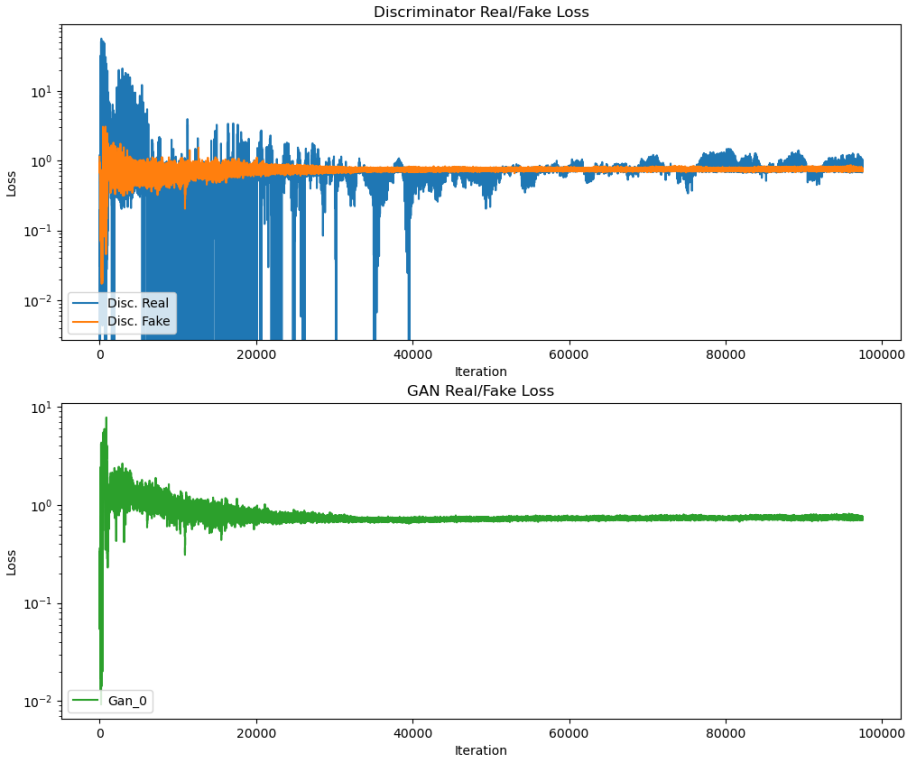

Figure 5.3: Modified ACGAN: Real Fake Loss vs Iterations

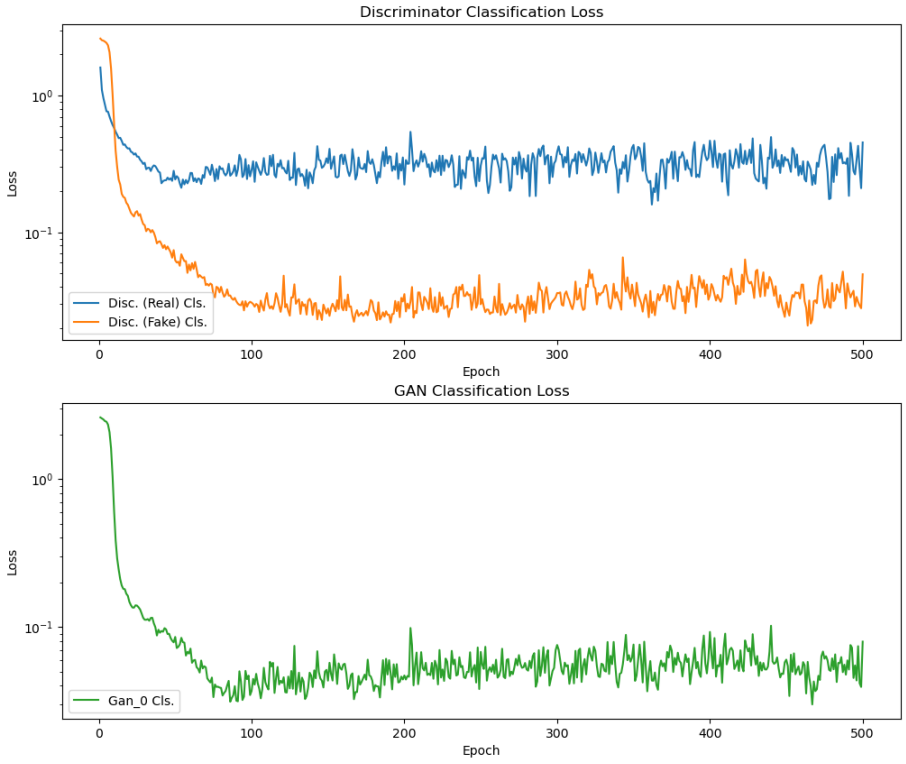

Figure 5.4: Modified ACGAN: Classification Loss vs Epoch

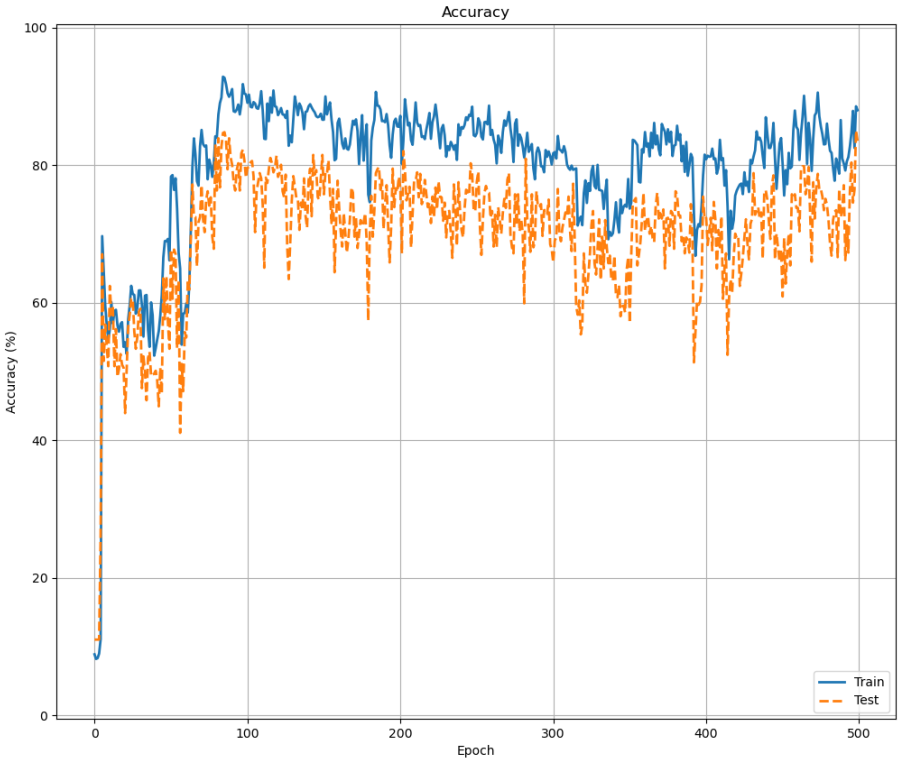

Figure 5.5: Modified ACGAN: Classification Accuracy vs Epoch

Figure 5.6 and Figure 5.7 provide plots of classification success of suggested method modified ACGAN. When the confusion matrix is examined, it is seen that especially the two classes are mixed with each other and most of the misclassified samples belong to these two classes. When the objects of these two classes are examined visually, it is seen that the night stand and dresser converge to a common cube shape geometrically. Figure 5.8 shows some examples that were mislabeled in the test set. In this figure, the actual class label and the estimated class label are given with probabilities that define belonging to that class for each misclassification made.

