



POLITÉCNICA



**UNIVERSIDAD POLITÉCNICA DE MADRID**  
**ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA**  
**AERONÁUTICA Y DEL ESPACIO**  
**GRADO EN INGENIERÍA AEROESPACIAL**

**TRABAJO FIN DE GRADO**

**Título del Trabajo Fin de Grado**

**AUTOR: Nombre APELLIDOS**

**ESPECIALIDAD: Denominación de la especialidad**

**COTUTOR: Nombre APELLIDOS**

**TUTOR DEL TRABAJO: Nombre APELLIDOS**

**Febrero de 2020**



# Agradecimientos

TITULACIÓN: Graduado en Ingeniería Aeroespacial

DEPARTAMENTO: Materiales y Producción Aeroespacial

TIPO DE TFG: Especial

INTENSIFICACIONES A LAS QUE SE OFERTA: Todas

TÍTULO DEL TRABAJO: Detección y análisis de impactos sobre estructuras de material compuesto con sensores integrados

## **CONTENIDOS Y OBJETIVOS DEL TRABAJO:**

En el marco de las estructuras inteligentes, las estructuras con sensores integrados permiten la determinación de la existencia de daños o de eventos mediante el tratamiento de las señales de los sensores, en lo que se conoce como SHM (Structural Health Monitoring). En este marco, se va a trabajar con una red de sensores PZT integrados en una placa de material compuesto con y sin rigidizadores. Se analizarán los impactos realizados sobre la misma y se estudiará su respuesta para tratar de discriminar no solo la posición del impacto, sino también discriminar la energía del mismo. Adicionalmente se tratará de discriminar entre la masa y la velocidad del impacto y, en caso de que se produzcan daños en la misma (delaminaciones, roturas, etc) cuantificarlos y caracterizarlos.

Con este objetivo, se desarrollarán redes neuronales (ANN-Artificial Neural Networks) que serán entrenadas con resultados experimentales y teóricos. Las ANN son unas excelentes clasificadoras y analizadoras de tendencias, por lo que se propone su uso para el estudio de las señales debidas al impacto.

Este TFG presenta un carácter marcadamente interdisciplinar, ya que abarca desde las estructuras de material compuesto, adquisición de medida mediante sensores y tratamiento de datos.

# Índice general

# Índice de figuras

# Índice de tablas

## Capítulo 1

# Introducción y motivación

Tanto en el sector aeronáutico como espacial, las aeronaves y sistemas son diseñados siguiendo un método basado en admisibles y reglas de diseño muy conservativas. Con esto se busca que, bajo una larga lista de condiciones de diseño que varían dependiendo del tipo de aeronave, se garantice la integridad estructural y su correcto funcionamiento durante el transcurso de cada de misión.

Estos admisibles de diseño pueden ir desde defectos estructurales complejos de gran importancia en la seguridad del conjunto de la aeronave, como puede ser la pérdida de un motor, hasta defectos con una pequeña influencia estructural pero que, con el tiempo, pueden convertirse en catastróficos, sirviendo de ejemplo la rotura de un pequeño remache.

La pérdida de un motor produce defectos estructurales cuya influencia se manifiesta en toda la estructura, siendo así fácilmente detectable. Sin embargo, cuando un remache se rompe su influencia es local y de pequeña magnitud.

Para poder detectar o cuantificar el tamaño de un defecto en una estructura se comparan magnitudes físicas de la estructura bajo estudio con un estado de la misma estructura sin defectos. La forma más sencilla de hacer esta comparación es realizando una inspección visual, pero este procedimiento solo permite detectar defectos que han producido cambios grandes y evidentes sobre la estructura, quedando ocultos aquellos que no se detectan a simple vista.

Al ser necesarias inspecciones más exhaustivas para detectar defectos se realizan mantenimientos preventivos. Durante los mantenimientos se utilizan técnicas de inspección no destructivas (NDI) con las que se recogen datos que se procesan para asegurar la integridad estructural de la aeronave.

Sin embargo, durante los últimos años, y como una evolución de las técnicas NDI, ha habido un gran desarrollo de sistemas cuya función es evaluar el estado de las estructuras de una aeronave o vehículo espacial durante la operación y a tiempo real mediante el uso sensores integrados en las propias estructuras. Estos son los llamados *Structural Health Monitoring Systems* (SHMS).

La utilización de estos sistemas tiene consecuencias muy importantes. Permiten la monitorización de las estructuras sin la intervención de inspectores y sin tener que desensamblarla para analizarla. Esto hace que el tiempo entre la detección de un defecto su reparación sea muy corto, haciendo que este no aumente y evitando que termine provocando un fallo catastrófico. Entonces, si el SHMS no detecta ningún tipo de defecto, se llega a la conclusión de que la aeronave no necesita ser revisada en profundidad con tanta

frecuencia y los periodos de mantenimiento pueden ser separados en el tiempo y, mientras tanto, la aeronave sigue operando cumpliendo con los admisibles de diseño.

