
REPUBLIC OF TÜRKİYE

ALTINBAŞ UNIVERSITY

Institute of Graduate Studies

Information Technologies

# DESIGN AND ANALYSIS OF A DATA COMPUTING SCENARIO FORECASTING SYSTEM BASED ON GENERATIVE ADVERSARIAL NETWORKS

**Ammar Raad Hashim HASHIM**

Supervisor

Asst.Prof. Timur INAN

Istanbul, 2023

# DESIGN AND ANALYSIS OF A DATA COMPUTING SCENARIO FORECASTING SYSTEM BASED ON GENERATIVE ADVERSARIAL NETWORKS

**Ammar Raad Hashim HASHIM**

Information Technologies

Master's Thesis

ALTINBAŞ UNIVERSITY

2023

The thesis titled DESIGN AND ANALYSIS OF A DATA COMPUTING SCENARIO FORECASTING SYSTEM BASED ON GENERATIVE ADVERSARIAL NETWORKS prepared by Ammar RAAD HASHIM HASHIM and submitted on 26/07/2023 has been **accepted unanimously** for the degree of Master of Science in Information Technologies

| | |
|---|---|
| Academic Title – First and Last Name of the Co-Supervisor | Academic Title – First and Last Name of the Supervisor |

Thesis Defense Committee Members:

| | | |
|---|---|---|
| Academic Title - First/Last Name | Department, University | _____ |
| Academic Title - First/Last Name | Department, University | _____ |
| Academic Title - First/Last Name | Department, University | _____ |
| Academic Title - First/Last Name | Department, University | _____ |
| Academic Title - First/Last Name | Department, University | _____ |

I hereby declare that this thesis meets all format and submission requirements of a Master's thesis.
Submission date of the thesis to Institute of Graduate Studies:____/___/____

I hereby declare that all information/data presented in this graduation project has been obtained in full accordance with academic rules and ethical conduct. I also declare all unoriginal materials and conclusions have been cited in the text and all references mentioned in the Reference List have been cited in the text, and vice versa as required by the abovementioned rules and conduct.

Ammar Raad Hashim HASHIM

Signature

# DEDICATION

I dedicate and promise this study to my supervisor Prof. Dr. Osman Nuri UÇAN, who was instrumental in helping me through the entire research process, as well as my family (my father is the role model, my beloved mother, my dear wife, my beloved children) and my beloved brother and sisters, who have always been my support during difficult times.

# ABSTRACT

## DESIGN AND ANALYSIS OF A DATA COMPUTING SCENARIO FORECASTING SYSTEM BASED ON GENERATIVE ADVERSARIAL NETWORKS

HASHIM, Ammar Raad Hashim

M.Sc., Information Technologies, Altınbaş University,

Supervisor: Asst. Prof. Dr. Timur INAN

Date: July / 2023

Pages: 63

One of the main drawbacks in using ML and DL based optimization techniques is that a massive amount of historical data is needed for the algorithms to converge to quasi-optimal solutions. In the DCs' scope, it is difficult, expensive, and sometimes even dangerous to gather data from all kinds of real situations, as the equipment's physical integrity may be at risk. Thus, the generation of realistic synthetic data in computing environments is crucial in enabling these optimization technologies. This project proposes to solve these challenges by generating synthetic data, forecasting possible realistic scenarios in a real Cloud DC environment. For this purpose, we will use a technique that has given extraordinary results in recent years, the Generative Adversarial Networks (GAN). These generative algorithms based on artificial neural networks have been commonly used to create realistic synthetic multimedia data (mainly images and videos). Some proposals in the state of the art use GAN to generate synthetic time-series data. However, this research has significant limitations, such as the limitation of handling multi-variable and categorical data in the generation of scenarios.

**Keywords:** Generative Adversarial Networks (GAN), Data Computing, Big Data , Artificial Intelligence (AI), Internet of Things (IoT).

# TABLE OF CONTENTS

**Pages**

# LIST OF TABLES

**Pages**

# LIST OF FIGURES

**Pages**

# ABBREVIATIONS

ANN : Artificial Neural Network

BCE : Binary Cross-Entropy

GAN : Generative Adversarial Network

LSTM : Long Short-Term Memory

MSE : Mean Squared Error

MSLE : Mean Squared Logarithmic Error

# 1. INTRODUCTION

We live in a data-driven world, and data is generated almost everywhere these days: sensor networks on Mars, submarines in the most profound sea, assessments of public sentiment on any subject, etc. A significant number of these certifiable applications have a similar issue: absent or inadequate information. Missing information can be brought about by various variables, like sensor or correspondence disappointments, asset requirements, or the straightforward reality that the rate at which various kinds of information can be obtained can differ enormously, bringing about data of interest in the tangible time series where just subsets of the information are accessible. In a modern trial, for instance, an outcome might be absent because of mechanical or electronic disappointments during the information procurement process. A few clinical trials can't be performed because the medical clinic misses the mark on essential clinical hardware, or the tests are unseemly for specific patients [1]. In a similar setting, another model could be a specialist's assessment, which incorporates different sorts of tests; some experimental outcomes might be accessible right away, while others might require a few days to finish.

These troubles can be overwhelmed by utilizing different AI, sifting, and assessment strategies. These information attribution procedures, be that as it may, are constantly founded on a series of expectations and will deliver great outcomes provided that these suppositions are generally right. Profound learning has made it conceivable to settle even the most complicated undertakings, like picture arrangement, picture age, etc. This has likewise been helped by a huge expansion in calculation power, which fills in as a spine while preparing profound brain networks through the improvement of cutting edge calculation units, for example, GPUs [2]. Therefore, we can address these missing information challenges utilizing profound learning and factual displaying information, as well as more current computational units to deliver quicker results.

The reason for this proposition is to resolve the issues that emerge while managing mind boggling, deficient time-series sensor information and to propose another generative brain network engineering to figure values in circumstances where portion of the information is absent or fragmented, by utilizing the rich construction in the accessible information.

## 1.1 DIGITAL ECONOMY & SOCIETY, GROWTH AND CONCERNS

This digital transformation is driven by ground-breaking technologies such as the Internet of Things (IoT), Big Data, and Artificial Intelligence (AI) as part of the development of Smart Cities and brand-new 5G communication networks. According to Cisco's Annual Internet Report [9] in 2018, 51% of the world's population used the Internet, and it is expected to reach 66% in 2023. The number of devices connected to IP networks was 1.6 times the world population in 2018 and will be 3.6 times in 2023. In Figure 1.1, we can observe the share of the population using the Internet in 2017.

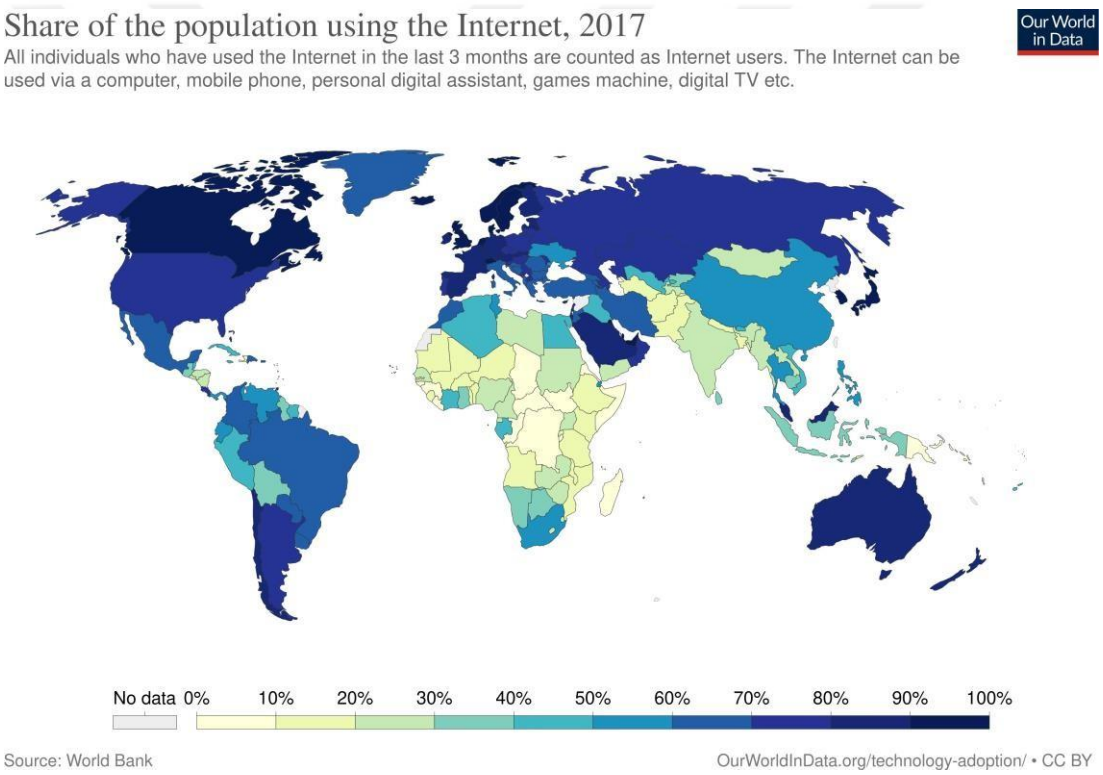



**Figure 1.1:** Share of the Population Using the Internet, 2017 [3]

All people who have involved the Web over the most recent three months are considered web clients. Source: World Bank Information

From an economic standpoint, the integration of Information and Communications Technology (ICT) brings additional innovations and organizational changes to businesses. According to the latest statistics published by the European Commission on the digital economy and society [10], 1 out of 5 EU companies made e-sales in 2018, and more than 1

in 3 used e-business integrations such as Enterprise Resource Planning and Customer Relationship Management.

Nevertheless, this digitalization trend is much more significant among households and individuals for two main reasons. Firstly, only 48% of European citizens between 16 and 74 used the Internet daily in 2014, although it reached 73% in 2019 [11]. Secondly, according to data from Sandvine [12], over 75% of worldwide Web traffic is created by video web based, gaming, and virtual entertainment applications. Figure 1.2 shows the evolution of the hours spent daily on digital media in the United States from 2008 to 2018.

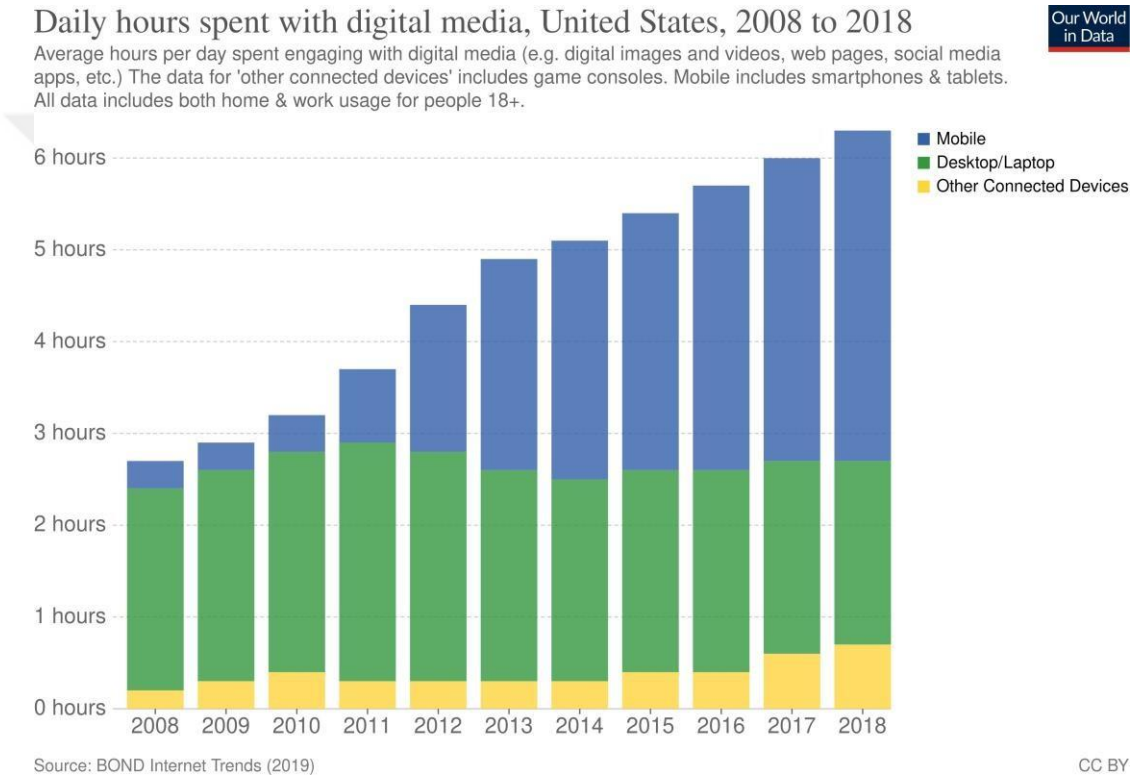



**Figure 1.2:** Daily Hours Spent with Digital Media, United States, 2008 to 2018. [11]

The crucial key we must understand from the above statements is that the vast majority of these technologies base their operations on the Cloud Computing (CC) paradigm, particularly the more complex and data-intensive ones (e.g., video streaming and social media). From 2014 to 2018, the use of Cloud solutions in business increased significantly, particularly in the largest companies (+21%) [13]. The ratio of traffic used by social networks and video streaming will continue to increase, both in terms of the number of users and the offered content's resolution.

This specific circumstance, along with the interest for novel applications in Web of Things (IoT), independent vehicles, and Brilliant Urban communities, will prompt an extraordinary measure of information traffic and demands for processing assets in the Cloud, making substantially more evident and stressing the shortcomings Server farm (DC)s have. [1] raise that some place in the scope of 2010 and 2018, the amount of DCs figure cases extended by 550% and IP traffic by 1,000%. Achieving overall energy.

## 1.2 MOTIVATION

There has been an explosion of time series data over the last decade. Time series can be found everywhere, from the financial sector to rivers and sea levels, but also in medical data and speech analysis. A time series is formed whenever a sequence of values is recorded in time order. As a result, a plethora of time series analysis methods for tasks such as classification, forecasting, anomaly detection, and others are emerging. However, a large amount of data is required to train, improve, and evaluate these methods. Large datasets are often very homogeneous, which makes training a model difficult. Furthermore, very little, if any, data is available for many domain-specific applications. As a result, new time series augmentation methods are developed. In this thesis, we organize the various methods and propose a method for evaluating them. We also present a tool to help with this task and demonstrate how to use it with various datasets and metrics.

## 1.3 PROBLEM DEFINITION

Many time series generation methods have been proposed in recent years. These methods use various techniques, and the results vary greatly ([Wen et al., 2020]). As a result, one naturally occurring question is: which method is the best? Answering this question is not as simple as it may appear, because it is necessary to first determine what constitutes a good generation. What format should the generated data take? Should it be the same as the original data? Should it be different but share some characteristics? Should one instead prioritize the preservation of certain features, and if so, which ones? What makes answering those questions difficult is that it is highly dependent on the task at hand. We want synthetic data that is similar to the original data when generating data to improve classification. Instead, if we want to improve an anomaly detection model, we need more diverse data. The goal of the generation is thus determined by the use we intend to make of the generated data.

Furthermore, a method that meets our requirements for one dataset may not meet our requirements for another. The best technique for a given situation is also determined by the properties of the input dataset (presence of a seasonal pattern, anomalies, etc.). In this paper, we look at various techniques for creating synthetic data and analyzing it in various ways (preservation of features, metrics score, visual comparison). To do so, we will first go over the main classes that are currently in use. Then, in (3 Generation Algorithms), we will show some implementations for each of the discussed classes and present a method for evaluating time series. The findings of the analysis are presented in (4 Experiments). Finally, in (5 Generation Tool), we will present a tool that we developed to facilitate the use and comparison of the techniques under consideration.

## 1.4 EXISTING WORK

Several algorithms for time series generation and augmentation have been implemented in recent years. The difference between generation and augmentation methods is that the latter begins with a small dataset and adds new data to it (in this work we will focus on these). Although their ultimate goal is the same, namely the generation of synthetic data to improve analysis models, they are very different. There are three main types of approaches: simple/basic approaches, model-based approaches, and machine learning approaches. The simplest method is to take an input dataset and generate new data by modifying/altering it. This can range from simple operations such as noise injection (for example, Gaussian noise, spikes, or trend) to more complex operations such as window cropping [10] and Dynamic Time Warping [11]. In general, we divide transformations into two types: time domain transformations and frequency domain transformations. The most common are time domain transformations, which, as the name implies, operate on the time aspect of time series. Most of them, but not all, operate directly on the original time series. [Gao et al., 2020] provides an example of this second case, in which operations on labels are used to improve anomaly detection. Frequency Domain transformations, on the other hand, work on the frequency aspect of time series. [12] provides another example.

