Redes Adversarias Generativas para el Aumento de Datos Secuenciales 1D. Aplicación a turbinas eólicas

Por
Víctor Ramos Fuentes

UNIVERSIDAD COMPLUTENSE
MADRID

Grado en Ingeniería informática
Facultad de Informática

Dirigido por
Matilde Santos Peñas, María José Gómez Silva

Generative Adversarial Networks for 1D Sequential Data Augmentation. Application to wind turbines

Madrid, 2021–2022

Autorización de difusión y uso

El autor de este trabajo autoriza a la Universidad Complutense de Madrid a emplear tanto el código como la memoria elaborada, únicamente con fines didácticos y mencionando al autor del mismo

Víctor Ramos Fuentes

Agradecimientos

En primer lugar, agradecer el apoyo de Matilde y Maria José por aportarme la idea y los datos necesarios para realizar este proyecto.

Por otro lado, a aquellas personas que han estado apoyándome durante la carrera y a lo largo de este proyecto, amigos y familiares, gracias por animarme siempre a continuar.

También me gustaría nombrar a todos aquellos profesores cuya dedicación en las asignaturas que cursé durante estos años me han ayudado a mejorar mis habilidades y ampliar mis conocimientos.

Por último, reconocer a la Universidad Complutense de Madrid el dar acceso a múltiple información, así como posibilitar estudios superiores de calidad en una universidad pública.

Índice general

Página

1. Introducción **1**
1.1. Motivación . . . 1
1.2. Energía Eólica . . . 1
1.3. GANs . . . 2
1.4. Objetivos . . . 3
1.5. Plan de Trabajo . . . 4
1.6. Asignaturas relacionadas y software utilizado . . . 4
1.7. Estructura de la memoria . . . 5

2. Introduction **1**
2.1. Motivation . . . 1
2.2. Wind Energy . . . 1
2.3. GANs . . . 2
2.4. Objectives . . . 3
2.5. Work Plan . . . 4
2.6. Related subjects and software used . . . 4
2.7. Structure of memory . . . 4

3. Fundamentos teóricos **6**
3.1. Turbinas Eólicas . . . 6
3.2. Red GAN . . . 8
3.2.1. Redes neuronales . . . 9
3.2.2. Entrenamiento de GAN . . . 11
3.2.3. Optimizadores GAN . . . 12
3.2.4. Funciones de pérdida o error . . . 13

4. Aplicación de una Red Generativa Adversaria **14**
4.1. Conjunto de datos inicial . . . 14
4.2. Arquitectura de la red GAN . . . 20
4.2.1. Generador . . . 20
4.2.2. Discriminador . . . 22
4.2.3. Optimizador y Función de pérdida . . . 24
4.2.4. Red GAN . . . 24
4.3. Entrenamiento de la red GAN . . . 25

5. Resultados **27**
5.0.1. Fichero Velocidad del viento = 0 . . . 28

5.0.2. Fichero Velocidad del viento = 20 . . . 32
5.0.3. Fichero Velocidad del viento = 40 . . . 35
5.0.4. Fichero Velocidad del viento = 60 . . . 37
5.0.5. Fichero Velocidad del viento = 80 . . . 39
5.0.6. Fichero con ángulo del pitch 3 . . . 41
5.0.7. Fichero con ángulo del pitch 5 . . . 43
5.0.8. Fichero con ángulo del pitch 8 . . . 45
5.0.9. Fichero con ángulo del pitch 10 . . . 47
5.0.10. Fichero con ángulo del pitch 11-4 . . . 49
5.0.11. Fichero con ángulo del pitch 13 . . . 51
5.0.12. Fichero con ángulo del pitch 17 . . . 53
5.0.13. Fichero con ángulo del pitch 20 . . . 55
5.0.14. Fichero con ángulo del pitch 21 . . . 57
5.0.15. Fichero con muestreo aleatorio 1 . . . 59
5.0.16. Fichero con muestreo aleatorio 2 . . . 61

6. Conclusiones y Trabajo futuro 63
6.1. Conclusiones . . . 63
6.2. Trabajo futuro . . . 64

7. Conclusions and Future Work 65
7.1. Conclusions . . . 65
7.2. Future Work . . . 66

9. Bibliografía y enlaces de referencia 71

Resumen

Las energías renovables son aquellas basadas en el uso de recursos naturales ilimitados. Se caracterizan por evitar la producción de gases de efecto invernadero (principal causa del cambio climático). Su desarrollo ha sido fuertemente impulsado en los últimos años. Dentro de las energías renovables se encuentra la energía eólica, que aprovecha el efecto de las corrientes del aire. Dicha energía se recoge en parques eólicos, donde se encuentran los aerogeneradores, máquinas que transforman la fuerza del viento o energía cinética en energía eléctrica. Los parques eólicos están compuestos de turbinas eólicas que pueden dividirse en diversas categorías, centrándonos en el proyecto en las turbinas eólicas flotantes, situadas generalmente en zonas marinas.

El objetivo de este proyecto es implementar una red generativa adversaria o GAN (Generative Adversarial Network), que a partir de unos datos de entrada reales que contienen parámetros de configuración de las turbinas, sea entrenada para autogenerar datos sintéticos similares a los reales. Estos datos sintéticos son requeridos debido a la falta de datos reales en las turbinas marinas flotantes, facilitando poder estudiar problemas relacionados con desgaste o roturas de las turbinas eólicas y entrenar otros modelos de control.

Una GAN, o red generativa adversaria, es una clase de red neuronal compuesta por un algoritmo discriminativo o discriminador que, una vez correctamente entrenado, es capaz de distinguir entre imágenes reales y otras sintéticas. Estas imágenes sintéticas son desarrolladas por el algoritmo generativo o generador, otra red neuronal que llega a generar imágenes sintéticas con un parecido tan idéntico a las reales que hace confundir al discriminador.

Palabras Claves

- Energías renovables
- Turbinas Eólicas Flotantes
- Datos sintéticos o artificiales
- Aprendizaje profundo
- Entrenamiento de redes
- Redes generativas adversarias (GAN)

Abstract

Renewable energies are those based on the use of unlimited natural resources. They are characterised by avoiding the production of greenhouse gases (the main cause of climate change). Their development has been strongly boosted in recent years. Renewable energies include wind energy, which harnesses the effect of air currents. This energy is collected in wind farms, where wind turbines, machines that transform the force of the wind or kinetic energy into electrical energy, are located. Wind farms are made up of wind turbines that can be divided into different categories, with the project focusing on floating wind turbines, which are generally located in marine areas.

The aim of this project is to implement a Generative Adversarial Network (GAN), which from real input data containing turbine configuration parameters, is trained to self-generate synthetic data similar to real data. This synthetic data is required due to the lack of real data on floating offshore turbines, making it easier to study problems related to wind turbine wear and tear and to train other control models.

A GAN, or generative adversarial network, is a class of neural network composed of a discriminative algorithm or discriminator that, once properly trained, is able to distinguish between real and synthetic images. These synthetic images are evolved by the generative or generative algorithm, another neural network that manages to generate synthetic images that are so similar to the real ones that they confuse the discriminator.

Keywords

- Renewable Energies
- Wind Turbines
- GAN
- Data Augmentation
- Deep Learning
- Network Training
- Generative adversarial networks

Capítulo 1

Introducción

1.1. Motivación

En la actualidad, para el correcto desarrollo de las turbinas y sus correspondientes módulos de control se emplea el modelado de datos. Una opción muy utilizada es el modelado de datos basándose en técnicas de Deep Learning. El problema que origina esta casuística es la imposibilidad de obtener grandes cantidades de datos. bien por su escasez, bien por el coste de obtención.

Por ello, la motivación principal de este proyecto es implementar una red generativa capaz de producir datos sintéticos que simulen el comportamiento de datos que puedan ser obtenidos por otras vías más costosas y así poder emplearlos en modelos desarrollados a futuro que faciliten el uso de este tipo de turbinas.

1.2. Energía Eólica

La energía eólica es una fuente de energía renovable, no se agota, no contamina y reduce el origen de emisiones de efecto invernadero que originan el calentamiento global. Por ello, la producción de electricidad a través de energía eólica, junto con otras energías renovables, y su utilización contribuyen al desarrollo sostenible. Se debe destacar que la energía eólica no emite sustancias tóxicas ni contaminantes al aire. Dichas sustancias pueden llegar a acidificar edificios y corromper ecosistemas terrestres y acuáticos. En el plano humano es reseñable la cantidad de enfermedades pulmonares y cardiovasculares que puede provocar dicha contaminación. La energía eólica no genera residuos ni contamina el agua, algo muy importante dado que en la actualidad la escasez de agua es un problema mundial.

Ahora bien, para transformar la energía del viento o energía cinética en energía eléctrica necesitamos aerogeneradores. Son la parte esencial de las turbinas eólicas. El viento golpea a las palas, lo cual hace que las palas giren y con ese movimiento mecánico del rotor produce energía cinética que se transforma en energía eléctrica. Esto se consigue porque el eje que rota está conectado a un generador que produce la energía eléctrica. Hay diversos factores que influyen en la cantidad de energía a producir, tales como lon-

gitud de las palas, dimensión de la turbina y del rotor, la ubicación del aerogenerador y, fundamentalmente, la velocidad del viento.

Estas turbinas se encuentran instaladas en parques eólicos en todo el mundo, siendo los más extensos los que encontramos en Estados Unidos. En cuanto a España, el parque eólico más importante es el 'Gecama', situado en la provincia de Cuenca, en Castilla La Mancha (figura 2.1). Cuenta con una extensión de 40 kilómetros cuadrados.

Figura 1.1: Parque Eólico de Gecama en Cuenca
[1]

Para sacar el máximo rendimiento de las turbinas es importante tener información de su funcionamiento, y datos tanto de su dinámica cómo de las perturbaciones externas a las que se ve sometida. Esta información no está siempre disponible o no es accesible al haber todavía pocas instalaciones de algunos tipos de parques eólicos marinos y más especialmente, de las turbinas flotantes que se encuentran en alta mar. Este tipo de turbina permite generar electricidad en aguas muy profundas, donde otro tipo de turbinas como las de cimientos fijos no son viables. Las turbinas flotantes tienen un gran potencial al aplicarlas en zonas con aguas poco profundas como Japón, puesto que incrementan significativamente el area marina disponible para granjas eólicas marinas. Situar dichas granjas en zonas alejadas de las costas tiene varias ventajas: reducir la contaminación visual, alcanzar vientos más consistentes y de mayor potencia, proveer mayor espacio a embarcaciones destinadas a la pesca...etc

La gran mayoría de turbinas eólicas flotantes se encuentran en una fase primeriza de su desarrollo. La instalación de los primeros prototipos se instalaron en 2007, siendo la única granja funcional a día de hoy, la granja *Hywind Scotland*[2] en 2017. Posee 5 turbinas con capacidad de 30 MW.

1.3. GANs

Las redes generativas antagónicas o GANS, son un tipo de algoritmo de inteligencia artificial (IA), planteado por Ian Goodfellow en 2014 [3]. En esencia, su funcionamiento se resume la combinación de dos redes neuronales que son entrenadas enfrentándose entre

sí. Esta técnica genera imágenes o datos que pueden ser tratadas como reales (figura 2.2).

Hasta ahora, dichas redes se han empleado en diversos campos, como diseño de interiores, de complementos, de ropa e industriales [[4]]. También se emplean para reconstruir modelos 3D a partir de modelos 2D [[4]]. La mayor aplicabilidad que tiene a día de hoy estas redes es rellenar información que falta". Por ejemplo, en diagnósticos clínicos podrían predecir un diagnóstico clínico al ayudar a conocer el comportamiento futuro de un paciente [[4]].

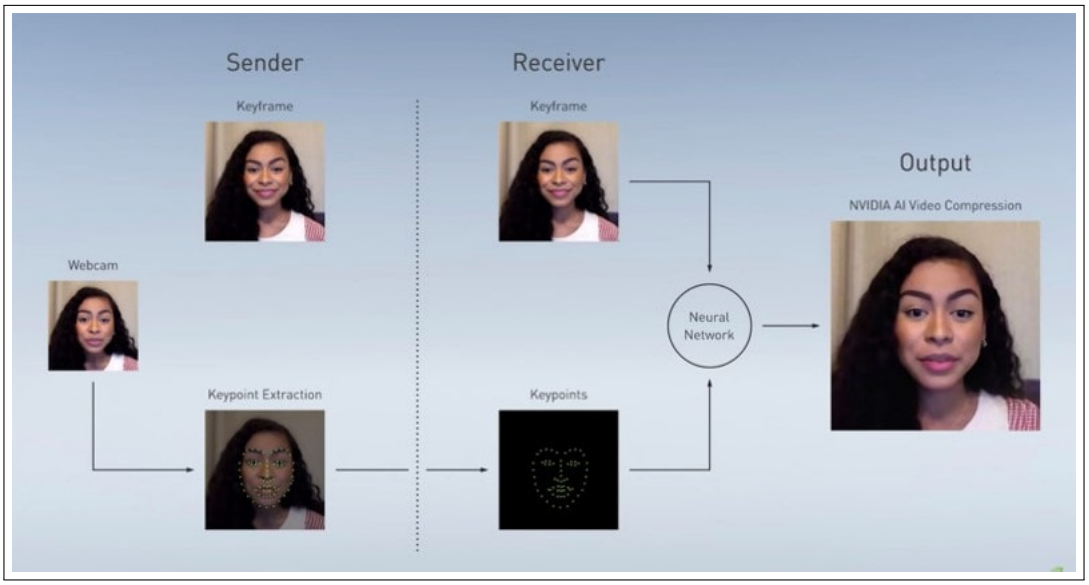

Figura 1.2: Imagen Real vs Imagen generada por GAN
[5]

1.4. Objetivos

El propósito de este trabajo es implementar una red GAN que, a partir de un conjunto de datos inicial, sea capaz de generar datos sintéticos similares a los reales. Para ello, la red constará de un discriminador que distinga entre ambos tipos de imágenes y un generador que produzca imágenes similares a las reales capaces de confundir al discriminador. Después se llevará a cabo un proceso de entrenamiento y validación para constatar el correcto funcionamiento de la red.

Estos datos artificiales deben comportarse igual que los datos suministrados al inicio, los cuales actuaran como entrada y salida de modelos posteriores (trabajo futuro).

Para ellos se han desarrollado los siguientes objetivos específicos:

- Estudio del funcionamiento de las GANs
- Implementación de una red GAN en lenguaje Python
- Estudio de los datos suministrados para el desarrollo de este proyecto
- Aplicación de dichos datos en la red GAN implementada para generar nuevos datos

- Entrenamiento de la red para generar unos datos sintéticos lo más similares posibles a los reales.
- Validación de los datos generados por la red mediante mediciones como funciones de pérdida y/o error.

1.5. Plan de Trabajo

Para el desarrollo completo de este trabajo se ha dinamizado el proyecto en fases, donde las tutoras resolvían las dudas que surgían de forma periódica.

Los pasos seguidos durante el proyecto han sido los siguientes:

a). En primer lugar se estableció una reunión donde se abarcaron los objetivos del proyecto y los pasos a seguir a lo largo de los meses, para dar lugar a una idea inicial.

b). Después de añadir un pequeño contexto sobre la temática del trabajo, el autor se formó en redes generativas adversarias a través de un curso de la plataforma Udemy, cuyo enlace quedará recogido en la bibliografía de este documento.

c). Tras el paso anterior, se suministró un dataset de varios ficheros Excel con los datos a emplear en el entrenamiento de la red.

d). El primer paso a nivel de programación fue desarrollar la red GAN para generar datos aleatorios.

e). El segundo paso de implementación de software fue desarrollar el discriminador, que se explicará en los siguientes capítulos.

f). En el tercer paso, se implementa la red uniendo los resultados del generador y el discriminador.

g). Después se lleva a cabo un proceso de entrenamiento y validación de los resultados de la red hasta que estos sean adecuados.

Para comunicarnos y llevar un control sobre las tareas a realizar nos hemos comunicado a través de correo electrónico y reuniones de Google Meet.

1.6. Asignaturas relacionadas y software utilizado

Durante el desarrollo de este trabajo de fin de grado se han empleado conocimientos adquiridos en diversas asignaturas del plan de estudios, destacando las siguientes:

- Aprendizaje automático y Big Data
- Tecnología de la programación
- Estructura de datos

En cuanto al software utilizado para implementar la red del proyecto, cabe destacar el lenguaje Python junto a varias de sus librerías más conocidas (Numpy, Matplotlib, Keras, Pandas, Tensorflow...etc). Todo el desarrollo ha sido en Jupyter Notebook dentro del entorno de Anaconda.

1.7. Estructura de la memoria

Se ha descrito el proyecto en un documento autocontenido dividido en 4 capítulos:

- El primer capítulo servirá como introducción al marco del trabajo, los objetivos definidos al inicio del proyecto, la motivación del mismo, el plan de trabajo llevado a cabo, las asignaturas cursadas a lo largo del grado cuyos conocimientos adquiridos han servido de base a la hora de comenzar el trabajo y la motivación para llevar a cabo este proyecto.
- El segundo capítulo presenta un inicio en los conceptos que engloba a una turbina eólica desde el punto de vista teórico, así como las ideas principales o básicas que comprenden una red generativa o GAN.
- El tercer capítulo detalla de forma extensa la implementación y funcionamiento de la red desarrollada en el trabajo. Se explica cada una de las partes de la red, así como su función de pérdida y optimizador. También se incluye un apartado que resume de forma breve el conjunto de datos utilizado para entrenar la red.
- El cuarto capítulo engloba los resultados obtenidos tras el entrenamiento y validación de la red.
- Por último, el quinto capítulo abarca las conclusiones del proyecto así como el trabajo a realizar en el futuro.

Capítulo 2

Introduction

2.1. Motivation

Nowadays, data modelling is used for the correct development of turbines and their corresponding control modules. One widely used option is data modelling based on Deep Learning techniques. The problem that originates in this case is the impossibility of obtaining large amounts of data, either because of its scarcity or because of the cost of obtaining it.

Therefore, the main motivation of this project is to implement a generative network capable of producing synthetic data that simulates the behaviour of data that can be obtained by other more costly means and thus be used in models developed in the future to facilitate the use of this type of turbines.

2.2. Wind Energy

Wind energy is a renewable energy source, does not run out, does not pollute and reduces the source of greenhouse gas emissions that cause global warming. Therefore, the production of electricity through wind energy, together with other renewable energies, and its use contribute to sustainable development. It should be noted that wind energy does not emit toxic substances or pollutants into the air. Such substances can acidify buildings and corrupt terrestrial and aquatic ecosystems. At the human level, it is worth noting the number of lung and cardiovascular diseases that can be caused by such pollution. Wind energy does not generate waste or pollute water, which is very important given that water scarcity is currently a global problem.

However, to convert wind energy or kinetic energy into electrical energy, we need wind turbines. They are the essential part of wind turbines. The wind hits the blades, which makes the blades rotate and with this mechanical movement of the rotor produces kinetic energy that is transformed into electrical energy. This is achieved because the rotating shaft is connected to a generator that produces the electrical energy. There are several factors that influence the amount of energy to be produced, such as the length of the blades, the size of the turbine and the rotor, the location of the wind turbine and, fun-

damentally, the wind speed.

These turbines are installed in wind farms all over the world, the most extensive being in the United States. In Spain, the most important wind farm is 'Gecama', located in the province of Cuenca, in Castilla La Mancha (figure 2.1). It covers an area of 40 square kilometres.

Figura 2.1: Parque Eólico de Gecama en Cuenca
[1]

To get the most out of turbines, it is important to have information on their operation, and data on both their dynamics and the external disturbances to which they are subjected. This information is not always available or accessible as there are still few installations of some types of offshore wind farms and more particularly, of floating offshore turbines. This type of turbine makes it possible to generate electricity in very deep waters, where other types of turbines such as fixed foundations are not viable. Floating turbines have great potential when applied in shallow water areas such as Japan, as they significantly increase the marine area available for offshore wind farms. Locating such farms far from the coast has several advantages: reducing visual pollution, achieving more consistent and stronger winds, providing more space for fishing vessels, etc.

The vast majority of floating wind turbines are at an early stage of development. The first prototypes were installed in 2007, with the only functional farm to date, the Hywind Scotland farm, being installed in 2017. It has 5 turbines with a capacity of 30 MW.

2.3. GANs

Generative antagonistic networks, or GANS, are a type of artificial intelligence (AI) algorithm proposed by Ian Goodfellow in 2014. In essence, it works by combining two neural networks that are trained against each other. This technique generates images or data that can then be treated as real (figure **??**).

So far, such networks have been used in various fields, such as interior, accessory, clothing and industrial design [[4]]. They are also used to reconstruct 3D models from 2D models

[[4]]. The main applicability of these networks today is to ”fill in missing information”. For example, in clinical diagnostics, they could predict a clinical diagnosis by helping to understand a patient's future behaviour [[4]].

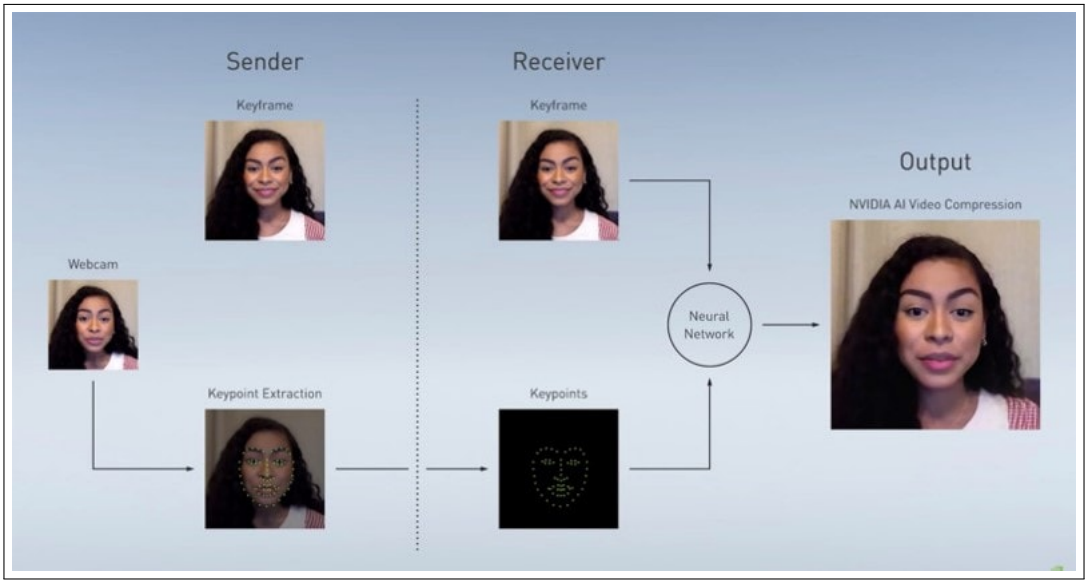

Figura 2.2: Imagen Real vs Imagen generada por GAN
[5]

2.4. Objectives

The purpose of this work is to implement a GAN network that, from an initial data set, is able to generate synthetic real-like data. For this purpose, the network will consist of a discriminator that distinguishes between both types of images and a generator that produces real-like images capable of confusing the discriminator. A training and validation process will then be carried out to verify the correct functioning of the network.

This artificial data must behave in the same way as the data supplied at the beginning, which will act as input and output for subsequent models (future work).

The following specific objectives have been developed for them:

- Study of the performing of GANs
- GAN implementation in Python language
- Study of the data supplied for the development of this project.
- Application of these data in the GAN network implemented to generate new data.
- Training of the network to generate synthetic data as similar as possible to the real data.
- Validation of the data generated by the network by means of measurements such as loss and/or error functions.

2.5. Work Plan

For the complete development of this work, the project has been dynamised in phases, where the tutors solved the doubts that arose periodically.