Por otra parte, la tendencia del sector aeroespacial desde hace varias décadas ha sido aumentar el porcentaje en peso del avión fabricado con materiales compuestos. Los materiales compuestos tienen unas propiedades específicas muy superiores a metales como el aluminio, por otro lado, presentan modos de fallo más complejos, variados y más difíciles de detectar comparados a los que sufren los materiales metálicos.

El aumento del uso de materiales compuestos, junto con el potencial de los SHMS ha sido el impulsor de este Trabajo Fin de Grado. El objetivo es desarrollar herramientas fiables basadas en algoritmos de Inteligencia Artificial (IA) para poder detectar defectos en estructuras aeronáuticas, más concretamente, en estructuras aeronáuticas con geometría compleja fabricadas con material compuesto.

Grandes empresas como Google, Facebook, Microsoft o Nvidia han invertido mucho esfuerzo en el campo de la IA durante los últimos años y han conseguido sorprendentes resultados, por ejemplo, el NVidia RTX Voice, un cancelador de ruido basado en algoritmos de Deep Learning (DL).

Siguiendo esta línea de investigación en DL, se va a explorar la posibilidad de utilizar este tipo de algoritmos para desarrollar dos herramientas de SHM. La primera tendrá que ser capaz de clasificar diferentes estados de una estructura basándose en medidas de deformación. A su vez, se pretende detectar el nivel de carga al que está sometida la estructura y a la temperatura que se encuentra.

Por otra parte, se desarrollará una segunda herramienta para localizar impactos y cuantificar la energía de estos en una costilla del A380. Esta herramienta se alimentará con la onda mecánica producida por el objeto impactador.

El lado negativo del uso de IA para SHM es que se requiere un conjunto de datos (Data Set, DS) muy grande para poder entrenar los algoritmos y conseguir un nivel de confianza elevado en los resultados. Usando de ejemplo la herramienta que detectará impactos y su energía, se va a calcular el orden de magnitud de los impactos que son necesarios para tener una cantidad de datos aceptable.

Dividiendo la placa en una malla de 10 x 10, seleccionando 10 niveles de energía diferentes y 100 repeticiones de cada impacto llegamos a tener una cantidad mínima de  $10^5$  impactos, lo cual es inviable para realizar de forma manual y rigurosa, por lo que su automatización es necesaria.

Para esta automatización se va a proponer el diseño de un Impactador por gravedad de control numérico. Con esta máquina se realizarán todos los impactos que sean necesarios de forma automática, a la vez que almacenará y preprocesará los datos para alimentar la herramienta de DL.

# Capítulo 2

## Estado del arte

En este capítulo se va a hacer una introducción teórica a los pilares en los que se asienta este trabajo, SHM e IA, y sus aplicaciones en la industria aeroespacial..

### 2.1. Structural Health Monitoring

Structural Helath Monitoring (SHM) es el proceso de identificar daños en estructuras de forma no destructiva mediante el uso de sensores integrados [SHM·Aero·3]. Por lo tanto, es necesario tener el concepto de daño bien definido. Se puede definir un daño como los cambios en el material y/o en las propiedades geométricas de un sistema estructural, incluyendo cambios en las condiciones de contorno y conectividad del mismo, que pueden afectar de manera adversa al funcionamiento presente y futuro del sistema [dam].

La gran ventaja de este proceso frente a las NDI de los que evoluciona es que se puede aplicar en tiempo real, incluyendo durante la operación del vehículo. Con esto se consigue, no solo tener un conocimiento total de la salud estructural, sino que también se puede llegar a monitorizar la magnitud de las cargas a las que las estructuras instrumentadas están sometidas.

El SHM se realiza en cuatro pasos principales:

1. **Evacuación operacional:** para aplicar la técnica de SHM, la evaluación operativa es el paso fundamental. Estudia los problemas y sus consecuencias que conducen a monitorizar las estructuras. La evaluación estructural trata de buscar solución a varias preguntas, tales como:
  - ¿qué tipo de daño se busca en la estructura monitorizada?
  - ¿a qué condiciones ambientales y operativas está sometido la estructura monitorizada mientras se lleva a cabo el proceso de SHM?
  - ¿Cuáles son las restricciones de la transferencia de datos a través del sistema SHM a causa de la operación del vehículo?
  - ¿cuáles son las ventajas del SHM en la vida segura del vehículo?
2. **Adquisición de datos:** el proceso de adquisición de datos es una parte crucial en SHM y se realiza a través de sensores integrados en la estructura. El tipo de sensores que se utilizarán está ligado a la técnica de SHM usada. En esta fase se estudia el número de sensores usados, su distribución en la estructura y el hardware que almacenará esta información.