A more elaborate method for generating time series consists in first building a model from existing data and then generating new data from this model [13]. [14], for example, focuses on developing a mixture of autoregressive models to generate time series covering the entire

feature space. Simple Autoregressive Models are more basic examples, in which each point is thought to be determined by the previous points. After determining the weight of each previous point, this model can be used to forecast future values. [13] provides an article on the subject. The third category is made up of (machine) learning methods. These methods employ machine learning models to learn existing data and then generate similar ones. [14] provides an example in which Variational Auto Encoders (VAEs) and Generative Adversarial Networks (GANs) are used to generate new data in order to improve anomaly detection. VAEs compress data to a latent space before reconstructing it. Reconstructing synthetic data will then be possible by randomly sampling from this latent space. GANs, on the other hand, are made up of two competing models, one creating synthetic data and the other attempting to distinguish original data from synthetic data. Other efforts to improve GAN adaptation to time series include [15].

## 1.5 THESIS OUTLINE

This MSc Thesis is divided as follows:

Chapter 2, analyzes the state of the art for synthetic data generation in DC and time-series data. Moreover, we will provide a short overview of the theory behind GANs.

Chapter 3 presents a concise explanation of the case study, covering the methodology used.

Chapter 4 provides a comprehensive analysis of the conducted experiments and the obtained results.

Chapter 5 reflects the proposal's conclusions, explores its limitations, and recommends coming steps to stay ahead..

# 2. LITERATURE SURVEY

In this section, we will go over the major techniques for evaluating time series. We will present a set of time series features and metrics based on these features that will be used to assess the generated time series. We'll explain how they work and why we think they're important.

## 2.1 SYNTHETIC DATA GENERATION

The essential utilization of manufactured information is as a method for information expansion, wherein new engineered tests are joined with existing information to mitigate either the issue of class irregular characteristics inside the information or the issue of information shortage. An original application for engineered information has been proposed. The idea is that manufactured information may be viable in gathering and keeping up with protection rules while sharing and distributing information [20]. Sharing data, like information, is fundamental for exploration to advance. In any case, while managing information that contains individual data, for example, wellbeing and monetary measurements, sharing data may be troublesome on the grounds that protection regulations might boycott it. Confidential data can be safeguarded by making engineered information and sharing it rather than veritable information [4]. Be that as it may, in this proposal, the justification for making manufactured information is to involve it for information expansion, fully intent on decreasing the expense and time expected for information assortment.

$$\mathbf{x}_t = \begin{bmatrix} x_{t1} \\ x_{t2} \\ \vdots \\ x_{tm} \end{bmatrix}, \quad t = 0, 1, 2, \dots . \tag{2.1}$$

## 2.2 TIME SERIES PROPERTIES AND FEATURES

The properties of time series can be characterized and classed. For example, a time series indicating daily temperature may have a "cyclical form" (greater in summer and lower in winter), while also exhibiting an increasing tendency (due to climate change). Then it may have a number of "spikes" (days with unusually high/low values) or the values could all be

inside a narrow range. All of this, as well as many other traits, may be mathematically defined by a set of features, which are summarized in part in this chapter.

### 2.2.1 Spectral Entropy

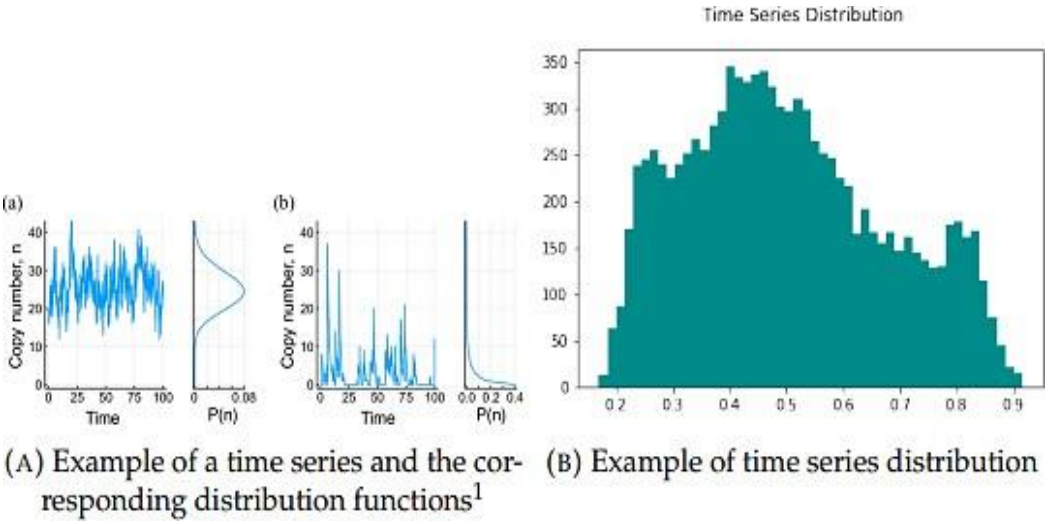
Shannon's Entropy is the quantity of information contained in a variable in Information Theory [21]. An intuitive approach to think about this is: how much storage would I need if I had to store the information provided by the variable? For example, if we needed to save the outcome of a non-biased coin flip, one bit would suffice. Because there are only two possible outcomes (head or coin), we could set the bit to 0 in the first case and 1 in the second. Similarly, if we had eight variables instead of two, we would want three bits. But what if the variables did not appear at the same rate? Assume we have three variables, a, b, and c. If a occurs 9 times out of 10, we might just keep a 0. If the variable is a b, we would store 10, and if it is a c, we would save 11. In this case, we would only need one bit 90% of the time and two bits 10% of the time. As a result, we require 1.1 bits on average. As a result, the Shannon's Entropy formula adds all the values with the logarithm of their frequency (the value will be negative because the log comprises a proportion). As a result, the result is multiplied by -1).

$$E = -\sum_{i=1}^{n} p_i \cdot log(p_i) \tag{2.1}$$

Shannon's Entropy is normalized as Spectral Entropy (between 0 and 1). It can be used to assess the disorder in a time series as well as its forecasting capacity. The higher the frequency of some values, the lower the Spectral Entropy, and thus it is easier to "predict" the next value.

### 2.2.2 Distribution

Another significant feature of a time series is the way its values are spread. If the values are normally distributed (Gaussian distribution), the mean and variance of this distribution are sufficient to completely characterize it. However, this is not always the case. To depict the distribution of a time series, we can divide the potential values into groups and count how many points exist in each category. The findings can then be plotted using a histogram, as shown in Figures 2.1a and 2.1b.




(A) Example of a time series and the corresponding distribution functions[1] (B) Example of time series distribution

**Figure 2.1:** Time Series distribution

### 2.2.3 Autocorrelation

Given two variables (in this case, two time series), the correlation between them is a measure of their similarity. It assesses how they differ in tandem. As defined by the correlation function,:

$$Corr(x,y) = \frac{Cov(x,y)}{\sigma(x)\sigma(y)} \tag{2.2}$$

Cov(x, y) indicates the covariance of x and y, and (x) means the standard deviation of x. It can go between - 1 (totally bad relationship) and +1 (entirely certain connection) (wonderful positive connection). A period series' autocorrelation is its association with itself, as the name suggests (moved by n steps). The autocorrelation function calculates its association with a copy of itself displaced by one step, two steps, three steps, and so on. Figure 2.2 depicts two autocorrelation functions.

## 2.3 TIME SERIES DATA

In this work, we principally center around working with sensor information that falls into the time series area. A period series is basically a bunch of information things that are requested by time. A period step is a solitary data of interest at any second in time. Information from a sensor keep stickiness and temperature values in a house consistently or

minute are instances of tactile time series, similar to a video recording, which is only a camera catching pictures at each occurrence of time, all the more formally known as casings. Essentially, an organization's stock value, which is recorded consistently, falls under the space of time series.

### 2.3.1 Time Series Forecasting

The most common way of extending information for future time steps is known as time series anticipating. It tends to be imagined as a component that cycles or examines past information and afterward utilizes that information to create forecasts about what's in store. Estimating the cost of an organization's stock the following day, for instance. There are two types of time series anticipating [22]:

a. One-step ahead prediction: This approach is simply worried about estimating the following time step in light of the time series. For instance, given 10 seconds of information, anticipate the worth of a sensor in the eleventh second.

b. Multi-step ahead prediction: This is engaged, as the name infers, with foreseeing a grouping of time steps given a period series. For instance, estimating the climate for the following 10 days in view of authentic information.

### 2.3.2 Regression Based Time Series Forecasting

Conventional factual strategies, for example, ARMA and ARIMA [4,] as well as some AI strategies, for example, Backing Vector Machines and fresher, further developed brain network-based determining methods, are commonly worried about anticipating (P(xt+1|X)), where xt+1 is the future expectation and X is the verifiable time series. These strategies, as a rule, follow mean relapse standards, and thus, they acquire the essential imperfection of these standards, to be specific, they do exclude varieties around the mean worth. Thus, now and again, the outcomes might deceive. Subsequently, we should apply a methodology that considers each of the varieties around the mean, which can be tackled utilizing an interaction known as probabilistic determining.

### 2.3.3 Missing Data

Missing qualities in a period series are a significant issue in genuine order and estimating applications. It is generally usually brought about by sensor disappointments, reboots, or human misstep while recording information, yet it can likewise be brought about by limited information securing assets or contrasting procurement paces of various sensor modalities. Missing information presents errors in information, which can prompt an unexpected circulation in comparison to the veritable information dissemination. However, contingent upon the beginning of the missing information, the holes in the information will have various circulations. For instance, in the event that missing information is brought about by changing sensor rates, the missing information time allotments will have a cyclic, rehashed design, however missing information brought about by sensor disappointments will have long, bordering blocks with capricious beginning times. Realizing these dispersions might be valuable since they can give more data about the missing information design. We will unequivocally plan to comprehend this example in the methodology introduced in this postulation, and we will portray how we might utilize this data to figure information when we have time series with deficient data.

In light of the measurable elements of the example with which information is missing, systems prompting missing information can be ordered into three gatherings.:

a. Missing completely at random (MCAR): There is no example in the manner the information is absent in MCAR. Missing data centers happen totally at irregular. This infers that there are two necessities: First, the probability of missing a discernment is unaffected by the advantages of different characteristics. Second, the probability of a discernment being missing is free of its worth. In the univariate case, this shows that the probability of a solitary sense being missing is free of its situation in the series. In the event that An is the vector showing whether every information thing will be absent or accessible, and Bobserved and Bmissing are the noticed and missing information thing values, respectively, then MCAR corresponds to the case where the probability of data items missing is independent of all data:

$$P(A|B_{observed}, B_{missing}) = P(A) \tag{2.3}$$

2. Missing aimlessly (Blemish): Like with MCAR, the probability of a perception being missing in Blemish is autonomous of the worth of the perception. It is, be that as it may, subject to different conditions. Since there can be no different boundaries other than time in a univariate time series, the gamble of missing a discernment in Flaw is subject to the planning of this insight.

### 2.3.4 Seasonal Time-Series Sata

An occasional impact in time series information is distinguished when the information acquires occasional factors like week by week, quarterly, or yearly redundancies. Occasional vacillations can be brought about by normal impacts like changing seasons or atmospheric conditions, or by man-made shows like design or propensities. Figure 2.2 is an example diagram that is occasionally repeating and displays a cyclic example.

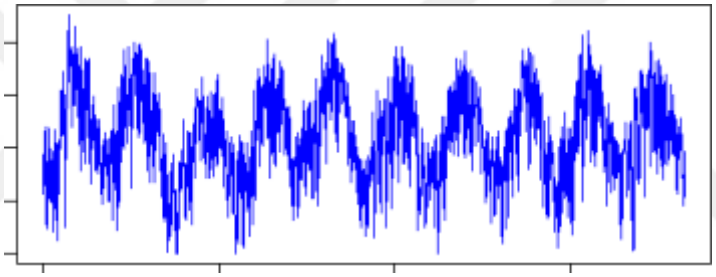


**Figure 2.2:** A sample seasonal time-series graph with cyclic variations [34]

### 2.3.5 Irregular Time-Series Data

Figure 2.3 portrays a period series diagram that follows no example and shows a great deal of clamor in the information. The factors found in the information in this proposal are frequently of this kind of sporadic information, making it hard to get a handle on and survey their circulation.

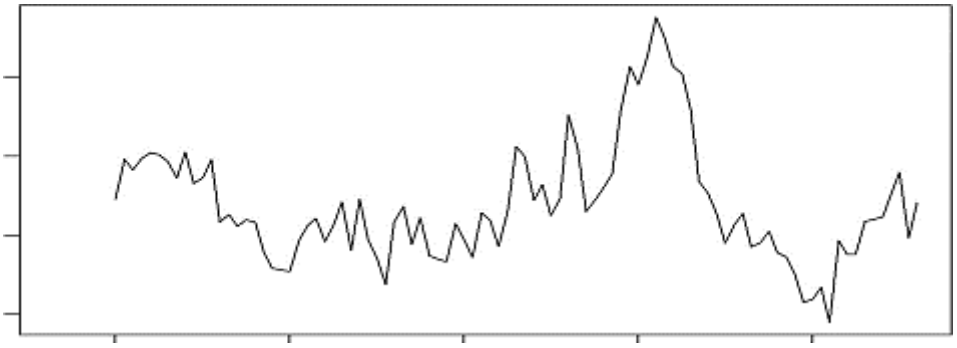


**Figure 2.3:** A Sample Irregular Time-Series Graph Without any Definite Patterns [23]

### 2.3.6 Univariate and Multivariate Data

Time-series data is in like manner gathered into two classes considering how much ascribes. Multivariate data is data that has more than one variable in the time-series data that is gotten at each time event. At the point when just a single variable shows up in the information at each time occasion, the information is alluded to as univariate time-series information. This is worried about multivariate information, which comprises of numerous segments of elements.

## 2.4 NEURAL NETWORKS

The neurons are every now and again portrayed as being stacked in layers, as displayed in Figure 2.4, with the organization engineering characterizing the sizes and association designs inside and between these layers [22]. A brain network is described as a solitary layer brain organization on the off chance that it has just two layers, one for input signs and one for yield signals. The organization is characterized as diverse in the event that there are at least one extra levels, known as covered up layers, between the info and result layers.

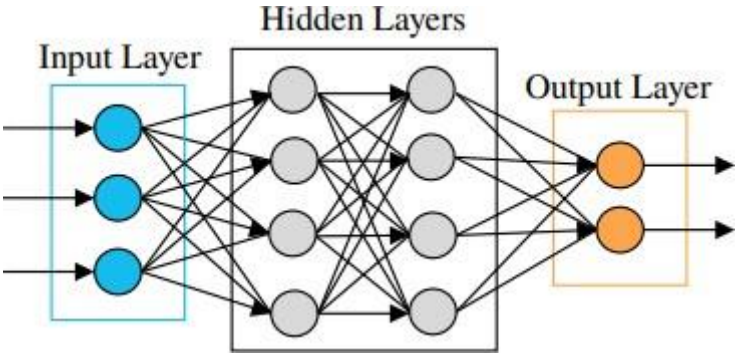



**Figure 2.4:** A Simple Illustration of A Neural Network [23].

The mind's association will doubtlessly get to know a nonlinear capacity that best deciphers an assortment of commitments to an assortment of results. The heaps of this association are changed during arrangement, ordinarily through backpropagation, until the best ones are found. The tendencies of a hardship capacity are recorded, and this setback or the model's presentation during readiness can be seen to decide when the ideal burdens have been reached. A neuron in a cerebrum association can be mathematically conveyed as,

$$y_k = \varphi(\sum_{j=1}^{m} w_{kj}x_j + b_k) \tag{2.4}$$

In this situation, x1, x2,..., xm are the information signals, wk1, wk2,..., wkm are the neuron loads, bk is the additional predisposition factor, is the actuation capability, and yk is the result signal [23]. The actuation capability determines how the weighted absolute of the data sources is converted into a result, or how the planning among sources of info and results shows up. The predisposition is utilized to change the enactment capability's net contribution by expanding or diminishing it. There are different sorts of initiation capabilities. The following are a couple of models:

### 2.4.1 Recurrent Neural Network

Feedforward and recurrent neural networks are two types of neural networks. The feedforward network only allows information to flow in one way, and that implies that sources of info are provided to the information layer and afterward to the result layer, perhaps through at least one secret layers. Something like one criticism circle in a repetitive brain organization (RNN) permits contributions to be taken care of back to past layers [23]. RNNs

are usually used to lead AI errands on successive information like time series, text, and sound. Since they catch the dependence of the consecutive data in the information, RNNs perform well in applications, for example, machine interpretation [43], time series determining [41], and discourse acknowledgment [19], among others. A RNN model's objective is equivalent to that of a feedforward brain organization: it figures out how to plan a bunch of contributions to a bunch of results. In a RNN, nonetheless, the preparation cycle can likewise utilize criticism to find the ideal loads. Figure 2.5 shows a basic portrayal of a RNN.