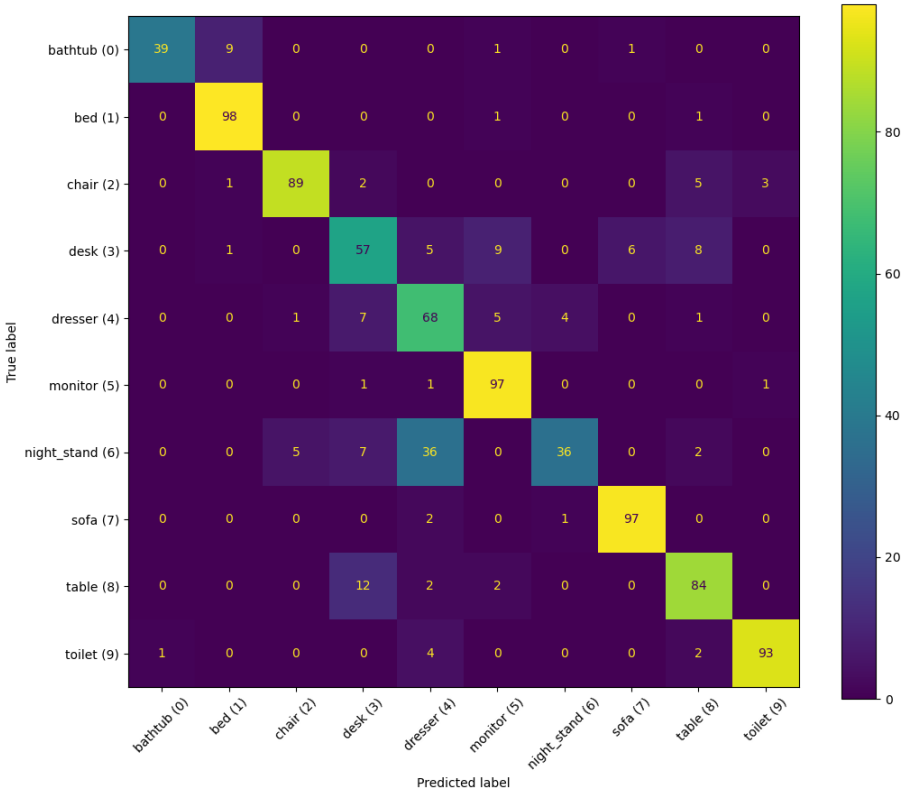

Figure 5.6: Modified ACGAN: Confusion Matrix

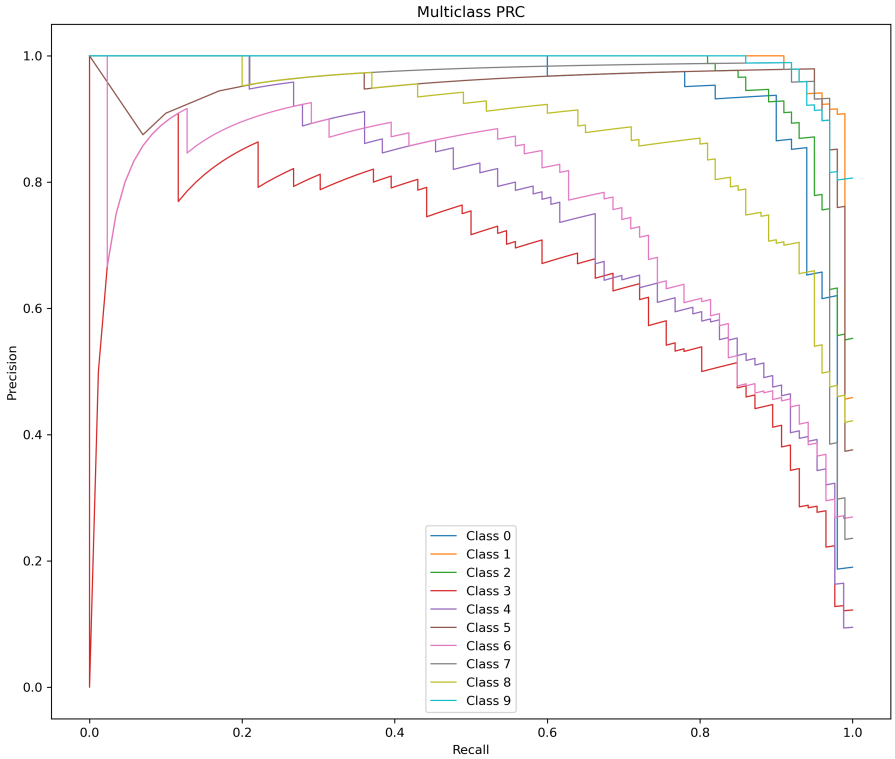

Figure 5.7: Modified ACGAN: Precision Recall Curve (PRC)

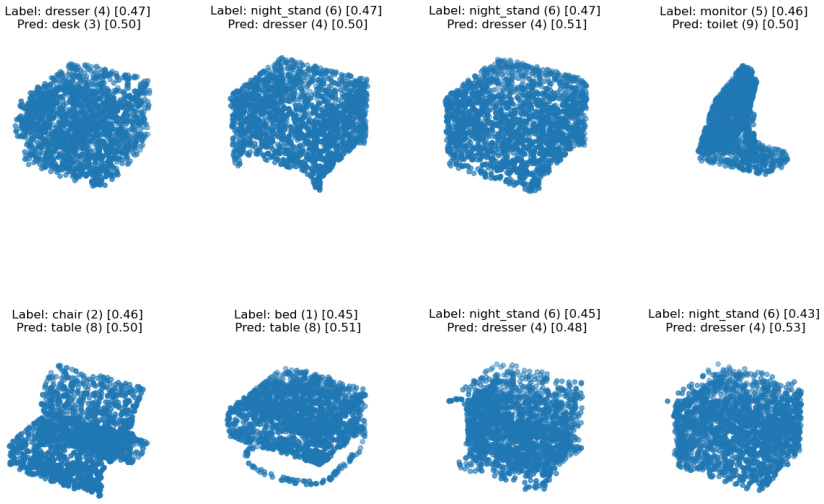

Figure 5.8: Modified ACGAN: Misclassified eight samples from test set. Label row for each sample indicates that real label of the sample and the value in brackets shows the assigned probability for that class. Pred row for each sample expresses the label assigned by classifier for that sample with probability to belonging that class

5.2.2 VACWGAN-GP

In the second proposed method, VACWGAN with GP, the GAN part of the existing VACGAN method was replaced with Wasserstein GAN with GP, and unlike ACGAN, the classifier network was kept separate from the GAN model. It was mentioned that with the use of the Chamfer Distance function in the calculation of the Gradient Penalty, the Generator model will produce fake data closer to the distribution in the real data set and take on the task of data augmentation that will feed the classifier with more data. Therefore, it is claimed that improved classification results should be achieved. Figure 5.9 presents the loss graph of the GAN section of the proposed VACWGAN-GP model. In Figure 5.10, the loss graph of the classification network is given. Similar to the Modified ACGAN method, it is seen that as the epochs in the training progress, the stability in the loss value for real and fake data increases

and decreases in parallel. Due to the WGAN nature, networks establish equilibrium in a slower but more stable manner. It is seen in Figure 5.11 that the classification accuracy has also increased decisively. Figure 5.12 and Figure 5.13 show confusion matrix after test set classification task and PRC plot, respectively. These figures prove our thesis which we argue. There is no doubt that the success of classification has increased compared to the other two base methods, PointNet and Modified ACGAN (ours). In Figure 5.14, we reveal misclassified samples, similar to what we did with the modified ACGAN method proposed. We would like to draw attention to the probabilities given to the correct classes for some of these incorrectly guessed examples. Our classifier, which comes close to correct classification, actually mislabels the data that may be difficult to label even visually.