3. **Procesado de señales y extracción de características debidas al daño:** la parte más crítica del SHM es extraer las inferencias a partir de las señales recogidas por los sensores. Este proceso se basa en descubrir qué propiedades y áreas son afectadas por el daño junto con como varían a medida que el daño evoluciona.
4. **Modelado estadístico para evaluación de las características:** el desarrollo de un modelo estadístico es el paso final del SHM. Este proceso incluye dos categorías:
  - Cuando los datos recopilados pertenecen tanto a partes dañadas como a no dañadas, el modelado estadístico hace una clasificación general también conocida como aprendizaje supervisado, como puede ser el análisis de regresión y clasificación grupal.
  - Cuando los datos proceden únicamente de estructuras dañadas, se habla de aprendizaje no supervisado.

Desde un punto de vista matemático, los sistemas SHM se pueden considerar como sistemas de control realimentado, diferenciándose de éstos en los tiempos característicos y el modo de actuación sobre la planta.

Las escalas temporales dependen de lo que el sistema busca controlar. En el caso de una grieta, los tiempos característicos son del orden de la vida operativa de la planta, en cambio, si se quiere un sistema de localización y detección de impactos, el tiempo característico se reduce a decenas de ms.

También cambian las características espaciales, dependiendo fundamentalmente de la criticidad del daño. La longitud característica será grande cuando la tolerancia al daño permita tamaños considerables, sin embargo, en las zonas donde ésta tolerancia sera pequeña, la longitud característica también lo tendrá que ser [Jaime·Tesis].

Para funcionar, un sistema SHM medirá todas las entradas y salidas que afectan a la planta monitorizada, entendiéndose como planta aquella cuya función sea soportar o transmitir cargas, siendo fija o móvil. Los parámetros de observación serían los parámetros característicos de operación propios y los relativos al entorno. La situación anómala será aquella que comprometa el funcionamiento presente y futuro de la planta, en su conjunto, dentro de las condiciones de diseño [civil].

Así pues, tenemos que el SHM es en esencia un sistema autónomo para inspeccionar y detectar daños en estructuras con una mínima intervención humana. En la Figura ?? se puede ver representada esta idea [Jaime·Tesis].

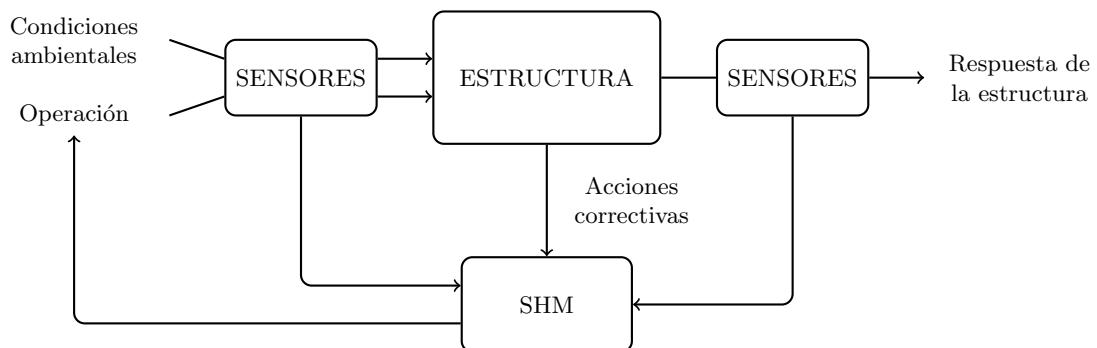


Figura 2.1: SHM como sistema realimentado

## **2.2. Monitorización de estructuras aeronáuticas - SHM 5 pags (SHM a día de hoy – futuros desarrollos)**

Dentro de la gran variedad de estructuras en las que se puede aplicar el SHM, este trabajo se va a centrar en estructuras aeronáuticas fabricadas con material compuesto. Como ya se ha introducido en el primer paso del SHM, *Evaluación operacional*, hay una serie de preguntas que se necesitan responder para elegir el sistema óptimo para monitorizar una estructura determinada dentro de todas las que componen una aeronave.

Será necesario dar respuesta a las siguientes cuestiones:

1. ¿Qué estructura se quiere monitorizar?
2. ¿Qué tipo de daño afecta a la integridad de dicha estructura?
3. ¿Cuál va a ser la característica sensitiva al daño, o Damage Sensitive Feature (DSF), la que se va a usar? El DSF tendrá que ser alguna característica o parámetro medible de la estructura que sea modificado por la presencia del daño.
4. ¿De qué modo el daño influye en la DSF?

Una vez que todas estas preguntas tengan respuesta, se tendrá toda la información necesaria para elegir una estrategia global de SHM apropiada para el objetivo que se ha fijado. A continuación, se van a responder estas cuestiones para el caso de estructuras aeronáuticas.