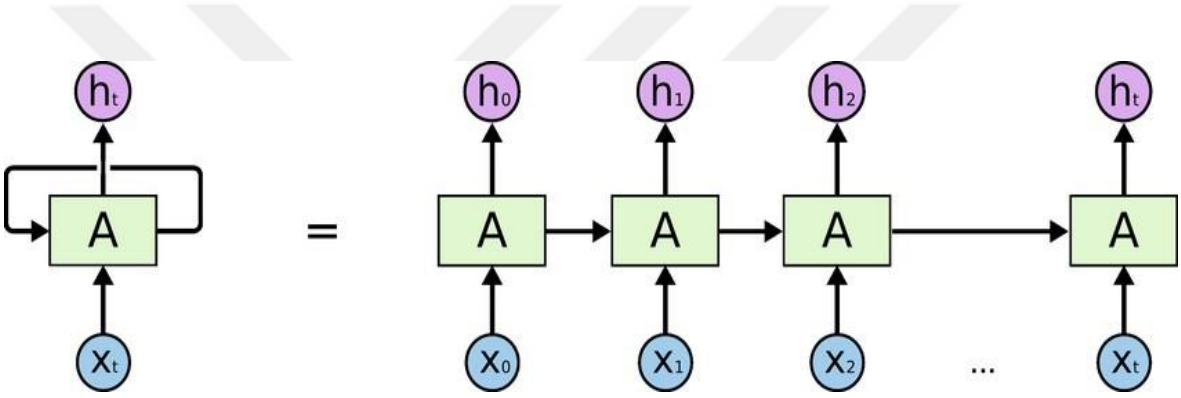



**Figure 2.5:** A Basic Outline of A RNN

### 2.4.2 Long Short-Term Memory

The fundamental issue with RNNs is that they have vanishing gradients, which means that when they are back-propagated, the angles by the same token "explode" (keep an eye on boundlessness) or "disappear" (will generally zero). [25] proposed the Long Transient Memory (LSTM), which is a RNN with a particular design that somewhat takes care of this issue. The LSTM is comprised of memory cells as well as three entryway units: an information door, a result entryway, and a neglect entryway. The cell stores values over the long run stretches, while the door units control the progression of data all through the hubs. The doors fundamentally figure out how to store the helpful data and kill the superfluous data. The LSTM, similar to the RNN, has a chain-like construction. The LSTM, on the other hand, includes four neural network layers rather than one. Figure 2.6 depicts an illustration of the LSTM unit.

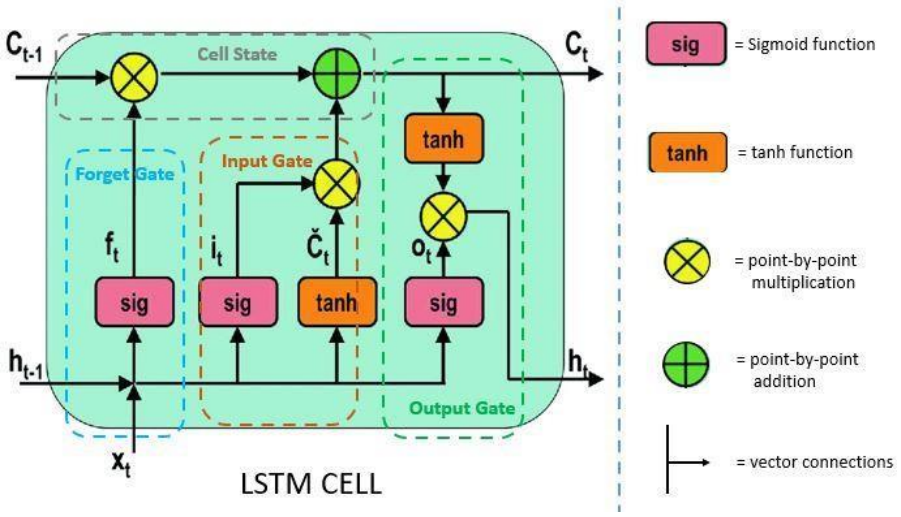



**Figure 2.6:** An Illustration of A LSTM Unit [38].

The info, neglect, and result entryways control the data put away in this cell. Each door is comprised of a sigmoid initiation capability, and a component wise duplication activity. [25]

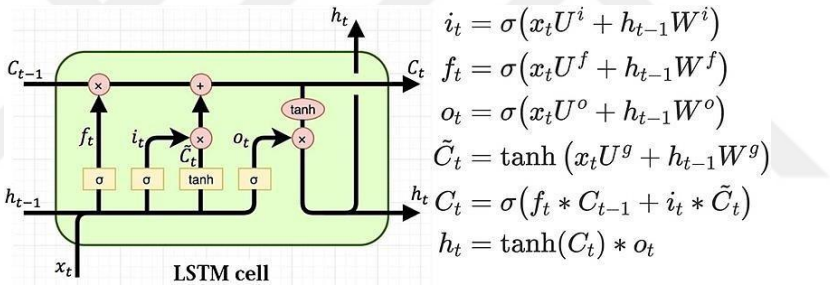



**Figure 2.7:** LSTM Equations

The forget gate, input gate, output gate, and cell activation vectors are represented by f, I o, and c, respectively. These are a similar size as the cell yield enactment vector h, as well as the weight and predisposition frameworks. shows the activity of component wise increase and the capabilities and mean the cell information and cell yield actuation capabilities, which are comparable to the tanh enactment capability [26]. The sigmoid initiation capability result can be seen as a size of how critical the data is, with 1 relating to critical and 0 comparing to really repetitive. By expanding ft by the ongoing data put away in memory cell ct1, the data passed judgment on excess by the sigmoid enactment capability is erased. New data can be added to the memory cell by adding this to its result and the new competitor valuesc t. At long last, the result door figures ot and afterward ht.

### 2.4.3 Autoencoder

Autoencoders are neural networks that have been trained to recreate their input [16]. The network is made up of two parts: an encoder capability and a decoder capability. The autoencoder is intended to inexact a duplicate of the info information and was generally utilized for dimensionality decrease or component learning. The autoencoder can be shown utilizing distribution [24], which analyzes the organization initiations on the first contribution to those on the recreations. Nonetheless, autoencoders are much of the time prepared involving similar methodologies as feedforward networks, normally backpropagation followed by inclination plunge. It is educated to diminish remaking mistakes, which are characterized as the distinction between input information x and recreations x. This is frequently alluded to as a reproduction misfortune, characterized as:

$$\mathcal{L} = \| \mathbf{x} - \tilde{\mathbf{x}} \|^2 \tag{2.5}$$

Autoencoders are portrayed into two sorts: undercomplete and overcomplete. When the encoder maps the commitment to a less conspicuous viewpoint than the data, the affiliation ought to be inadequate. The affiliation ought to be overcomplete when the encoder maps the obligation to a more unmistakable point of view while considering everything. An autoencoder is displayed in Figure 2.8. The encoder changes over the information x into latent codes h, which can have a lower or higher need. The decoder then moves these codes to the multiplications x in the essential point.

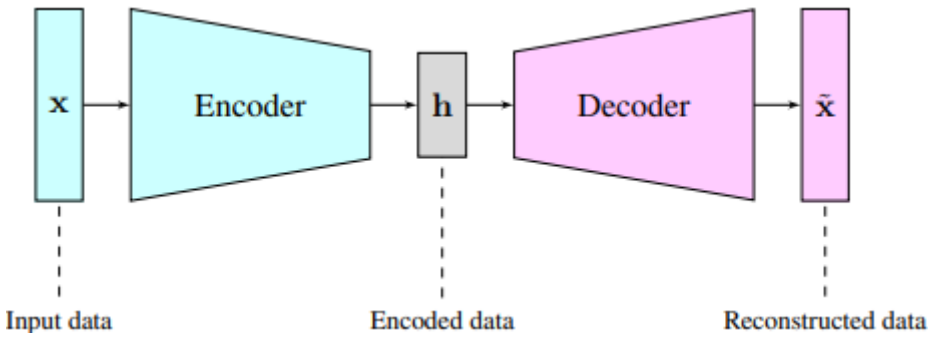



**Figure 2.8:** General Structure of An Autoencoder.

## 2.5 GENERATIVE MODELS

Generative models are those that unequivocally or certainly get familiar with the dissemination of both information and result information [5]. A generative model for pictures, for instance, may discover that a nose is probably not going to happen wherever other than in the center of a face, or that a canine's tail is probably going to show up toward the finish of their body. A discriminative model, then again, just figures out how to recognize a picture of a canine and a picture of whatever isn't a canine by drawing In the data space, a detaching line. Accepting the line is accurately drawn, the discriminative model can recognize pictures of "canine" and "not canine" by seeing which side of the line the picture is on in information space. Figure 2.9 portrays a reasonable illustration of this.

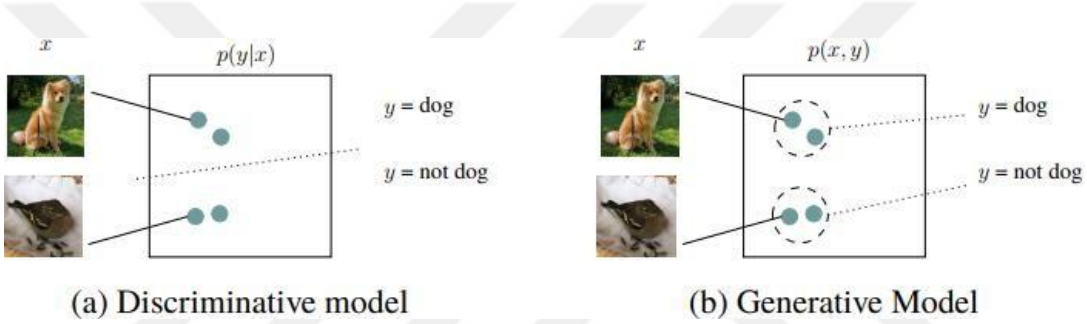



**Figure 2.9:** Simple Illustration of A Discriminative Model.

## 2.6 GENERATIVE ADVERSARIAL NETWORKS

[17] at first proposed the generative ill-disposed network (GAN) in 2014 as another system for assessing generative models through an ill-disposed approach.

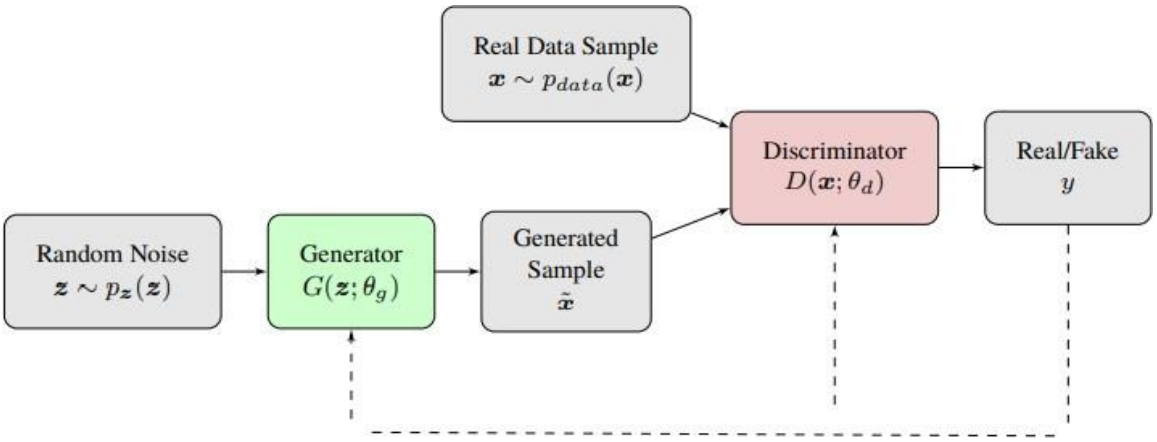



**Figure 2.10:** General Design of a GAN.

Hued boxes address the brain network parts in the model, while dim blocks address their bits of feedbacks and results as a clamor grouping, an information test, or a characterization. The strong lines address inclination forward spread, while the specked lines address slope backpropagation. Figure 2.10 portrays the GAN structure, which integrates two brain organizations, the generator and the discriminator. The objective is to prepare these two models simultaneously, with the generator endeavoring to catch the certified dissemination and the discriminator deciding if an example has a place with the genuine dispersion or was examined from the generator [27].

The conveyance of generators Pg is learned over information x by first making an earlier on input clamor factors pz(z), and afterward letting the generator to foster a planning capability from this before information space as G(z; g). The discriminator is prepared to amplify the probability of accurately marking the examples, or at least, to recognize whether the example came from the certified dissemination or from the generator.

## 2.7 TIME SERIES GAN

This model, dubbed TimeGAN (Time Series Generative Adversarial Network), combines the freedom of the unsupervised GAN framework with the control provided by supervised training in autoregressive models.

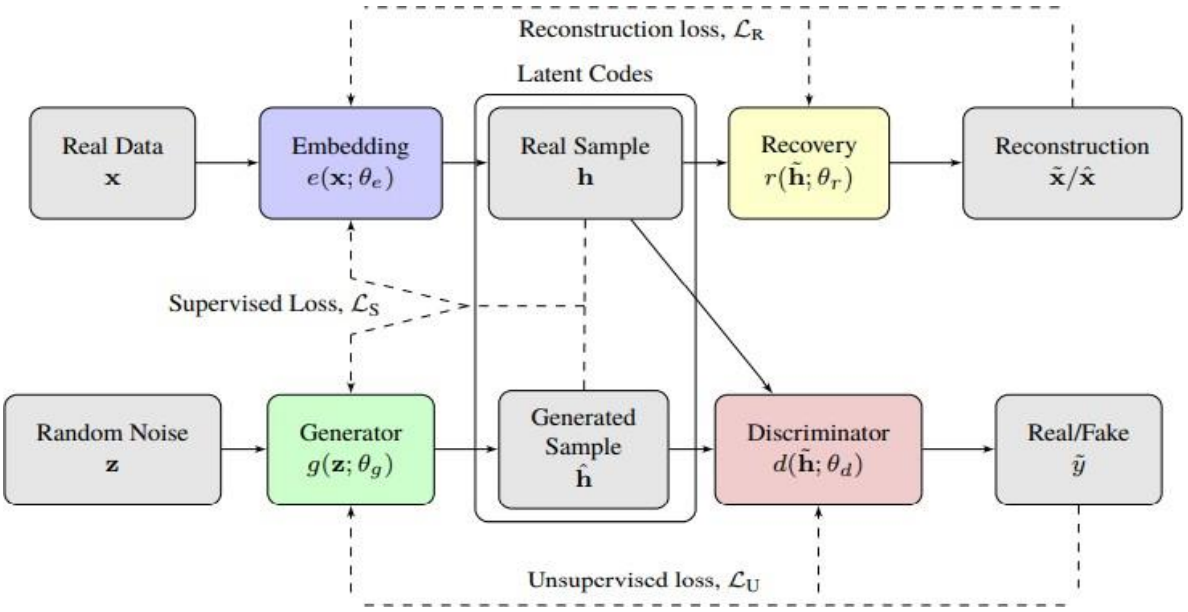



**Figure 2.11:** General Structure of TimeGAN. [30]

Tinted blocks address the model's cerebrum network parts, while dim blocks address their pieces of criticism, bringing about an uproar gathering, a data test in one or the other feature or inactive space, a diversion, or a portrayal. The solid lines are worried about point forward spread, though the spotted lines are worried about incline backpropagation.

The TimeGAN consolidates an additional two brain organizations, the auto-encoding parts implanting and recuperation capabilities, notwithstanding the ill-disposed parts (generator and discriminator) of a standard GAN. These auto-encoding parts can be seen as a solitary autoencoder that gives mappings among highlight and dormant space and is prepared pair with the ill-disposed parts. Subsequently, the TimeGAN figures out how to encode qualities, develop portrayals, and emphasize through time all simultaneously [31]. Figure 2.11 portrays a worked on portrayal of this worldview.