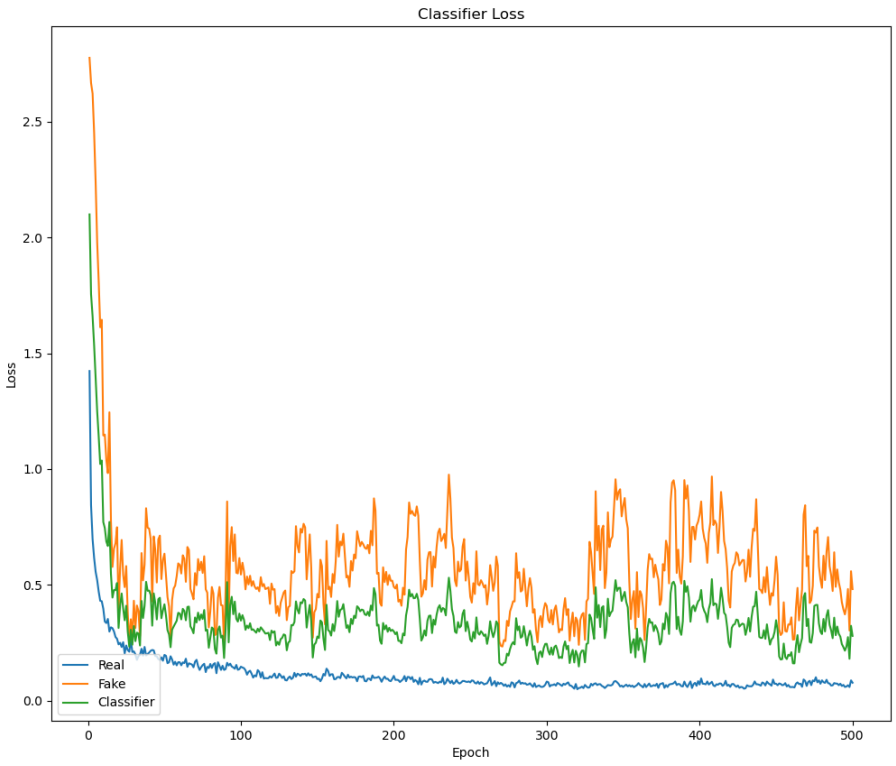

Figure 5.9: VACWGAN-GP: Classification Loss vs Epoch

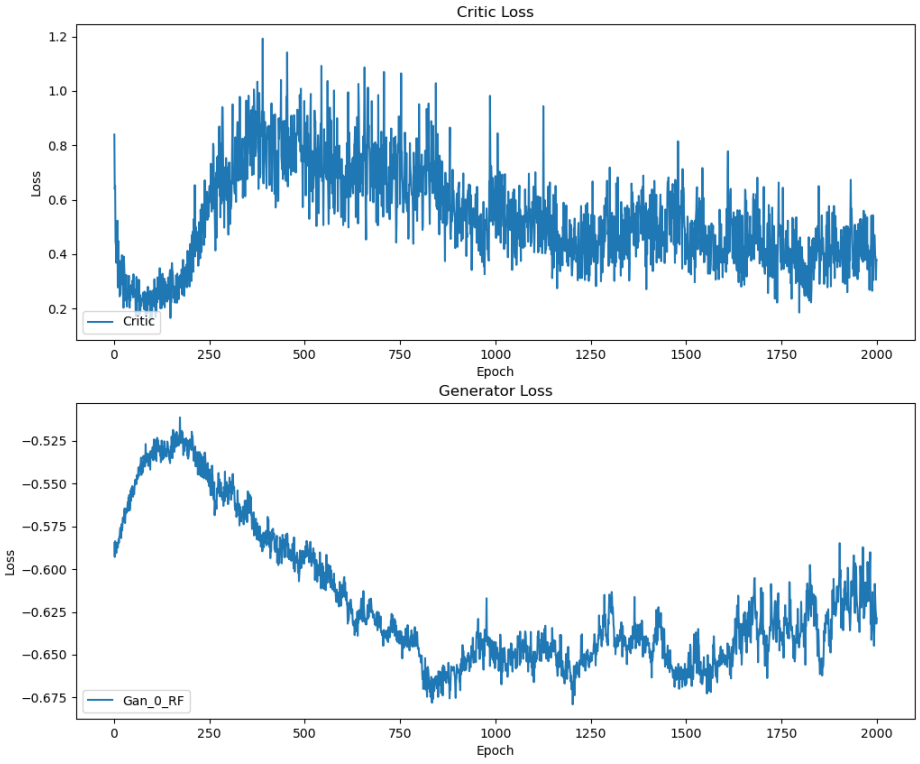

Figure 5.10: VACWGAN-GP: GAN Loss vs Epoch

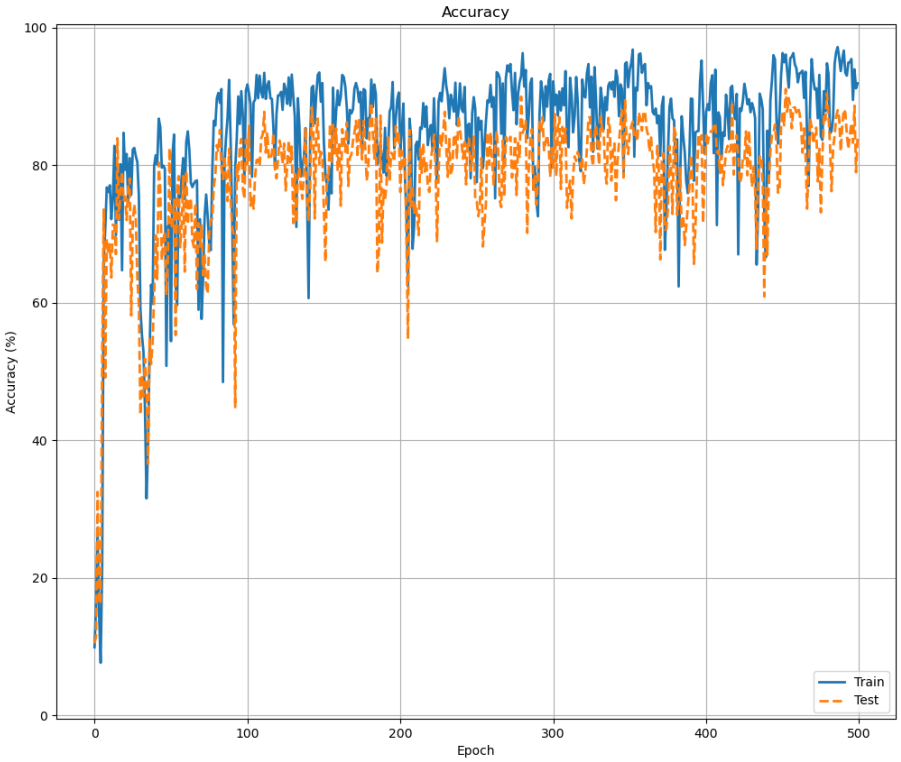

Figure 5.11: VACWGAN-GP: Classification Accuracy

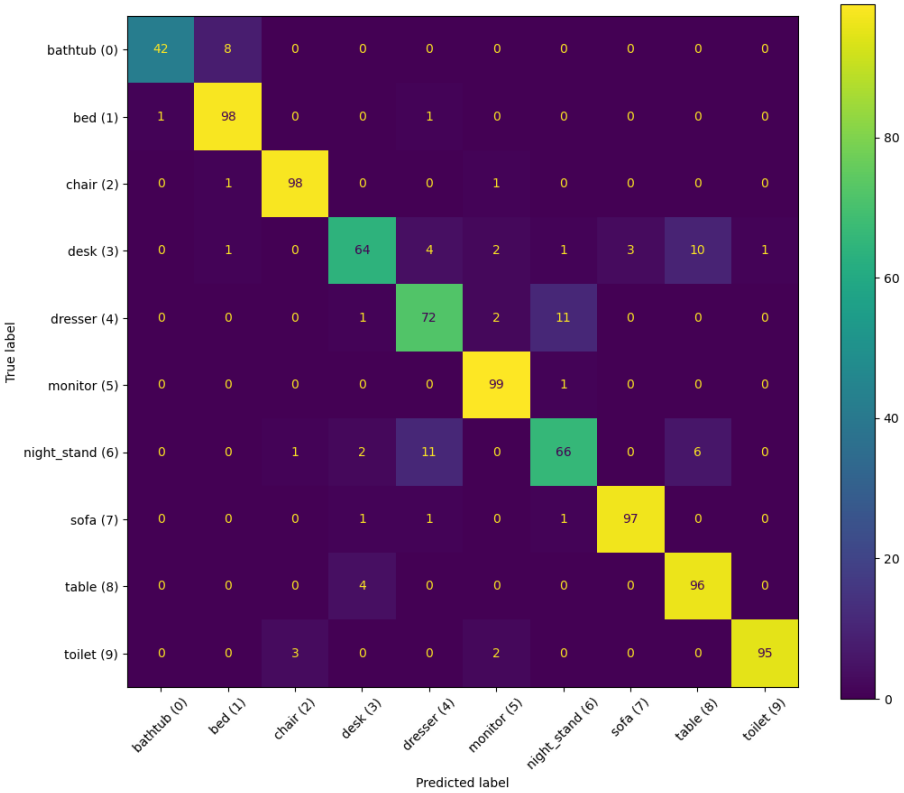

Figure 5.12: VACWGAN-GP: Confusion Matrix

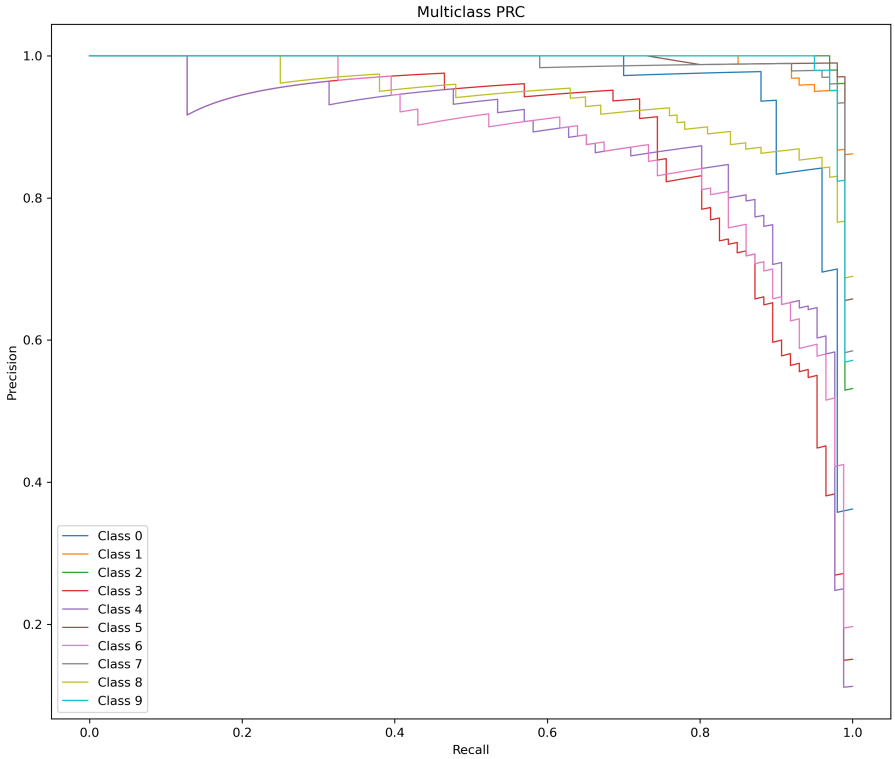

Figure 5.13: VACWGAN-GP: Precision Recall Curve (PRC)

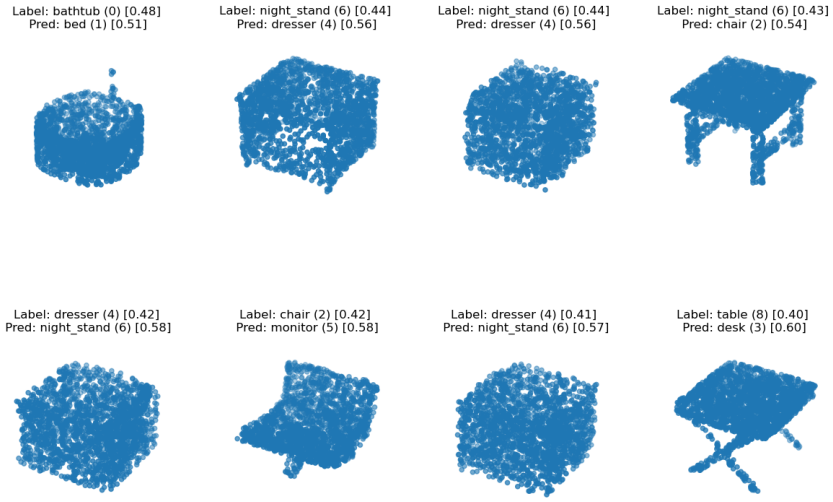

Figure 5.14: VACWGAN-GP: Misclassified eight samples from test set. Label row for each sample indicates that real label of the sample and the value in brackets shows the assigned probability for that class. Pred row for each sample expresses the label assigned by classifier for that sample with probability to belonging that class

5.3 Impact of Latent Vector Dimension

During the experiments, we employed some tests that chose different sizes for latent vector in order to observe how latent vector dimension affects model performance. Thus, we repeated our tests with only updating the latent vector size to 50, 100, 256, and 512 while leaving everything else the same. We would like to draw attention to the fact that we are considering the generator network structure when choosing the number 256 as the limit. Generator, which takes the class value as an input, embeds this input to be the same size as the latent vector, and combines this layer with the latent vector. We do not want the output of this layer to be larger than the successor layer. As suggested in other studies, choosing a high latent vector size in our experiments also had negative effects such as late convergence or failure to converge at all. It has been observed that there is no significant change that will affect

the classification results among other values. Figure 5.15 shows the results of these experiments. As a result, 256 is chosen as a latent vector size throughout the tests.