### **2.2.1. Estructuras aeronáuticas**

No todas las estructuras que componen una aeronave están sometidas al mismo tipo de cargas y, por lo tanto, no presentan los mismos problemas ni se usan los mismos criterios de diseño para toda la aeronave.

En aeronáutica, la inmensa mayoría de partes se diseñan a fatiga. Esto quiere decir que, para un número determinado de ciclos y con un espectro determinado de cargas, la pieza o estructura no compromete la integridad global de la aeronave.

Dentro del diseño a fatiga hay otras dos tendencias, el diseño a vida segura o a tolerancia al daño. En las estructuras diseñadas bajo el criterio de vida segura garantizan que, mientras no se supere el espectro de carga, la integridad estructural no peligra. En cambio, el diseño de tolerancia al daño tiene en cuenta que en el propio proceso de fabricación se generan grietas en el material y asume que estas grietas van a crecer, por lo que toma medidas para asegurar que no se produzcan fallos catastróficos [**criterios**].

Usando las Figuras ?? y ?? se puede tener una idea general de la estructura bajo estudio y responder a la primera pregunta. El siguiente paso será diferenciar entre los distintos daños que pueden sufrir estas estructuras.

### **2.2.2. Daños en estructuras aeronáuticas**

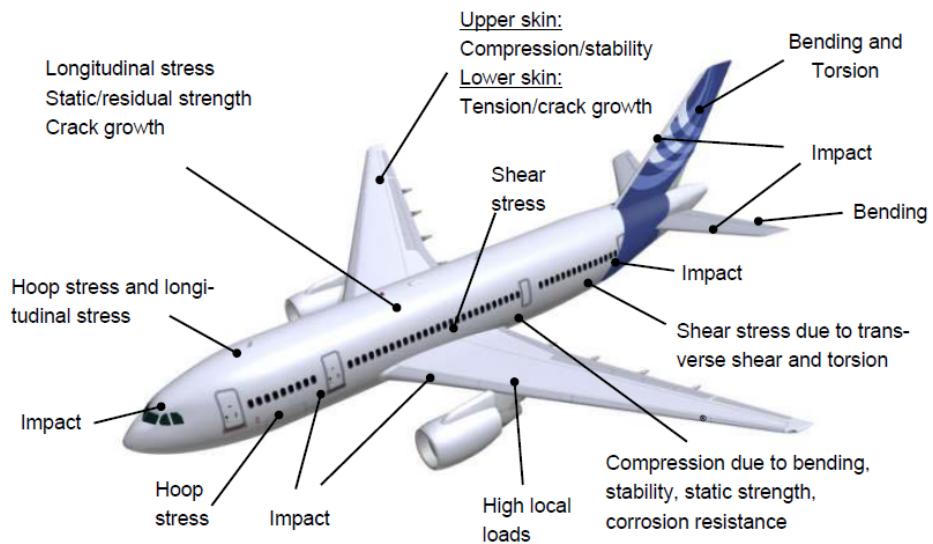


Figura 2.2: Problemas asociados a las estructuras de una aeronave

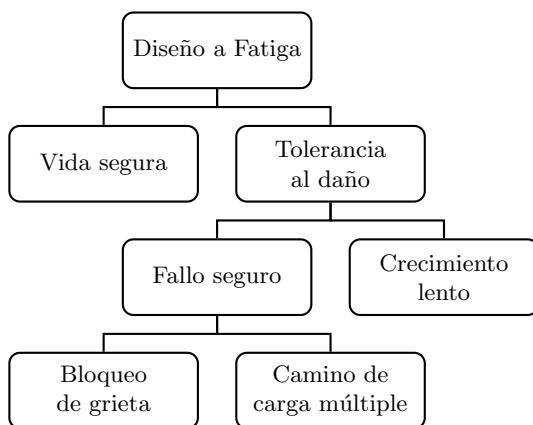


Figura 2.3: Criterios de diseño [criterios]

## 2.3. Inteligencia Artificial aplicada a SHM

2-3 pags

1. Explicar que es la inteligencia artificial y los tipos que hay
2. Diferencias que hay entre Machine learning y Deep learning y por qué decidimos que el Deep Learning es mejor para nuestro objetivo
3. Comentar las distintas redes neuronales dentro de DL y argumentos para elegir las Recurrent Neural Networks - RNN
4. Ejemplos de IA aplicada en SHM

### 2.3.1. Representación gráfica del funcionamiento de una Red Neuronal

Aunque el campo del DL lleva unos años en auge, para la mayoría de usuarios sigue siendo una simple herramienta, es desconocido cómo funciona y mucho menos se tiene una representación gráfica de qué hace y cómo llega a los resultados.