## 2.8 TIME SERIES EVALUATION METRICS

When describing time series on their own, features are in handy. When comparing various time series, we have another tool: metrics. The features are used in metrics to define the relationship between two (or more) time series. In our instance, this is very valuable for comparing generated data to the original.

### 2.8.1 Mutual Information

The Mutual Information is the first major statistic used to compare the similarity of two collections of data ([Dionisio, Menezes, and Mendes, 2004]). Given two variables, v1 and v2, the Mutual Knowledge evaluates how much information we can learn about v2 from v1 (and the other way around, as it is symmetric).

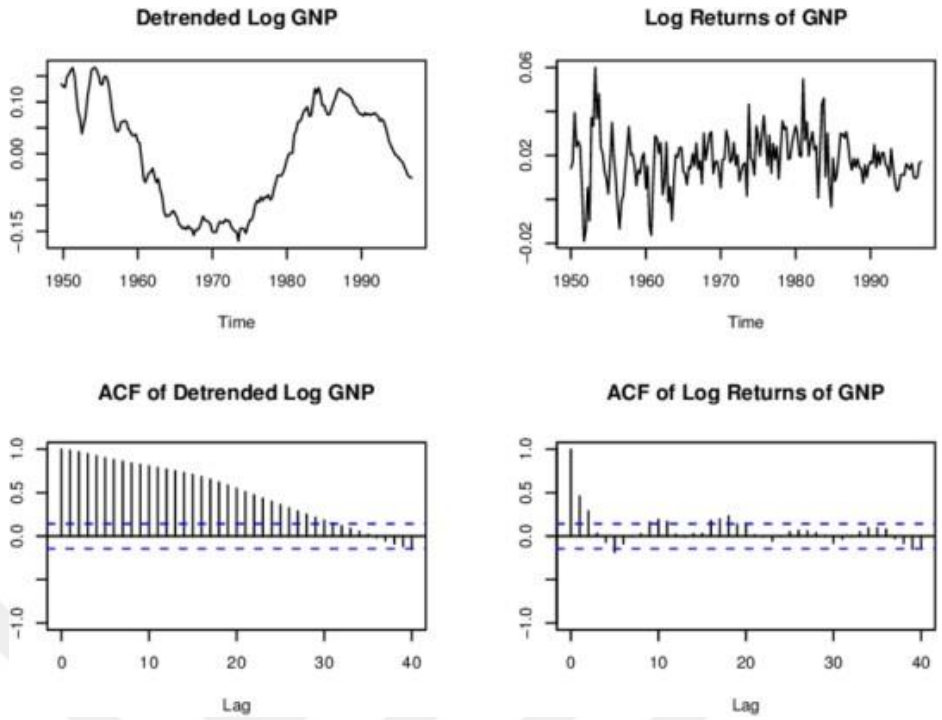



**Figure 2.12:** Example of Autocorrelation Function [32].

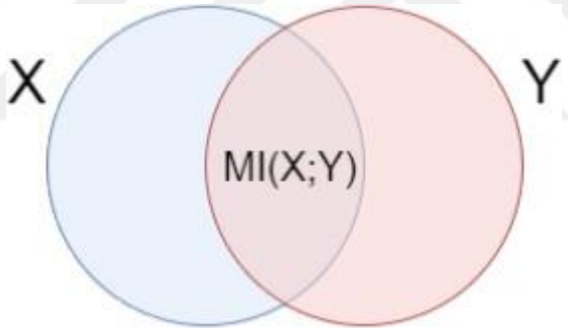



**Figure 2.13:** Mutual Information [32].

Given two time series s1 and s2 (the original and the synthetic), Mutual Information can be used to quantify their similarity, hence evaluating the creation technique.

### 2.8.2 Correlation

Correlation is a measure of how comparable two variables are (in this case, between two time series). It is calculated using:

$$Corr(x,y) = \frac{Cov(x,y)}{\sigma(x)\sigma(y)} \tag{2.6}$$

Cov(x, y) denotes the covariance of x and y, and (x) denotes the standard deviation of x. Its value spans from +1 to -1, with +1 indicating that x and y vary together, -1 indicating that they vary completely oppositely, and 0 indicating that they vary separately from each other. Figure 2.14 depicts three instances of correlation.

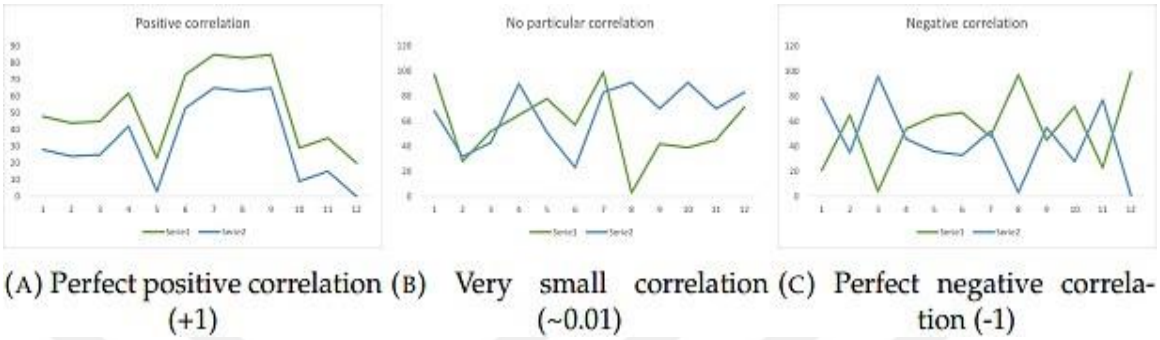



**Figure 2.14:** Examples of Correlation

### 2.8.3 Root Mean Squared Error (RMSE)

The Root Mean Squared Error (or Deviation), as illustrated in Figure 2.15, is another significant statistic for evaluating a generation model. The RMSE adds the squares of the differences between each point in the time series and the model given a model and a time series. Squaring them guarantees that the values are positive (otherwise, positive and negative errors would compensate for each other) and gives larger errors a higher weight. Taking the root at the end guarantees that the result has the same scale as the data.

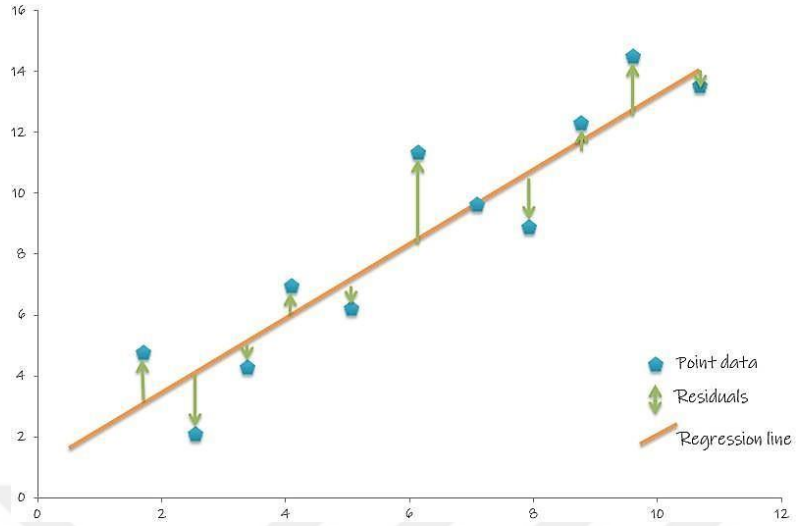



**Figure 2.15:** Root Mean Squared Error

As depicted behind the scenes area, GANs have filled in ubiquity and utility, exhibiting their capacity to produce top notch information in areas where information is scant. While even the most developed GAN models[18, 2, 3] are normally used to produce pictures, GANs are not restricted to visual information. There have been various articles in which GANs have been utilized to create consecutive information with great results[26, 27, 28, 29]. While managing non-time series information that has missing qualities, the underlying step is to finished the information utilizing techniques like zero attribution, mean ascription, or mode ascription. [30] presents an inside and out assessment of missing information attribution draws near. There are a few GAN-based information attribution structures too. Acquire [31] was made with the restrictive type of GAN. The generator yields a completed form of the first information in light of the first information and a cover (saying which values are noticed or missing), the two of which are provided to it as a framework. This completed information is taken care of into the discriminator, whose goal is to yield probabilities for every section in the grid as opposed to the whole attributed lattice. They report RMSE values, which seem to beat customary ascription draws near.

# 3. PROPOSED SYSTEM

## 3.1 INTRODUCTION

The fuse of a high extent of inexhaustible age into power frameworks requires a rising need to demonstrate the dubious and discontinuous nature of these assets. Situation age, which gives the framework administrator a bunch of conceivable future acknowledge, is a significant strategy for portraying the way of behaving of inexhaustible assets. In contrast with deterministic point conjectures or probabilistic figures [1,] situation estimates couldn't educate clients regarding future vulnerability, yet in addition mirror the fleeting reliance of sustainable power age. [2], [3]. The data given by these produced situations is helpful for an assortment of navigation and stochastic streamlining issues, including sustainable power dispatch [4], [5], unit responsibility [6], [7], and numerous others. Thus, numerous calculations have been created lately for different applications going from load anticipating to wind and sun based power age.

One of the most troublesome parts of situation anticipating is demonstrating and learning the hidden stochastic cycles that drive environmentally friendly power age [8]. Past factual or actual strategies, for example, the first-request autoregressive model [9], group techniques, and Gaussian Copula [2-10], required areas of strength for either presumptions or definite actual estimations and demonstrating. Besides, most of these techniques are worried about catching the peripheral circulation of every individual schedule opening of the determining skyline, while trying to ignore transient relationships in the situations [12].

## 3.2 GAN TRAINING IMPROVEMENTS

As made sense of in segment 2.2, preparing GANs depend on a minimax game between the Generator and Discriminator organizations. This approach can be ventured into a more extensive objective for the whole GAN, which is to make the created information conveyance seem to be like the genuine information dispersion. Practically speaking, preparing a GAN is a troublesome and unpredictable cycle. Several recommendations have been presented in recent years to address some of its most common issues.

One of the most challenging GAN failure problems is the vanishing gradients, which consists of the following. The task to be performed by the Generator network is much more complicated, so in most cases, the Discriminator becomes better promptly. In the beginning, this is not a severe problem because the Discriminator can give useful feedback to the Generator. However, if the Discriminator becomes too good at doing its job, the Binary Cross-Entropy (BCE) cost function approaches flat regions (Figure 3.1). Thus the gradients are very close to 0, and the feedback is no longer useful to the Generator.

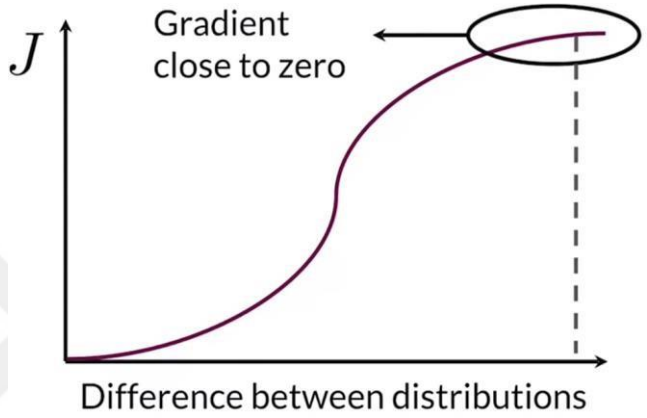



**Figure 3.1:** Vanishing Gradients Problem.

Other main problems suffered by GANs are those related to mode collapse. A mode in a data distribution is an area with a high concentration of observations. Real-world datasets are usually multimodal. Mode collapse occurs when the Generator gets stuck around a single-mode and cannot generate samples from the other modes. Mode dropping happens when the Generator fails to represent all the modes from the real data, meaning that some of them are missing. Figure 3.15 shows an example of a mode collapse.

The ilustrations in Figure 3.2 represent the Generator's heat map along with several steps during the training, and the final column presents the real data distribution. We can notice how the Generator rotates through the different modes trying to fool the Discriminator without ever converging or producing samples of different modes simultaneously. The use of the BCE cost function in GAN training leads to critical problems such as vanishing gradients and mode collapse. In the following section, we will review some of the most successful proposals to mitigate these issues.

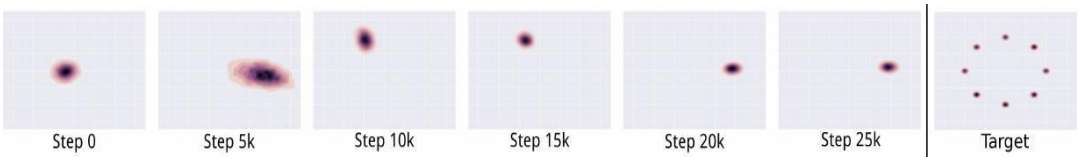



**Figure 3.2:** Mode Collapse in a GAN. The Last Column Shows the Real Data Distribution.

### 3.2.1 Wasserstein GAN with Gradient Penalty

With the W-Loss, there is no longer a limit between 0 and 1 on the outputs of the Discriminator. Instead, it continues to grow regardless of how apart the distributions are, and it does not have flat regions. The W-Loss solves the vanishing gradients problem and reduces the likelihood of mode collapse. Figure 3.3 provides an illustrative comparison between the BCE and W-Loss cost functions.

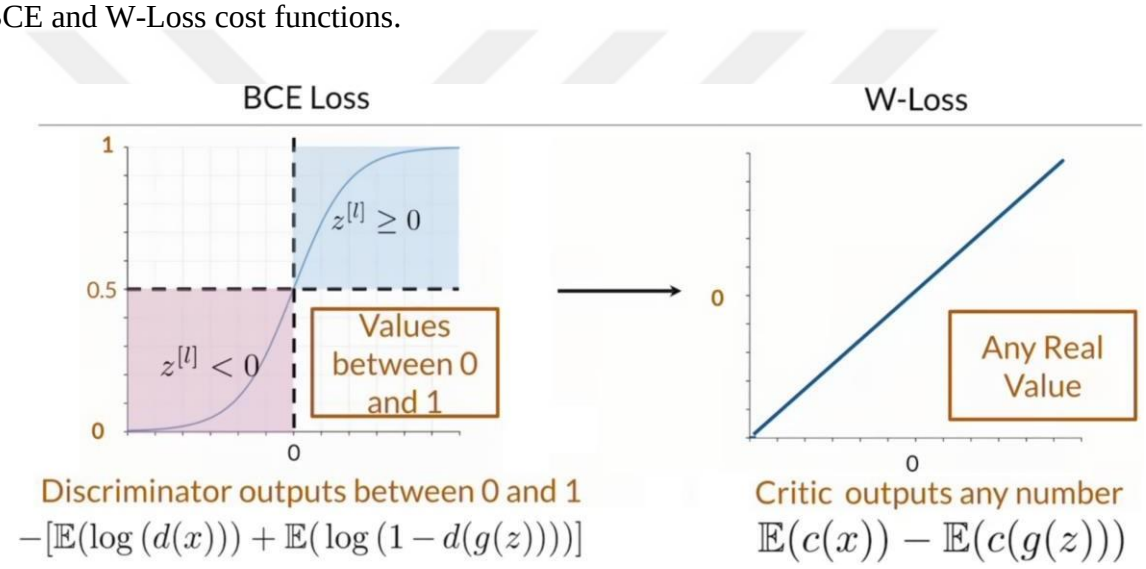



**Figure 3.3:** Comparison of BCE Loss and W-Loss.

It is crucial to note that WGAN requires a particular condition on the Discriminator network for stable training: the W-Loss cost function must be 1- Lipschitz continuous. For a function to be 1-Lipschitz continuous, the gradient norm (i.e., slope) must be less than or equal to 1 at all points (Figure 3.4). This condition is critical because it ensures that the W-Loss function is continuous, differentiable, grows linearly, and correctly approximates the EMD. If this condition is met, the training will remain stable.

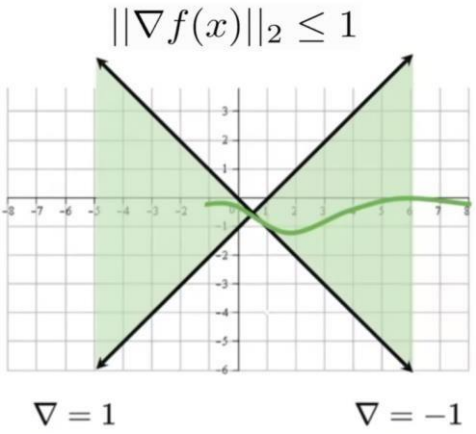



**Figure 3.4:** 1-Lipschitz Continuity Condition of A Function.

The WGAN authors proposed the Weight Clipping method [38] to assure 1-Lipschitz continuity. They propose constraining the weights of the Discriminator network to a set interval after updating with the gradient descent approach. It requires extensive hyperparameter adjustment to function properly. The Empirically Gradient Penalty approach produces higher quality outcomes more consistently. Its implementation requires simply the addition of a regularization factor to the W-Loss cost function.