The steps followed during the project were as follows:

a). First of all, a meeting was established where the objectives of the project and the steps to be followed throughout the months were covered, in order to come up with an initial idea.

b). After adding a little context on the subject matter of the work, the author was trained in generative adversarial networks through a course on the Udemy platform, the link to which will be included in the bibliography of this document.

c). After the previous step, a dataset of several Excel files was provided with the data to be used in the training of the network.

d). The first programming step was to develop the GAN network to generate random data. second software implementation step was to develop the discriminator, which will be explained in the following chapters. the third step, the network is implemented by linking the results of the generator and the discriminator. a training and validation process of the network results is carried out until the results of the network are adequate.

To communicate and keep track of the tasks to be performed we have communicated by email and Google Meet meetings.

2.6. Related subjects and software used

During the development of this final degree project, knowledge acquired in various subjects of the syllabus has been used, the most important of which are the following:

- Machine Learning and Big Data
- Programming Technology
- Data Structure

As for the software used to implement the project's network, it is worth mentioning the Python language together with several of its most well-known libraries (Numpy, Matplotlib, Keras, Pandas, Tensorflow...etc). All the development has been done in Jupyter Notebook within the Anaconda environment.

2.7. Structure of memory

The project has been described in a self-contained document divided into 4 chapters:

- The first chapter will serve as an introduction to the framework of the work, the objectives defined at the beginning of the project, the motivation of the project, the work plan carried out, the subjects taken throughout the degree whose acquired knowledge have served as a basis for starting the work and the motivation to carry out this project.

- The second chapter presents an introduction to the concepts that encompass a wind turbine from a theoretical point of view, as well as the main or basic ideas that comprise a generative network or GAN.

  third chapter extensively details the implementation and operation of the network developed in this work. Each part of the network is explained, as well as its loss function and optimiser. It also includes a section that briefly summarises the dataset used to train the network.

  fourth chapter covers the results obtained after training and validation of the network.

- Finally, the fifth chapter covers the conclusions of the project as well as the work to be done in the future.

Capítulo 3

Fundamentos teóricos

3.1. Turbinas Eólicas

La energía eólica es la segunda fuente de generación eléctrica en nuestro país, con una media de 27.000 MW de potencia acumulada al año [1]. Una forma de obtención de esta energía es mediante el uso de aerogeneradores. Funcionan bajo un principio sencillo, utilizan la energía del viento (mecánica) para producir electricidad.

La fuerza del viento mueve las palas que forman una hélice, las cuales están unidas a un rotor que gira y transmite ese movimiento al generador, que es el verdadero encargado de producir la electricidad. Cuando las palas son movidas por el efecto del viento, la presión del aire disminuye por un lado y aumenta por otro. Esta diferencia de presión en ambos lados produce el movimiento de elevación y arrastre. La resistencia (inercia) del rotor es superada y por ello es obligado a moverse. Por eso hay una velocidad mínima del viento para que comience a moverse y a producir energía. Gracias a la conexión del rotor con el generador, la fuerza aerodinámica del rotor hace que el generador produzca la electricidad.

Las turbinas se pueden clasificar en función de su eje. Las turbinas de eje horizontal son las más habituales en los parques eólicos que hay distribuidos en grandes extensiones, mientras que las turbinas con el eje vertical son menos populares ya que su producción de energía es algo menor con respecto a las horizontales. Se suelen usar en ambientes urbanos. Estas últimas son omni-direccionales, es decir, no necesitan estar orientadas a la dirección de procedencia del viento para producir energía. Sin embargo, las turbinas de eje vertical, sobre todo las de tamaño mediano y pequeño, sí tienen un sistema de control que gira la góndola de la turbina, donde está el rotor con las palas, para que se enfrente al viento y lo aproveche al máximo.

En la figura 3.1 se muestran los componentes principales de una turbina eólica en su interior destacando las palas o álabes del rotor, el buje, que es la parte que sujeta las palas junto al resto de componentes. El generador, que extrae la fuerza del viento recogida por las palas para producir energía, el anemómetro, un aparato empleado para conocer la velocidad del viento...etc

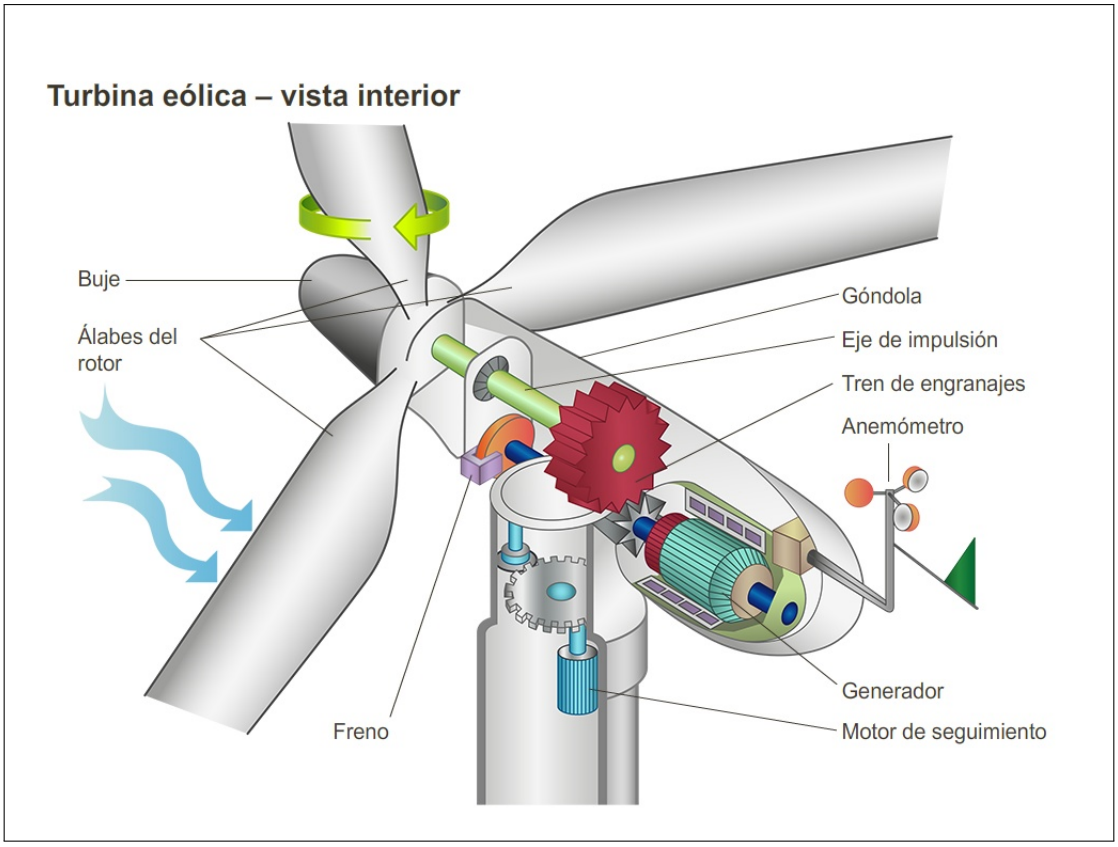

Figura 3.1: Visión interior de una turbina
[6]

Las turbinas eólicas pueden estar ubicadas en cualquier localización, ya sea en lugares marítimos (offshore), costeros o en alta mar o, en lugares terrestres interiores (onshore). Las turbinas terrestres suelen tener un tamaño pequeño o mediano, produciendo entre los 100 kilovatios hasta pocos megavatios. Cuanto mayor sea su superficie, más rentables son y se agrupan en parques eólicos. Las más pequeñas se instalan cerca de la ubicación donde se aprovechará la energía producida, por debajo de los 100 kilovatios, y suelen emplearse en residencias o pequeñas granjas agrícolas. Por otro lado, las turbinas offshore tienden a ser mucho más masivas, con una altura que ronda los 100 metros y una producción de MW. Pueden ser costeras, ancladas al lecho marino, o flotantes, en alta mar. El mayor desafío que plantean estas últimas es su transporte para la instalación, que se hace por barco. Como ventaja destaca su facilidad para aprovechar el viento oceánico, más potente y estable que el terrestre debido a la falta de obstáculos geográficos, generando mayor cantidad de energía.

3.2. Red GAN

Las redes generativas adversarias son un tipo de red neuronal que se ha empezado a aplicar en los últimos años. Fueron descubiertas por Ian Goodfellow en 2014 [3] y, desde entonces han supuesto un gran avance en el campo del aprendizaje no supervisado. Consisten esencialmente en un algoritmo basado en dos redes neuronales denominadas generador y discriminador que compiten entre sí.

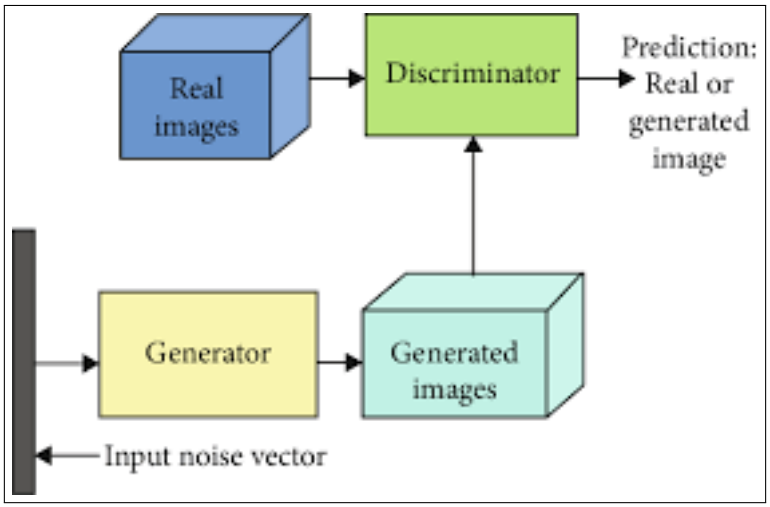

Figura 3.2: Esquema básico del funcionamiento de una red GAN [7]

En la figura 3.2 vemos de forma básica cómo funciona una red GAN. Primero, el algoritmo generador a partir de un vector de números aleatorios denominado ruido, que es su entrada, genera un conjunto de datos sintéticos que evalúa el discriminador. Este último posee las imágenes verdaderas y emite un resultado que puede ser de dos tipos: o bien la imagen generada es prácticamente igual a la real, o no se parece y es desechada.

3.2.1. Redes neuronales

Tanto el generador como el discriminador son redes neuronales, cuya arquitectura se describe en esta sección. Una red neuronal es una técnica de la Inteligencia Artificial que procesa datos. Este tipo de aprendizaje se denomina aprendizaje profundo o Deep Learning. Se basa en emplear neuronas o nodos interconectados a una estructura de capas generando un sistema dinámico que se adapta según necesidad, es decir, según avanza la ejecución de la red, ésta aprende de sus errores y mejora sus resultados, aunque no siempre es así. Se debe evitar el sobreajuste, un fenómeno que ocurre cuando la red se entrena en demasía generando datos erróneos. Así, los retos a los que se enfrentan las redes neuronales son cada vez más complejas, como reconocer rostros con precisión o intentar simular el lenguaje humano.

Una aplicación muy útil de las redes neuronales (NN) es la visión artificial. Permiten extraer información y conocimiento de vídeos e imágenes tal y como lo haría un ser humano. Algunos ejemplos de aplicación de visión artificial son: reconocimiento de señales de tráfico, reconocimiento facial para escáneres o la eliminación de contenido sensible en ciertas páginas webs [8].

Las redes neuronales emulan la estructura del cerebro del ser humano. Una red neuronal artificial está formada por neuronas que trabajan de forma cooperativa con el objetivo de resolver un problema. La arquitectura de la red se puede observar en la figura 3.3.

a). Capa de entrada

En esta primera capa, la red neuronal coge la información de entrada, es decir, los datos de entrenamiento que se le suministran. Las neuronas de esta capa los procesan y mandan la información a la siguiente capa.

b). Capa oculta

Las redes neuronales pueden tener N capas ocultas o intermedias, cuya función es analizar la información enviada por la capa anterior, la procesa y la envía a la siguiente capa.

c). Capa de salida

En la última capa se proporciona el resultado de todo el procesamiento llevado a cabo en los pasos mencionados anteriormente. La salida puede ser un valor único como 1, 0, Sí, No...etc o múltiples valores si estamos ante un problema multiclase.

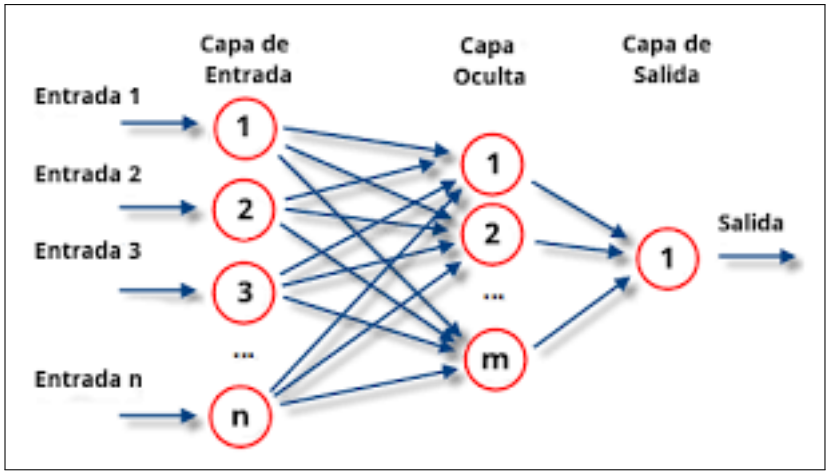

Figura 3.3: Estructura de una red neuronal
[9]

3.2.2. Entrenamiento de GAN

El proceso de entrenamiento de una red consiste en suministrar al discriminador un lote de imágenes sintéticas y otro lote de imágenes reales indicándole junto a la imagen un bit (0 si es sintético, 1 si es real) para que este aprenda a identificar diferencias. El algoritmo generativo será entrenado generando imágenes sintéticas que intenten confundir en su toma de decisión a un discriminador ya entrenado. Estos dos pasos se repiten de forma iterativa un número definido de veces, ya que cuantas más veces sea entrenada la red mejores resultados aportarán.

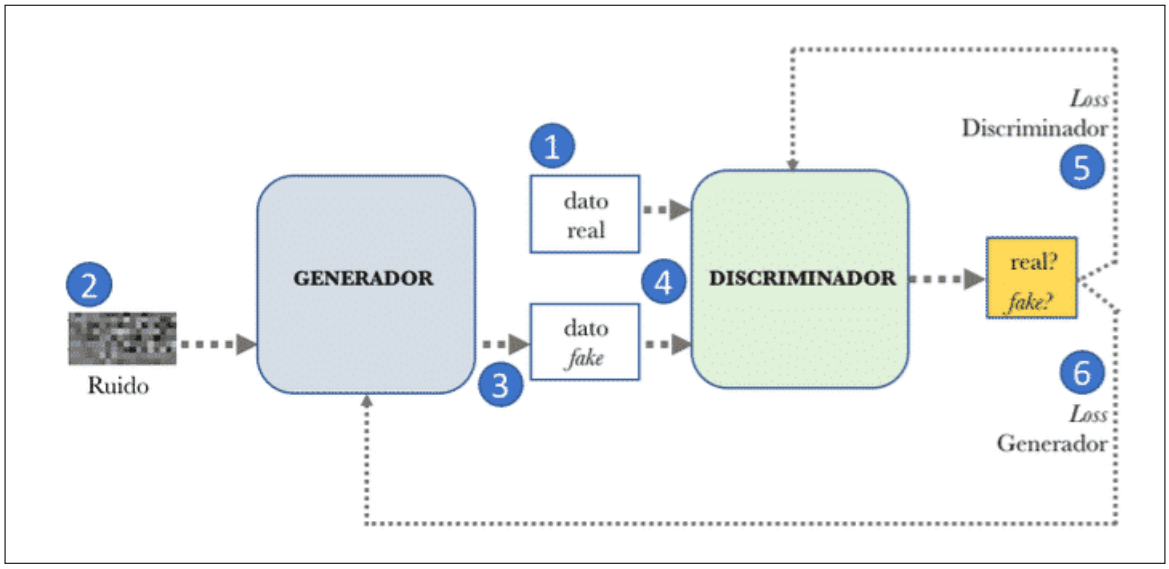

Figura 3.4: Entrenamiento de una red GAN
[10]

En la figura 3.4 se aprecian los datos reales (número 1) que debe aprender a emular lo mejor posible el generador, que son la entrada del algoritmo discriminatorio; los datos aleatorios o ruido (número 2), que emplea el algoritmo generador como inicio para producir nuevos datos. Estos datos artificiales son también datos de entrada al discriminador (número 3). Una vez que este algoritmo recibe ambos conjuntos de datos como entrada, emite un resultado (número 4) dentro del intervalo 0 a 1 para determinar, según su criterio, si el valor producido es real o no. Por último, cada resultado emitido por el discriminador es utilizado para ajustar de forma iterativa los parámetros de la red de cada algoritmo (número 5).

En esencia, aumentando la probabilidad de acierto del discriminador se obtendrán mejores resultados creados por el generador. El entrenamiento simultáneo de ambas redes neuronales hará que ambos resultados evolucionen positivamente, de ahí el nombre de redes adversarias. Los pasos que sigue el algoritmo discriminador durante el entrenamiento son los siguientes:

- Coge un conjunto de datos reales aleatorio del dataset de entrenamiento suministrado.
- Recibe los datos creados por el generador, también conocidos como datos sintéticos.
- El discriminador clasifica los datos reales y los sintéticos.

- Se calculan los valores que después adoptan ambos algoritmos para mejorar su funcionamiento.

El proceso de entrenamiento del generador se lleva a cabo según los pasos mencionados, pero basándose de forma exclusiva en los valores que ha asignado el discriminador a los datos sintéticos que produjo anteriormente.

3.2.3. Optimizadores GAN

El campo del Deep Learning evolucionó con respecto a los algoritmos iniciales de Machine Learning debido a que las soluciones que emplean estos algoritmos deben considerar no sólo la arquitectura de la red, sino también sus parámetros, pre-procesamiento de datos, hiper-parámetros...etc, para alcanzar un resultado satisfactorio. Uno de estos parámetros clave para mejorar el funcionamiento de una red GAN son los optimizadores, que ayudan a reducir el error del algoritmo. El optimizador más básico actualizaría los parámetros de forma equitativa dentro de la red, mientras que un optimizador más complejo asignaría los valores en función del peso que tenga cada parámetro de cara al resultado final.

Ejemplos de los optimizadores más utilizados en la actualidad son [11]:

- Adagrad
- Adadelta
- RMSprop
- Adam
- Nadam
- Stochastic Gradient Descent (SGD)

Algunos de los optimizadores mencionados han sido desarrollados basándose en otros ya existentes, con lo cual su funcionamiento es superior y son los más utilizados. Estos optimizadores son SGD, Adam y RMSProp.

3.2.4. Funciones de pérdida o error

Los algoritmos tanto de Machine Learning como, dentro de ese contexto, de Deep Learning, pueden ser empleados para solucionar problemas de clasificación, es decir, generar predicciones sobre un problema y dar un resultado que sea uno o cero en función de si es correcto o no. Pero durante el entrenamiento de estos algoritmos se necesita medir si la predicción es correcta o no para averiguar si el entrenamiento está siendo satisfactorio. La forma en la que se puede conocer la calidad del resultado del algoritmo es comparando dichos resultados con los datos de entrenamiento de los que se dispone. Este último paso se lleva a cabo mediante el empleo de funciones de error. Después se ajustan los parámetros del algoritmo para minimizar el error con las funciones de pérdidas. Algunas de las funciones de pérdida más reseñables son [12]:

- Error absoluto medio
- Error cuadrático medio
- Pseudo-Huber
- Logaritmo del error cuadrático medio
- Binary Cross Entropy Loss
- Cross Entropy Loss

En este proyecto se va a aplicar la función Binary Cross Entropy Loss, puesto que este tipo de función es la más indicada para problemas donde la solución debe ser cero o uno. La fórmula detallada 3.1 es:

$$-\frac{1}{n}\sum j = 1^{n}\sum i = 1^{c}[y_i \log(p_i) + (1 - y_i)\log(1 - p_i)] \tag{3.1}$$

Donde:

- y es la etiqueta, puede ser 1 si la imagen se predice como real o 0 si es sintética.
- p es el porcentaje de probabilidad dado a la imagen para que sea real.
- n es el número de imágenes en el lote suministrado.

Capítulo 4

Aplicación de una Red Generativa Adversaria

4.1. Conjunto de datos inicial

Se ha suministrado un conjunto de ficheros con formato Excel tratados como datos reales de entrada en la red GAN a implementar. Los dieciséis ficheros de datos comparten la misma estructura: 72000 filas y 97 columnas. Cada columna corresponde a una magnitud física. Los datos provienen de simulaciones de comportamiento de turbinas eólicas flotantes.
De todas la columnas existentes, hay 6 claves para el proyecto dentro del desarrollo de la red que conformarán los datos de salida del algoritmo:

- Wind1VelX: Velocidad del viento en la componente X
- Wind1VelY: Velocidad del viento en la componente Y
- Wind1VelZ: Velocidad del viento en la componente Z
- BldPitch1: Ángulo de la pala en el eje X de la turbina
- BldPitch2: Ángulo de la pala en el eje Y de la turbina
- BldPitch3: Ángulo de la pala en el eje Z de la turbina

El resto de columnas se refieren a otros parámetros del modelo de una turbina, como por ejemplo GenPwr y GenTq, que son, respectivamente, la potencia del generador y el torque del mismo. Pero las columnas sobre las cuales debe aprender principalmente la red son las 6 listadas anteriormente. Es decir, la red que se implementará en este proyecto generará un vector fila de 1x6, 1 fila y 6 columnas.

Además, podemos clasificar los ficheros aportados en 3 categorías:

- **Test22pulsetrendWindpitchN**: estos ficheros recogen las señales de pulsos o escalones para el viento con un valor pitch N. Esta N puede tomar distintos valores (0, 20, 40, 60 y 80), cada valor distinto corresponde a un fichero distinto, con lo cual tendremos 5 ficheros distintos de este tipo. A continuación, se expondrá un ejemplo de este tipo de ficheros, tomando como referencia el fichero Excel Test22pulsetrendWindpitch0, donde la variable N toma el valor 0.

	A	B	C	D	E	F	G
1	Time	Wind1VelX	Wind1VelY	Wind1VelZ	BldPitch1	BldPitch2	BldPitch3
2	0	1	0	0	0	0	0
3	0,0125	1	0	0	0	0	0
4	0,025	1	0	0	0	0	0
5	0,0375	1	0	0	0	0	0
6	0,05	1	0	0	0	0	0
7	0,0625	1	0	0	0	0	0
8	0,075	1	0	0	0	0	0
9	0,0875	1	0	0	0	0	0
10	0,1	1	0	0	0	0	0
11	0,1125	1	0	0	0	0	0
12	0,125	1	0	0	0	0	0
13	0,1375	1	0	0	0	0	0
14	0,15	1	0	0	0	0	0
15	0,1625	1	0	0	0	0	0
16	0,175	1	0	0	0	0	0

Figura 4.1: Contenido fichero Test22pulsetrendWindpitch0

En la figura 4.1 se aprecian las 16 primeras filas del archivo con las 6 columnas claves ya mencionadas, siendo cada fila una imagen que se le suministrará al discriminador dentro del lote de imágenes reales a la hora de llevar a cabo el entrenamiento del algoritmo. En este fichero las columnas pertenecientes a los ángulos del pitch se mantendrán en un valor constante igual a 0, dado que es el valor N que aparece en la máscara del fichero. Cabe destacar que la información de cada una de las filas ha sido tomada en un instante de tiempo distinto, es decir, son datos secuenciales. Por ello la primera columna es *Time*, que recoge el instante temporal donde se recolectaron los datos.