Dado que este trabajo se va a centrar en tecnología de DL, es conveniente hacer una breve explicación de cómo una NN es capaz de clasificar diferentes grupos de datos. Para ello, esta explicación se ha basado en el gran trabajo realizado por el equipo de *TensorFlow - playground* [[TensorFlow playground](#)]. Un entorno perfecto para la visualización del comportamiento de los diferentes parámetros que componen una Red Neuronal.

Lo primero de todo es saber la forma de los argumentos de entrada que, como para casi todo algoritmo, sus *inputs* son números. Dependiendo del problema que se busque resolver estos datos tendrán una forma u otra, más complejos o más simples, pero en esta introducción se va a utilizar un dataset de 2 dimensiones ( $x_1, x_2$ ) y dos clases diferentes (azul y naranja).

Los datos que se van a intentar clasificar están representados en la Figura ??

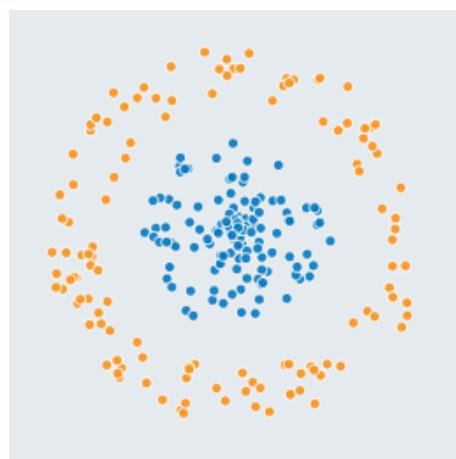


Figura 2.4: Representación de las dos clases diferentes en dos colores

Una vez se conocen los dos grupos que se busca clasificar se va a comenzar la descripción de los elementos que componen una NN y como interactúan entre ellos. Estos elementos son:

### • Neurona

Es la unidad básica de procesamiento que compone una Red Neuronal. Estas neuronas tienen conexiones a través de las que reciben los valores de entrada y realizan una suma ponderada con ellos. Cada una de las entradas es multiplicada por un valor, llamado peso, que definirá a cual de los valores de entrada se le da más importancia. A este cálculo se le añadirá un sesgo o *bias*, dicho de otro modo, se le sumará un escalar.

Si se observa detenidamente la Figura ?? (primera neurona llamada *perceptron*) se puede ver que, matemáticamente, una neurona es equivalente a una regresión lineal ( $w_1 \cdot x_1 + w_2 \cdot x_2 + b = 0$ ).

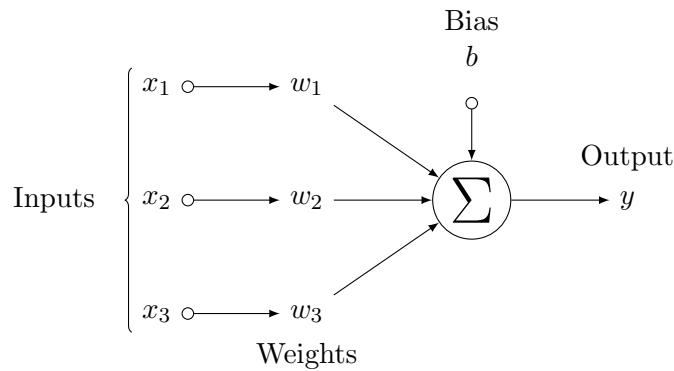


Figura 2.5: Representación del *perceptron*

En una regresión lineal bidimensional, ajustando los pesos ( $w_1, w_2$ ) se modifica la recta que separa el plano ( $x_1, x_2$ ) en dos regiones. En el ejemplo que se está resolviendo, por muy bien que se ajusten estos pesos, con una neurona solo se puede llegar a separar el plano en dos regiones, como se ve en la Figura ??.

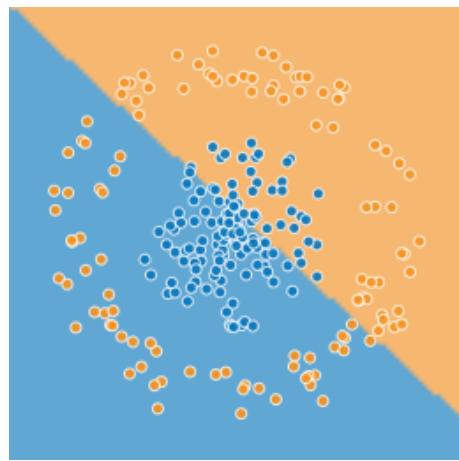


Figura 2.6: Clasificación de datos con una sola neurona

Para definir que región del plano es asignada a la clase azul o naranja se utiliza la

siguiente expresión:

$$w_1 \cdot x_1 + w_2 \cdot x_2 + b = y \begin{cases} y \geq 0, & \text{naranja} \\ y < 0, & \text{azul} \end{cases} \quad (2.1)$$

Por lo tanto, la neurona se puede sintetizar una función. Como una neurona no consigue separar las dos regiones de puntos de forma efectiva, hace que sea necesario dar un paso más.