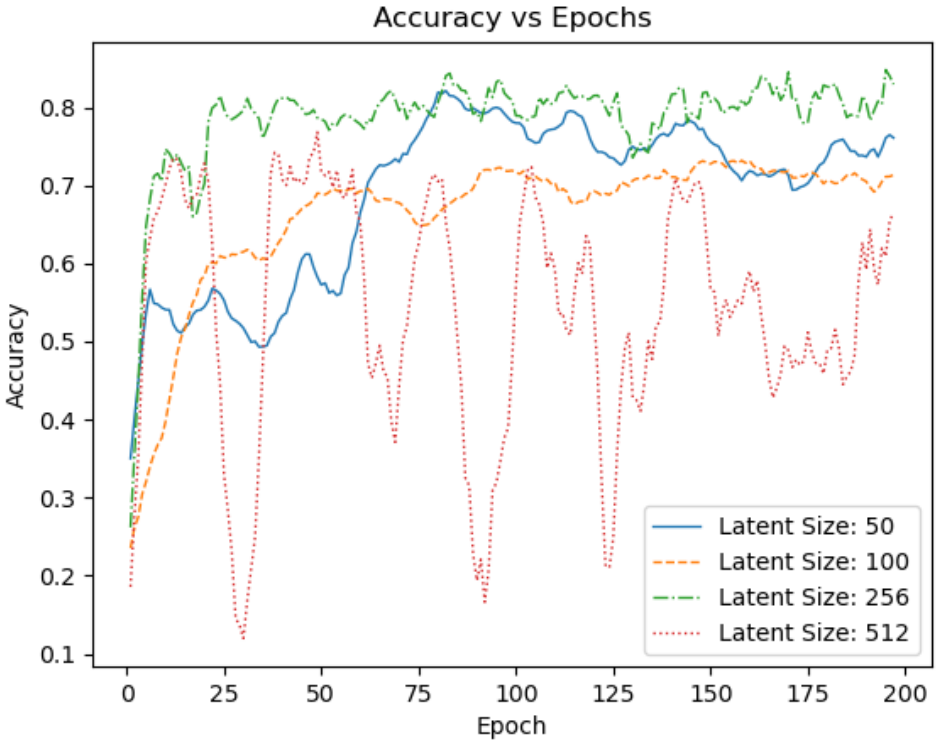

Figure 5.15: The effect of the selected latent vector dimension on the classification accuracy of the proposed method VACWGAN-GP. Choosing a large latent vector size that is not compatible with the size of the network layers has negative impact on classification.

5.4 Impact of Point Size

The number of points that can be sampled through 3D meshes may be subject to some restrictions. The most crucial of these may be performance concerns, or choosing high numbers for uncomplicated problems may result in overdosing. In addition, it may not always be possible to reach the desired number of points in the data directly sampled by the sensors. Both of the models we proposed can take sampled point size as a configuration parameter and be able to work with specified point size. However, like with the latent vector size, it may be helpful to take into consideration

network layer's structure and dimension. In order to demonstrate our flexibility in the manner of sampled point size, we repeated our tests by sampling from the data set at various densities. We chose 128, 256, 512, 1024 and 2048 as candidate sizes. The comparative results are shown in Figure 5.16. Our tests demonstrate that while the number of samples goes down between these levels, the success of classification drops by roughly 3% on average. Since difficulty of expressing the shapes with details at 128 and below increases, drastic decreases are observed in the manner of classification accuracy. In order to be able to make a fairer comparison with other studies, we conducted our tests on the number of 2048 points, taking PointNet as a reference.

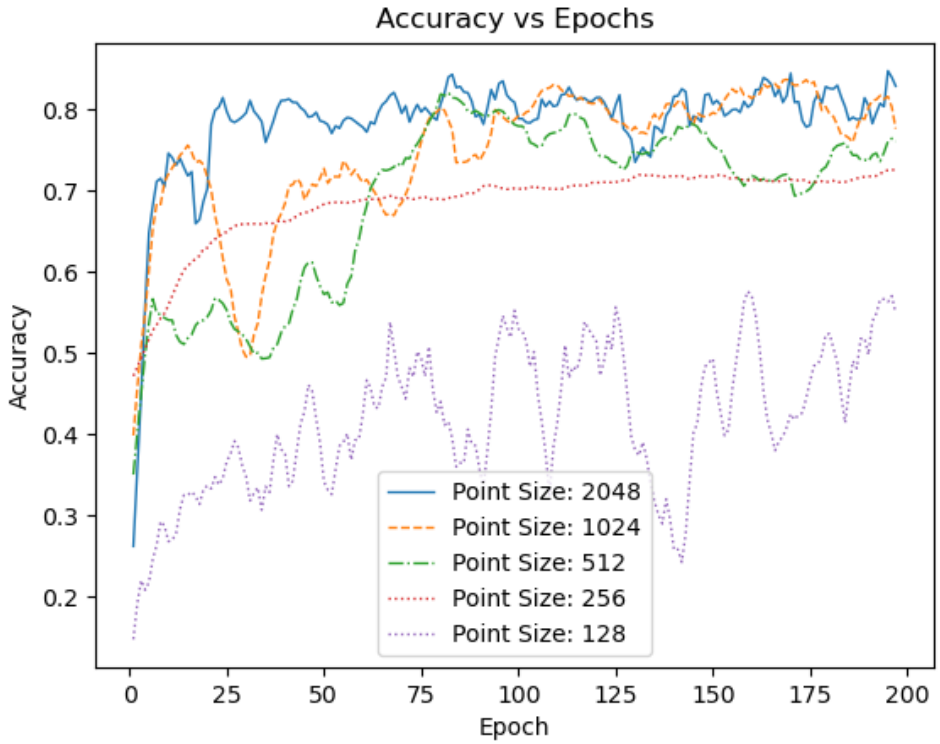

Figure 5.16: The effect of the selected point size on the classification accuracy of the proposed methods. Choosing a small point size has a negative effect as it causes difficulties in expressing the shapes. During the tests, the number of points was chosen as 2048.

5.5 Comparison

Evaluation metrics used in order to measure performance of classification task is given Equation 5.1. Note that, those metrics are defined for binary classification problems and can be extended to multi class classification task with One vs Rest approach where TP, TN, FP and FN are defined as follows:

True Positives (TP) : Number of correctly positive labeled data
True Negatives (TN) : Number of correctly negative labeled data
False Positives (FP) : Number of incorrectly positive labeled data
False Negatives (FN) : Number of incorrectly negative labeled data

$$
\begin{aligned}
Precision &= \frac{TP}{TP + FP} \\
Recall &= \frac{TP}{TP + FN} \\
F1score &= \frac{2 \times Precision \times Recall}{Precision + Recall} \\
Accuracy &= \frac{TP + TN}{TP + FP + TN + FN}
\end{aligned}
\tag{5.1}
$$

Detailed classification reports of proposed methods is given Table 5.3. The table shows that the two methods we propose perform quite well in classifying the dataset. Our VACWGAN-GP method, in particular, outperforms our Modified ACGAN method and competes with state-of-the-art researches. While the modified ACGAN method is more inaccurate in categorizing samples belonging to Desk, Dresser and Night Stand classes, VACWGAN-GP produces decent results. Table 5.4 compares the classification accuracy of our methods with previous studies. The table consists of previous studies that accept various input types. During the comparison, one should consider this difference, which affects the way of solving the problem and the difficulty of the solution and also compatibility to applying on possible datasets.