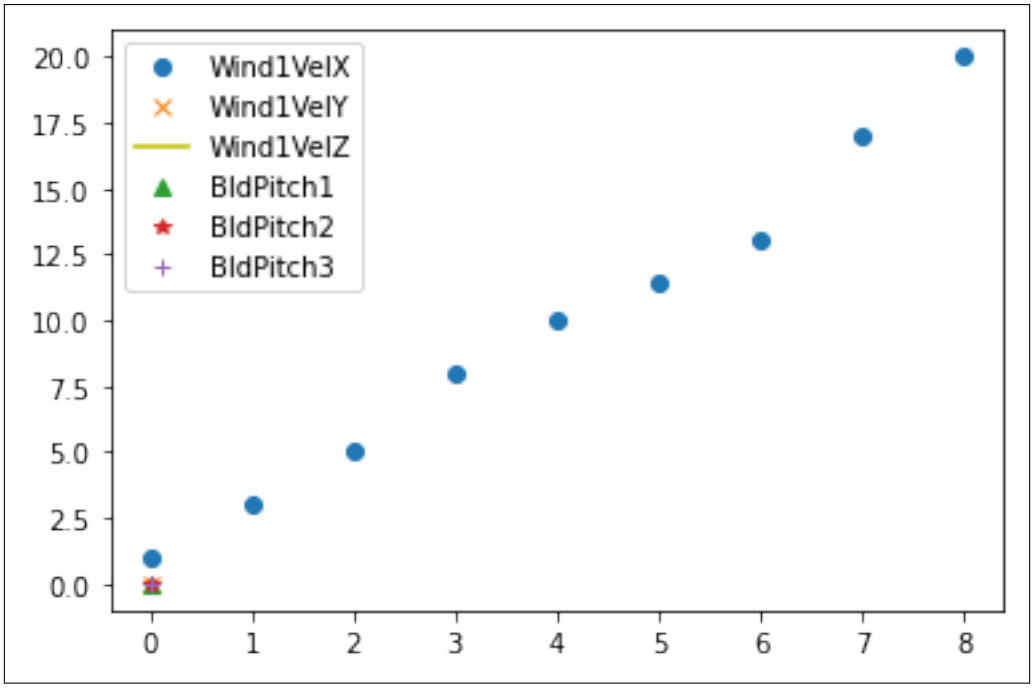

Figura 4.2: Gráfica valores Test22pulsetrendWindpitch0

En la gráfica 4.2 se aprecia el intervalo de valores de cada columna en este fichero. La columna Wind1VelX modifica su valor en las distintas filas del fichero mientras que el resto de columnas se mantiene constante en 0.

- **Test22WindKpitchpulsetrend**: ficheros que recogen las señales de pulsos para el pitch y la velocidad el viento K. La variable K puede tomar los valores 3, 5, 8, 10, 11-4, 13, 17, 20 y 21. Por ello, de este tipo de ficheros tendremos 9 distintos.

	A	B	C	D	E	F	G
1	Time	Wind1VelX	Wind1VelY	Wind1VelZ	BldPitch1	BldPitch2	BldPitch3
2	0	3	0	0	0	0	0
3	0,0125	3	0	0	0	0	0
4	0,025	3	0	0	0	0	0
5	0,0375	3	0	0	0	0	0
6	0,05	3	0	0	0	0	0
7	0,0625	3	0	0	0	0	0
8	0,075	3	0	0	0	0	0
9	0,0875	3	0	0	0	0	0
10	0,1	3	0	0	0	0	0
11	0,1125	3	0	0	0	0	0
12	0,125	3	0	0	0	0	0
13	0,1375	3	0	0	0	0	0
14	0,15	3	0	0	0	0	0
15	0,1625	3	0	0	0	0	0
16	0,175	3	0	0	0	0	0

Figura 4.3: Contenido fichero Test22Wind3pitchpulsetrend

En la figura 5.21 se aprecia en detalle la información que contiene este tipo de ficheros con un valor para K de 3, es decir, la velocidad del viento tiene valor 3. Los valores para las columnas Pitch aunque en la imagen aparezca siempre a 0 en los registros o imágenes posteriores tiene valores distintos.

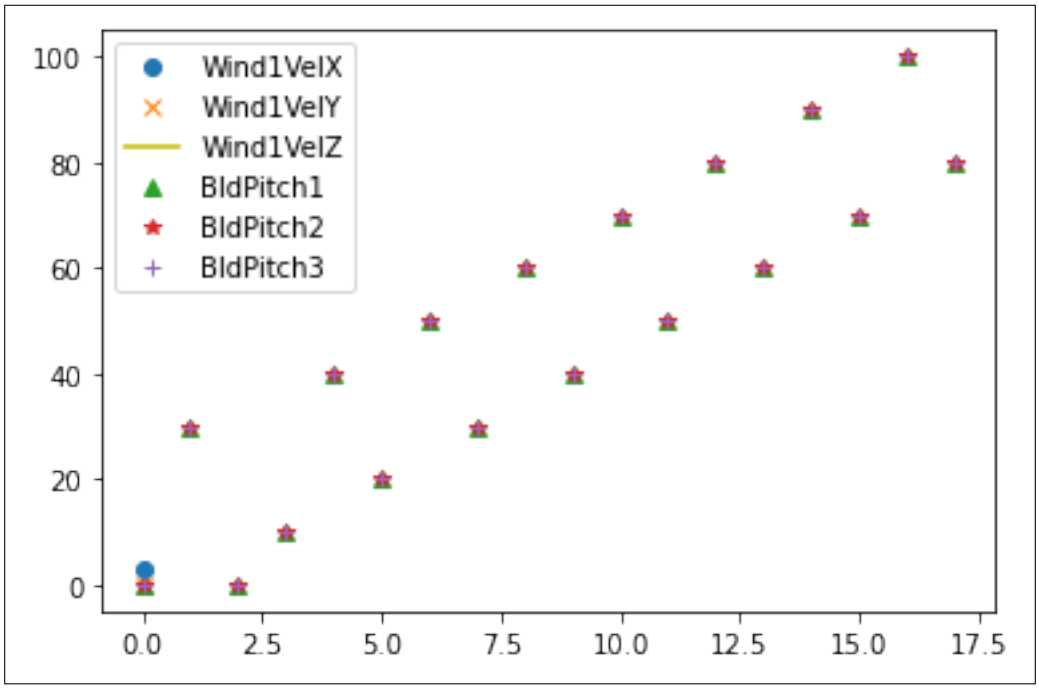

Figura 4.4: Gráfica Valores Test22Wind3pitchpulsetrend

En la gráfica 4.4 se observan los distintos valores que adoptan cada columna del fichero en todas las entradas del mismo. Las columnas BldPitch1, BldPitch2 y BldPitch3 cambian su valor mientras que el resto de columnas se mantienen constantes en 0.

- **Test22WindrandompitchrandomJ**: estos archivos recogen también las señales de pulsos con valores aleatorios tanto para la velocidad del viento como para el pitch. Estos valores aleatorios vienen definidos por la variable J, que puede adoptar valores de 0 y 1. Tendremos 2 ficheros de este tipo.

	A	B	C	D	E	F	G
1	Time	Wind1VelX	Wind1VelY	Wind1VelZ	BldPitch1	BldPitch2	BldPitch3
2	0	18,7867527	-1,13406801	0,61409909	0	0	0
3	0,0125	18,759079	-1,30654716	0,46431336	3229,20483	3229,20483	3229,20483
4	0,025	18,7314072	-1,4790318	0,31452316	-4,75676203	-4,75676203	-4,75676203
5	0,0375	18,7199917	-1,61619818	0,16595873	1071,64514	1071,64514	1071,64514
6	0,05	18,7844143	-1,5886122	0,02306262	1072,85376	1072,85376	1072,85376
7	0,0625	18,8488255	-1,561028	-0,11982476	1084,30396	1084,30396	1084,30396
8	0,075	18,9132462	-1,53344309	-0,26271647	1083,64795	1083,64795	1083,64795
9	0,0875	18,981884	-1,52623749	-0,37602168	1087,3407	1087,3407	1087,3407
10	0,1	19,0701885	-1,61410952	-0,35127297	1091,7561	1091,7561	1091,7561
11	0,1125	19,158493	-1,70198655	-0,32652265	1098,82751	1098,82751	1098,82751
12	0,125	19,2467957	-1,78985858	-0,30177397	1102,76025	1102,76025	1102,76025
13	0,1375	19,3149223	-1,86808455	-0,28508627	1107,81934	1107,81934	1107,81934
14	0,15	19,2888737	-1,90128732	-0,30601725	1109,41089	1109,41089	1109,41089
15	0,1625	19,2628231	-1,93448818	-0,32694691	1098,28271	1098,28271	1098,28271
16	0,175	19,2367744	-1,96769094	-0,34787792	1102,18567	1102,18567	1102,18567

Figura 4.5: Contenido fichero Test22Windrandompitchrandom

En la figura 5.48 se muestran algunos valores que contienen los ficheros con valores aleatorios. Como se puede observar, aunque cada fila tenga unos datos cogidos en el tiempo en un momento determinado posterior al anterior, los datos tienen valores sin correlación entre sí.

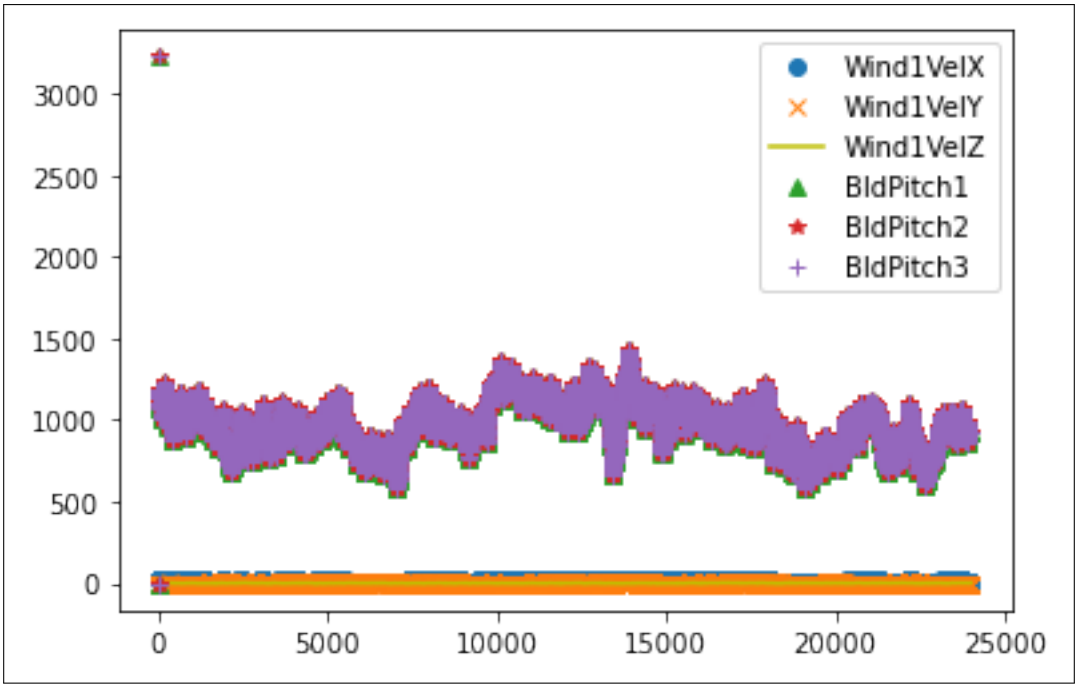

Figura 4.6: Gráfica Valores Test22Windrandompitchrandom

En la gráfica 4.6 se detalla los distintos valores aleatorios que adopta cada columna en este tipo de ficheros. Los valores de las columnas asociadas a los ángulos de palas de las turbinas oscilan en su mayoría entre 500 y 1500, mientras que la mayoría de los valores de las columnas acordes a los ángulos del viento oscilan entre 0 y 500.

Para emplear este conjunto como datos de entrada a a red se deben realizar una serie de pasos que se detallan a continuación:

- Tras cargar el dataset en una variable de la librería Pandas, éste se transforma en un array de la librería Numpy fijando el tipo de dato a Float para poder ejecutar los pasos siguientes.
- Una vez tengamos el array de Numpy listo, se deberán transformar las dimensiones añadiendo en primer lugar el número de filas que contiene el array, así como una cuarta dimensión donde se recoja el valor 1, que facilitará el tratamiento de imágenes en blanco y negro en problemas de clasificación. Esto es debido a que las liibrerías de Deep Learning utilizan tensores de 4 dimensiones y este proyecto emplea datos 1 dimensión, con lo cual las dimensiones restantes tendrán un valor de 1. La red GAN se adaptará a 1D.

4.2. Arquitectura de la red GAN

Una vez estudiado el dataset de entrada, se procede a implementar una red generativa en tres fases, una para implementar el generador, otra para el desarrollo del discriminador y una tercera para la implementación de la red GAN con los algoritmos anteriores. Además se definirán los métodos para el optimizador Adam y la función de pérdida Binary Cross Entropy.

4.2.1. Generador

El generador estará preparado para recibir como entrada un vector de número aleatorios, y, a través de las seis capas que se van a implementar, terminará aportando una imagen de 1x6. En el proceso de transformación de estos datos, el vector atravesará 6 capas distintas definidas en el algoritmo.

En cada capa el vector se irá pareciendo cada vez más al resultado esperado. La primera capa denominada *Dense*, la cual se encarga de recoger el vector con ruido aleatorio. Cada una de las capas tiene una función de comprobación para que las dimensiones de salida sean iguales a las de entrada de la siguiente capa, asegurando evitar cualquier problema en este sentido. Todas las capas, exceptuando la última de salida, contienen ciertas funciones que ayudan a pulir su correcto funcionamiento. Dichos métodos son:

- BatchNormalization: permite normalizar las convoluciones para obtener unos resultados más satisfactorios en menor tiempo, es decir, garantiza que los valores generados en cada capa tendrán un valor medio similar a cero y una desviación estándar igual a 1.
- LeakyReLU: es una función de activación que hace que la red neuronal no sea lineal. Elimina la linealidad en la función definida por la red.

Se debe destacar también el uso de la función Conv2DTranspose, que permite realizar convoluciones inversas, es decir, agrandar la imagen en cada capa para obtener el resultado preciso.

En la figura 4.7 se muestra un resumen del algoritmo generador. El generador comienza con una capa del tamaño del vector de entrada (6144 componentes), conectándose a una capa de 1024x6x1 que disminuye su tamaño a la mitad en cada capa, pasando así por 512x6x1, 256x6x1, 128x6x1 y 64x6x1, redimensionándose en cada paso, para obtener en la última capa

la dimensión esperada 1x6x1.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 6144)              36864

 batch_normalization (BatchN  (None, 6144)             24576
 ormalization)

 leaky_re_lu (LeakyReLU)     (None, 6144)              0

 reshape (Reshape)           (None, 1, 6, 1024)        0

 conv2d_transpose (Conv2DTra  (None, 1, 6, 512)        13107200
 nspose)

 batch_normalization_1 (Batc  (None, 1, 6, 512)        2048
 hNormalization)

 leaky_re_lu_1 (LeakyReLU)   (None, 1, 6, 512)         0

 conv2d_transpose_1 (Conv2DT  (None, 1, 6, 256)        3276800
 ranspose)

 batch_normalization_2 (Batc  (None, 1, 6, 256)        1024
 hNormalization)

 leaky_re_lu_2 (LeakyReLU)   (None, 1, 6, 256)         0

 conv2d_transpose_2 (Conv2DT  (None, 1, 6, 128)        819200
 ranspose)

 batch_normalization_3 (Batc  (None, 1, 6, 128)        512
 hNormalization)

 leaky_re_lu_3 (LeakyReLU)   (None, 1, 6, 128)         0

 conv2d_transpose_3 (Conv2DT  (None, 1, 6, 64)         204800
 ranspose)

 batch_normalization_4 (Batc  (None, 1, 6, 64)         256
 hNormalization)

 leaky_re_lu_4 (LeakyReLU)   (None, 1, 6, 64)          0

 conv2d_transpose_4 (Conv2DT  (None, 1, 6, 1)          1600
 ranspose)

 activation (Activation)     (None, 1, 6, 1)           0

 reshape_1 (Reshape)         (None, 1, 6, 1)           0

=================================================================
Total params: 17,474,880
Trainable params: 17,460,672
Non-trainable params: 14,208
_________________________________________________________________
```

Figura 4.7: Resumen por capas del generador

4.2.2. Discriminador

Una vez implementado el algoritmo generador de imágenes, la red requiere otro algoritmo que pueda clasificar y diferenciar las imágenes reales de las sintéticas. Ahí es donde entra el discriminador, el cual será implementado con las mismas 6 capas que el generador pero con las dimensiones invertidas, es decir, es vez de agrandar la imagen, ésta se verá reducida en tamaño.

En este caso, el discriminador requerirá como entrada una imagen del mismo tamaño que las imágenes producidas por el generador, o bien una imagen real extraída del dataset suministrado. De igual forma, las dimensiones requeridas son 1x6x1. El algoritmo dispondrá también de seis capas con los métodos BatchNormalization y LeakyReLU idénticos a los utilizados en el algoritmo anterior. Esta vez el discriminador empleará el método Conv2D en vez de Conv2DTranspose para empequeñecer el tamaño de la imagen.

Cabe destacar que en la última capa del discriminador se usa el **método Flatten** para convertir el resultado de salida de la capa anterior en un formato unidimensional. En la salida de la última capa se incluye la función Dense para indicar la categoría a la que pertenece la imagen. A diferencia del generador, la función de activación será la sigmoide. El modelo es compilado incluyendo el optimizador Adam mencionado anteriormente y la función de pérdida Binary Cross Entropy.

En la figura 4.8 se pueden observar las capas del discriminador que van desde las dimensiones 1x6x1 hasta las dimensiones 1x6x1024, las mismas que tiene la primera capa del generador, obviando la capa de entrada del vector.

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 1, 6, 64)          1600

 leaky_re_lu_5 (LeakyReLU)   (None, 1, 6, 64)          0

 conv2d_1 (Conv2D)           (None, 1, 6, 128)         204800

 batch_normalization_5 (Batc  (None, 1, 6, 128)        512
 hNormalization)

 leaky_re_lu_6 (LeakyReLU)   (None, 1, 6, 128)         0

 conv2d_2 (Conv2D)           (None, 1, 6, 256)         819200

 batch_normalization_6 (Batc  (None, 1, 6, 256)        1024
 hNormalization)

 leaky_re_lu_7 (LeakyReLU)   (None, 1, 6, 256)         0

 conv2d_3 (Conv2D)           (None, 1, 6, 512)         3276800

 batch_normalization_7 (Batc  (None, 1, 6, 512)        2048
 hNormalization)

 leaky_re_lu_8 (LeakyReLU)   (None, 1, 6, 512)         0

 conv2d_4 (Conv2D)           (None, 1, 6, 1024)        13107200

 batch_normalization_8 (Batc  (None, 1, 6, 1024)       4096
 hNormalization)

 leaky_re_lu_9 (LeakyReLU)   (None, 1, 6, 1024)        0

 flatten (Flatten)           (None, 6144)              0

 dense_1 (Dense)             (None, 1)                 6145

=================================================================
Total params: 17,423,425
Trainable params: 0
Non-trainable params: 17,423,425
_________________________________________________________________
```

Figura 4.8: Resumen por capas del discriminador

4.2.3. Optimizador y Función de pérdida

La función encargada de optimizar la calidad del resultado de la red es, como se ha mencionado anteriormente, Adam. Esta funcionalidad está definida con un hyperparámetro de entrada llamado *learning rate* o tasa de aprendizaje, cuyo valor marcará la facilidad de aprendizaje de la red. El valor seleccionado ha sido 0.001 ya que es la cifra mencionada para obtener unos resultados buenos en la relación calidad y tiempo.

En cuanto a la función de pérdida o error implementada, se han definido dos funciones, una para corroborar el error del generador, donde se comparan el número de imágenes falsas generadas con aquellas que son muy similares o iguales a las reales y podrían llegar a confundir al discriminador.

La segunda función engloba los errores producidos por el discriminador. Se calcula como la suma del error al comparar las imágenes reales con las que el discriminador cree que son reales y las imágenes falsas con las que el discriminador cree que son falsas.

La fórmula utilizada para el cálculo del error o pérdida es la 3.1 ya mencionada en el capítulo anterior.

4.2.4. Red GAN

Una vez implementados ambos algoritmos, se procede a implementar la red. Para ello primero se ha creado un contenedor mediante el método *Sequential* para albergar los algoritmos. Tras esto, se añade el generador y se determinan los coeficientes de entrenamiento del discriminador. Después se añade el discriminador, el optimizador y la función de pérdida.

En la figura 4.9 se ven ambos algoritmos de la GAN, el generador con unas dimensiones de salida tal y como se espera, y un discriminador que devuelve un único valor numérico que indica si la imagen es real con 1 o si es falsa con un 0.