### • Capas

Para modelizar conocimiento complejo no es suficiente con utilizar una única neurona, es necesario concatenar varias de ellas. Una forma de colocar las neuronas sería una debajo de otra en forma de columna, lo que se define como, en la misma capa.

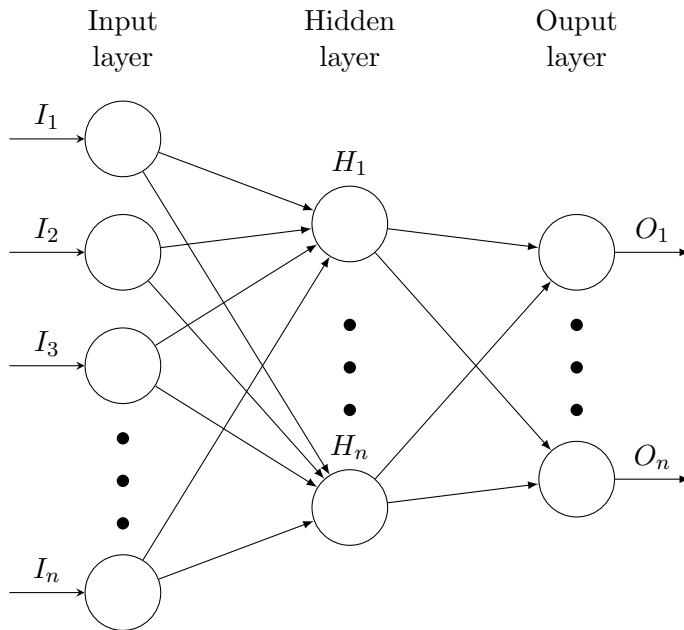


Figura 2.7: Representación de las capas de una red

Como se representa en la Figura ??, las capas se pueden concatenar de manera secuencial, una detrás de otra y dependiendo de dónde estén colocadas tendrán un nombre u otro. La capa que recibe primero los argumentos de entrada se llama *Input layer*, la que calcula el argumento de salida se llama *Output layer* y todas las capas que estén entre estas dos, se llamarán *Hidden layers*.

La concatenación secuencial de las capas hace que la red sea capaz de aprender conocimiento jerarquizado y es lo que da nombre al Deep Learning (Aprendizaje Profundo). Dicho de otro modo, si a la red se le introduce una imagen de un coche, es capaz de aprender que un coche no es un objeto único, una capa sintetiza lo que es una rueda, la siguiente lo que es una puerta y las últimas capas son capaces de juntar toda esa información para construir la idea de coche como un conjunto de objetos.

Las neuronas dentro de la misma capa lo que permiten es diversificar el conocimiento. Cada neurona se especializa en una parte de la información ajustando los pesos para dar más importancia a un canal de entrada u otro.

Juntando lo explicado anteriormente se entiende mejor este concepto de aprendizaje. Una red parte de una cantidad grande de datos, los píxeles de una imagen, por ejemplo. La capa de entrada será la que primero reciba la información, por ello necesitará muchas neuronas para procesarla y extraer de ella qué zonas de la imagen son más importantes. A medida que se va avanzando, cada una de las capas tiene que desglosar cada vez más la información sintetizada que le ha llegado de la capa anterior, por lo que necesitará menos neuronas, su nivel de abstracción de conocimiento aumenta.

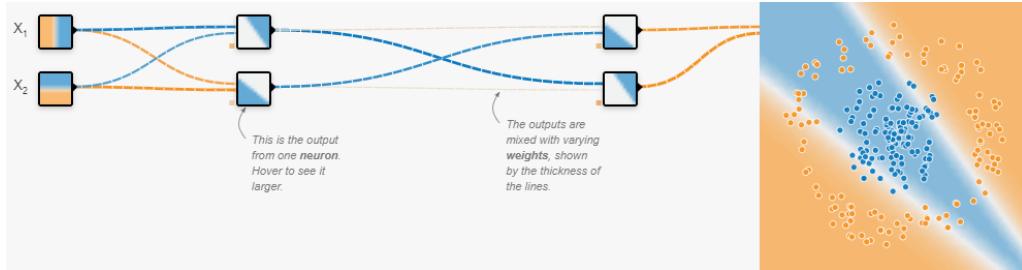


Figura 2.8: Red con dos *Hidden layers* y dos neuronas en cada capa

La aplicación de lo explicado anteriormente se representa en la Figura ???. Se puede observar en la caja que representa cada neurona que, en esencia, todas hacen una regresión lineal y que no pueden crear una frontera circular para separar el espacio. Esto se debe a que la operación matemática de concatenación de regresiones lineales termina colapsando en una regresión lineal, no puede dibujar curvas. Para conseguirlo, es necesario añadir un elemento más.