### 3.2.2 Additional Training Improvements

[35] presented Spectral Normalization as another effective strategy for improving the stability of GAN training. Spectral Normalization improves stability and avoids vanishing gradient issues like mode collapse.

The Two Time-Scale Update Rule was proposed by [36]. (TTUR). This strategy entails increasing the learning rate in the Discriminator over the learning rate in the Generator. The preparation is more predictable and approaches Nash's balance quicker. The creators show that this technique beats the first WGAN-GP proposition exactly., which used more training steps in the Discriminator but had the same learning rate in both networks.

[37] study co-authors gave a session titled "How to Train a GAN?" at Neural Information Processing Systems (NIPS) 2016. [39] summarizes various tips for stable GAN training. Some of the tips we will employ in this task are as follows: I standardize inputs between - 1 and 1; (ii) example commotion from Gaussian dispersions; (iii) make separate bunches for valid and misleading information; (iv) utilize the LeakyReLU enactment capability; (v)

utilize the Adam enhancer [40]; (vi) use Implanting layers for discrete factors; and (vii) use regularization techniques like Dropout and Group Standardization .

### 3.2.3 GAN Data Generation Improvements

In the original GAN proposal, only the Generator network is used to produce new samples after the training phase. To this end, random noise vectors from the latent space are introduced into the Generator. If the training has produced a perfect Generator, its probability density function pG should be the same as that of the actual data. Unfortunately, GANs do not converge to the same distribution of the actual data in practice [16]. Thus, some of the generated samples will not look similar to the original training data.

The inconsistency of pG leads us to consider a different probability density function, the one implied by the Discriminator concerning the Generator pD. Typically this distribution is closer to the actual data distribution because the Discriminator has learned information that can be useful in many cases to correct the Generator. Nevertheless, generating samples of this new distribution is not as straightforward as when using only the Generator. To address this challenge, [8] propose the Metropolis-Hastings Generative Adversarial Network (MH-GAN). A sampling method based on a Markov Chain Monte Carlo (MCMC) approach. In this work, we will implement this proposal.

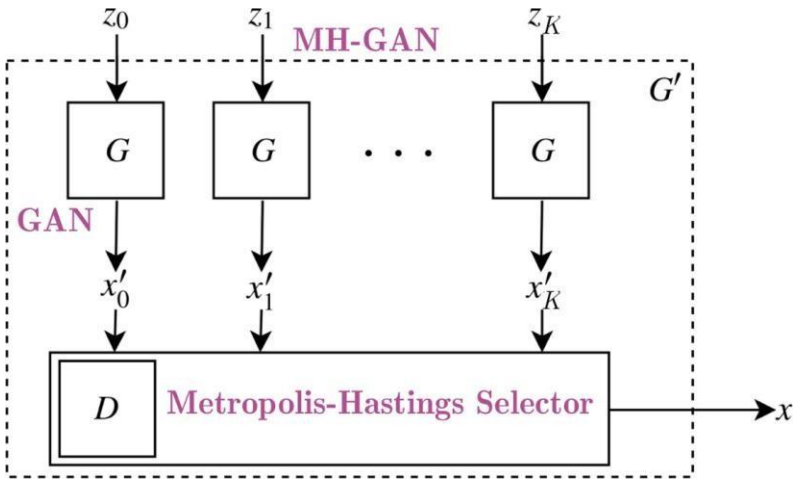



**Figure 3.5:**Metropolis-Hastings Generative Adversarial Network. [8]

The Metropolis-Hastings approach works as follows. We take $K$ samples from the proposed distribution (i.e., the Generator distribution). Then, we choose one of these samples and sequentially decide whether to accept the next sample or keep the previous one, based on an acceptance rule (Figure 3.5). The decisive feature of MH-GAN is that the acceptance probability can be computed through the probability densities ratio $^{p}D$ , and this is available at the Discriminator output

In practice, to avoid long burn-in periods, the authors recommend initializing the sampling process with a randomly chosen real data sample. It is also highly desirable to use a calibration method (e.g., isotonic regression model) for the Discriminator outputs since it is impossible to achieve a perfect Discriminator empirically. Through this simple post-training method, we can significantly improve the fidelity of the generated data samples.

## 3.3 PROPOSED SYSTEM

Having studied the methods for developing our models satisfactorily (a priori), we will explain the proposed architecture and data management in depth. We find the workflow that has been applied for the development of all models. It consists of three main phases: (i) Data Wrangling; (ii) Training; (iii) Scenario Generation. Between the second and third phase, there is a feedback loop focused on hyperparameter tuning.

In the scenario generation phase, we will use two approaches. One of them is to explore the latent space randomly. Since most of the dataset does not contain any anomalies, this random generation will presumably be conservative in most cases and will occasionally produce less

usual outputs. The second approach is based on MH-GAN for high-fidelity sampling, as explained in the previous section.

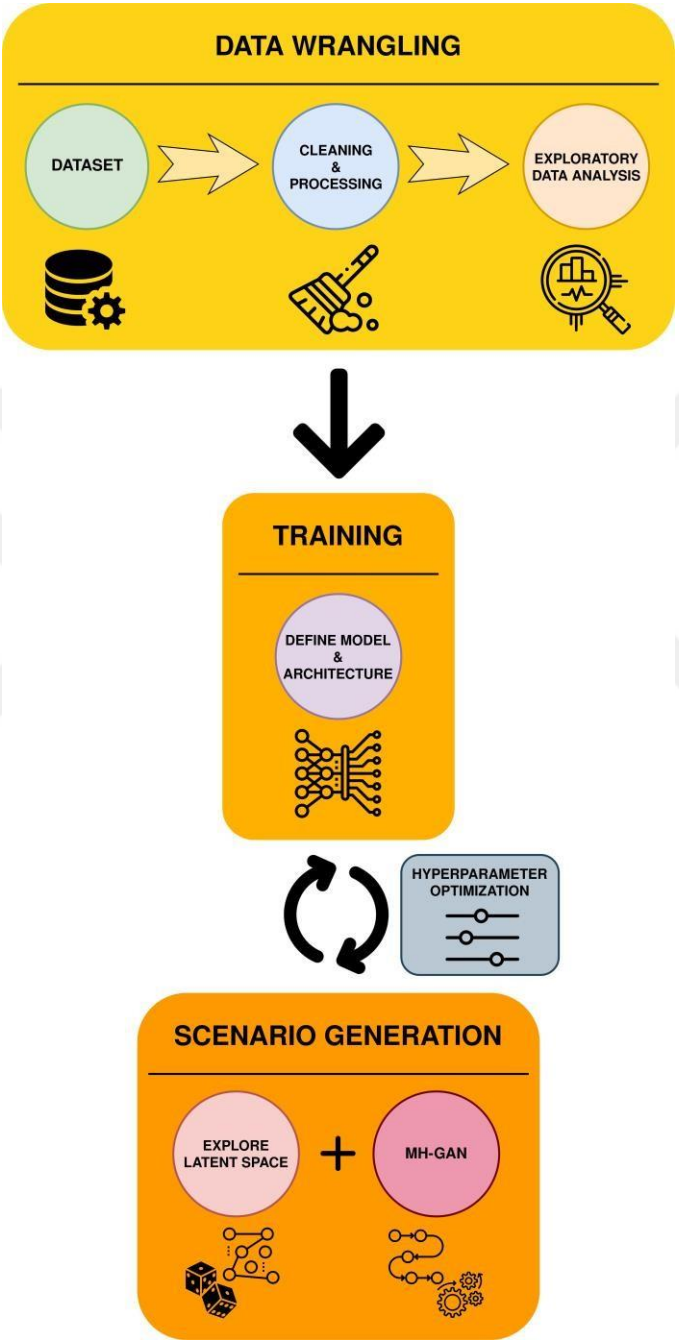



**Figure 3.6:** Model Development Workflow.

Concerning the first phase of data wrangling, the dataset provided by TycheTools has been processed and cleaned for its use in the neural network models. The data has been normalized between values of -1 and 1, as advised by [8]. Figure 3.7 illustrate the proposed Generative adversarial network, specifying each model's inputs and outputs, and data management during training. The Generator network has three inputs. The sensor ID is introduced in the first one, and an Embedding layer will treat it since it is a categorical variable. In the second one, the historical time-series data of the sensor being studied is introduced, which can be treated using LSTM or CNN layers. Finally, in the last entry, a random noise vector sampled from a multidimensional Gaussian distribution is introduced. Figure 3.7 provides an example of the time-series data fed into the neural models and the ground truth next step.

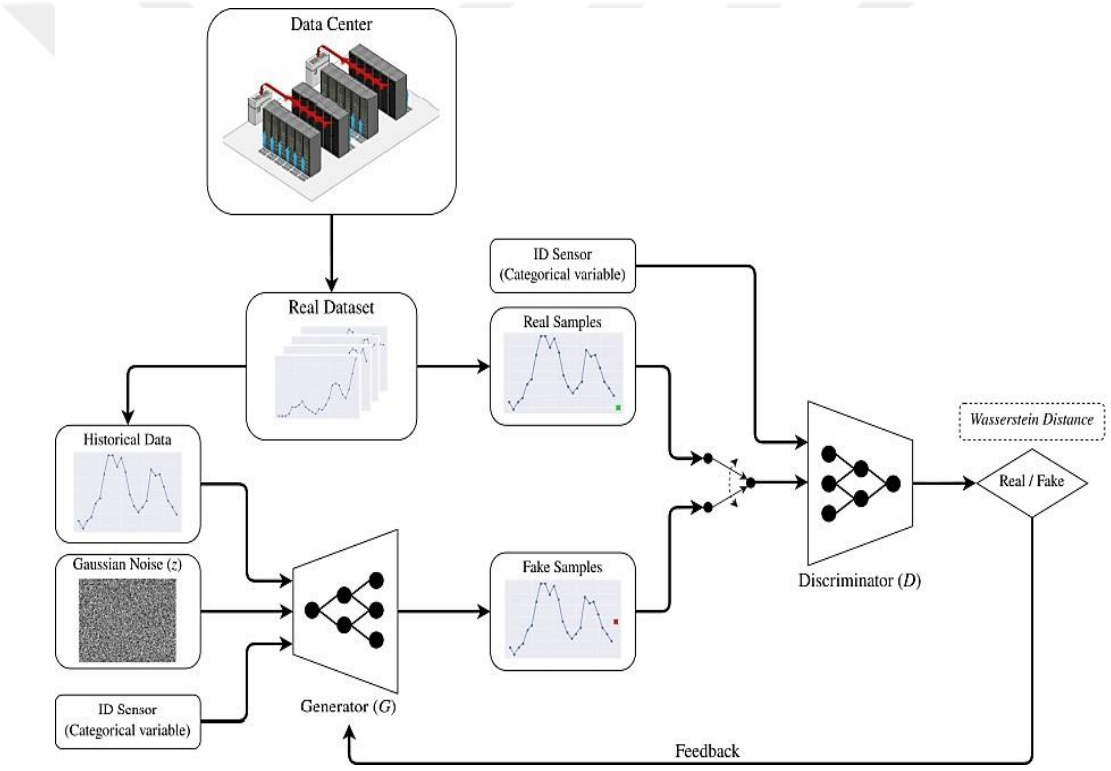



**Figure 3.7:** Complete GAN Architecture.

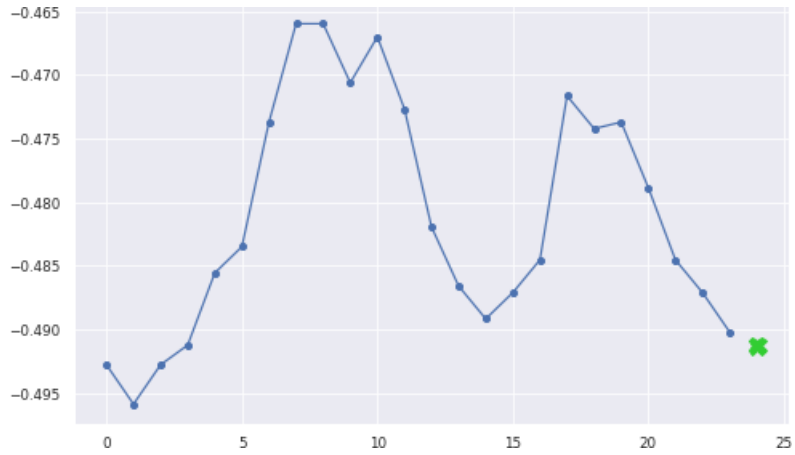



**Figure 3.8:** Time-Series Input Example. the Green Cross Represents the Ground Truth Next Step in The Time Series.

It is more hard to approve the nature of produced situations than it is to gauge the exhibition of point figures. Situations, then again, should be adequately reasonable to mirror the underlying and between factor association of foreseeing values at various expectation skylines. They should, be that as it may, be adequately factor to give helpful extra data. In the cutting edge, the most usually utilized approach is to look at the autocorrelation of the produced information with that of the genuine information and carry out a visual assessment to guarantee that the situations "look sensible." This study proposes a clever methodology that utilizes three particular measurements: Mean Squared Error, Kullback-Leibler Divergence, and Pinball Loss Function.

## 3.4 SCENARIOS

The main purpose of the generation tool is to make it easier to generate synthetic time series using various techniques, as well as to visualize and analyze the results. The models demonstrate how to use the tool to select the optimal algorithm for a certain case, and the list below provides a more detailed discussion of the key application situations:

Scenario 1: Make datasets visible. The first thing a user might want to do is visualize a dataset. In this case, the user can upload their own dataset to the framework and view it using the tools supplied. He can also see the distribution of the dataset as well as the distribution of each individual time series in it. The same is true for autocorrelation, which can be generated as a single plot for the entire dataset or as a separate plot for each time series.

Scenario 2: Examine datasets. In this case, a user examines a dataset using a set of attributes and metrics. The mean and spectral entropy can be calculated. It is also possible to define two datasets and compute metrics between them, such as mutual information and correlation. All of these findings are saved in a csv file and may be visualized with gnuplot using another script provided.

Scenario 3: Enhance datasets. In this case, the user has a dataset that he wishes to improve. He can add this dataset to the framework, then choose an algorithm and tweak the parameters. For example, for ML-based approaches, he can select the number of epochs or the frequency and kind of anomalies for anomalies injection. You can optionally select the size of the generated data (number of time series and their length).

## 3.5 DATA VISUALISATION

Before moving on to the real machine learning challenge, data visualization is essential. Information representation gives bits of knowledge into the conveyance of different highlights present in the information. Contingent upon the sort of information, information can be addressed in different ways, including a guide, a chart, or a plot. Long haul examples might be analyzed, and exceptions in information can be effortlessly spotted, on account of information representation apparatuses.

### 3.5.1 Graphs using Plotly

Various data features are plotted as pulse graphs in Python using the plotly module. The beat charts utilize exceptional varieties to separate highlights and plot each component across time in the x hub. The conveyance scope of every property is addressed by the y hub. Clients can choose a particular perspective in the information and spotlight exclusively on that component for assessment in the constant graphical dashboard.