Table 5.3: Classification Report for Proposed Methods

Modelnet10	**Modified ACGAN**			**VACWGAN-GP**			**Support**
	Precision	Recall	F1-Score	Precision	Recall	F1-Score	
Bathtub (0)	0.97	0.78	0.87	0.98	0.84	0.90	**50**
Bed (1)	0.90	0.98	0.94	0.91	0.98	0.94	**100**
Chair (2)	0.94	0.89	0.91	0.96	0.98	0.97	**100**
Desk (3)	0.66	0.66	0.66	0.89	0.74	0.81	**86**
Dresser (4)	0.58	0.79	0.67	0.81	0.84	0.82	**86**
Monitor (5)	0.84	0.97	0.90	0.93	0.99	0.96	**100**
Night Stand (6)	0.88	0.42	0.57	0.82	0.77	0.80	**86**
Sofa (7)	0.93	0.97	0.95	0.97	0.97	0.97	**100**
Table (8)	0.82	0.84	0.83	0.86	0.96	0.91	**100**
Toilet (9)	0.96	0.93	0.94	0.99	0.95	0.97	**100**
Accuracy			**0.83**			**0.91**	**908**
Macro Avg.	0.85	0.82	0.82	0.91	0.90	0.91	**908**
Weighted Avg.	0.85	0.83	0.83	0.91	0.91	0.91	**908**

Table 5.4: Classification Accuracy Comparison

Method	**Input Type**	**Classification Acc. Modelnet10**	**Classification Acc. Modelnet40**
3DShapeNets	Volume	83.5%	77%
VoxNet	Volume	92%	83%
LP-3DCNN	Volume	94.4%	92.1%
Primitive-GAN	Volume	86.4%	92.2%
DeepPano	2D (Panoromic)	85.45%	77.63%
ACNN	2D	92.52%	89.11%
PointNet	Point	77.6%	89.2%
PointNet++	Point	-	90.7%
DGCNN	Point	-	92.2%
STRL + DGCNN	Point	-	93.1%
Pointwise CNN	Point	87.07	-
Ours Baseline	Point	70.1%	63.2%
Modified ACGAN (ours)	Point	85.2%	75.4%
VACWGAN-GP (ours)	Point	91.74%	81.3%

5.6 Shape Retrieval Application

In order to adapt these proposed classifiers to real life problems, we have developed a shape retrieval application where these classifiers can be used. Working as an example-based application, it aims to bring similar shapes in the same class from the data set labeled by this classifier by processing the query data as a point cloud. In Figure 5.17 and Figure 5.18, we show five examples for each class that we collected from the shape database. To implement this task, a sample for each class is randomly selected from the test set as query data. Using this selected query data, similar shapes are retrieved from the data set. These figures are the results of the application using classifiers in the two methods we propose, respectively.

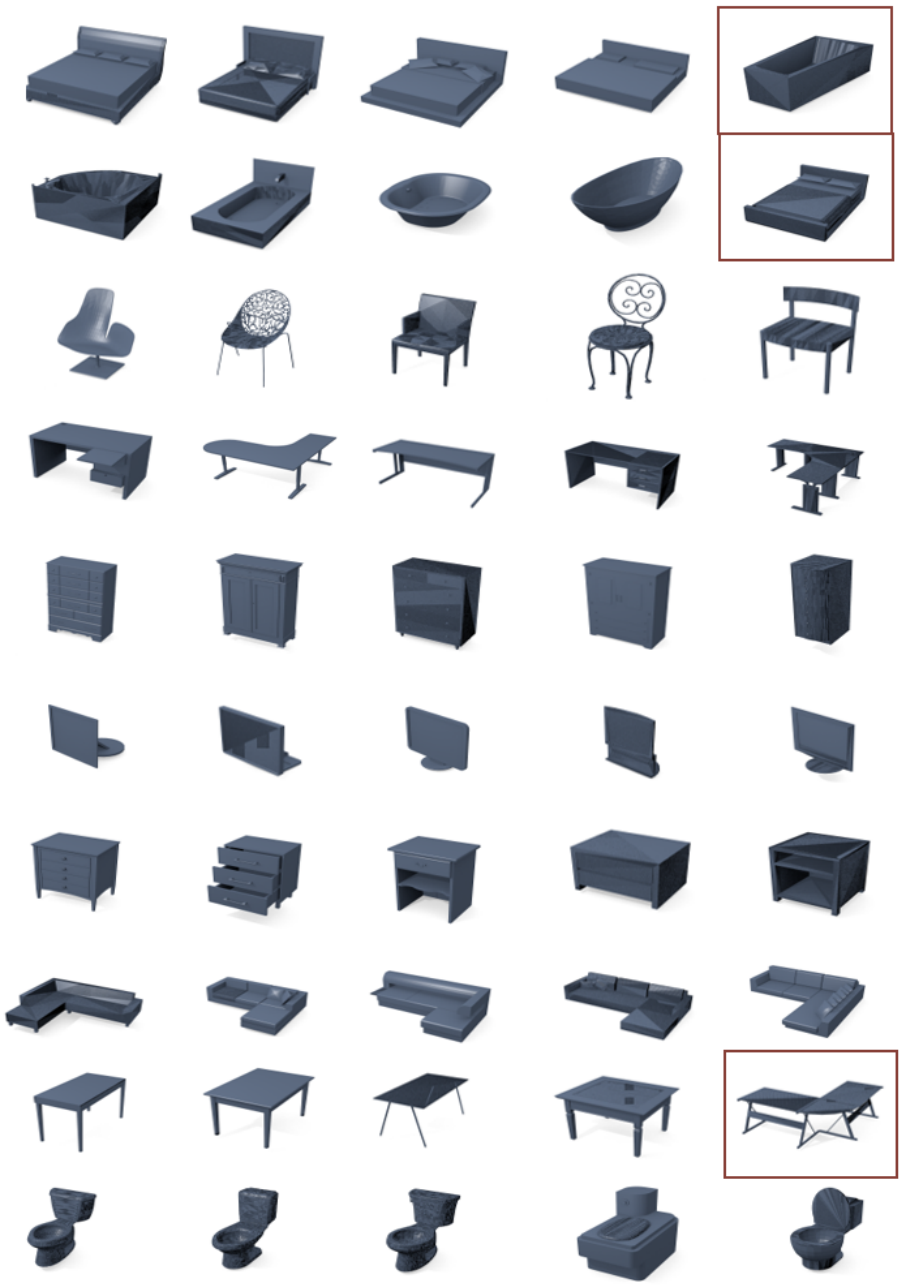

Figure 5.17: Modified ACGAN: Shape Retrieval Results
For every class label, randomly selected query shapes and wrong classified examples pointed with red squares. Each row corresponds to class labels and columns are samples from those labels

Figure 5.18: VACWGAN-GP: Shape Retrieval Results

For every class label, randomly selected query shapes and wrong classified examples pointed with red squares. Each row corresponds to class labels and columns are samples from those labels

5.7 Missing Data

It may not always be possible to have the desired number of points for the 3D point clouds to be queried. In particular, it is possible to encounter this situation in the data produced by the sensors. In order to examine the behavior of the proposed methods when applied with missing data and to show their strength, we conducted experiments with different rates of missing data. In this context, the coordinate information of randomly selected points from each sample in the point cloud data generated from the test set in the modelnet10 data has been cleared. The results of the experiments with different data reduction percentages can be accessed from the Table 5.5. Percentage represents the number of points cleared in each point cloud sample. The results show that both methods we propose are data loss resistant and stable.