#### • Función de activación

La necesidad de aplicar una distorsión no lineal a la salida de las neuronas convierte en indispensables a las funciones de activación. Con ellas se podrá encadenar de forma efectiva la computación de varias neuronas.

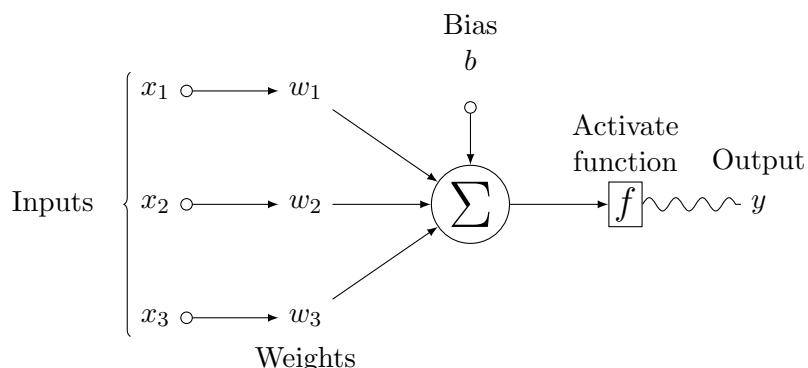


Figura 2.9: Efecto de la función de activación sobre el output de una neurona

En cierto modo ya se estaba usando una función de activación antes de nombrarlas. En la Ecuación ?? se había usado la función escalón para separar las dos regiones del plano, pero esta función no provoca no linealidades y no es muy efectiva cuando se requieren fronteras complejas.

Las funciones de activación más conocidas están representadas en la Figura ???. Su

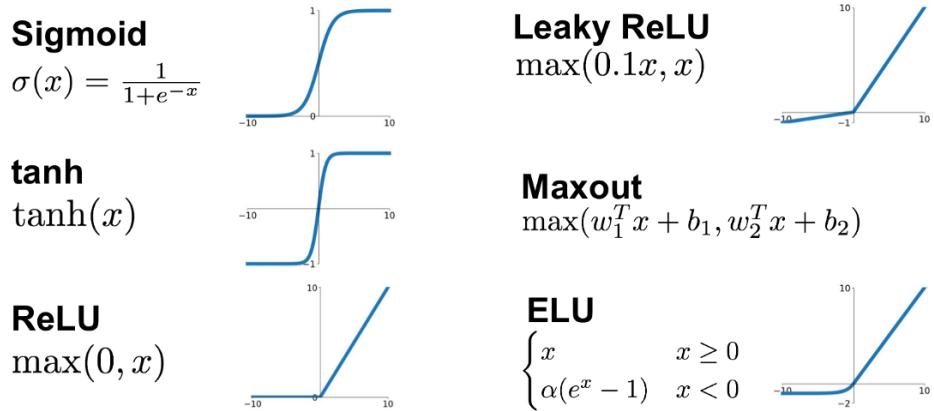


Figura 2.10: Funciones de activación más populares

función principal es hacer que los valores muy grandes o muy pequeños se saturen a determinado valor, dependiendo de la función elegida.

Para ver el efecto de distorsión geométrica que tienen estas funciones sobre el *output* de las neuronas se usa de ejemplo la Figura ???. La nueva función de salida se convierte en una superficie, continua y derivable, que para cada *input*  $(x_1, x_2)$  le corresponde una altura  $y$  entre 0 y 1.

Tal y como se ha hecho anteriormente, se selecciona un valor umbral de  $y$  a partir del cual se clasificarán los puntos como azules o naranjas. En la Figura ?? este valor umbral se representa como un plano a una altura de 0,5, por lo tanto, los puntos  $(x_1, x_2)$  que tras ser evaluados en la neurona saquen un valor de  $y$  inferior a 0,5 se clasificarán como *naranjas* y los que den un valor superior o igual a 0,5 se clasificarán como *azules*. Esto recuerda a la Figura ??.