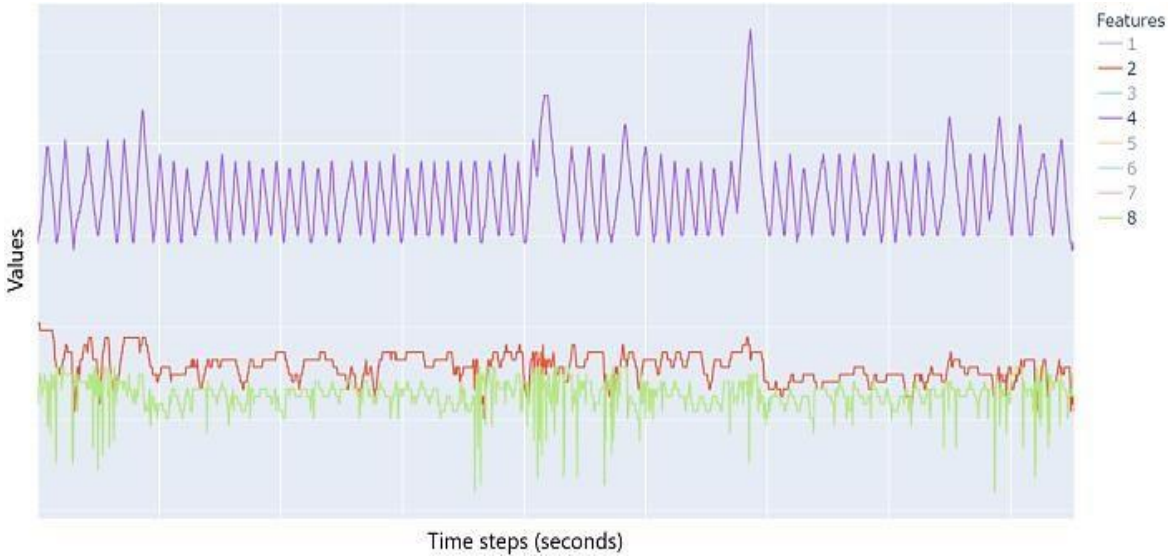



**Figure 3.9:** Unnormalized Time-Series Data Features Plotted Using Plotly.

### 3.5.2 3D Plot using Principal Component Analysis

One technique for deciding the connection between various elements is to run Head Part Examination (PCA) on the given information and afterward present the outcomes in a 2-layered or 3-layered Disperse plot. This sort of perception is utilized to dissect the information's huge angles and afterward look at the dispersions of unique and created information. The PCA in this proposition is performed exclusively for visual assessment and not for possible application in the AI challenge.

Since the information has eight highlights, seeing it in 8-layered space becomes muddled. To work on the issue, the aspects are decreased to three, permitting the data of interest in each time period to be displayed in a 3D plot. Figure 3.10 portrays the PCA 3-layered plot, which portrays the three most recognizable components of solid and broken information. The detachment of these two arrangements of information should be an element performed by highlight 3.10 in the preparation information, with continuous vacillations in broken information and less variances in solid information.

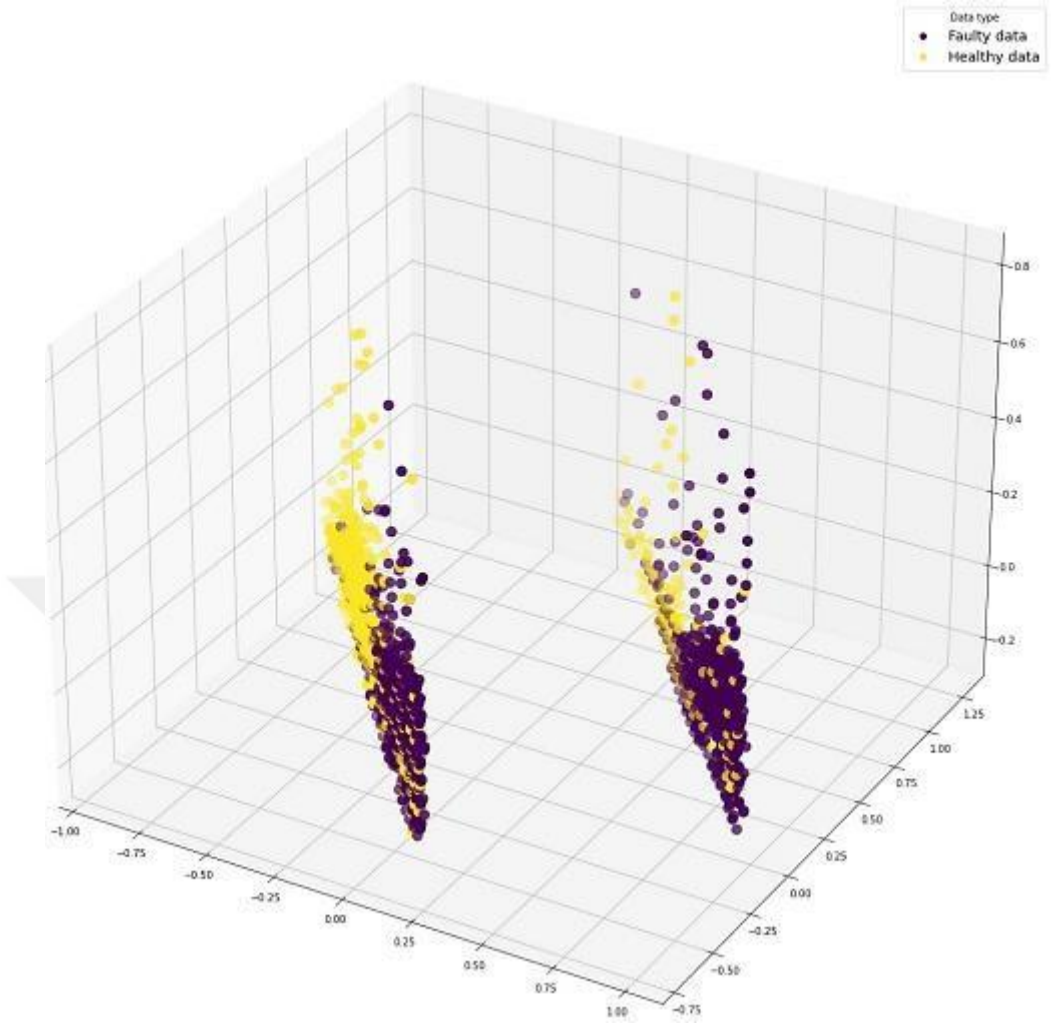



**Figure 3.10:** 3-Dimensional Graph Plotted Using PCA for Training Data.

It is possible to lose some information while performing dimensionality reduction. A statistic known as cumulative explained variance ratio is used to calculate the percentage of original data that is kept. There are three significant parts in this postulation (since decreasing the information to 3-aspects). To start, the difference shown by every essential part is separated by the complete fluctuation. Afterward, the proportion of every essential part is added aggregately to create the action.

The combined made sense of difference ranges somewhere in the range of 0 and 1, with zero demonstrating that no data is saved and one showing that all data is safeguarded. The aggregate made sense of change proportion is 0.988 when PCA is led on preparing information for visual experiences, showing that 98.8 percent of the first information is safeguarded all through dimensionality decrease.

## 3.6 DATA PRE-PROCESSING

Pre-handling information is a basic stage in the AI cycle. Information readiness is an assortment of tasks directed to clean the information, eliminate missing qualities from the information, encode and change the information in view of the utilization case. These strategies are done in light of the fact that the crude information gained from sensors is habitually conflicting, fragmented, and may contain errors.

### 3.6.1 Column Dropping

The informational indexes for this task are in Comma Isolated Worth (CSV) design, with the main segment being the 'chronic number' and the subsequent section containing the 'Time' in seconds for every data of interest. These two sections are eliminated utilizing the 'drop' capability since they are ill-advised for the time-series information creation process and dial back the preparation combination.

### 3.6.2 Conditional Shifting

Restrictive moving is a strategy for supplanting a useful piece of information on the off chance that it is erroneous or missing. Each element in the information outline is doled out to a predetermined conveyance range, and no 'NaN' values ought to be available in the information. Every piece of information (each line) of the component is checked to eliminate such mistaken and missing qualities. Assuming an erroneous information point is found, it is supplanted with the following nearest suitable worth in that element.

### 3.6.3 Normalization

Standardization is the most common way of organizing an information assortment with the goal that it appears to be reliable across all fields. The standardization step as a rule further develops the information quality. Standardization is performed across each component in the informational index to integrate such elements into the preparation information. The condition gives the numerical recipe to normalizing.

$$X_{\text{Normalized}} = (X - X_{\text{min}}) / (X_{\text{max}} - X_{\text{min}}) \tag{3.1}$$

For this situation, Xmin addresses the data of interest with the most reduced esteem and Xmax addresses the data of interest with the greatest worth of every specific characteristic over the entire dataset. Figure 3.11 portrays the information after standardization, where the information qualities are all inside the scope of (0,1).

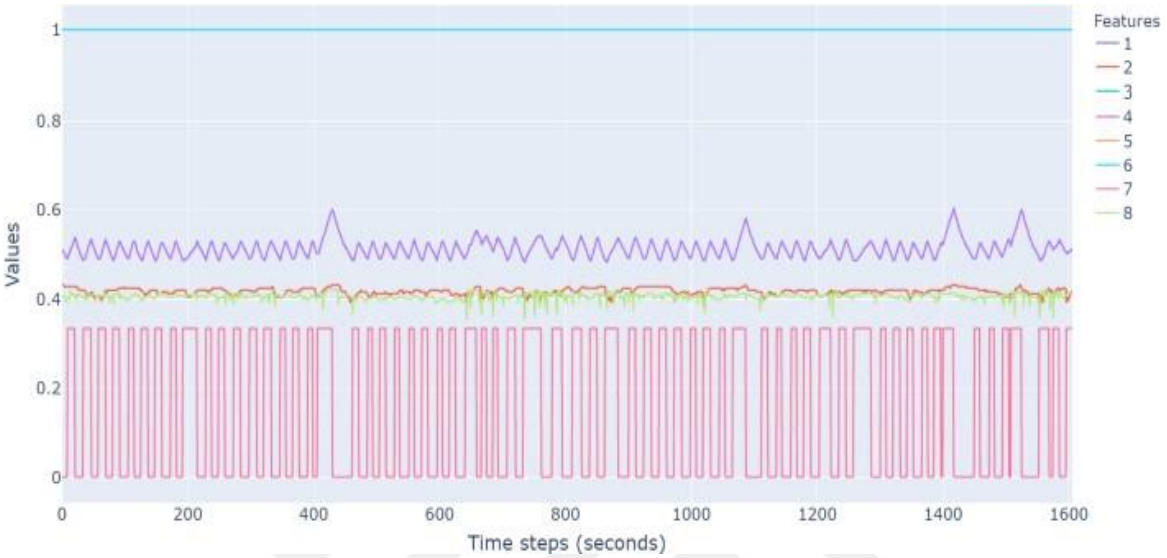



**Figure 3.11:** Normalized Training Time-Series Data Plotted Using Plotly.

# 4. RESULT AND DISCUSSION

## 4.1 EXPERIMENT SETUP

This part makes sense of the investigations and talks about the nature of the outcomes gathered. To start with, we will depict the strategy for finding the ideal hyperparameters for the recommended GAN engineering. Accordingly, in this work, we involved different proposals from the best in class in GANs, joined with an experimental heuristic pursuit, to characterize some really fantastic beginning hyperparameters, as displayed in Table 4.1.

**Table 4.1:** Initial GAN Hyperparameters

| Hyperparameters | | | | | Results Metrics | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Optimizer | Skip Connection | Output Activation | TTUR | Dropout | KL Divergence [bits] | Pinball Loss Function | | MSE | | |
| | | | | | | Temp. | Humid. | Temp. [ž$C^2$] | Humid. [%$^2$] | |
| Adam | C | *linear* | C | C | 1.888 | 0.971 | 0.506 | 1.943 | 1.011 | |
| Adam | C | *linear* | C | ✓ | 1.801 | 1.179 | 0.546 | 2.359 | 1.092 | |
| Adam | C | *linear* | ✓ | C | 1.771 | 1.103 | 0.389 | 2.206 | 0.778 | |
| Adam | C | *linear* | ✓ | ✓ | 1.825 | 1.234 | 0.422 | 2.468 | 0.844 | |
| Adam | C | *tanh* | ✓ | C | 2.431 | 1.102 | 0.493 | 2.203 | 0.987 | |
| Adam | ✓ | *linear* | ✓ | C | 2.358 | 1.122 | 0.361 | 2.244 | 0.721 | |
| AdaBelief | C | *linear* | C | C | 2.375 | 1.567 | 0.542 | 3.134 | 1.083 | |
| AdaBelief | C | *linear* | C | ✓ | 2.013 | 0.736 | 0.331 | 1.473 | 0.662 | |
| AdaBelief | C | *linear* | ✓ | C | 1.707 | 0.68 | 0.291 | 1.359 | 0.583 | |
| AdaBelief | C | *linear* | ✓ | ✓ | **1.432** | **0.488** | **0.219** | **0.977** | **0.438** | |
| AdaBelief | C | *tanh* | ✓ | ✓ | 1.872 | 0.656 | 0.209 | 1.317 | 0.419 | |
| AdaBelief | ✓ | *linear* | ✓ | ✓ | 2.018 | 0.778 | 0.593 | 1.556 | 1.187 | |

Following the first hyperparameter selection procedure, we decided to use Long Short-Term. The architectures of both networks are depicted in Figures 4.1 and 4.2. The

Discriminator network has around 735,000 trainable parameters, while the Generator network has approximately 165,000. We adjust the following hyperparameters once the architectures are fixed:

(i) Optimizer: (ii) Skip-Connection: A modification to the Discriminator design that connects the last step of the time series with the Fully Connected block; (iii) Generator output activation function: (iii) Linear or tanh; (iv) TTUR; (v) Dropout The outcomes of this hyperparameter change are shown in Table 4.1.

## 4.2 TRAINING AND TESTING

First, we will explain the searching process to obtain the best hyperparameters for the proposed GAN architecture. Then, we will find some examples of data generated using the random latent space exploration method. After that, we will use the high-fidelity sampling method MH-GAN and compare the results with those. Finally, we will present the approach to generate on-demand anomalous situations and present some examples. As we examined in past segments, GAN preparing is a complicated and fragile cycle. Preparing solidness is subject to numerous hyperparameters that should be physically set and have complex between connections. The picture age research local area has endeavored to recognize hyperparameters that produce improved results and more steady preparation. Thus, in this work, we involved numerous suggestions from the best in class in GANs, alongside an observational heuristic hunt, to set some sensibly great beginning hyperparameters reflected. We decided to use Long Short Term Memory (LSTM) neurons in the Generator and 1D Convolution neurons in the Discriminator after the first hyperparameter determination process, in view of best in class models and a few experimental tests. In Figures 2 and 3, we find the architectures of both networks.

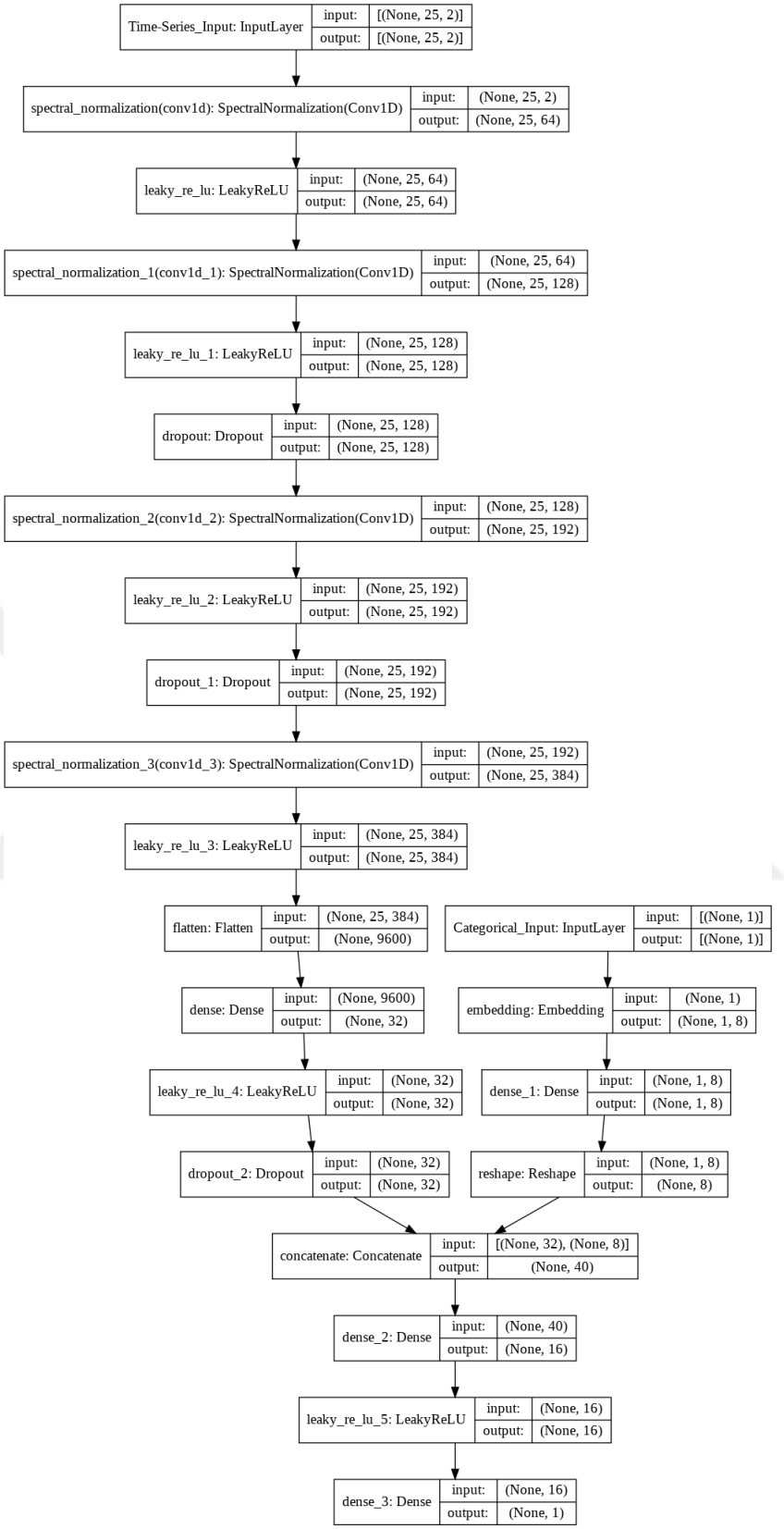



**Figure 4.1:** Discriminator Network Architecture. The Question Marks on the Shapes Indicate the Batch Size, Which is Undefined in the Network Architecture.