Table 5.5: Missing Rate vs Classification Accuracy

Missing	**Classification Accuracy**	
Percentage	*ACGAN*	*VACWGAN-GP*
0%	85.2	91.74
5%	82.37	90.08
10%	82.26	90.3
20%	82.92	90.41
25%	82.48	90.52
30%	82.15	90.3
50%	81.27	90.52
75%	79.95	90.08
80%	79.18	89.97
85%	79.07	89.64
90%	75.77	87.55
95%	70.81	80.72
99%	43.61	51.43
100%	11.01	11.01

5.8 Time and Space Complexity

Table 5.6 shows the space complexity of the models proposed. The number of the trainable parameters in the networks in each model is stated in separate columns. Note that, since the classifier and discriminator are integrated in the proposed ACGAN model, the number of parameters in the Discriminator and classifier expressed with a single cell. Space complexity is $O(N)$ where N is point size of the samples.

Table 5.6: Space Complexity

Models	**# params**			
	Generator	Discriminator + Classifier	Critic	Classifier
ACGAN	13.99M	0.49M	-	-
VACWGAN-GP	15.34M	-	0.08M	0.20M

Empirical time results for both classification and shape retrieval application are given in the Table 5.6 in seconds. Same hardware combination with training phase which given in Section 5 is employed. First column group in the table shows the classification time on the test set of the specified dataset. While the middle column expresses the duration of determining the label of a single mesh in query, the last column group states the running time of the developed shape retrieval application on the given dataset.

Table 5.7: Run Time Results of Classification and Shape Retrieval Application

Time (s)	**Classification on Testset**		**Classification on Instance**	**Shape Retrieval**	
	Modelnet10	Modelnet40		Modelnet10	Modelnet40
ACGAN	0.9944	2.534	0.0010	0.4008	0.9642
VACWGAN-GP	1.1831	2.879	0.0013	0.4119	0.9868

CHAPTER 6

CONCLUSIONS

In this thesis, we propose two different methods to classify 3D shapes. In order to accomplish this task, we are offering to classify 3D meshes in 3D space only by their coordinates by purging the connection information of vertexes. With this proposed methods, we use 3D point cloud data that can even be produced from sensors. Thus, by not using voxel-based approaches which are more distant from real-life scenarios, we propose a lighter method with easier data set availability. Although there are many 3D data produced today, it is not always possible to access the labeled ones or it may be difficult to label them. Considering the fact that 3D shapes will increase even more, the idea of labeling shapes becomes even more important. Taking these two major factors into consideration, we proposed classifiers to label 3D shapes and we preferred to use the power of generative models to make these classifiers better. Based on the difficulty of finding labeled data and the dilemma that machine learning techniques to be proposed to label the data still need labeled data, we set out with the motivation to augment an already labeled small dataset with generative models. With the help of generative models, we are able to enlarge data sets which are very similar to the data set at hand, enabling the classifier to better model the dataset distribution and thus make a better generalization. The first of the two methods we suggested is modified ACGAN, which is already designed to perform the classification task of 2D images, suitable for 3D Point Clouds and employing it in labeling. Unlike 2D images, the points in 3D point cloud data are unordered and no direct neighborhood relationship can be defined with respect to each other. This makes this task even more challenging. The biggest challenge here was when setting up the network layers. The training process of GAN models, which takes a long time and can cause problematic results, like mode collapse, depending on the situation, pushed us to improve this

method. In a solution system where Classifier is separated from Discriminator and formed by two separate networks, we proposed replacing the GAN model part with Wasserstein GAN. On top of the Wasserstein GAN structure, we applied the Gradient Penalty, which can also solve problems such as Vanishing Gradients, mode collapse etc. However, we have replaced Gradient Penalty calculation proposed for 2D images with the Chamfer Distance to suit the nature of the 3D point cloud problem. Thus, by avoiding potential mode collapse problems, we have produced a model that can enlarge the dataset with fake data that is close to the distribution of the real data. Thus, we have proposed using the generative model as a data augmenter. With this powerful tool we have acquired, we have further improved our classifier and achieved our main goal. As far as we know, we have implemented such a use of Wasserstein-GAN with Chamfer Distance Gradient penalty as a data augmenter in 3D point cloud classification as an innovation. The methods we propose are acceptably sensitive to the number of points selected, as long as they are compatible with the network layer structure and comply with the physical boundaries. Furthermore, the models we propose are resistant to working with incomplete data, which means they perform better against data sets that may be incompletely generated by sensors. As a conclusion, while the first of the two models we recommend can be preferred with its compact design, our second method stands out for higher performance. We also developed a shape retrieval application that uses these proposed classifiers and used our solution to solve real-life problems in reasonable run time.

As a continuation of this study, application areas such as repairing deformed models or completing mostly missing data samples by using the generative model we propose, producing new shapes by combining data belonging to two different or the same classes without getting too far from the real data set, may be interesting. Extending the classifiers we recommend to make point cloud segmentation would be a worthwhile area to work on. With today's changing focus, 3D models will become more important and many problems will arise that will require generation. It should always be kept in mind as an option, without negating the power of generative models in this area.

REFERENCES

[1] Hemmes, T. (2018). Classification of large scale outdoor point clouds using convolutional neural networks.

[2] Liu, L. (2021). Point cloud classification by nearest samples of random rays (Doctoral dissertation, University of British Columbia).

[3] Tangelder, J. W., & Veltkamp, R. C. (2008). A survey of content based 3D shape retrieval methods. Multimedia tools and applications, 39(3), 441-471.

[4] Sahillioğlu, Y., & Sezgin, M. (2017). Sketch-based articulated 3d shape retrieval. IEEE computer graphics and applications, 37(6), 88-101.

[5] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Bengio, Y. (2014). Generative adversarial nets. Advances in neural information processing systems, 27.

[6] Grilli, E., Menna, F., & Remondino, F. (2017). A review of point clouds segmentation and classification algorithms. The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, 42, 339.

[7] Besl, P. J., & Jain, R. C. (1988). Segmentation through variable-order surface fitting. IEEE Transactions on pattern analysis and machine intelligence, 10(2), 167-192.

[8] Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM, 24(6), 381-395.

[9] Ballard, D. H. (1981). Generalizing the Hough transform to detect arbitrary shapes. Pattern recognition, 13(2), 111-122.

[10] Nguyen, A., & Le, B. (2013, November). 3D point cloud segmentation: A survey. In 2013 6th IEEE conference on robotics, automation and mechatronics (RAM) (pp. 225-230). IEEE.

[11] Su, H., Maji, S., Kalogerakis, E., & Learned-Miller, E. (2015). Multi-view convolutional neural networks for 3d shape recognition. In Proceedings of the IEEE international conference on computer vision (pp. 945-953).

[12] Zhou, Y., & Tuzel, O. (2018). Voxelnet: End-to-end learning for point cloud based 3d object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4490-4499).