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 sequential (Sequential)     (None, 1, 6, 1)           17474880

 sequential_1 (Sequential)   (None, 1)                 17423425

=================================================================
Total params: 34,898,305
Trainable params: 17,460,672
Non-trainable params: 17,437,633
_________________________________________________________________
```

Figura 4.9: Resumen por capas del generador

4.3. Entrenamiento de la red GAN

Tras implementar la red hay que entrenarla, es decir, ejecutarla un número N de veces para que los resultados mejoren progresivamente y se obtenga una salida del algoritmo de la mejor calidad posible. Para llevar a cabo el entrenamiento primero será entrenado el algoritmo discriminador y después el generativo. El motivo por el cual primero se entrena el discriminador es el siguiente: una vez el discriminador es capaz de distinguir entre datos reales y sintéticos, el generador utilizará al discriminador para entrenarse, ya que los datos producidos por el generador llegarán al discriminador para intentar engañarlo. Una vez conseguido este último paso, el generador estará entrenado. En resumen, sin tener un discriminador entrenado no se puede entrenar al generador.

Para entrenar al discriminador se le suministrará un lote de imágenes reales y otro de imágenes sintéticas. Estas últimas se generan a partir de un vector de N números aleatorios que son la entrada al generador, el cual desarrollará las imágenes del tamaño definido en su implementación. Para obtener las imágenes reales, partimos de N números aleatorios que sirvan como índice en el dataset, escogiendo así unas imágenes reales aleatorias dentro de la muestra. Una vez conseguidos los lotes pertinentes, se activarán los coeficientes de entrenamiento del discriminador, desactivados a la hora de entrenar la red GAN. Con los coeficientes activados, se le suministra al discriminador las imágenes reales indicándole que son las reales bajo un vector de *numpy* de 1s y las imágenes falsas junto a otro vector de *Numpy* de 0s. Tras llevar a cabo esta etapa, se ejecutan las funciones de pérdi-

da del discriminador para cada tipo de imagen suministrada, obteniendo los valores con los que optimizar los parámetros. Finalmente, se vuelven a desactivar los coeficientes de entrenamiento del discriminador para así posibilitar el entrenamiento del algoritmo generador.

A la hora de entrenar el generador se vuelve a generar otro vector de números aleatorios de tamaño N que se suministra a la red, con la diferencia de incluir ahora un vector de 1s que indique que las imágenes que produzca el generador serán ahora reales.

Este proceso se repite un número determinado de iteraciones para favorecer el aprendizaje de la red a la hora de diferenciar las imágenes. Para determinar el número de iteraciones que deben ejecutarse a la hora de entrenar la red, se harán distintas pruebas variando este número hasta encontrar el valor que mejor tasa de acierto aporte en menor tiempo de ejecución (Visible en el subapartado 4.0.1).

Capítulo 5

Resultados

En este capítulo se mostrarán los resultados obtenidos para cada fichero suministrado al inicio del proyecto. Para cada fichero escogido se incluirá una tabla con los resultados numéricos obtenidos en el proceso de entrenamiento con el método *train on batch* una imagen generada por la GAN y un resumen explicativo de los datos obtenidos. El concepto *batch* mencionado anteriormente, se refiere al lote de imágenes escogido para llevar a cabo el proceso de entrenamiento. Para determinar el número de ejecuciones necesarias para cada tipo de fichero, se harán pruebas con 5, 10, 20, 50 y 100 iteraciones para corroborar el número más indicado. De cada ejecución se mostrarán los 5 últimos resultados del proceso simplificando así la cantidad de valores a mostrar.

Las columnas que engloban la tabla de resultados que se muestra para cada fichero son:

- Real Discriminador: pérdida del discriminador al ser entrenado con un lote de imágenes reales.
- Sintético Discriminador: pérdida del discriminador al ser entrenado con un lote de imágenes sintéticas.
- Discriminador: pérdida global del discriminador. Este valor se calcula sumando los dos resultados anteriores.
- Generador: pérdida global del generador.

5.0.1. Fichero Velocidad del viento = 0

- Número de iteraciones = 5

Real Discriminador	Sintético Discriminador	Discriminador	Generador
0.0	0.349	0.349	6.297
0.0	0.0	0.0	10.172
0.0	1.088	1.088	8.149
0.0	0.005	0.005	8.149
0.0	0.015	0.015	9.658

Los resultados mostrados en la tabla 5.0.1 indica que para una ejecución de 5 iteraciones, el discriminador ha entrenado lo suficiente como para identificar imágenes reales, para las sintéticas los resultados podrían ser más óptimos puesto que oscilan entre 0 y 1 sin relación. El generador requiere de un mayor número de ejecuciones para disminuir su error.

Figura 5.1: Resultado fichero Test22pulsetrendWindpitch0 para 5 iteraciones

En la figura 5.1 se aprecia la imagen generada por la red tras 5 iteraciones. La primera celda de color amarillo representa la única columna de este fichero que cambia su valor en el fichero, el resto se mantiene constante a 0 y a la red le es muy sencillo imitar este resultado puesto que el valor es como ya se ha mencionado, constante.

- Número de iteraciones = 10

Real Discriminador	Sintético Discriminador	Discriminador	Generador
0.0	0.0	0.0	13.617
0.0	0.0	0.0	15.209
0.0	0.0	0.0	12.528
0.0	0.0	0.0	13.738
0.0	0.0	0.0	14.061

Con 10 iteraciones ejecutadas los resultados referidos al discriminador indican que ya estaría preparado para distinguir entre imágenes ficticias y reales, sin embargo, con respecto al generador los valores de pérdida siguen mostrando un requerimiento de mayor entrenamiento, su valor sigue comprendido entre 10 y 15. Valores apreciados en la tabla 5.0.1

Figura 5.2: Resultado fichero Test22pulsetrendWindpitch0 para 10 iteraciones

En la figura 5.2 se muestra la imagen generada tras 10 iteraciones. Al igual que en el caso anterior, la primera columna es aquella en la que el valor se modifica en el fichero y la red genera un resultado muy parecido al que debe confundir al discriminador. El resto de celdas aparecen negras ya que la red únicamente debe imitar el valor 0 y lo hace correctamente.

- Número de iteraciones = 20

Real Discriminador	Sintético Discriminador	Discriminador	Generador
0.0	0.0	0.0	2.958
0.0	0.0	0.0	3.101
0.0	0.0	0.0	3.205
0.0	0.0	0.0	2.913
0.0	0.0	0.0	2.938

Los resultados generados tras 20 iteraciones reflejan un descenso drástico del valor pérdida del generador, lo cual es positivo y se extrae que hasta el momento un buen número de ejecuciones de entrenamiento para la red es 20. Los valores del discriminador siguen siendo 0 con lo cual está bien entrenado. Valores reflejados en la tabla 5.0.1

Figura 5.3: Resultado fichero Test22pulsetrendWindpitch0 para 20 iteraciones

La figura 5.3 muestra la imagen generada a finalizar la iteraciones correspondiente, la imagen es muy similar a las ya mostradas y no merece la pena ahondar en la misma explicación.

- Número de iteraciones = 50

Real Discriminador	Sintético Discriminador	Discriminador	Generador
0.0	0.0	0.0	3.16
0.0	0.0	0.0	3.116
0.0	0.0	0.0	3.118
0.0	0.0	0.0	3.104
0.0	0.0	0.0	3.161

Los resultados obtenidos en la tabla 5.0.1 tras 50 iteraciones determinan una nula mejoría con respecto a ejecutar el proceso de entrenamiento 20 veces. El coeficiente de error del generador sigue en 3 aproximadamente. Los valores del discriminador se mantienen constantes.

Figura 5.4: Resultado fichero Test22pulsetrendWindpitch0 para 50 iteraciones

La figura 5.4 muestra el resultado visual tras 50 iteraciones con una apariencia igual al paso anterior de 20 iteraciones.

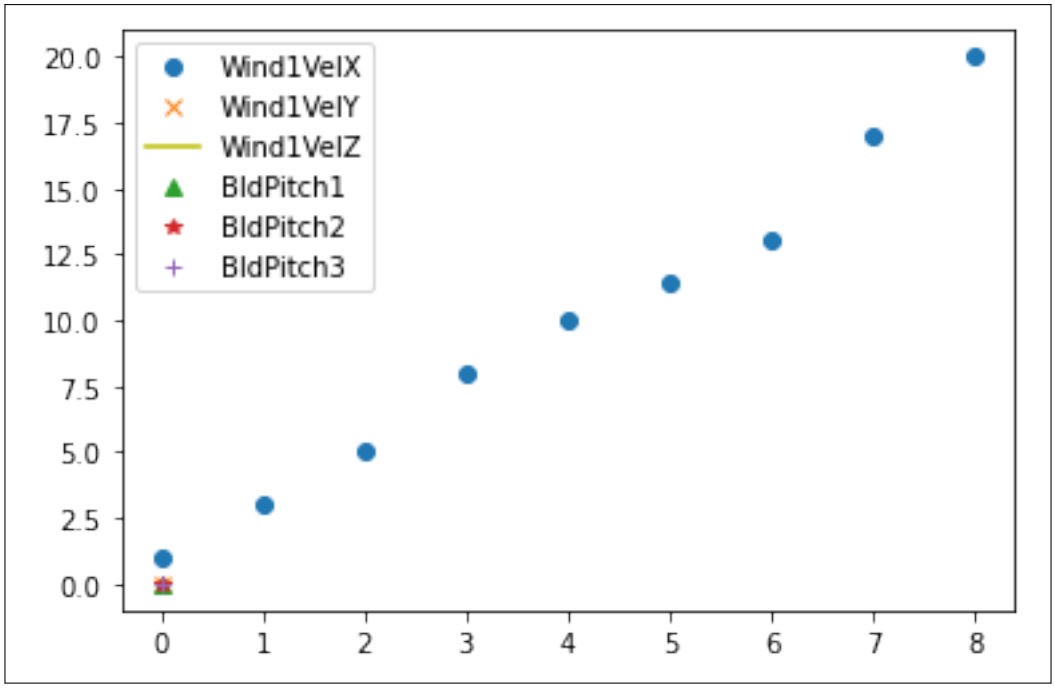

Figura 5.5: Distribución valores reales fichero Test22pulsetrendWindpitch0

La figura 5.5 muestra la distribución de valores datos del fichero suministrado.

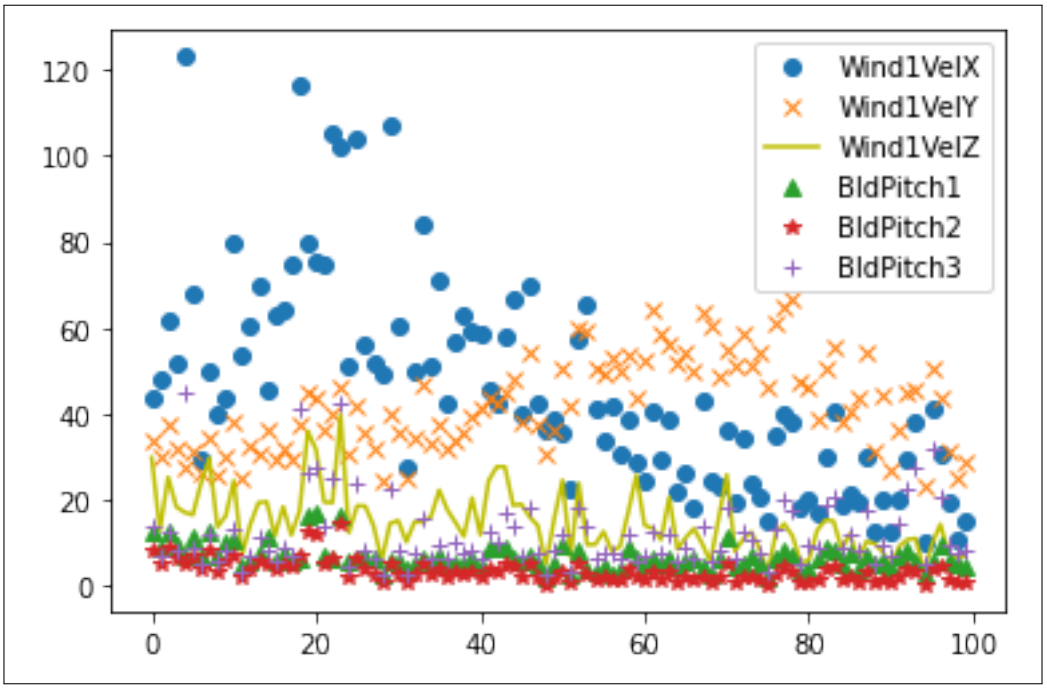

Figura 5.6: Distribución valores sintéticos fichero Test22pulsetrendWindpitch0

En la figura 5.6 se observa la distribución de valores sintéticos que genera el generador para este fichero. A destacar, la precisión casi

exacta de similitud entre los ángulos de las palas (BldPitch) y entre los ángulos de velocidad del viento (Wind1Vel).

- Número de iteraciones = 100

Real Discriminador	Sintético Discriminador	Discriminador	Generador
0.0	0.0	0.0	2.854
0.0	0.0	0.0	2.89
0.0	0.0	0.0	2.829
0.0	0.0	0.0	2.779
0.0	0.0	0.0	2.823