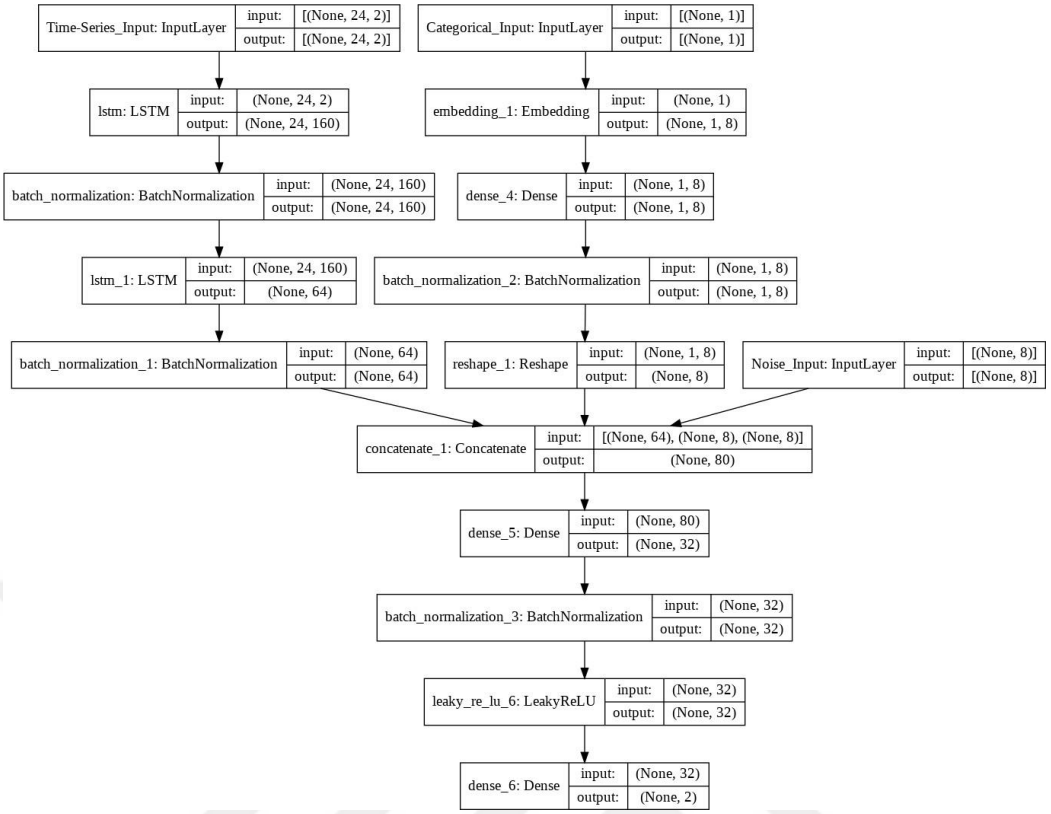



**Figure 4.2:** Generator Network Architecture. the Question Marks on the Shapes Indicate the Batch Size, Which is Undefined in The Network Architecture.

As shown by the experiment results reflected in figures below, the hyperparameters that offer the best metrics are: (i) AdaBelief Optimizer ; (ii) Do not use the Skip-Connection architecture; (iii) Linear output activation function in the Generator; (iv) Use the TTUR training technique [81]; and (v) Use the Dropout regularization method [. Figure 4.3 portrays a correlation of the disseminations of the Generator's situation expectations for 24 time-steps (4 hours) ahead and a ground truth haphazardly chose approval set. Hence, the generated distributions from the predictions are consistent with the available real data. In figures 4.4 and 4.5, we observe the comparison between the distributions of the real validation data and the generated data 24 time-steps (4 hours) ahead. Therefore, we confirm the hypothesis that we have managed to generate data that simulate real data characteristics. In the following sections, we will explore the Generator network’s latent space and find examples of realistic scenarios and on-demand anomalous situations.


**Figure 4.3:** Comparison Between Real and Generated Data Distributions**.**

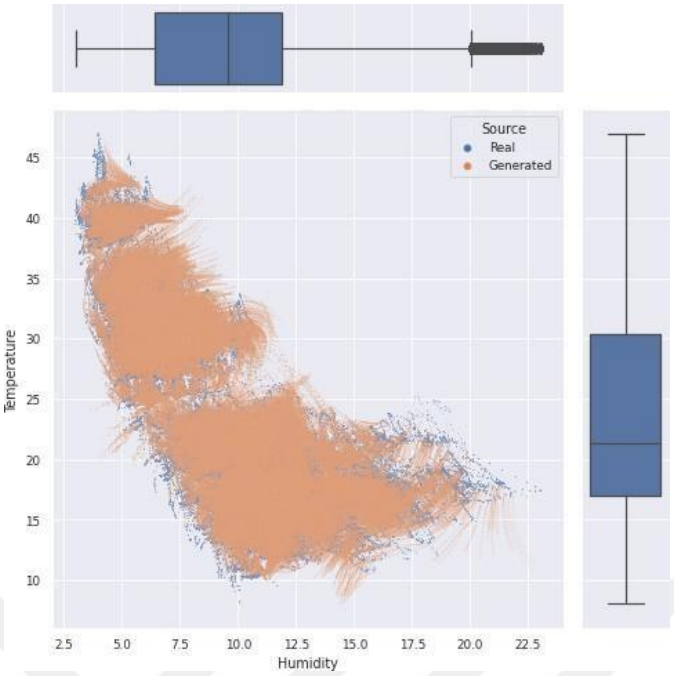



**Figure 4.4:** Plot of Information From the Real and Generated Dispersions of Relative Dampness and Temperature.

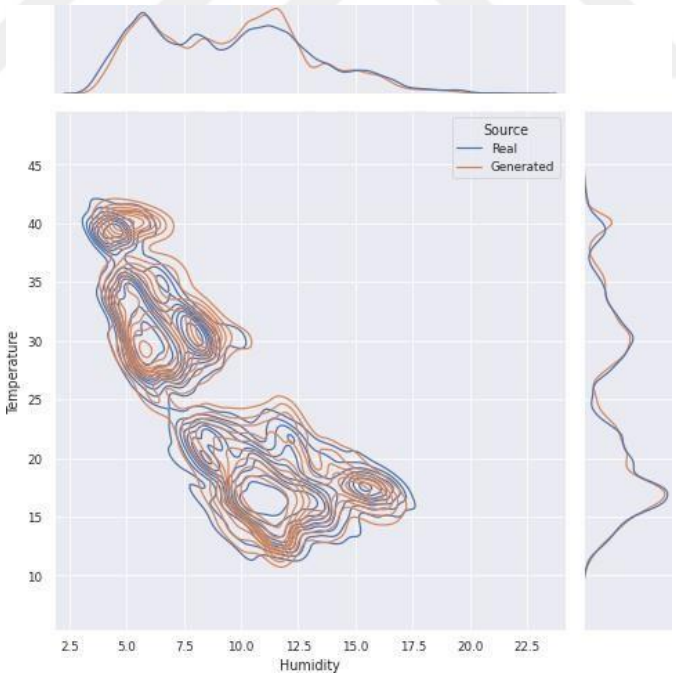



**Figure 4.5:** Plot of Part Thickness Assessment With Genuine and Created Information Dispersions**.**

## 4.3 SCENARIO GENERATION: RANDOM EXPLORATION OF THE LATENT SPACE

In this section, we will explore scenario generation through the method of random exploration of latent space. Since this method is the most straightforward to produce GAN outputs, during the training phase we have used it to obtain the result metrics, obtaining a KL divergence of 1.432 bits in the validation set, and an RMSE accuracy error of 0.988ºC for temperature and 0.661% for humidity.

To demonstrate the consistency of the proposed architecture, we have carried out experiments in a test set. Following are some instances of scenarios provided by some sensors, which aid in visualizing the quality of the obtained findings. In each case, six alternative situations have been constructed, with a total duration of 24 time-step concatenated forecasts (which represents a prediction of 4 hours ahead). Figure 4.6 depicts a case in which the scenarios have low uncertainty, indicating that the Generator is more confident in the results. Figure 4.9 depicts a case in which the anticipated scenarios' uncertainty is significantly high even from the initial phases. The uncertainty captured in the generation of random scenarios gives us useful information about the algorithm’s confidence when making a prediction. Furthermore, all the generated scenarios, despite those with high uncertainty, can be considered realistic and therefore useful to augment the dataset synthetically.


**Figure 4.6:** Scenario Generation Example of 24 Time-Steps (4 hours).

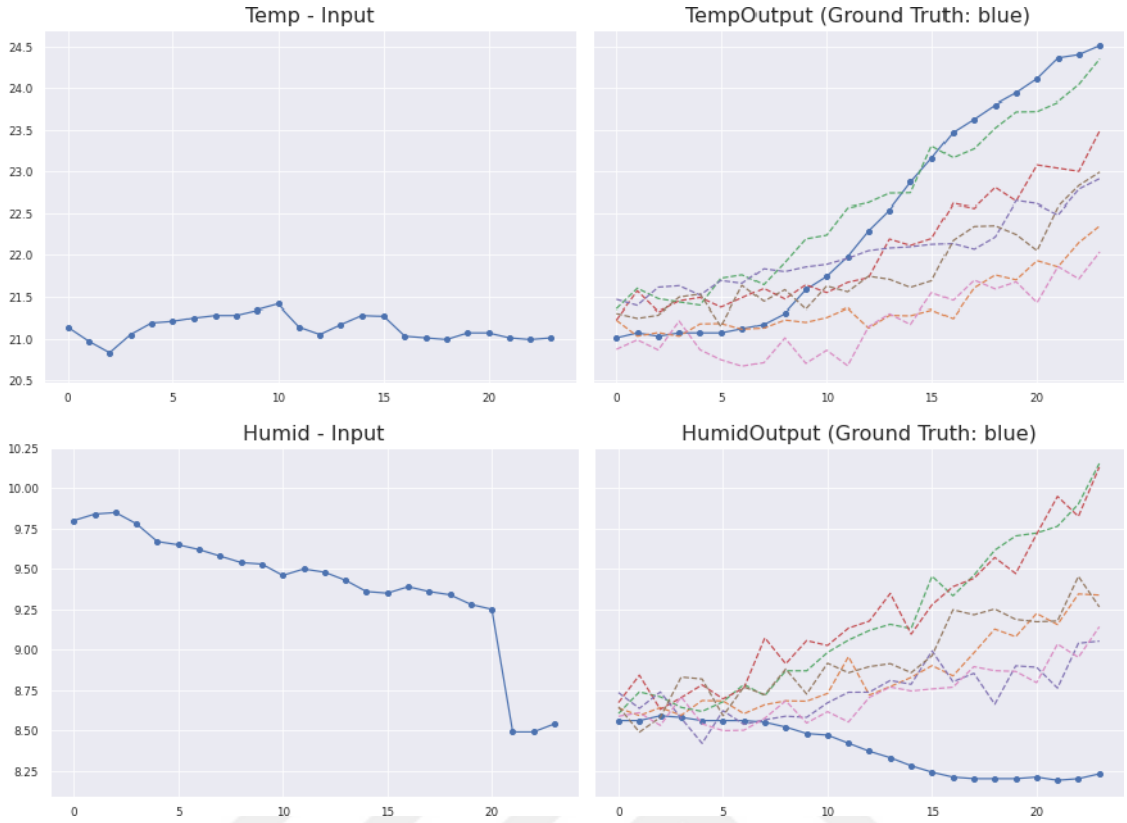



**Figure 4.7:** Scenario Generation Example of 24 Time-Steps (4 hours)

To provide an overview of the scenarios generated in all sensors, in Figures 4.8 and 4.9, we find heat maps of the temperature and humidity, where each cell represents a different sensor. Specifically, Figure 4.9 shows the heat maps for an example scenario where one prediction step has been made, comparing the real data with the fake data generated. Figure 4.8 shows the same comparison but after 24 time-step predictions (4 hours ahead). The differences between sensors are due to the distribution of temperatures and humidity in a DC is not homogeneous since there are regions with greater intensity of cooling (inlets) and other zones where the heat produced by many servers accumulates (outlets).

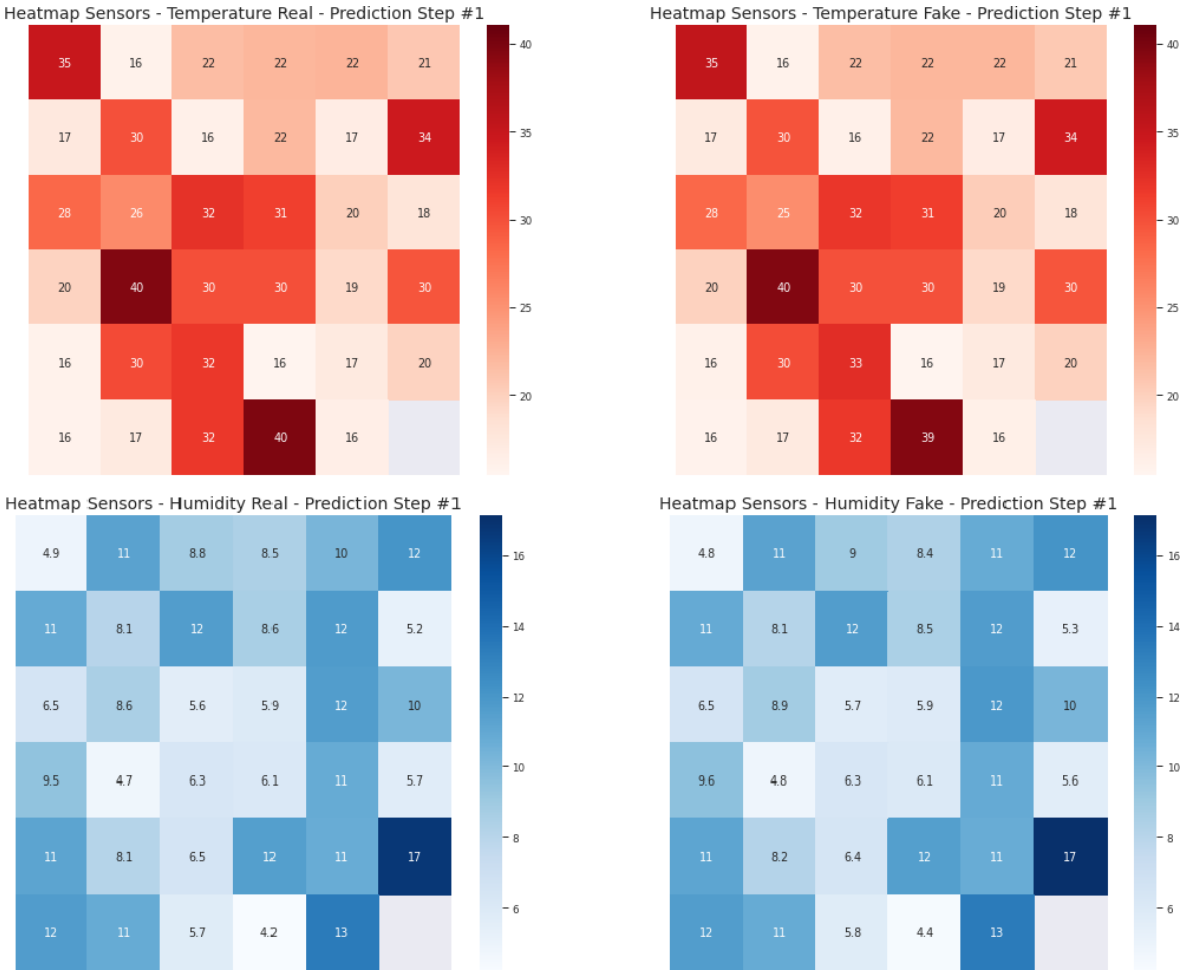



**Figure 4.8:** Heat Maps of Temperatures and Humidities in The 35 Available Sensors.

Each cell represents a different sensor. The figures on the left show the real data, and the ones on the right show the generated fake data using the random exploration of the latent space method.