[13] Caltagirone, L., Scheidegger, S., Svensson, L., & Wahde, M. (2017, June). Fast LIDAR-based road detection using fully convolutional neural networks. In 2017 ieee intelligent vehicles symposium (iv) (pp. 1019-1024). IEEE.

[14] Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 652-660).

[15] Kim, E., & Medioni, G. (2011). Urban scene understanding from aerial and ground LIDAR data. Machine Vision and Applications, 22(4), 691-703.

[16] Anguelov, D., Taskarf, B., Chatalbashev, V., Koller, D., Gupta, D., Heitz, G., & Ng, A. (2005, June). Discriminative learning of markov random fields for segmentation of 3d scan data. In 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) (Vol. 2, pp. 169-176). IEEE.

[17] Socher, R., Huval, B., Bath, B., Manning, C. D., & Ng, A. (2012). Convolutional-recursive deep learning for 3d object classification. Advances in neural information processing systems, 25, 656-664.

[18] Rabbani, T., Van Den Heuvel, F., & Vosselmann, G. (2006). Segmentation of point clouds using smoothness constraint. International archives of photogrammetry, remote sensing and spatial information sciences, 36(5), 248-253.

[19] Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., & Xiao, J. (2015). 3d shapenets: A deep representation for volumetric shapes. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1912-1920).

[20] Han, Z., Liu, Z., Han, J., Vong, C. M., Bu, S., & Li, X. (2016). Unsupervised 3D local feature learning by circle convolutional restricted Boltzmann machine. IEEE Transactions on Image Processing, 25(11), 5331-5344.

[21] Bu, S., Liu, Z., Han, J., Wu, J., & Ji, R. (2014). Learning high-level feature by deep belief networks for 3-D model retrieval and recognition. IEEE Transactions on Multimedia, 16(8), 2154-2167.

[22] Mirza, M., & Osindero, S. (2014). Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784.

[23] Beltrán, J., Guindel, C., Moreno, F. M., Cruzado, D., Garcia, F., & De La Escalera, A. (2018, November). Birdnet: a 3d object detection framework from lidar information. In 2018 21st International Conference on Intelligent Transportation Systems (ITSC) (pp. 3517-3523). IEEE.

[24] Nie, D., Trullo, R., Lian, J., Petitjean, C., Ruan, S., Wang, Q., & Shen, D. (2017, September). Medical image synthesis with context-aware generative adversarial networks. In International conference on medical image computing and computer-assisted intervention (pp. 417-425). Springer, Cham.

[25] Costa, P., Galdran, A., Meyer, M. I., Abramoff, M. D., Niemeijer, M., Mendonça, A. M., & Campilho, A. (2017). Towards adversarial retinal image synthesis. arXiv preprint arXiv:1701.08974.

[26] Xue, Y., Xu, T., Zhang, H., Long, L. R., & Huang, X. (2018). Segan: Adversarial network with multi-scale l 1 loss for medical image segmentation. Neuroinformatics, 16(3), 383-392.

[27] Chuquicusma, M. J., Hussein, S., Burt, J., & Bagci, U. (2018, April). How to fool radiologists with generative adversarial networks? A visual turing test for lung cancer diagnosis. In 2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018) (pp. 240-244). IEEE.

[28] Karras, T., Aila, T., Laine, S., & Lehtinen, J. (2017). Progressive growing of gans for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196.

[29] Lucic, M., Kurach, K., Michalski, M., Gelly, S., & Bousquet, O. (2017). Are gans created equal? a large-scale study. arXiv preprint arXiv:1711.10337.

[30] White, T. (2016). Sampling generative networks: Notes on a few effective techniques. CoRR, abs/1609.04468, 7.

[31] Kang, M., Shim, W., Cho, M., & Park, J. (2021). Rebooting ACGAN: Auxiliary Classifier GANs with Stable Training. Advances in Neural Information Processing Systems, 34.

[32] Cho, J., & Yoon, K. (2020). Conditional Activation GAN: Improved Auxiliary Classifier GAN. IEEE Access, 8, 216729-216740.

[33] Zhang, Y., & Rabbat, M. (2018, April). A graph-cnn for 3d point cloud classification. In 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 6279-6283). IEEE.

[34] Özdemir, E., Remondino, F., & Golkar, A. (2019). Aerial point cloud classification with deep learning and machine learning algorithms. The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, 42, 843-849.

[35] Yao, X., Guo, J., Hu, J., & Cao, Q. (2019). Using deep learning in semantic classification for point cloud data. IEEE Access, 7, 37121-37130.

[37] Li, R., Li, X., Heng, P. A., & Fu, C. W. (2020). Pointaugment: an auto-augmentation framework for point cloud classification. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 6378-6387).

[36] Bazrafkan, S., & Corcoran, P. (2018). Versatile auxiliary classifier with generative adversarial network (vac+ gan), multi class scenarios. arXiv preprint arXiv:1806.07751.

[37] Arjovsky, M., Chintala, S., & Bottou, L. (2017, July). Wasserstein generative adversarial networks. In International conference on machine learning (pp. 214-223). PMLR.

[38] Radford, A., Metz, L., & Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434.

[39] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. (2017). Improved training of wasserstein gans. arXiv preprint arXiv:1704.00028.

[40] Liu, Y., Li, Z., Zhou, C., Jiang, Y., Sun, J., Wang, M., & He, X. (2019). Generative adversarial active learning for unsupervised outlier detection. IEEE Transactions on Knowledge and Data Engineering, 32(8), 1517-1528.

[41] Kazhdan, M., Funkhouser, T., & Rusinkiewicz, S. (2003, June). Rotation invariant spherical harmonic representation of 3 d shape descriptors. In Symposium on geometry processing (Vol. 6, pp. 156-164).

[42] Maturana, D., & Scherer, S. (2015, September). Voxnet: A 3d convolutional neural network for real-time object recognition. In 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 922-928). IEEE.

[43] Shi, B., Bai, S., Zhou, Z., & Bai, X. (2015). Deeppano: Deep panoramic representation for 3-d shape recognition. IEEE Signal Processing Letters, 22(12), 2339-2343.

[44] Qi, C. R., Yi, L., Su, H., & Guibas, L. J. (2017). Pointnet++: Deep hierarchical feature learning on point sets in a metric space. Advances in neural information processing systems, 30.

[45] Kumawat, S., & Raman, S. (2019). Lp-3dcnn: Unveiling local phase in 3d convolutional neural networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 4903-4912).

[46] Khan, S. H., Guo, Y., Hayat, M., & Barnes, N. (2019). Unsupervised primitive discovery for improved 3D generative modeling. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 9739-9748).

[47] Xu, S., Zhou, X., Ye, W., & Ye, Q. (2022). Classification of 3D Point Clouds By A New Augmentation Convolutional Neural Network. IEEE Geoscience and Remote Sensing Letters.

[48] Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, & J. M. Solomon (2019). Dynamic Graph CNN for Learning on Point Clouds. arXiv:1801.07829 [cs], arXiv: 1801.07829

[49] Huang, S., Xie, Y., Zhu, S. C., & Zhu, Y. (2021). Spatio-temporal self-supervised representation learning for 3d point clouds. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 6535-6545).

[50] Song, W., Liu, Z., Tian, Y., & Fong, S. (2021). Pointwise CNN for 3D Object Classification on Point Cloud. Journal of Information Processing Systems, 17(4), 787-800.