La tabla 5.0.1 que muestra los resultados tras 100 iteraciones denota una ligera mejoría en la pérdida del generador, descendiendo de 3 a 2. Sin embargo, al tratarse de una mejora tan leve con respecto a los valores obtenidos tras 20 y 50 iteraciones, se establecerá como valor por defecto de iteraciones de entrenamiento 50.

Figura 5.7: Resultado fichero Test22pulsetrendWindpitch0 para 100 iteraciones

La figura 5.4 muestra una imagen idéntica al paso anterior. Las columnas que deben aparecer negras por tener un valor 0 son correctas y la única columna o celda con valor variable muestra un resultado correcto.

5.0.2. Fichero Velocidad del viento = 20

Real Discriminador	Sintético Discriminador	Discriminador	Generador
0.0	0.0	0.0	9.489
0.0	0.0	0.0	9.251
0.0	0.0	0.0	9.06
0.0	0.0	0.0	9.395
0.0	0.0	0.0	9.026

Los resultados de este fichero (tabla 5.0.2) muestran un discriminador perfectamente entrenado para distinguir las imágenes reales de las sintéticas, pero el generador produce valor de error más alto que en el caso anterior. En las figuras siguientes se aprecia bien este detalle.

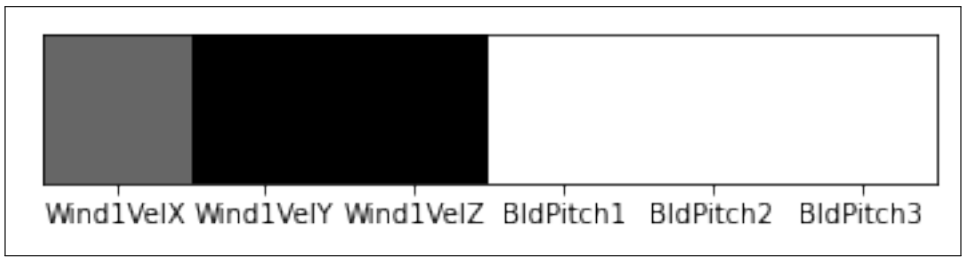

Figura 5.8: Resultado fichero Test22pulsetrendWindpitch20 Caso 1

Figura 5.9: Resultado fichero Test22pulsetrendWindpitch20 Caso 2

Las dos imágenes anteriores son aquellas que genera la red alternativamente durante las iteraciones. Por un lado la imagen 5.8 denota un valor ambiguo (no del todo correcto) para la coordenada X del viento (primera celda), mientras que en la segunda imagen 5.9 directamente muestra un valor igual a las 3 últimas celdas (ángulo de la pala). En resumen, el generador es capaz de generar valores correctos en 5 de las 6 columnas.

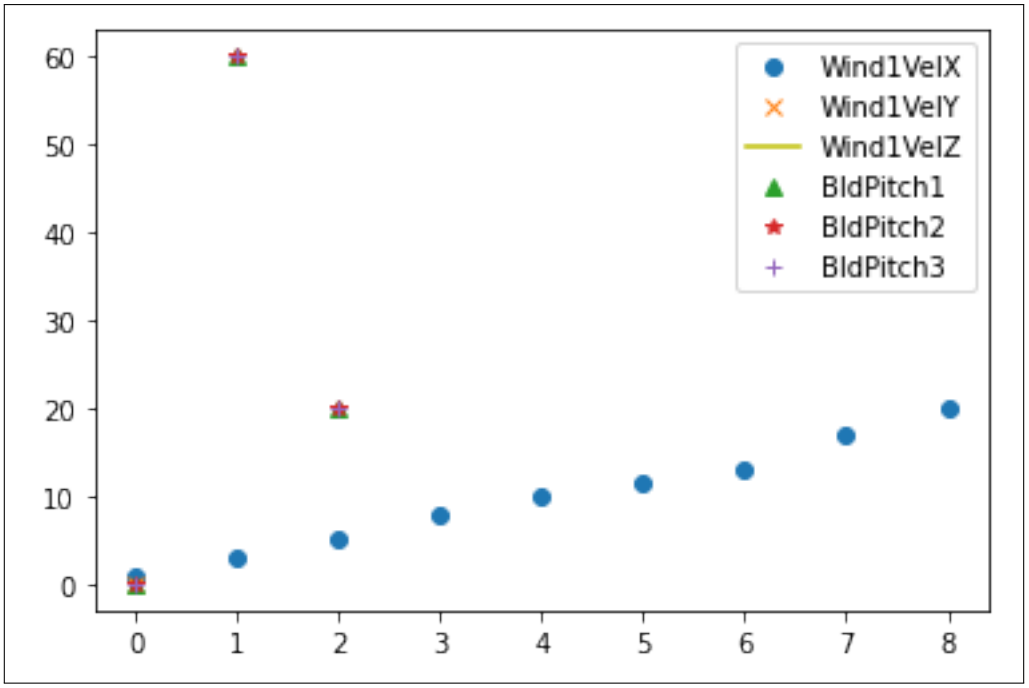

Figura 5.10: Distribución valores reales fichero Test22pulsetrendWindpitch20

La figura 5.10 detalla la distribución de datos reales del fichero suministrado.

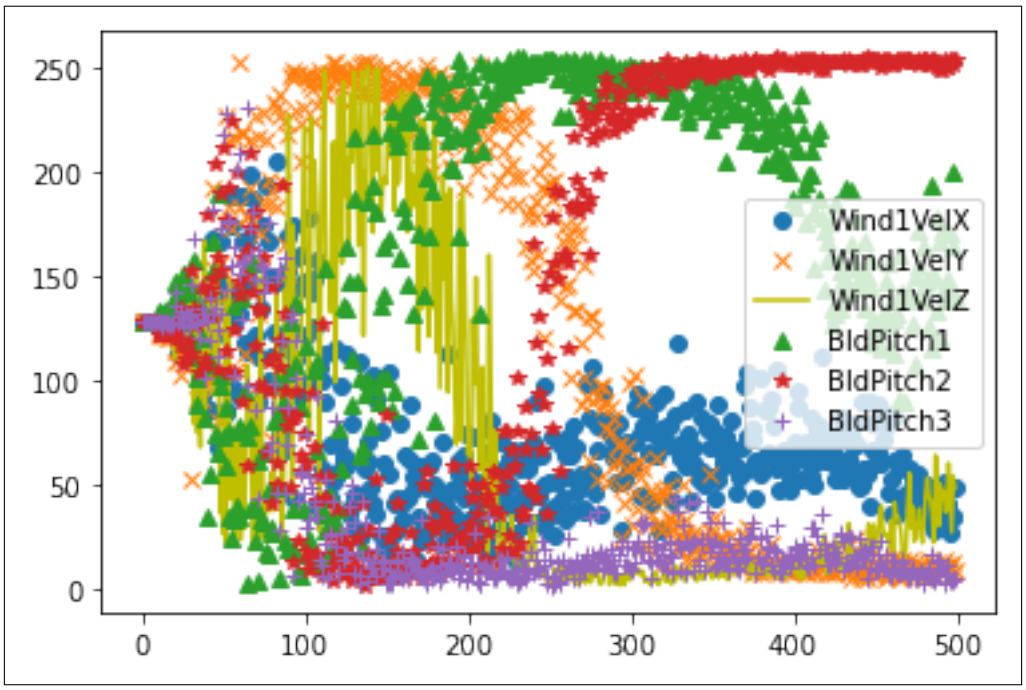

Figura 5.11: Distribución valores sintéticos fichero Test22pulsetrendWindpitch20

La figura 5.11 muestra la distribución de valores sintéticos obtenidos para este fichero. A destacar, la correlación de valores entre BldPitch1 y BldPitch2.

5.0.3. Fichero Velocidad del viento = 40

Real Discriminador	Sintético Discriminador	Discriminador	Generador
0.002	0.001	0.003	8.141
0.003	0.002	0.005	6.526
0.003	0.002	0.005	5.086
0.003	0.009	0.012	7.239
0.003	0.002	0.005	0.385

Los resultados de este fichero (tabla5.0.3) muestran un discriminador muy bien entrenado para detectar diferencias entre imágenes, con una pérdida de error mínima. Mientras que el generador mejora al calcular el valor de la primera columna, en el resto sigue identificando el valor a generar de forma correcta.

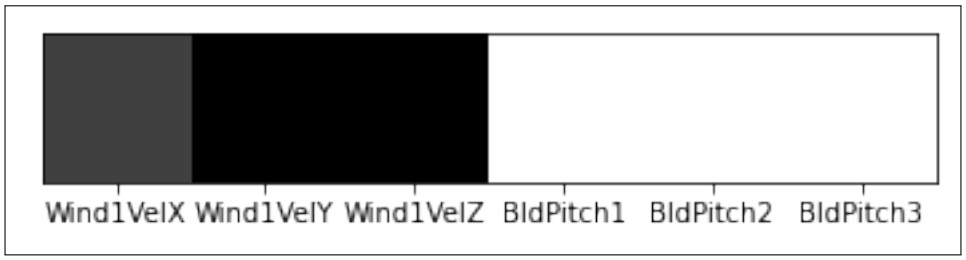

Figura 5.12: Resultado fichero Test22pulsetrendWindpitch40

La figura 5.12 muestra la imagen generada por la red. Esta vez es más precisa que en la subsección anterior como también se ha podido comprobar en los valores de error.

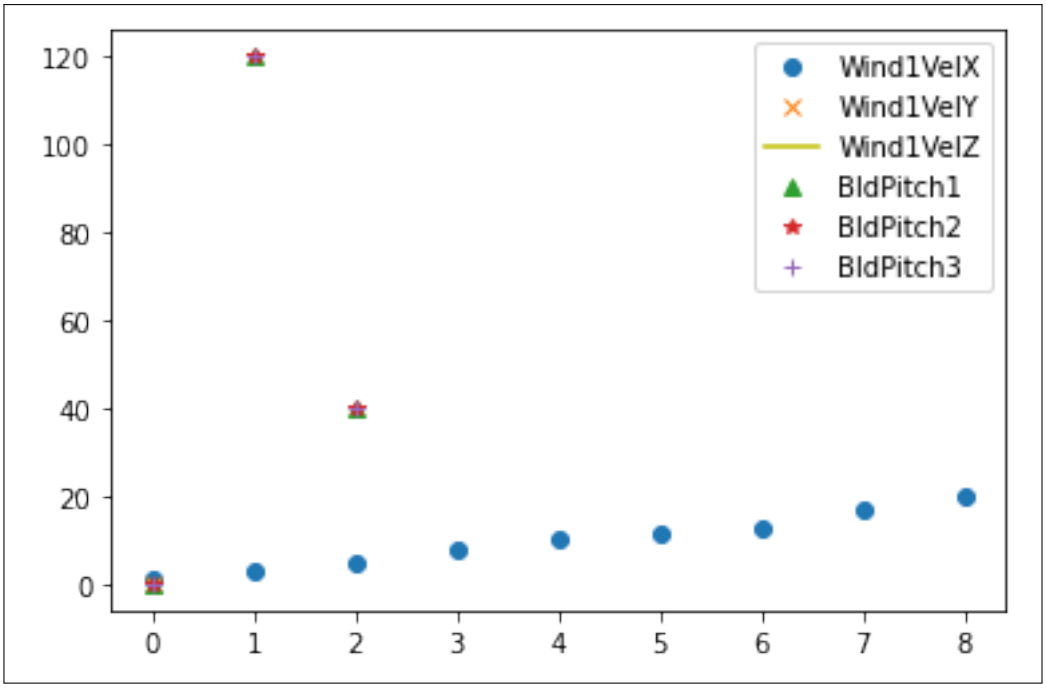

Figura 5.13: Distribución valores reales fichero Test22pulsetrendWindpitch40

La figura 5.13 expone la distribución de datos reales del fichero suministrado.

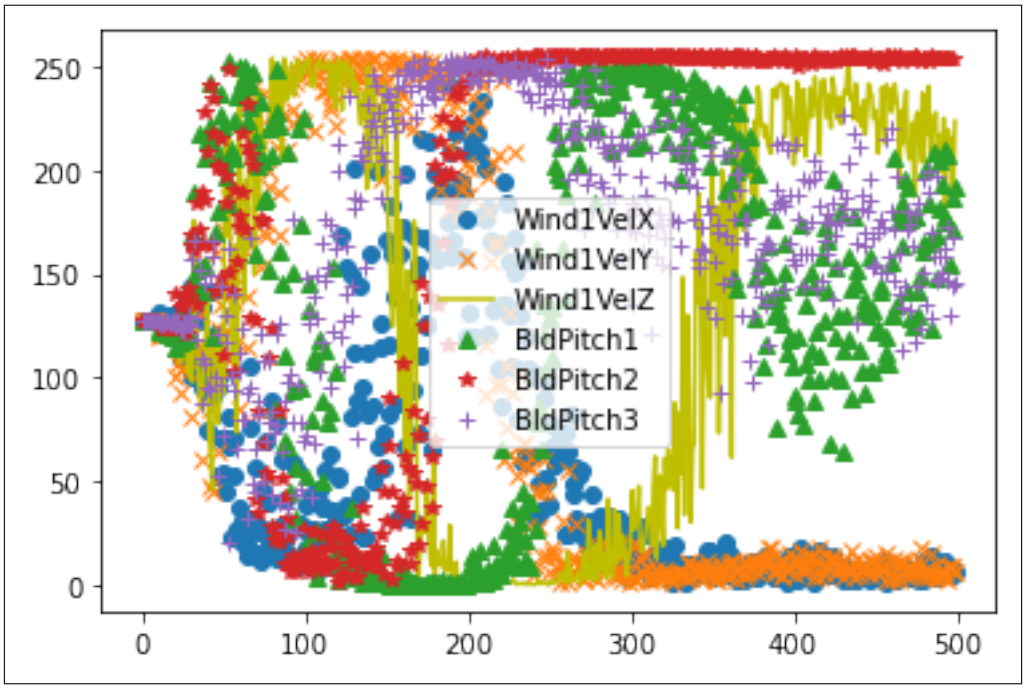

Figura 5.14: Distribución valores sintéticos fichero Test22pulsetrendWindpitch40

La figura 5.14 muestra los valores que genera el generador durante la etapa de entrenamiento. Destaca la correlación de valores entre Wind1VelX,

Wind1VelY y Wind1VelZ. También se extraen similitudes entre BLD-Pitch1, BldPitch2 y BldPitch3.

5.0.4. Fichero Velocidad del viento = 60

Real Discriminador	Sintético Discriminador	Discriminador	Generador
0.0	0.0	0.0	14.613
0.0	0.0	0.0	14.63
0.0	0.0	0.0	14.313
0.0	0.0	0.0	14.017
0.0	0.0	0.0	13.309

Los resultados de este fichero muestran un discriminador perfectamente entrenado para distinguir entre imágenes reales y sintéticas. Con respecto al generador, el valor de error aumenta con respecto a los anteriores ficheros ejecutados. Valores en la tabla 5.0.4

Figura 5.15: Resultado fichero Test22pulsetrendWindpitch60

En la imagen 5.15 se muestra la figura generada por la red al final de las ejecuciones de entrenamiento. Se aprecia como de las 3 primeras celdas con valores (correspondientes a los 3 ejes de velocidad del viento X,Y y Z) tienen distinto resultado. En las coordenadas Y y Z los valores están bien generados, el color de la variable X no termina de definirse en un color concreto obteniendo una apariencia grisácea.

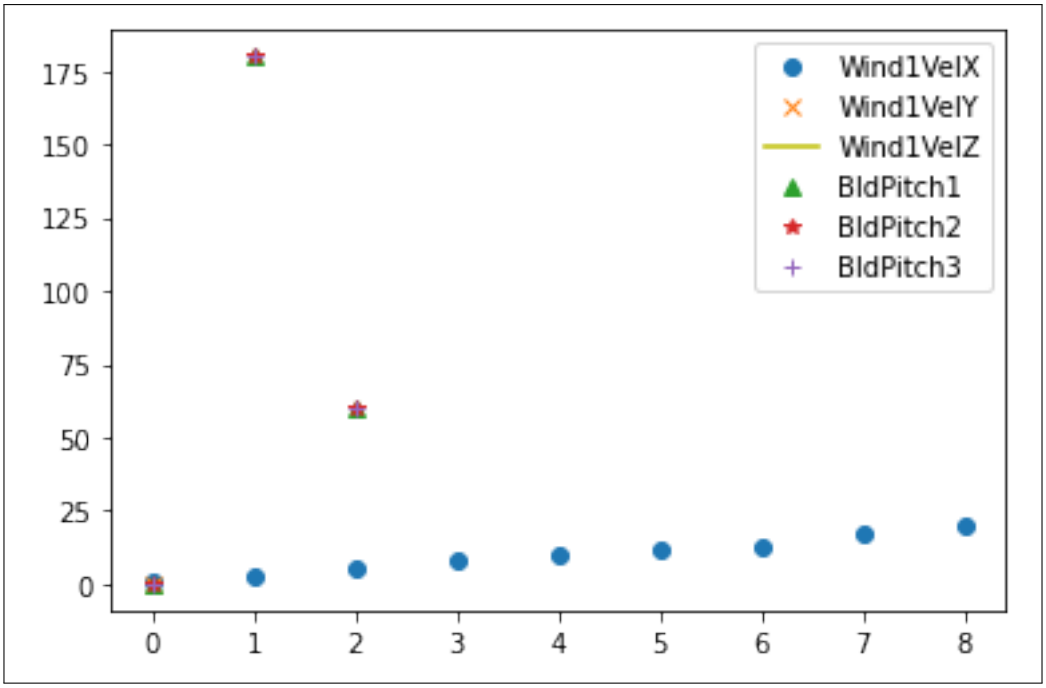

Figura 5.16: Distribución valores reales fichero Test22pulsetrendWindpitch60

La figura 5.16 expone la distribución de datos reales del fichero suministrado.

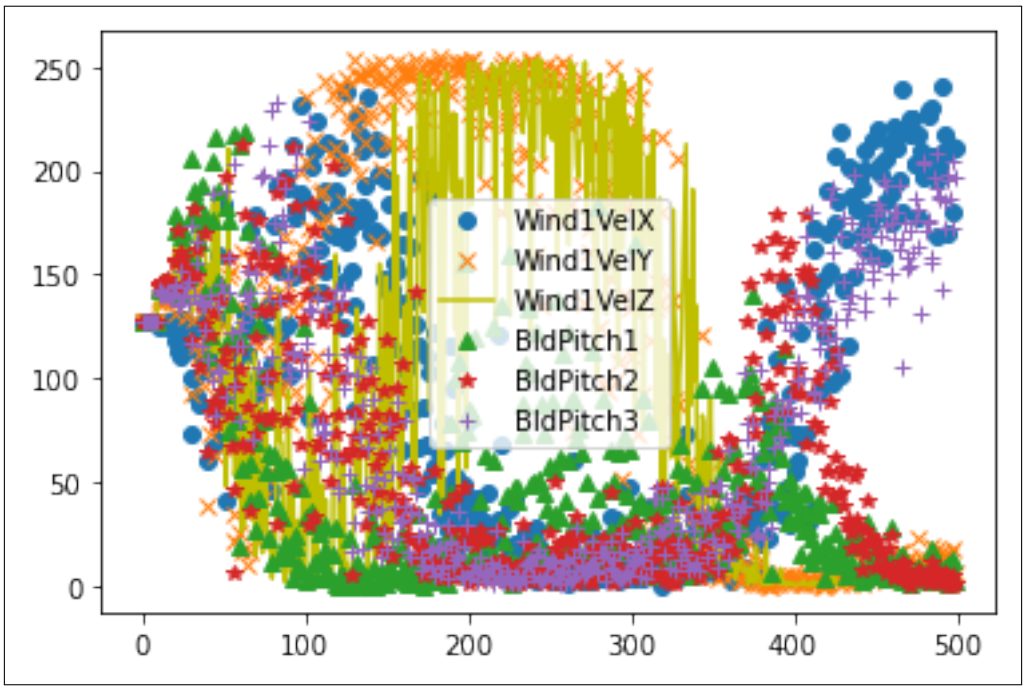

Figura 5.17: Distribución valores sintéticos fichero Test22pulsetrendWindpitch60

La figura 5.17 muestra la distribución de valores asignados en cada columna por el generador. Se puede observar una fuerte correlación entre los valores de ángulos de las palas (BldPitch).

5.0.5. Fichero Velocidad del viento = 80

Real Discriminador	Sintético Discriminador	Discriminador	Generador
0.0	0.0	0.0	6.283
0.0	0.0	0.0	6.278
0.0	0.0	0.0	5.799
0.0	0.0	0.0	5.361
0.0	0.0	0.0	5.192

Los resultados de este fichero (5.0.5) muestran un discriminador perfectamente entrenado para distinguir entre imágenes reales y sintéticas. Con respecto al generador, el valor de error disminuye con respecto al fichero anterior.

Figura 5.18: Resultado fichero Test22pulsetrendWindpitch80

En la imagen 5.18 se muestra la figura generada por la red al final de las ejecuciones de entrenamiento. Se aprecia como mejora el valor de la coordenada X del viento con respecto al fichero anterior.

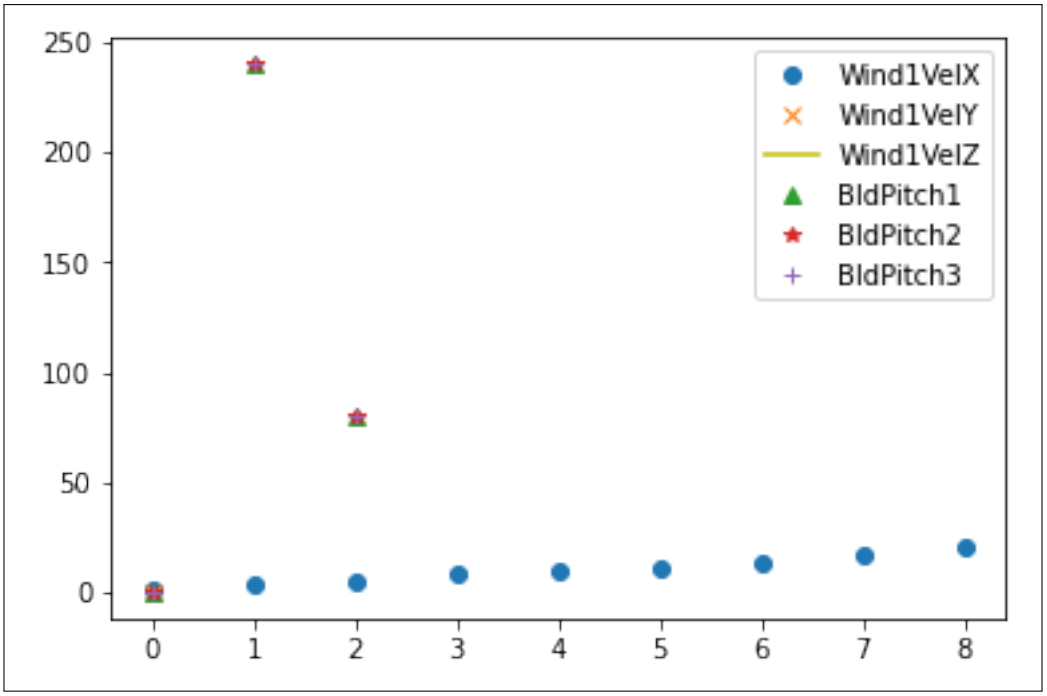

Figura 5.19: Distribución valores reales fichero Test22pulsetrendWindpitch80

La figura 5.19 detalla la distribución de datos reales del fichero suministrado.

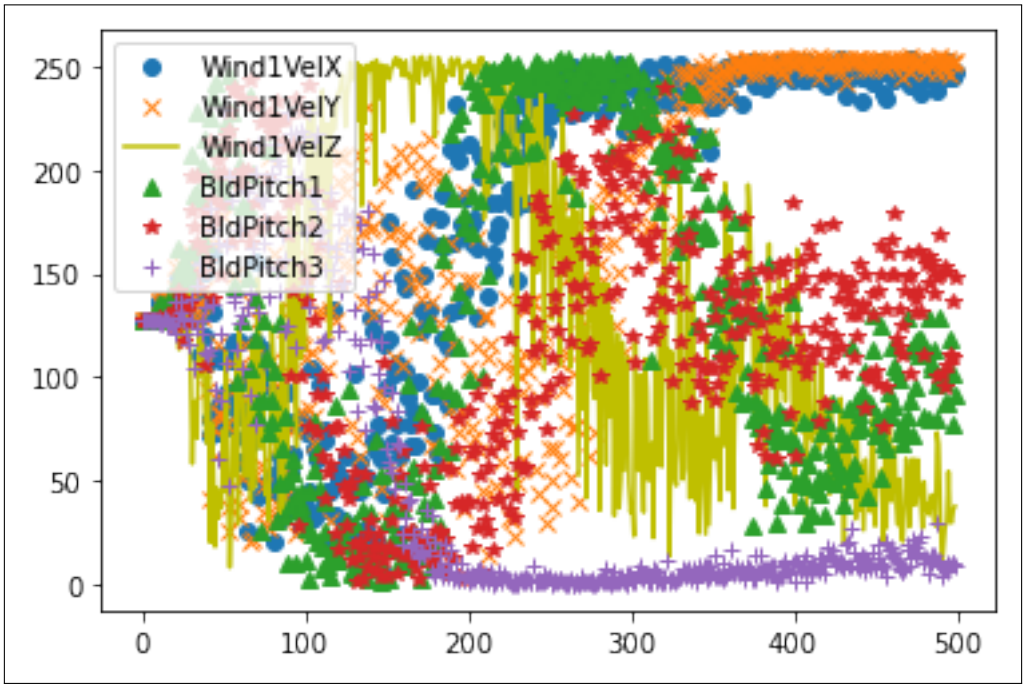

Figura 5.20: Distribución valores sintéticos fichero Test22pulsetrendWindpitch80

La figura 5.20 muestran la distribución de valores sintéticos para este fichero. A destacar, la fuerte correlación entre los valores de las columnas Wind1VelX y Widn1VelY, y BldPitch1 y BldPitch2.

5.0.6. Fichero con ángulo del pitch 3

Real Discriminador	Sintético Discriminador	Discriminador	Generador
0.0	0.001	0.001	5.31
0.0	0.001	0.001	5.212
0.0	0.0	0.0	4.775
0.0	0.0	0.0	4.873
0.0	0.0	0.0	5.018

Los resultados de este fichero (tabla5.0.6) muestran un discriminador casi perfectamente entrenado para distinguir entre imágenes reales y sintéticas, con un mínimo error casi imperceptible al detectar imágenes sintéticas. Con respecto al generador, el valor de pérdida se sitúa entre 4 y 5.

Figura 5.21: Resultado fichero Test22Wind3pitchpulsetrend

La imagen 5.21 muestra la imagen generada para el ángulo del pitch o pala igual a 3. Los valores generados que dan a lugar a la imagen son bien reconocidos por la red.

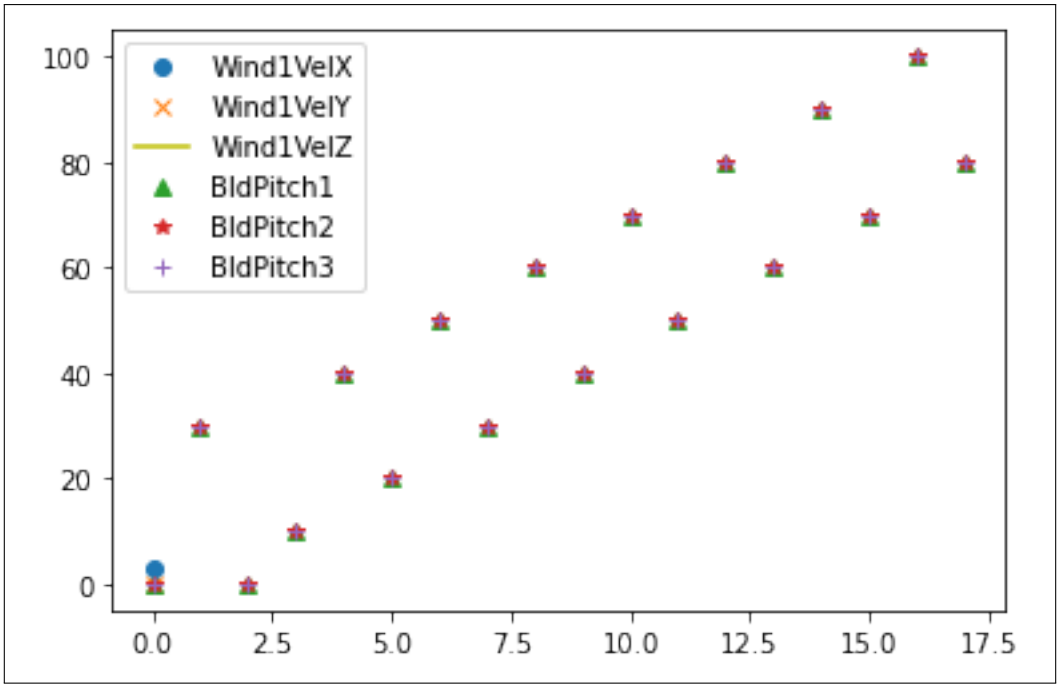

Figura 5.22: Distribución valores reales fichero Test22Wind3pitchpulsetrend

La figura 5.22 expone la distribución de datos reales del fichero suministrado.

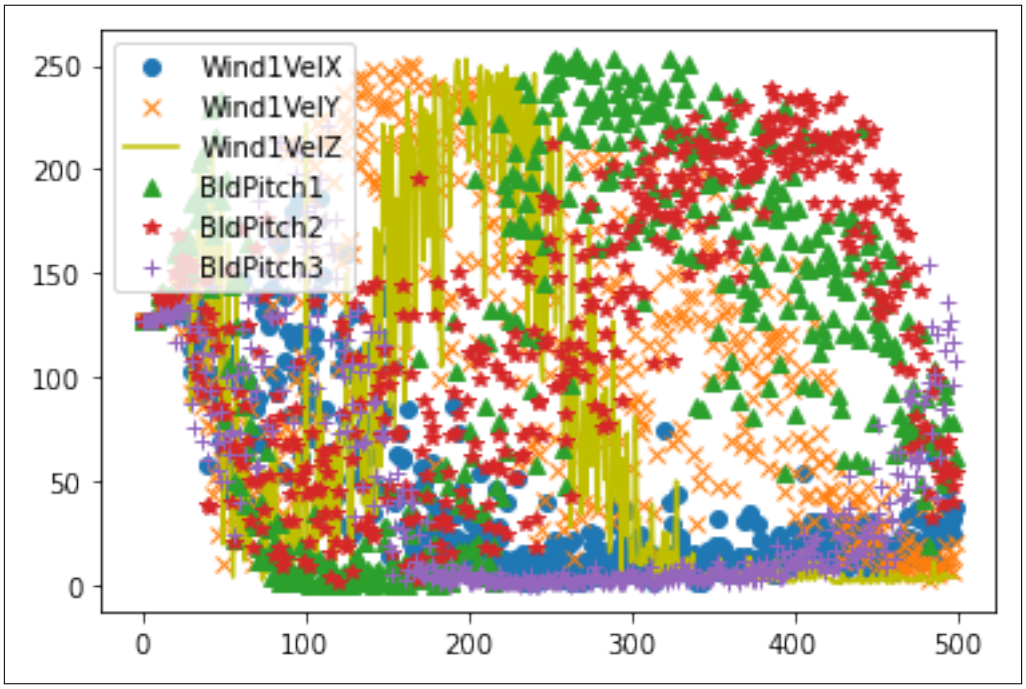

Figura 5.23: Distribución valores sintéticos fichero Test22Wind3pitchpulsetrend

En la figura 5.23 se observa la distribución de datos sintéticos proporcionados por el generador. Se oberva una correlación entre los valores de

BldPitch1 y BldPitch2, así como entre las coordenadas Z e Y de la velocidad del viento.

5.0.7. Fichero con ángulo del pitch 5

Real Discriminador	Sintético Discriminador	Discriminador	Generador
0.0	0.0	0.0	10.499
0.0	0.0	0.0	10.934
0.0	0.0	0.0	10.698
0.0	0.0	0.0	11.16
0.0	0.0	0.0	11.066

Los resultados de este fichero muestran un discriminador perfectamente entrenado para distinguir entre imágenes reales y sintéticas. Con respecto al generador, el valor de error prácticamente duplica a los obtenidos en el fichero anterior. Valores en la tabla 5.0.7.

Figura 5.24: Resultado fichero Test22Wind5pitchpulsetrend

La figura 5.24 muestra la imagen generada por la red en este caso, donde se aprecia el aumento del valor de error del generador, puesto que la imagen es de menor calidad que la anterior en la primera celda, la coordenada X del viento.

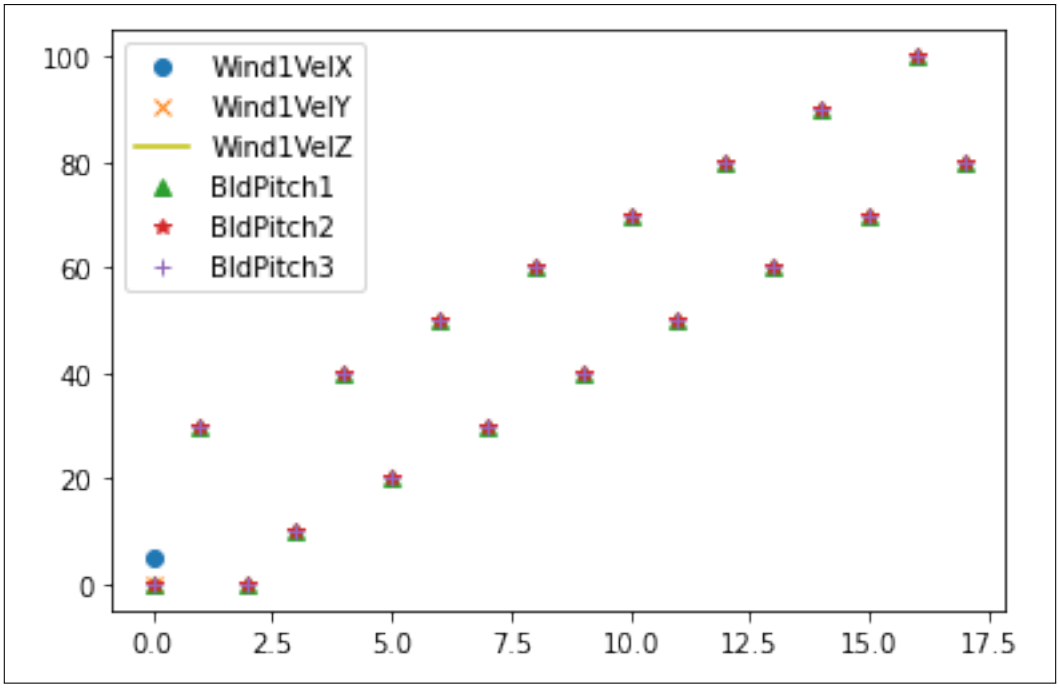

Figura 5.25: Distribución valores reales fichero Test22Wind5pitchpulsetrend

La figura 5.25 expone la distribución de datos reales del fichero suministrado.

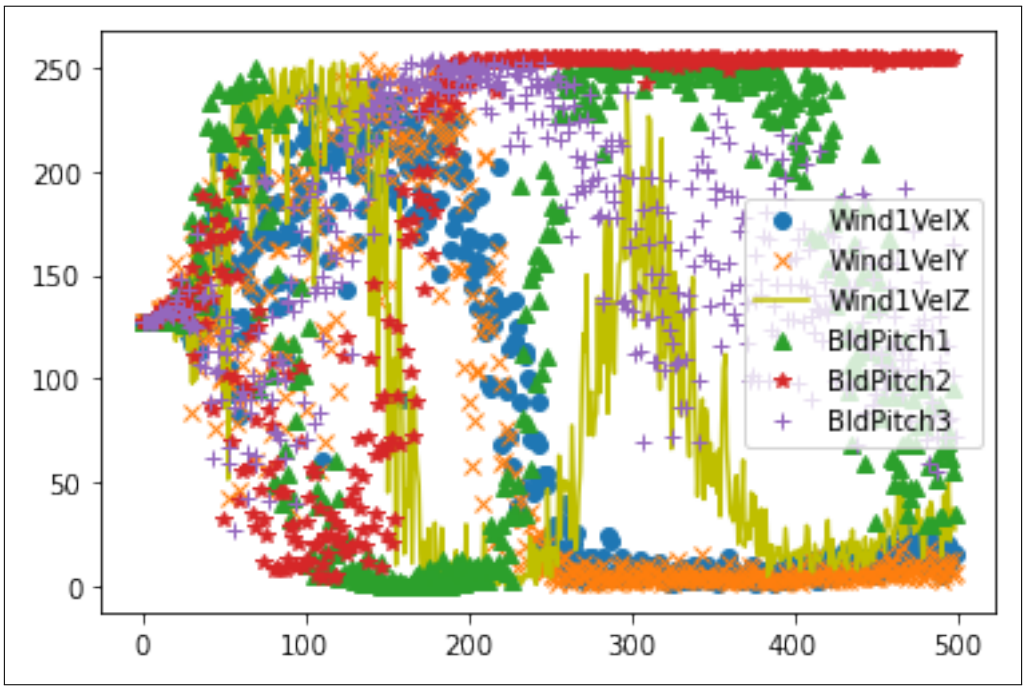

Figura 5.26: Distribución valores sintéticos fichero Test22Wind5pitchpulsetrend