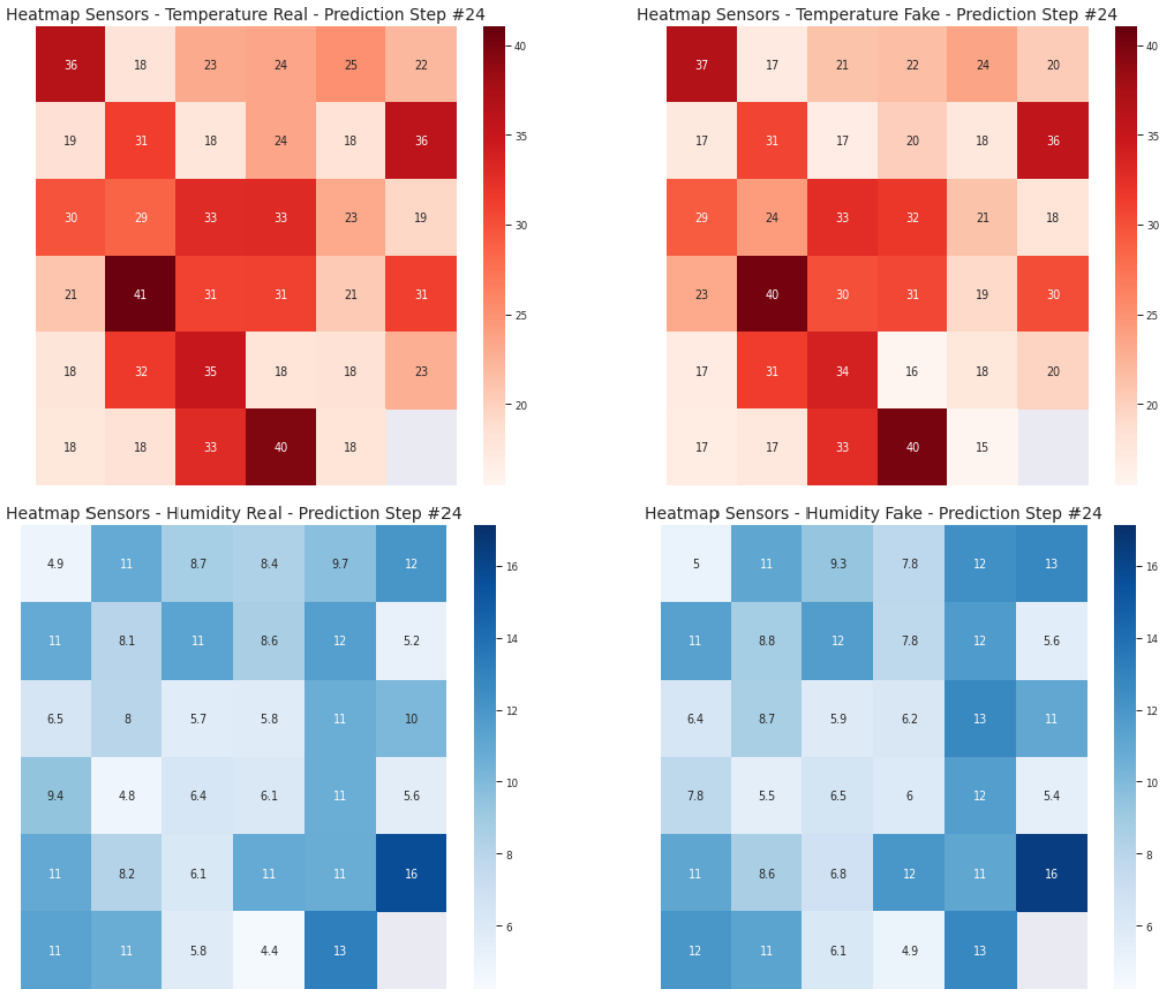



**Figure 4.9:** Heat Maps of Temperatures and Humidities in The 35 Available Sensors, 24 Prediction Steps Ahead (4 Hours).

Each cell represents a different sensor. The figures on the left show the real data, and the ones on the right show the generated fake data using the random exploration of the latent space method.

## 4.4 CONCLUSION

Development in energy use has eased back, attributable to the proficiency acquires that brilliant approaches can assist with keeping up with for the time being. In any case, a lot bigger public information and displaying limits are earnestly expected to comprehend and screen DC energy use, and plan powerful streamlining strategies. Data-driven optimization methods have proven to be the most accurate for modeling, predicting, controlling, and optimizing complex systems. We addressed this topic in the our research.

# 5. CONCLUSIONS AND FUTURE WORKS

## 5.1 CONCLUSIONS

Chapter 3 considers a multi-variable sensor data scenario in a real DC facility, used to verify our work's usefulness. Synthetic data generation is presented as an alternative with a huge potential to boost optimization in DCs, where data gathering with significant variability is expensive, and the equipment's physical integrity may be at risk. A comprehensive exploratory analysis of the available data is performed. We also describe the methodology applied in the experiments, the evaluation metrics, and the improvements implemented to stabilize the training and results' obtention.

Finally, Chapter 4 describes the experiments carried out, where we perform an intensive hyperparameter tuning to find the architecture that produces better results. Eventually, we obtained satisfactory results that demonstrate the Generator's consistency. In particular, the best result obtained from scenario generation in the random validation set is a KL divergence of the 1.432 bits and an RMSE accuracy error of 0.988ºC for temperature and 0.661% for humidity. Afterward, we present scenario generation and on-demand anomaly examples from a randomly selected test set, demonstrating the proposed architecture's reliability.

This thesis gives an original generative model to guaging time series with missing information. This model predicts the estimating circulation without first doing attribution, and it is the main model to create total appropriation gauges in the setting of time series information with missing data, supposedly. We confronted with two issues in this undertaking. It first endeavors to adapt to missing information, and afterward it means to deliver great probabilistic gauges for time series information with missing qualities. Our examinations have yielded promising results. The utilization of an Assistant GAN to prepare the veil appropriations significantly helps information creation.

The discoveries of this study make a philosophy for broadening engineered time-series information applications by considering the fuse of all out factors and the creation of multi-variable situations at specific moments. On-request creation of bizarre circumstances presents high information inconstancy without requiring extra work or endangering the actual trustworthiness of electronic gear. Moreover, the proposed approach is appropriate to

some other time-series-like issues in different fields of utilization. GANs are a type of generative calculation that has a life expectancy of just five years. Subsequently, the advancements that will be accomplished in this area before very long can be utilized to work on the got discoveries and conquered its cutoff points.

## 5.2 FUTURE WORKS

In the last section of this document, some possible new steps for the future of this research are discussed. This MSc Thesis extends the application of GANs for use in time-series data augmentation, but these algorithms have only five years of life. Hence the advances that will be made in the next years in this field can be implemented to improve the obtained results and address its limitations. The possible future lines can be divided as follows:

a. As previously said, GANs have received little attention for uses other than image production. As a result, a thorough and consistent investigation of the optimum hyperparameters necessary for robust training in time-series data is critical.

b. For the proposed use case, DC optimization, the use of only three variables (temperature, relative humidity, and the analyzed sensor ID) is sufficient to demonstrate the results' utility. Nevertheless, it is necessary to empirically demonstrate the proposed architecture's scalability with a larger number of variables.

c. We have demonstrated the realism of the data generated in this work. Yet further research is needed on the real utility for other Deep Learning algorithms (e.g., Train on Synthetic Data - Test on Real Data)

# REFERENCES


[1] M. Lei, L. Shiyan, J. Chuanwen, L. Hongling, and Z. Yan, “A review on the forecasting of wind speed and generated power,” Renewable and Sustainable Energy Reviews, vol. 13, no. 4, pp. 915–920, 2009.

[2] P. Pinson, H. Madsen, H. A. Nielsen, G. Papaefthymiou, and B. Klockl, ¨ “From probabilistic forecasts to statistical scenarios of short-term wind power production,” Wind energy, vol. 12, no. 1, pp. 51–62, 2009.

[3] Y. Wang, Y. Liu, and D. S. Kirschen, “Scenario reduction with submodular optimization,” IEEE Transactions on Power Systems, vol. 32, no. 3, pp. 2479–2480, 2017.

[4] C. Tang, Y. Wang, J. Xu, Y. Sun, and B. Zhang, “Economic dispatch considering spatial and temporal correlations of multiple renewable power plants,” arXiv preprint arXiv:1707.00237, 2017.

[5] Y. Gu and L. Xie, “Stochastic look-ahead economic dispatch with variable generation resources,” IEEE Transactions on Power Systems, vol. 32, no. 1, pp. 17–29, 2017.

[6] Y. Feng, I. Rios, S. M. Ryan, K. Spurkel, J.-P. Watson, R. J.-B. Wets, ¨ and D. L. Woodruff, “Toward scalable stochastic unit commitment. part 1: load scenario generation,” Energy Systems, vol. 6, no. 3, pp. 309–329, 2015.

[7] G. Osorio, J. Lujano-Rojas, J. Matias, and J. Catal ´ ao, “A new scenario ˜ generation-based method to solve the unit commitment problem with high penetration of renewable energies,” International Journal of Electrical Power & Energy Systems, vol. 64, pp. 1063–1072, 2015.

[8] G. Giebel, R. Brownsword, G. Kariniotakis, M. Denhard, and C. Draxl, “The state-of-the-art in short-term prediction of wind power: A literature overview,” ANEMOS. plus, Tech. Rep., 2011.

[9] R. Barth, L. Soder, C. Weber, H. Brand, and D. J. Swider, “Methodology ¨ of the scenario tree tool,” Wilmar Deliverable, vol. 6, 2006.

[10] S. Delikaraoglou and P. Pinson, “High-quality wind power scenario forecasts for decision-making under uncertainty in power systems,” in 13th International Workshop on Large-Scale Integration of Wind Power into Power Systems as well as on Transmission Networks for Offshore Wind Power (WIW 2014), 2014.

[11] T. Wang, H.-D. Chiang, and R. Tanabe, “Toward a flexible scenario generation tool for stochastic renewable energy analysis,” in Power Systems Computation Conference (PSCC), 2016. IEEE, 2016, pp. 1–7.

[12] X.-Y. Ma, Y.-Z. Sun, and H.-L. Fang, “Scenario generation of wind power based on statistical uncertainty and variability,” IEEE Transactions on Sustainable Energy, vol. 4, no. 4, pp. 894–904, 2013.

[13] Y. Chen, Y. Wang, D. S. Kirschen, and B. Zhang, “Model-free renewable scenario generation using generative adversarial networks,” IEEE Transactions on Power Systems, 2018.

[14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in Advances in neural information processing systems, 2014, pp. 2672– 2680.

[15] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein GAN,” arXiv preprint arXiv:1701.07875, 2017. [16] C. Villani, Optimal transport: old and new. Springer Science & Business Media, 2008, vol. 338.

[16] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” arXiv preprint arXiv:1312.6114, 2013.

[17] C. Wan, Z. Xu, P. Pinson, Z. Y. Dong, and K. P. Wong, “Probabilistic forecasting of wind power generation using extreme learning machine,” IEEE Transactions on Power Systems, vol. 29, no. 3, pp. 1033–1044, 2014.

[18] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, “On the importance of initialization and momentum in deep learning,” in International conference on machine learning, 2013, pp. 1139–1147.

[19] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin et al., “Tensorflow: Large-scale machine learning on heterogeneous distributed systems,” arXiv preprint arXiv:1603.04467, 2016.

[20] Kay Gregor Hartmann, Robin Tibor Schirrmeister, and Tonio Ball. “EEG-GAN: Generative adversarial networks for electroencephalograhic (EEG) brain signals”. In: arXiv preprint arXiv:1806.01875 (2018).

[21] Mohamad H Hassoun et al. Fundamentals of artificial neural networks. MIT press, 1995. [23] S.S. Haykin. Neural Networks and Learning Machines. Neural networks and learning machines v. 10. Prentice Hall, 2009. isbn: 9780131471399

[22] Geoffrey E Hinton and James L McClelland. “Learning representations by recirculation”. In: Neural information processing systems. Vol. 1987. 1988, pp. 358–366.

[23] Sepp Hochreiter and Jürgen Schmidhuber. “Long short-term memory”. In: Neural computation 9.8 (1997), pp. 1735–1780.

[24] Plotly Technologies Inc. Collaborative data science. 2015. url: https: //plot.ly.

[25] Ricardo Jesus et al. “Stream Generation: Markov Chains vs GANs.” In: IoTBDS. 2019, pp. 177–184.

[26] Yanghua Jin et al. Towards the Automatic Anime Characters Creation with Generative Adversarial Networks. 2017.

[27] Tero Karras et al. Progressive Growing of GANs for Improved Quality, Stability, and Variation. 2018. [30] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: arXiv preprint arXiv:1412.6980 (2014).

[28] Diederik P Kingma and Max Welling. “Auto-encoding variational bayes”. In: arXiv preprint arXiv:1312.6114 (2013).

[29] Alex Lamb et al. “Professor forcing: A new algorithm for training recurrent networks”. In: arXiv preprint arXiv:1610.09038 (2016).

[30] Arthur Le Guennec, Simon Malinowski, and Romain Tavenard. “Data augmentation for time series classification using convolutional neural networks”. In: ECML/PKDD workshop on advanced analytics and learning on temporal data. 2016.

[31] Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. “Time-series Generative Adversarial Networks”. In: Advances in Neural Information Processing Systems. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc., 2019. url: www.shorturl.at/evGT5.

[32] Yi Zheng et al. “Time Series Classification Using Multi-Channels Deep Convolutional Neural Networks”. In: Web-Age Information Management. Ed. by Feifei Li et al. Cham: Springer International Publishing, 2014, pp. 298–310. isbn: 978-3-319-08010-9.

[33] Jun-Yan Zhu et al. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. 2020. arXiv: 1703.10593 [cs.CV].

[34] Crist´obal Esteban, Stephanie L. Hyland, and Gunnar R¨atsch. Real-valued (medical) time series generation with recurrent conditional gans, 2017.

[35] A. Koochali, P. Schichtel, A. Dengel, and S. Ahmed. Probabilistic forecasting of sensory data with generative adversarial networks – forgan. IEEE Access, 7:63868–63880, 2019. [40] Ian Goodfellow. Nips 2016 tutorial: Generative adversarial networks, 12 2016.

[36] Colin Sparrow. The Lorenz Equations: Bifurcations, Chaos, and Strange Attractors. Springer, New York, NY, 2012.

[37] Stephen H. Kellert. In the Wake of Chaos: UNPREDICTABLE ORDER IN DYNAMICAL SYSTEMS. University of Chicago Press, 1993.

[38] MC Mackey and L Glass. Oscillation and chaos in physiological control systems. Science, 197(4300):287–289, 1977.

[39] Rasoul Rahmani, Irene Moser, and Mohammadmehdi Seyedmahmoudian. A complete model for modular simulation of data centre power load. arXiv preprint arXiv:1804.00703, 2018.

[40] Giorgia Ramponi, Pavlos Protopapas, Marco Brambilla, and Ryan Janssen. T-cgan: Conditional generative adversarial network for data augmentation in noisy time series with irregular sampling. arXiv preprint arXiv:1811.08295, 2018.

[41] Sima Siami-Namini, Neda Tavakoli, and Akbar Siami Namin. A comparison of arima and lstm in forecasting time series. In 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), pages 1394–1401. IEEE, 2018.

[42] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research, 15(56):1929–1958, 2014.

[43] Cecilia Summers and Michael J. Dinneen. Improved mixed-example data augmentation. In 2019 IEEE Winter Conference on Applications of Computer Vision (WACV), pages 1262–1270, 2019.

[44] Mike West. Bayesian forecasting of multivariate time series: scalability, structure uncertainty and decisions. Annals of the Institute of Statistical Mathematics, 72(1):1–31, February 2020

[45] Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. Time-series Generative Adversarial Networks. In Advances in Neural Information Processing Systems, volume 32, pages 5508–5518. Curran Associates, Inc., 2019.