En la figura 5.26 muestra la distribución de datos sintéticos proporcionados por el generador. Se observan correlaciones entre las columnas Wind1VelX y Wind1VelY. También entre las columnas BldPitch1 y BldPitch2.

5.0.8. Fichero con ángulo del pitch 8

Real Discriminador	Sintético Discriminador	Discriminador	Generador
0.0	0.0	0.0	7.176
0.0	0.0	0.0	7.112
0.0	0.0	0.0	7.126
0.0	0.0	0.0	7.106
0.0	0.0	0.0	7.26

Los resultados obtenidos en la tabla 5.0.8 indican un algoritmo discriminativo correctamente entrenado y un algoritmo generativo con un valor de error de aproximadamente 7.

Figura 5.27: Resultado fichero Test22Wind8pitchpulsetrend

La figura 5.27 muestra la imagen generada por el generador con unos resultados bastante parecidos entre sí y de una calidad mayor a los anteriores subapartados.

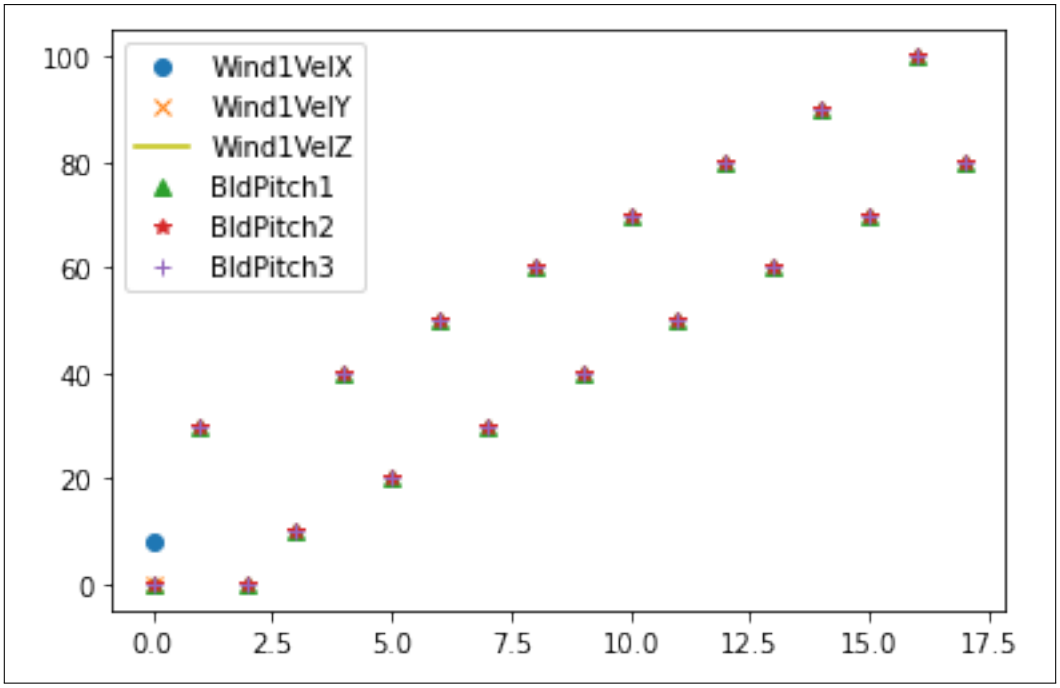

Figura 5.28: Distribución valores reales fichero Test22Wind8pitchpulsetrend

La figura 5.28 expone la distribución de datos reales del fichero suministrado.

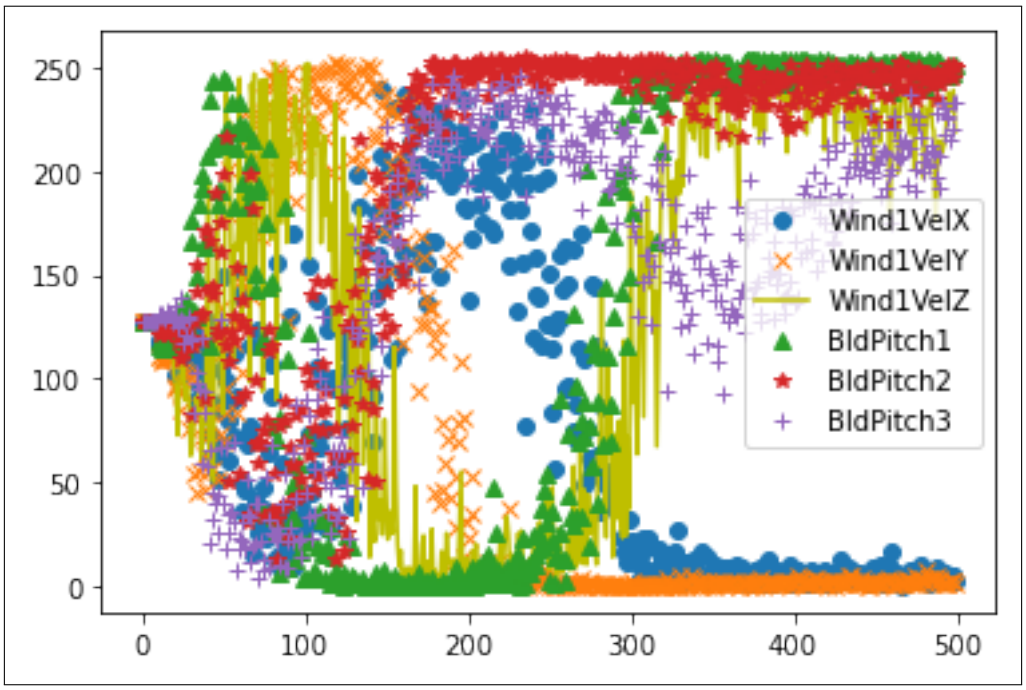

Figura 5.29: Distribución valores sintéticos fichero Test22Wind8pitchpulsetrend

En la figura 5.29 muestra la distribución de datos sintéticos proporcionados por el generador.Se observan correlaciones de datos entre Wind1Velx y Wind1VelY, BldPitch1, BldPitch2 y BldPitch3.

5.0.9. Fichero con ángulo del pitch 10

Real Discriminador	Sintético Discriminador	Discriminador	Generador
0.0	0.0	0.0	7.901
0.0	0.0	0.0	8.247
0.0	0.0	0.0	7.992
0.0	0.0	0.0	8.133
0.0	0.0	0.0	8.166

Los resultados de este fichero en la tabla 5.0.9 muestran un discriminador perfectamente entrenado para distinguir entre imágenes reales y sintéticas. En cuanto al generador, el valor de error es menor que en el fichero anterior pero mayor al primero.

Figura 5.30: Resultado fichero Test22Wind10pitchpulsetrend

En la figura 5.30 se extrae la imagen resultante en la última iteración de entrenamiento. Se aprecia un valor semi-erróneo en la coordenada X del viento, de ahí su color grisáceo.

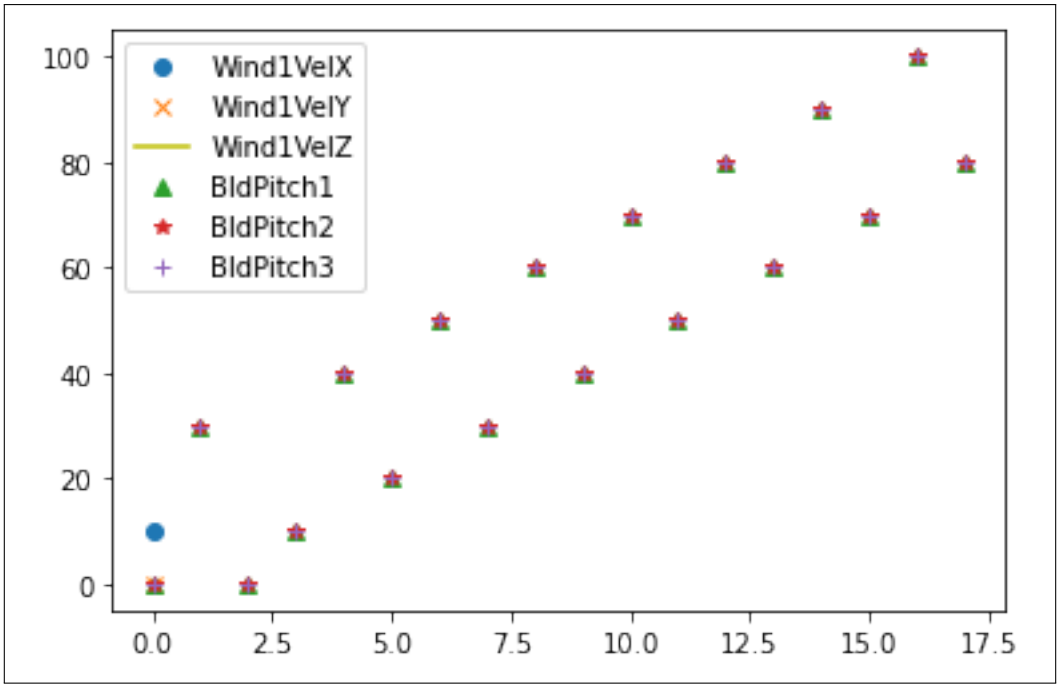

Figura 5.31: Distribución valores reales fichero Test22Wind10pitchpulsetrend

La figura 5.31 expone la distribución de datos reales del fichero suministrado.

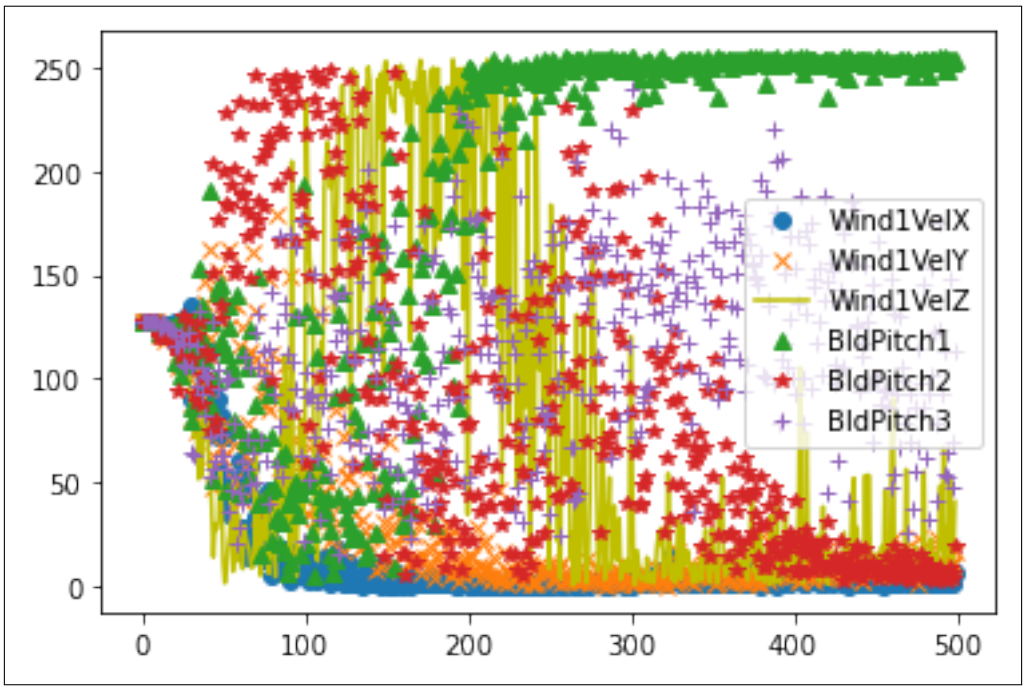

Figura 5.32: Distribución valores sintéticos fichero Test22Wind10pitchpulsetrend

En la figura 5.32 se observa la distribución de datos sintéticos proporcionados por el generador. Destacar valores similares entre las columnas Wind1VelX y Wind1VelY.

5.0.10. Fichero con ángulo del pitch 11-4

Real Discriminador	Sintético Discriminador	Discriminador	Generador
0.0	0.0	0.0	5.271
0.0	0.0	0.0	5.459
0.0	0.0	0.0	4.984
0.0	0.0	0.0	5.77
0.0	0.0	0.0	5.543

Los resultados de este fichero (tabla 5.0.10) muestran un discriminador perfectamente entrenado para distinguir entre imágenes reales y sintéticas. Con respecto al generador, el valor de error continúa mejorando en comparación a los anteriores subapartados.

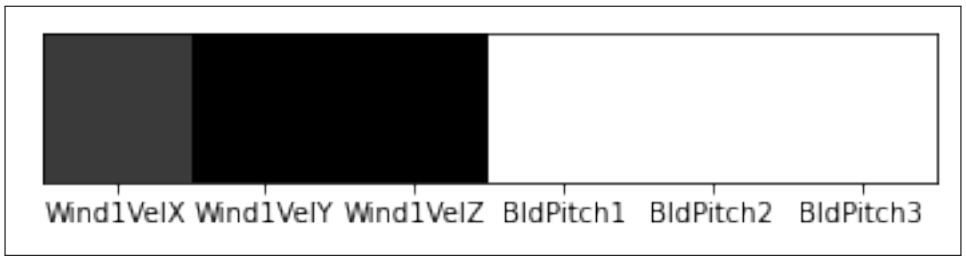

Figura 5.33: Resultado fichero Test22Wind11-4pitchpulsetrend

La figura 5.33 expone los resultados tras la ejecución dando lugar a una imagen de buena calidad donde los valores 0 (blanco) son correctos y el resto obtienen un color bastante similar a lo buscado.

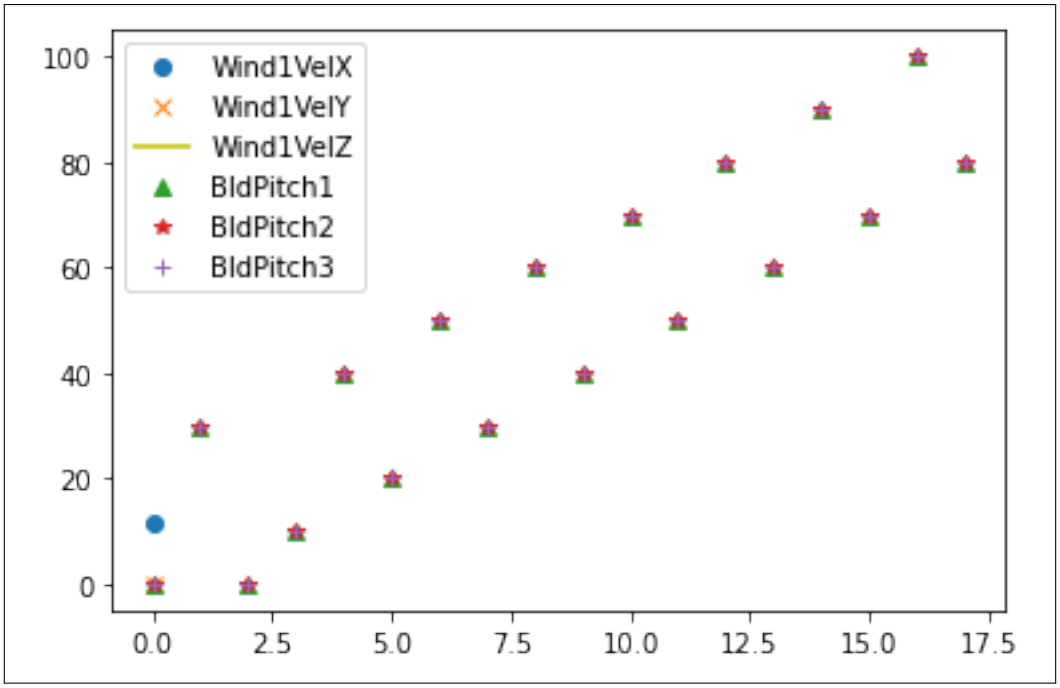

Figura 5.34: Distribución valores reales fichero Test22Wind11-4pitchpulsetrend

La figura 5.34 expone la distribución de datos reales del fichero suministrado.

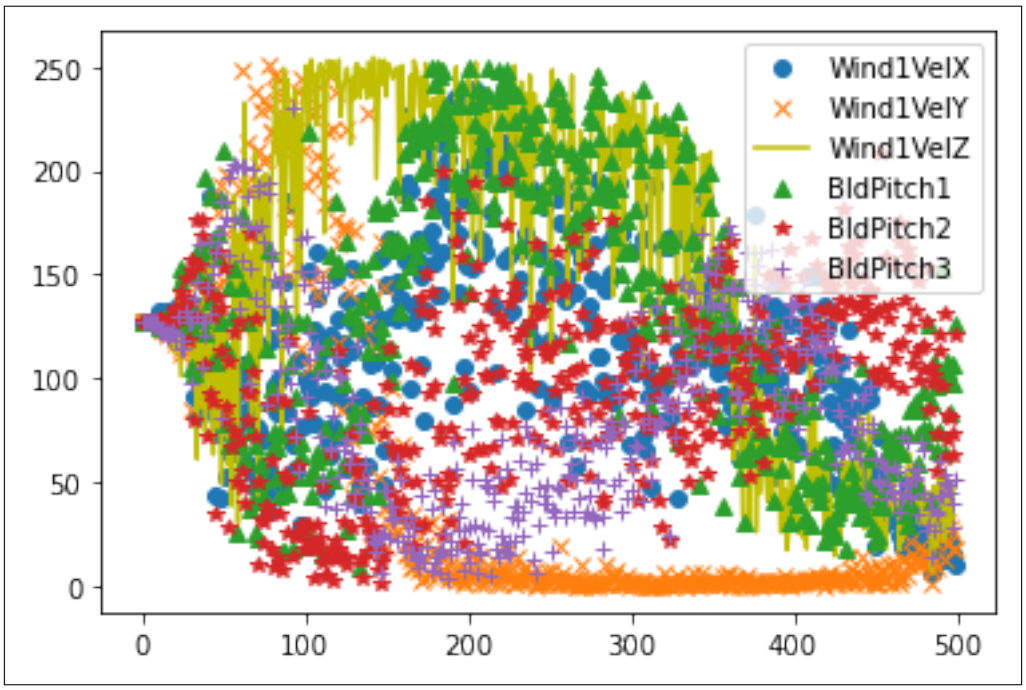

Figura 5.35: Distribución valores sintéticos fichero Test22Wind11-4pitchpulsetrend

En la figura 5.35 se observa la distribución de datos sintéticos aportados por el generador. Destaca la correlación entre los valores de coordenadas X y Z de la velocidad del viento y BldPitch1 y BldPitch2.

5.0.11. Fichero con ángulo del pitch 13

Real Discriminador	Sintético Discriminador	Discriminador	Generador
0.0	0.0	0.0	17.362
0.0	0.0	0.0	17.31
0.0	0.0	0.0	17.009
0.0	0.0	0.0	16.879
0.0	0.0	0.0	16.775

Con estos resultados obtenidos en la tabla 5.0.11, se extrae que el discriminador está correctamente entrenado para difenciar ambos lotes de imágenes pero el generador obtiene un valor de error demasiado alto en comparación con el resto de ficheros analizados hasta ahora.

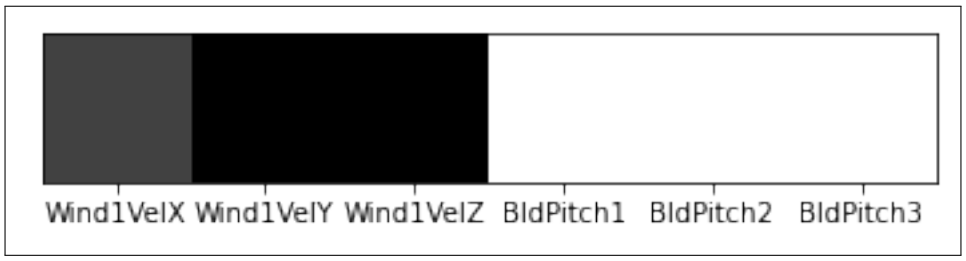

Figura 5.36: Resultado fichero Test22Wind13pitchpulsetrend

La figura 5.36 muestra la imagen generada por la red al final de las iteraciones donde se aprecia un resultado poco coherente con respecto a los valores de error.

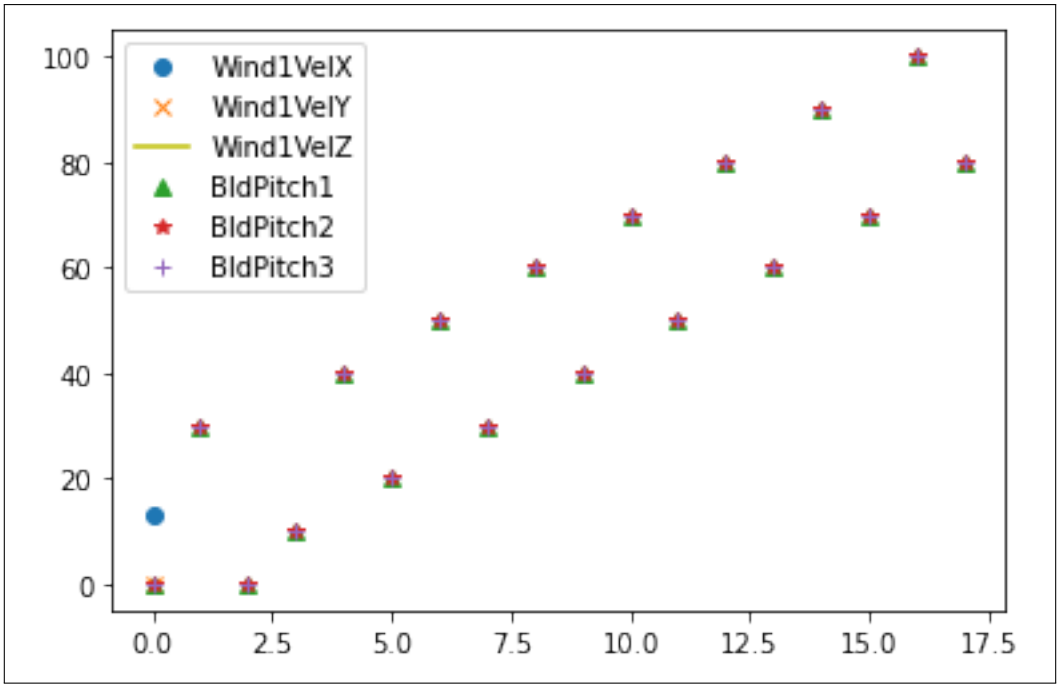

Figura 5.37: Distribución valores reales fichero Test22Wind13pitchpulsetrend

La figura 5.37 muestra la distribución de datos reales del fichero suministrado.

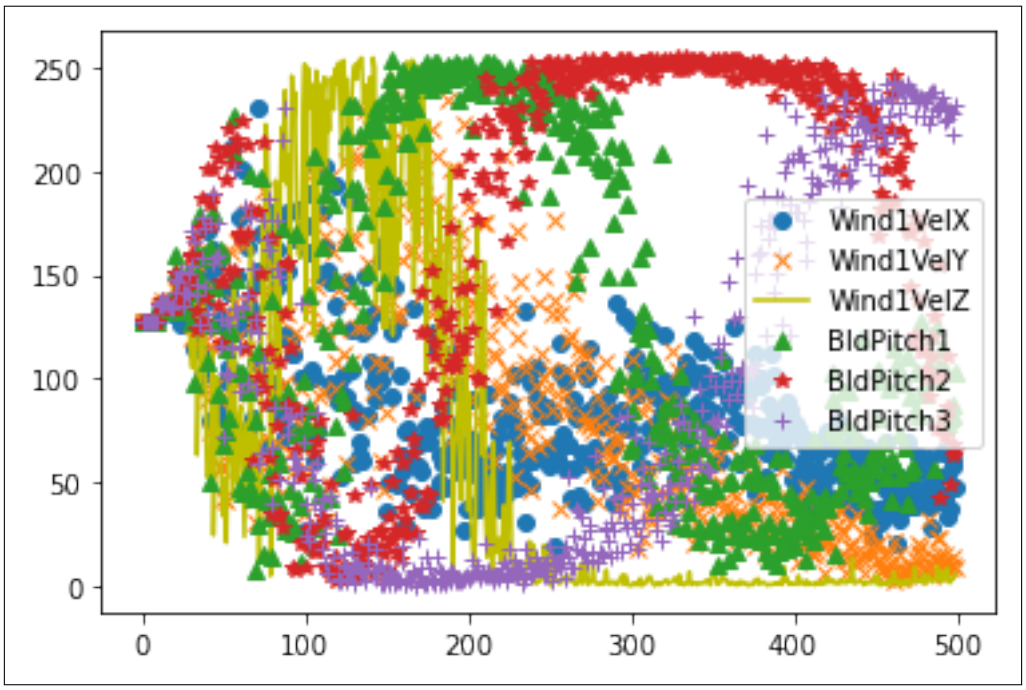

Figura 5.38: Distribución valores sintéticos fichero Test22Wind13pitchpulsetrend

En la figura 5.38 muestra la disposición da datos sintéticos generados con este fichero. A destacar, la correlación de los valores de Wind1VelX y Wind1VelZ. También, la correlación entre BldPitch1 y BldPitch2.

5.0.12. Fichero con ángulo del pitch 17

Real Discriminador	Sintético Discriminador	Discriminador	Generador
0.0	0.0	0.0	4.705
0.0	0.0	0.0	5.083
0.0	0.0	0.0	4.685
0.0	0.0	0.0	4.606
0.0	0.0	0.0	4.158

Los resultados de este fichero (tabla 5.0.12) muestran un discriminador perfectamente entrenado para distinguir entre imágenes reales y sintéticas. Con respecto al generador, los valores de error mejoran sustancialmente el caso anterior.

Figura 5.39: Resultado fichero Test22Wind17pitchpulsetrend

La figura 5.39 enseña la imagen producida por la red, esta vez con un resultado de mayor coherencia con respecto al caso anterior. El valor de error se genera en la primera casilla, la coordenada X del viento que la red no llega a dilucidar correctamente aunque se aproxima.

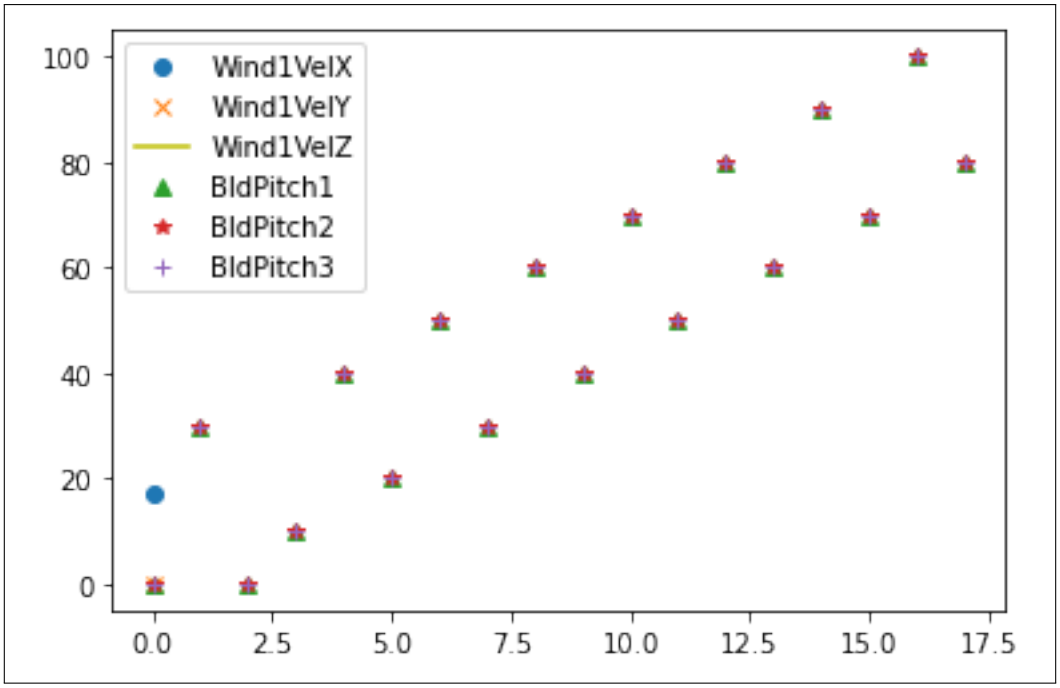

Figura 5.40: Distribución valores reales fichero Test22Wind17pitchpulsetrend

La figura 5.40 muestra la distribución de datos reales del fichero suministrado.

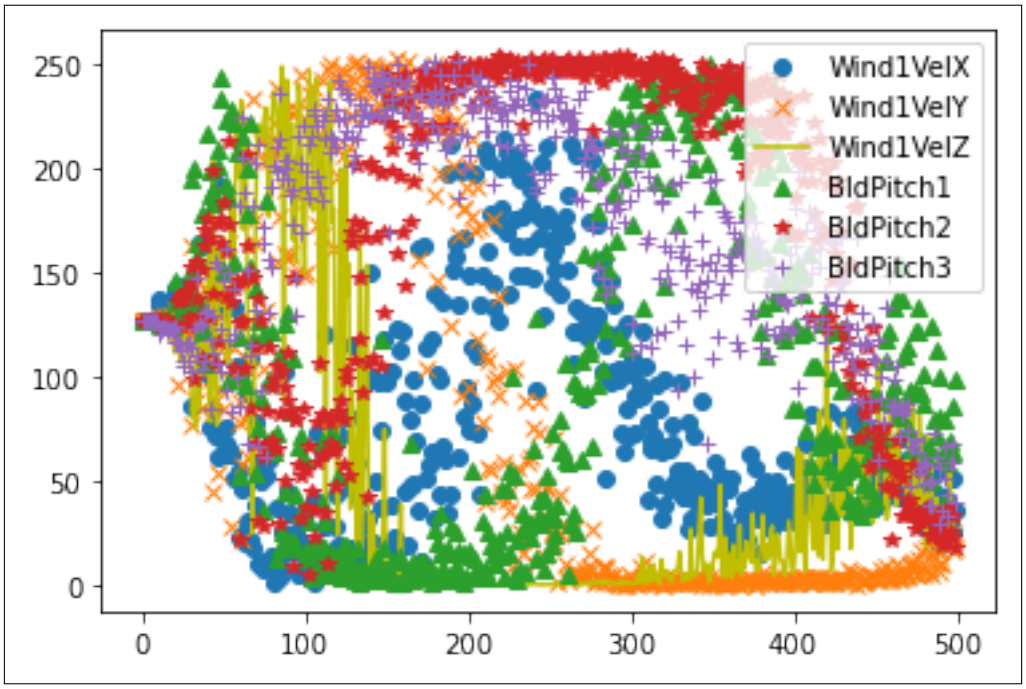

Figura 5.41: Distribución valores sintéticos fichero Test22Wind17pitchpulsetrend

En la figura 5.41 muestra la correlación de valores sintéticos entre las columnas Wind1VelX, Wind1VelY y Wind1VelZ. También se aprecia entre

las columnas de los ángulos de palas de la turbina (BldPitch1, BldPitch2, BldPitch3).

5.0.13. Fichero con ángulo del pitch 20

Real Discriminador	Sintético Discriminador	Discriminador	Generador
0.0	0.0	0.0	2.995
0.0	0.0	0.0	2.644
0.0	0.0	0.0	3.012
0.0	0.0	0.0	3.009
0.0	0.0	0.0	2.995

Los resultados de este fichero muestran un discriminador perfectamente entrenado para distinguir entre imágenes reales y sintéticas. El generador esta vez obtiene unos valores bajísimos siendo un buen indicativo de haber sido entrenado de manera eficaz. Valores en la tabla 5.0.13

Figura 5.42: Resultado fichero Test22Wind20pitchpulsetrend

La figura 5.42 es la imagen generada por la red y se aprecia como las tres primeras celdas tienen un color muy parecido, indicativo de tener unos valores correctos.

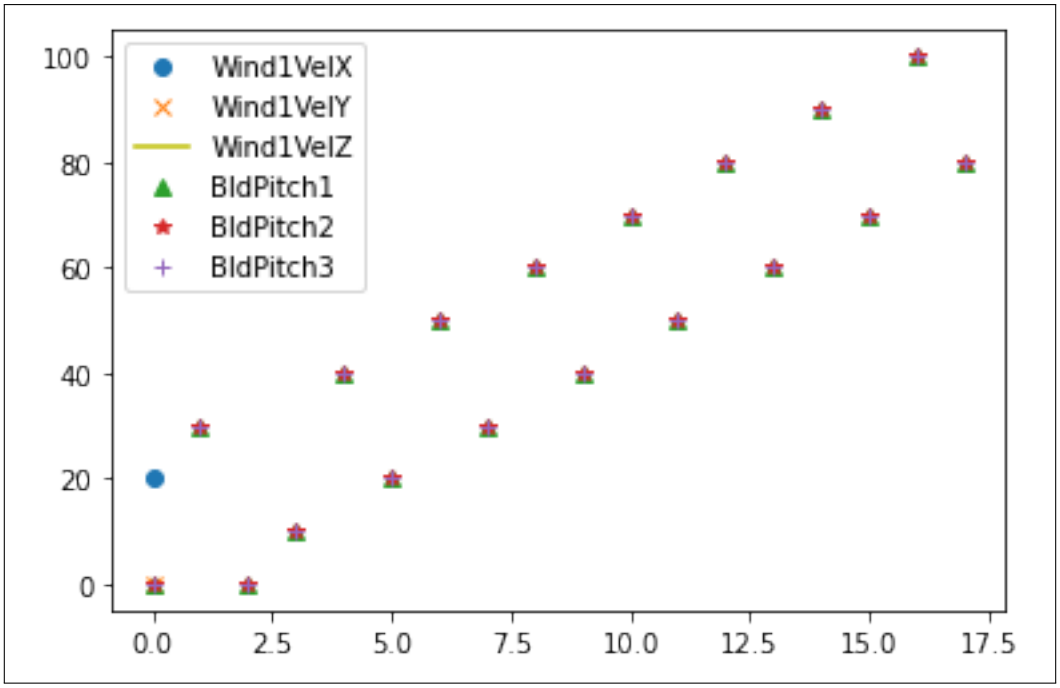

Figura 5.43: Distribución valores reales fichero Test22Wind20pitchpulsetrend

La figura 5.43 muestra la distribución de datos reales del fichero suministrado.

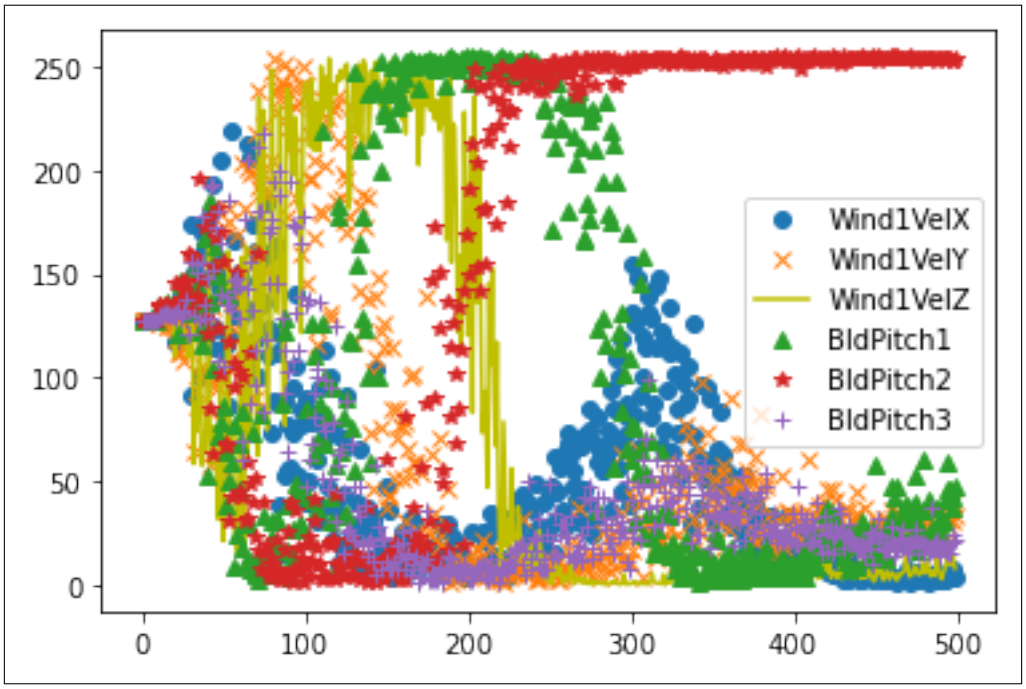

Figura 5.44: Distribución valores sintéticos fichero Test22Wind20pitchpulsetrend

En la figura 5.44 muestra una distribución de valores sintéticos generados por el generador al ser entrenado con este fichero. Sobresale de forma

clara una correlación entre las columnas BldPitch1 y BldPitch2. Además se extrapola otra similitud de patrón entre Wind1VelX y Wind1VelY.

5.0.14. Fichero con ángulo del pitch 21

Real Discriminador	Sintético Discriminador	Discriminador	Generador
0.0	0.0	0.0	5.236
0.0	0.0	0.0	1.902
0.0	0.0	0.0	7.26
0.0	0.0	0.0	16.139
0.0	0.0	0.0	8.48

Los resultados de este fichero (tabla 5.0.14) muestran un discriminador perfectamente entrenado para distinguir entre imágenes reales y sintéticas. Sin embargo, el generador muestra unos valores de error muy dispares, sin aparente relación entre sí en cada iteración, posiblemente provocado por incongruencias de datos en el fichero.

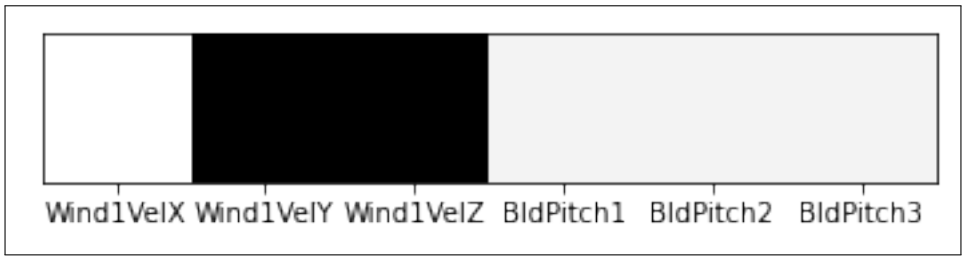

Figura 5.45: Resultado fichero Test22Wind21pitchpulsetrend

La figura 5.45 denota perfectamente los valores *anárquicos* obtenidos como pérdida. La red es incapaz de definir un patron de comportamiento.

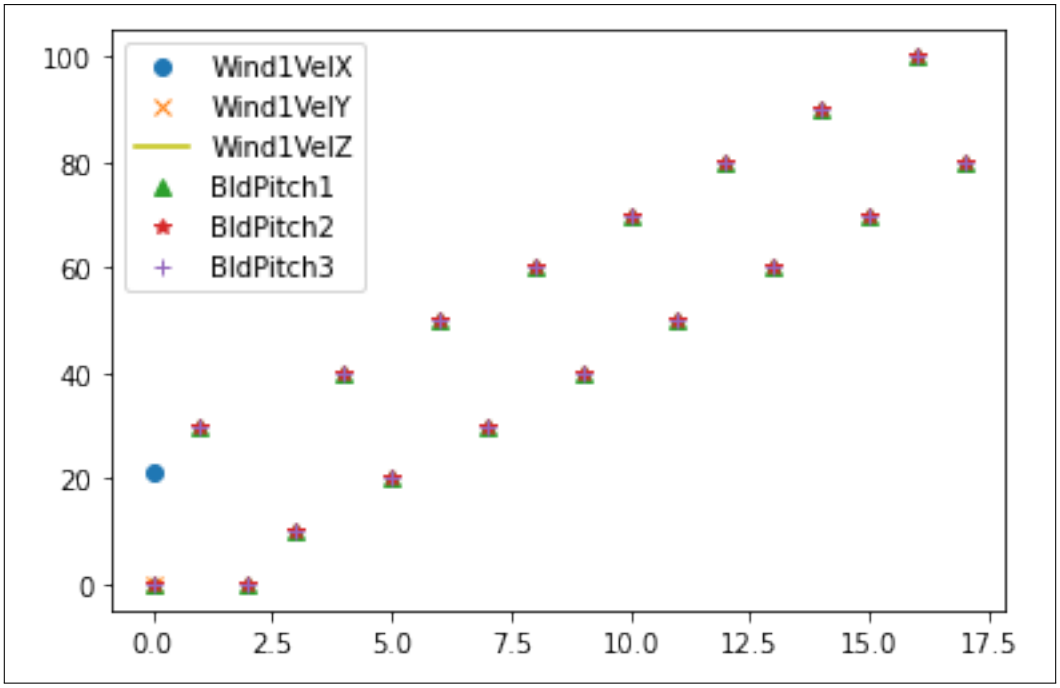

Figura 5.46: Distribución valores reales fichero Test22Wind21pitchpulsetrend

La figura 5.46 muestra la distribución de datos reales del fichero suministrado.

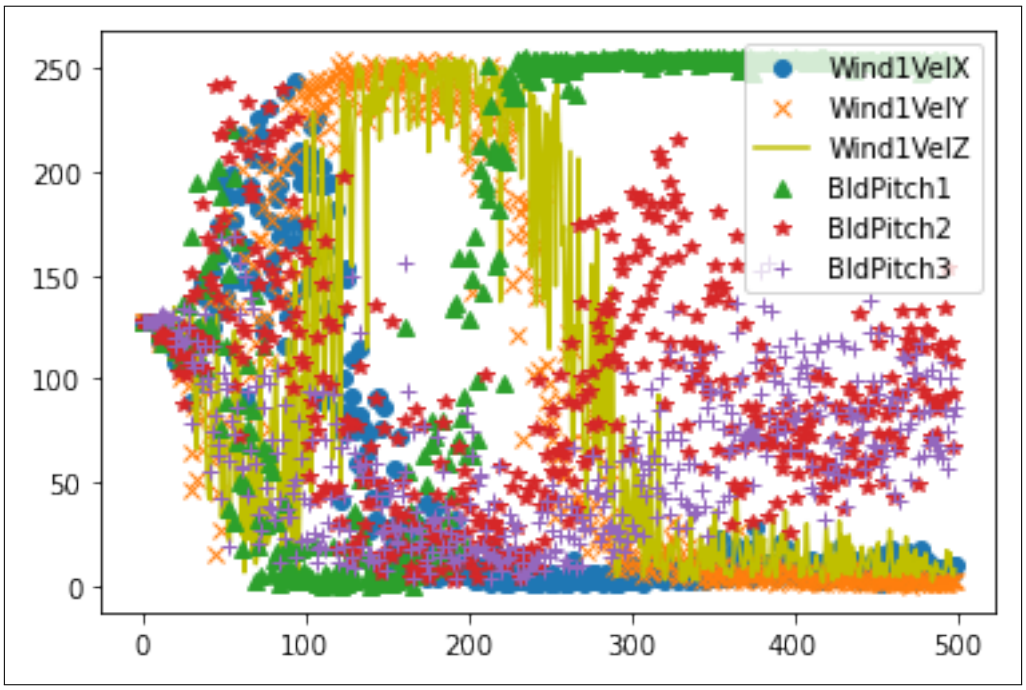

Figura 5.47: Distribución valores sintéticos fichero Test22Wind21pitchpulsetrend

La figura 5.47 corresponde a una correlación de valores sintéticos del generador. Cabe destacar la similtud de patrón entre las columnas Wind1VelX y Wind1VelY, además de BldPitch2 y BldPitch3.

5.0.15. Fichero con muestreo aleatorio 1

Real Discriminador	Sintético Discriminador	Discriminador	Generador
0.0	0.0	0.0	0.202
0.0	0.0	0.0	0.3
0.0	0.0	0.0	0.266
0.0	0.0	0.0	0.167
0.0	0.0	0.0	0.229

Los resultados de este fichero muestran un discriminador perfectamente entrenado para distinguir entre imágenes reales y sintéticas. Con respecto al generador, la red ha sido entrenada casi a la perfección y reporta un valor de error mínimo. Valores en la tabla 5.0.15

Figura 5.48: Resultado fichero Test22Windrandompitchrandom

La figura 5.48 muestra la imagen generada por la red cuando intenta imitar los valores del fichero aleatorio. La imagen está perfectamente definida ya que el generador aprendió bien los valores a generar.

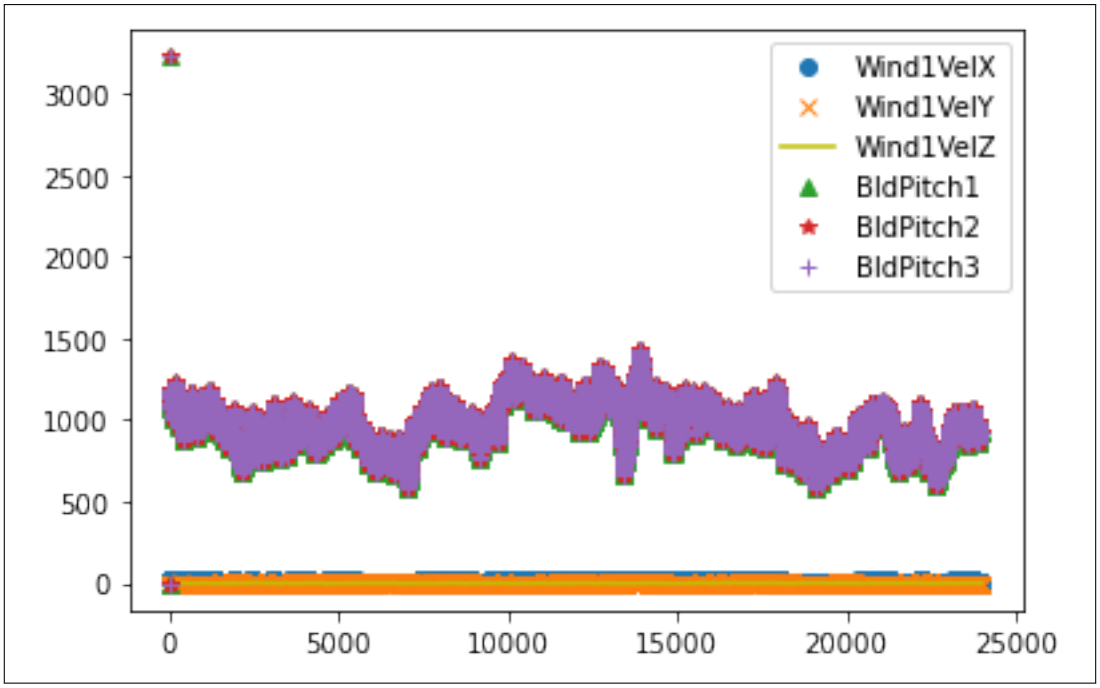

Figura 5.49: Distribución valores reales fichero Test22Windrandompitchrandom

La figura 5.49 muestra la distribución de datos reales del fichero suministrado.

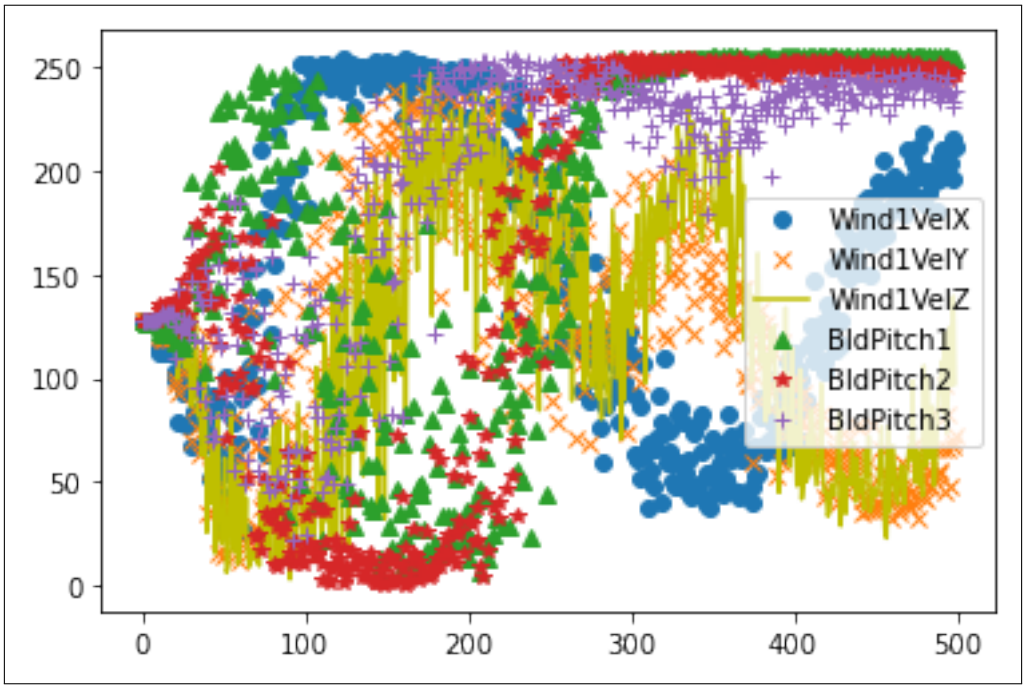

Figura 5.50: Distribución valores sintéticos fichero Test22Windrandompitchrandom

La figura 5.50 muestra la distribución de valores sintéticos generados para este fichero. Se observan correlaciones entre las 3 columnas de ángulos

de velocidad del viento. Además resaltan similitudes en las trazas de BldPitch1 y BldPitch2.

5.0.16. Fichero con muestreo aleatorio 2

Real Discriminador	Sintético Discriminador	Discriminador	Generador
0.0	0.0	0.0	9.05
0.0	0.0	0.0	9.431
0.0	0.0	0.0	9.485
0.0	0.0	0.0	9.755
0.0	0.0	0.0	10.121

Los resultados de este fichero (tabla 5.0.16) muestran un discriminador perfectamente entrenado para distinguir entre imágenes reales y sintéticas. Con respecto al generador, el valor de error es muy distinto con respecto a los valores del fichero anterior también generado por semillas aleatorias. En este caso el error se mueve en un intervalo de 9.

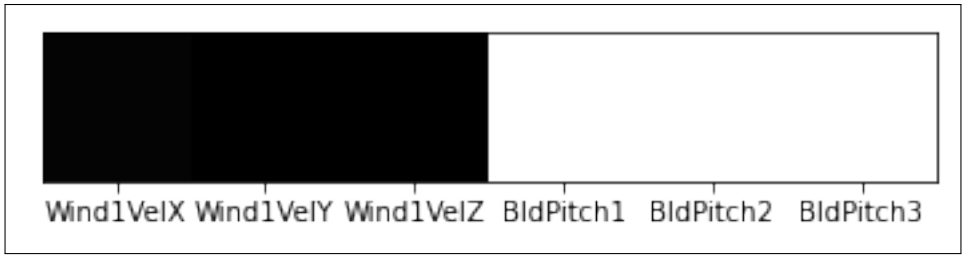

Figura 5.51: Resultado fichero Test22Windrandompitchrandom2

La figura 5.51 muestra la imagen generada por la red con el otro fichero aleatorio. La imagen es muy similar a la generada en el apartado anterior.

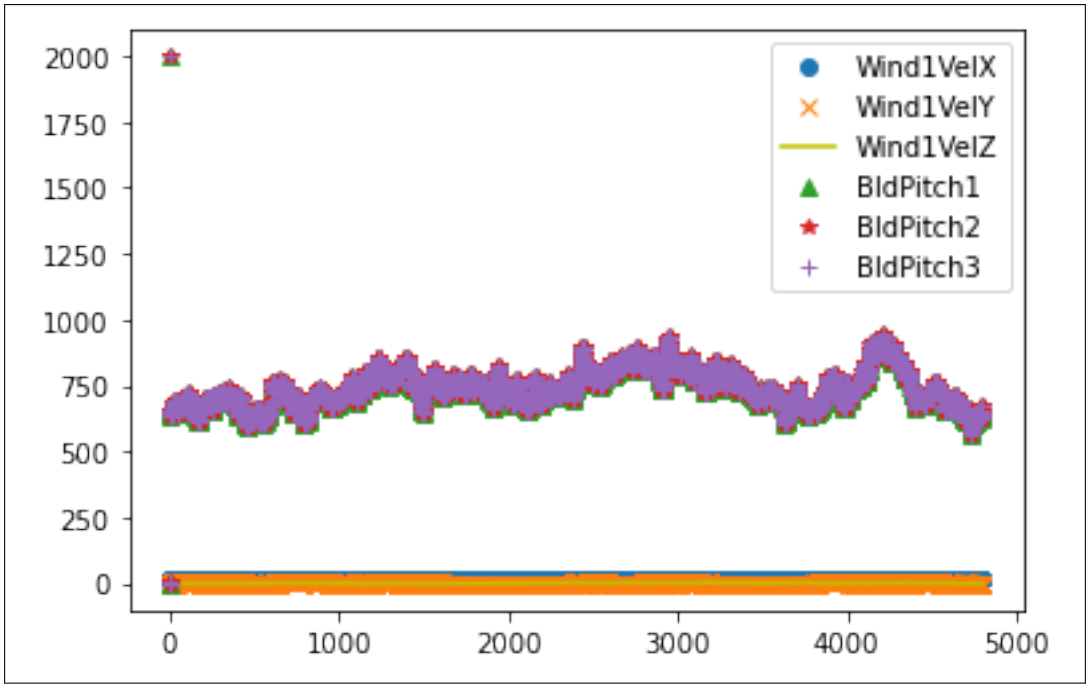

Figura 5.52: Distribución valores reales fichero Test22Windrandompitchrandom2

La figura 5.52 muestra la distribución de datos reales del fichero suministrado.

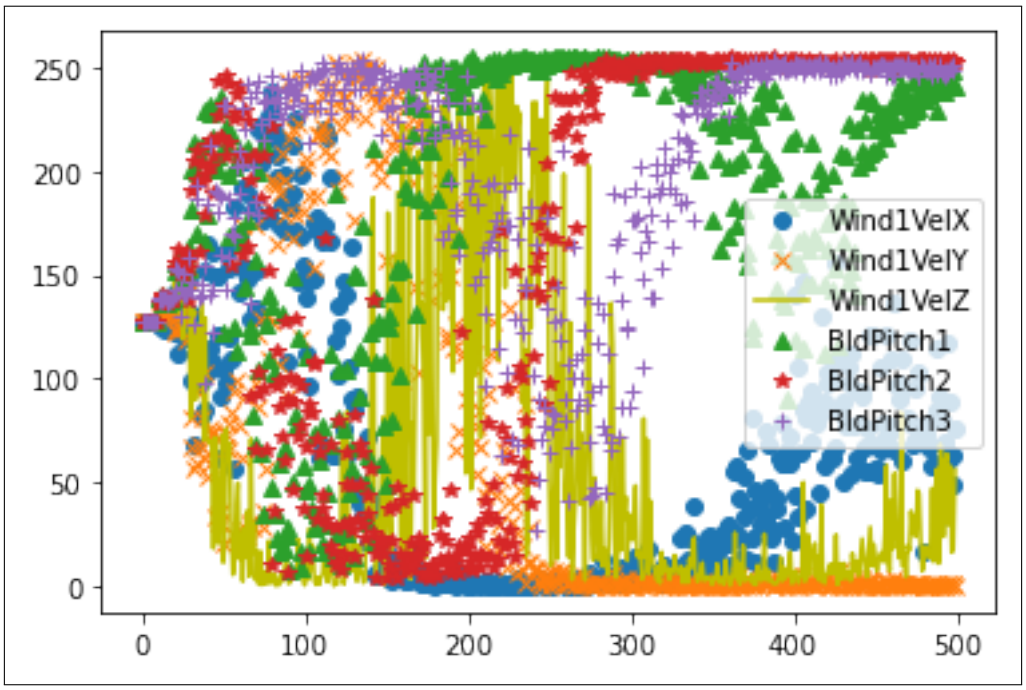

Figura 5.53: Distribución valores sintéticos fichero Test22Windrandompitchrandom2

La figura 5.53 muestra la distribución de valores sintéticos. Destacan correlaciones de valores entre las columnas Wind1VelX y Wind1VelY, y entre las columnas de ángulos de palas de turbinas.

Capítulo 6

Conclusiones y Trabajo futuro

6.1. Conclusiones

Una vez mostrados los resultados de todos y cada uno de los ficheros con los que se inició el proyecto, se extraen las siguientes conclusiones:

- La red es capaz de generar imágenes unidimensionales a partir de un fichero dado, adopta los valores de cada fila formando una imagen visible y después, con los valores de esa imagen es capaz de generar otra imagen similar.
- Los ficheros que muestran distintas velocidades del viento, es decir, los primeros mostrados, indican un patrón de valor de pérdida ascendente según se iba aumentando la velocidad, aunque cabe recalcar el fichero con velocidad 40, donde en la última iteración el resultado es distinto, probablemente debido a ruido en los datos del fichero.
- Sobre el segundo tipo de ficheros mostrado, en función del ángulo de la pala, los resultados no permiten extrapolar un patrón ascendente o descendente como en el caso anterior, aunque los resultados son buenos en su mayoría.
- Por último, en el caso de los ficheros generados a partir de una semilla aleatoria, los resultados son muy dispares. Con el primer fichero se obtienen valores de pérdida ínfimos, suceso que podría conllevar un error de interpretación si no se observase el segundo fichero, ya que éste genera unos valores muy altos. El razonamiento lógico de esta secuencia induce a un comportamiento extraño de la red, donde no es capaz de asimilar la información que contiene y produce imágenes aleatorias, a veces cercanas a una imagen real y en otras ocasiones

muy alejada de la misma.

En definitiva, el proyecto ha tenido un enfoque novedoso, la arquitectura y diseño de la red se ha adaptado para trabajar con datos unidimensionales. Estos valores corresponden a magnitudes físicas, con lo cual, al no tratarse de valores de color como pueden ser los píxeles, la relación entre variables es diferente. Este factor facilitaba la impredecibilidad del comportamiento de la red. Tras este trabajo, se puede afirmar el potencial de la idea inicial, la red aprende a generar datos sintéticos.

6.2. Trabajo futuro

Una vez obtenidos los resultados de este proyecto se plantean varias posibilidades de desarrollos futuros.

Los datos artificiales generados podrían ser empleados en varios modelos a desarrollar que adoptaran como entrada los campos resultantes para, con ellos, generar valores de velocidad angular de rotor y potencia del generador para ajustar los parámetros de arranque de una turbina eólica.

Por otro lado, podrían aplicarse técnicas de limpieza y mejora de la calidad del dato para que la red los recibiera con una mayor precisión, así como eliminar aquellos valores que puedan ser clasificadas como anómalas o ruido.

Además, sería interesante entrenar la red con más de una fila por imagen, puesto que mejoraría la cantidad de información a manejar por la red durante el aprendizaje aunque supusiera un mayor tiempo de ejecución. Inclusive, la red podría aprender a generar datos sintéticos con una consistencia temporal que en este proyecto desarrollado no ha sido capaz.

Capítulo 7

Conclusions and Future Work

7.1. Conclusions

Having shown the results of each and every one of the files with which the project began, the following conclusions can be drawn:

- The network is able to generate one-dimensional images from a given file, it adopts the values of each row forming a visible image and then, with the values of that image, it is able to generate another similar image. Of the 3 types of files available, depending on wind speed, blade angle and random values, the best percentage of loss or error obtained by the network is in the range of 2-3. Although values between 0 and 1 have been shown, these have been obtained without any correlation, i.e. either in cases where the file has been obtaining disparate values (very high or low) or in the random value files where the network has not been able to find a defined pattern and has generated random values.
- The files that show different wind speeds, i.e. the first ones shown, indicate a pattern of increasing loss values as the speed increases, although it is worth highlighting the file with speed 40, where in the last iteration the result is different, probably due to noise in the data of the file.
- Regarding the second type of files shown, depending on the angle of the blade, the results do not allow extrapolating an ascending or descending pattern as in the previous case, although the results are mostly good.

- Finally, in the case of the files generated from a random seed, the results are very different. With the first file, very low loss values are obtained, an event that could lead to an error of interpretation if the second file were not observed, since it generates very high values. The logical reasoning of this sequence leads to a strange behaviour of the network, where it is not able to assimilate the information it contains and produces random images, sometimes close to a real image and sometimes very far from it.

In fact, the project has had a novel approach, the architecture and design of the network has been adapted to work with one-dimensional data. These values correspond to physical magnitudes, which means that, as they are not colour values such as pixels, the relationship between variables is different. This factor facilitated the unpredictability of the behaviour of the network. After this work, the potential of the initial idea can be affirmed: the network learns to generate synthetic data.

7.2. Future Work

Following the results of this project, several possibilities for future developments are being considered.

The artificial data generated could be used in various models to be developed that adopt the resulting fields as input to generate rotor angular velocity and generator power values to adjust the start-up parameters of a wind turbine.

On the other hand, data cleaning and data quality improvement techniques could be applied to improve the accuracy of the data received by the network, as well as to eliminate those values that could be classified as outliers or noise.

Furthermore, it would be interesting to train the network with more than one row per image, since it would improve the amount of information to be handled by the network during learning, even though it would mean a longer execution time. Furthermore, the network could learn to generate synthetic data with a temporal consistency that it has not been able to do in this project.

Índice de figuras

1.1. Parque Eólico de Gecama en Cuenca 2
1.2. Imagen Real vs Imagen generada por GAN 3

2.1. Parque Eólico de Gecama en Cuenca 2
2.2. Imagen Real vs Imagen generada por GAN 3

3.1. Visión interior de una turbina 7
3.2. Esquema básico del funcionamiento de una red GAN . . . 8
3.3. Estructura de una red neuronal 10
3.4. Entrenamiento de una red GAN 11

4.1. Contenido fichero Test22pulsetrendWindpitch0 15
4.2. Gráfica valores Test22pulsetrendWindpitch0 16
4.3. Contenido fichero Test22Wind3pitchpulsetrend 17
4.4. Gráfica Valores Test22Wind3pitchpulsetrend 17
4.5. Contenido fichero Test22Windrandompitchrandom 18
4.6. Gráfica Valores Test22Windrandompitchrandom 19
4.7. Resumen por capas del generador 21
4.8. Resumen por capas del discriminador 23
4.9. Resumen por capas del generador 25

5.1. Resultado fichero Test22pulsetrendWindpitch0 para 5 iteraciones 28
5.2. Resultado fichero Test22pulsetrendWindpitch0 para 10 iteraciones 29
5.3. Resultado fichero Test22pulsetrendWindpitch0 para 20 iteraciones 30
5.4. Resultado fichero Test22pulsetrendWindpitch0 para 50 iteraciones 30
5.5. Distribución valores reales fichero Test22pulsetrendWindpitch0 31
5.6. Distribución valores sintéticos fichero Test22pulsetrendWindpitch0 31

5.7. Resultado fichero Test22pulsetrendWindpitch0 para 100 iteraciones 32
5.8. Resultado fichero Test22pulsetrendWindpitch20 Caso 1 . . 33
5.9. Resultado fichero Test22pulsetrendWindpitch20 Caso 2 . . 33
5.10. Distribución valores reales fichero Test22pulsetrendWindpitch20 34
5.11. Distribución valores sintéticos fichero Test22pulsetrendWindpitch20 34
5.12. Resultado fichero Test22pulsetrendWindpitch40 35
5.13. Distribución valores reales fichero Test22pulsetrendWindpitch40 36
5.14. Distribución valores sintéticos fichero Test22pulsetrendWindpitch40 36
5.15. Resultado fichero Test22pulsetrendWindpitch60 37
5.16. Distribución valores reales fichero Test22pulsetrendWindpitch60 38
5.17. Distribución valores sintéticos fichero Test22pulsetrendWindpitch60 38
5.18. Resultado fichero Test22pulsetrendWindpitch80 39
5.19. Distribución valores reales fichero Test22pulsetrendWindpitch80 40
5.20. Distribución valores sintéticos fichero Test22pulsetrendWindpitch80 40
5.21. Resultado fichero Test22Wind3pitchpulsetrend 41
5.22. Distribución valores reales fichero Test22Wind3pitchpulsetrend 42
5.23. Distribución valores sintéticos fichero Test22Wind3pitchpulsetrend 42
5.24. Resultado fichero Test22Wind5pitchpulsetrend 43
5.25. Distribución valores reales fichero Test22Wind5pitchpulsetrend 44
5.26. Distribución valores sintéticos fichero Test22Wind5pitchpulsetrend 44
5.27. Resultado fichero Test22Wind8pitchpulsetrend 45
5.28. Distribución valores reales fichero Test22Wind8pitchpulsetrend 46
5.29. Distribución valores sintéticos fichero Test22Wind8pitchpulsetrend 46
5.30. Resultado fichero Test22Wind10pitchpulsetrend 47
5.31. Distribución valores reales fichero Test22Wind10pitchpulsetrend 48
5.32. Distribución valores sintéticos fichero Test22Wind10pitchpulsetrend 48
5.33. Resultado fichero Test22Wind11-4pitchpulsetrend 49
5.34. Distribución valores reales fichero Test22Wind11-4pitchpulsetrend 50
5.35. Distribución valores sintéticos fichero Test22Wind11-4pitchpulsetrend 50
5.36. Resultado fichero Test22Wind13pitchpulsetrend 51
5.37. Distribución valores reales fichero Test22Wind13pitchpulsetrend 52
5.38. Distribución valores sintéticos fichero Test22Wind13pitchpulsetrend 52
5.39. Resultado fichero Test22Wind17pitchpulsetrend 53
5.40. Distribución valores reales fichero Test22Wind17pitchpulsetrend 54
5.41. Distribución valores sintéticos fichero Test22Wind17pitchpulsetrend 54
5.42. Resultado fichero Test22Wind20pitchpulsetrend 55
5.43. Distribución valores reales fichero Test22Wind20pitchpulsetrend 56
5.44. Distribución valores sintéticos fichero Test22Wind20pitchpulsetrend 56

5.45. Resultado fichero Test22Wind21pitchpulsetrend 57
5.46. Distribución valores reales fichero Test22Wind21pitchpulsetrend 58
5.47. Distribución valores sintéticos fichero Test22Wind21pitchpulsetrend 58
5.48. Resultado fichero Test22Windrandompitchrandom 59
5.49. Distribución valores reales fichero Test22Windrandompitchrandom 60
5.50. Distribución valores sintéticos fichero Test22Windrandompitchrandom 60
5.51. Resultado fichero Test22Windrandompitchrandom2 61
5.52. Distribución valores reales fichero Test22Windrandompitchrandom2 62
5.53. Distribución valores sintéticos fichero Test22Windrandompitchrandom2 62

Índice de cuadros

Bibliografía

[1] "Parques eólicos." https://www.antala.es/parque-eolico-mas-grande-espana/.

[2] "Ventajas de turbinas eólicas flotantes." https://es.wikipedia.org/wiki/Turbina_e%C3%B3lica_flotante.

[3] "Introducción a las gans." https://es.wikipedia.org/wiki/Red_generativa_antag%C3%B3nica.

[4] "Aplicaciones red gan." https://www.genbeta.com/a-fondo/que-como-funcionan-gan-esas-redes-neuronales-capaces-crear-rostros-personas-qu

[5] "Contexto de turbinas eólicas flotantes." https://alcanzia.es/blog/turbinas-eolicas-flotantes-windfloat/.

[6] "Imagen comparativa resultados gan." https://www.iartificial.net/redes-neuronales-generativas-adversarias-gans/.

[7] "Imagen imagen gan básica." https://spj.sciencemag.org/journals/plantphenomics/2020/8309605/.

[8] "Redes neuronales generativas adversarias (gans)." https://www.iartificial.net/redes-neuronales-generativas-adversarias-gans/.

[9] "Imagen estructura red neuronal." https://es.wikipedia.org/wiki/Perceptr%C3%B3n_multicapa.

[10] "Entrenamiento de una red gan imagen." https://torres.ai/generative-adversarial-networks/.

[11] "Introducción a los optimizadores." https://datasmarts.net/es/que-es-un-optimizador-y-para-que-se-usa-en-deep-learning/.

[12] "Funciones de pérdida." https://rubialesalberto.medium.com/data-scientist-explicaci%C3%B3n-y-diferencias-entre-m%C3%A9tricas-de-error-con-python-f7d71dbee634.

[13] "Curso de udemy realizado para este trabajo." https://www.udemy.com/certificate/UC-6b8a2f87-3967-46de-adeb-71509cc5f0f9/.

[14] "Material de latex." https://ondiz.github.io/cursoLatex/Contenido/04.Imagenes.html.

[15] "Generative adversarial networks: Build your first models in python." https://realpython.com/generative-adversarial-networks/.

[16] "Exploratory data analysis." https://www.youtube.com/watch?v=E-q24gS8cqg.

[17] "Cómo crear una red generativa antagónica (gan) en python." https://anderfernandez.com/blog/como-crear-una-red-generativa-antagonica-gan-en-python/.

[18] "Deep convolutional generative adversarial network." https://www.tensorflow.org/tutorials/generative/dcgan?hl=en.

[19] "Gans en python." https://www.codificandobits.com/blog/redes-adversarias-explicacion-y-tutorial-python/.

[20] "Información energía eólica." https://www.bbva.com/es/sostenibilidad/que-es-la-energia-eolica-la-importancia-del-viento-como-renovable

[21] "Información energía eólica 2." https://www.enelgreenpower.com/es/learning-hub/energias-renovables/energia-eolica.

[22] "Introducción a la energía eólica." https://www.acciona.com/es/energias-renovables/energia-eolica/#:~:text=La%20energ%C3%ADa%20e%C3%B3lica%20es%20una,que%20causan%20el%20calentamiento%20global.

[23] "Tipos de turbinas." https://forococheselectricos.com/2021/10/energia-eolica-como-funciona-un-aerogenerador.html#:~:text=Una%20turbina%20e%C3%B3lica%20convierte%20la,un%20lado%20de%20esta%20disminuye.

[24] "Binary cross entropy loss." https://rubialesalberto.medium.com/funciones-de-error-con-entropia-cross-entropy-y-binary-cross-entropy-8df8442cdf

Víctor Ramos Fuentes
16/09/2022
Ult. actualización 16 de septiembre de 2022
LaTeX lic. LPPL & powered by TEFLON CC-ZERO