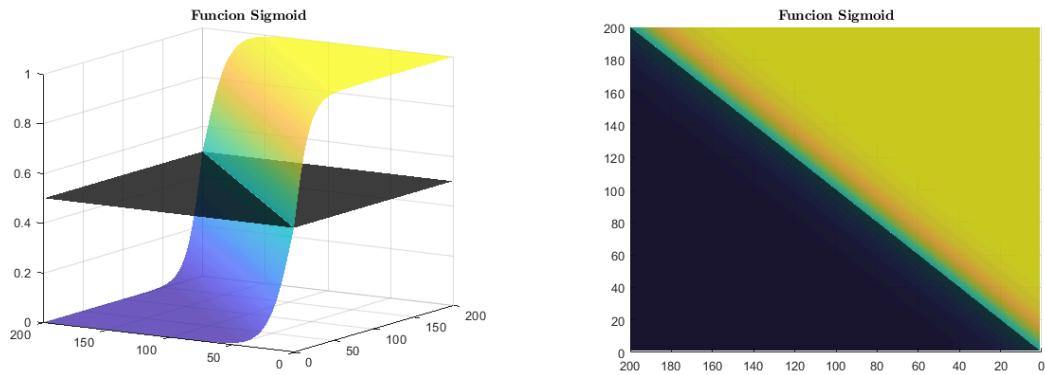


Figura 2.11: Combinación de funciones de activación

Ahora ya se puede utilizar una *hidden layer* con varias neuronas de forma efectiva. La combinación de varias neuronas hace que se sumen superficies (funciones) como la representada en la Figura ??, pero con diferentes orientaciones, generando superficies más complejas, Figura ???. Igual que antes, los puntos  $(x_1, x_2)$  que tras ser evaluados tienen un  $y$  mayor o igual a 0,5 pertenecerán a la clase azul y los menores a la

naranja.

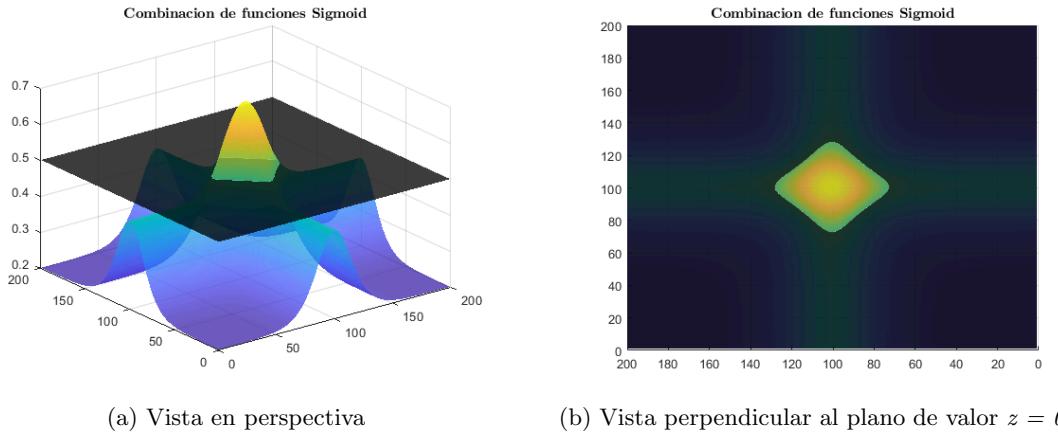


Figura 2.12: Combinación de funciones de activación

Finalmente se puede ver en la Figura ?? la aplicación de todo lo explicado. Gracias a las funciones de activación se consiguen las no linealidades necesarias para generar la superficie que separa exitosamente las dos clases de datos diferentes.

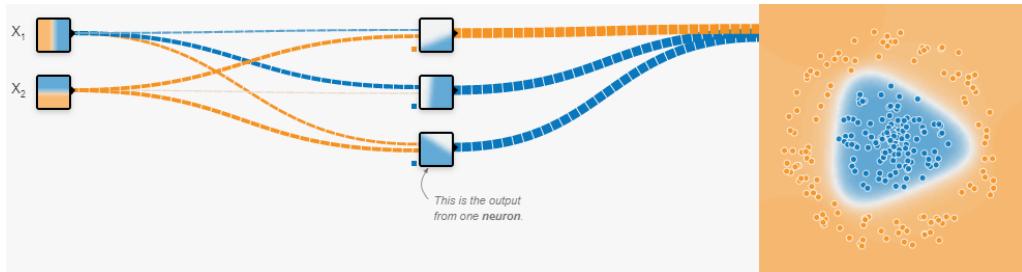


Figura 2.13: Resultado final de la clasificación

Las neuronas de la Figura ?? están unidas mediante unas líneas discontinuas de diferente color y grosor. Esto representa qué pesos son mayores y qué zona perturban más. Los pesos son los elementos que ajustan las funciones de activación para conseguir la superficie final que mejor realice la clasificación. Pero, ¿cómo se realiza el ajuste de los pesos?

#### • Entrenamiento

Se llama entrenamiento al proceso de aprendizaje automático que utilizan las redes neuronales para ajustar sus parámetros internos. El algoritmo que hizo posible que la optimización de parámetros en DL fuera eficiente se llama *backpropagation* y es el responsable del final del *Invierno de la Inteligencia Artificial*

Para llevar a cabo el proceso de optimización es necesario calcular las derivadas parciales de cada uno de los parámetros de la red con respecto a la función de error. Una vez se consigan las derivadas, se utilizará el algoritmo de descenso del gradiente para variar los parámetros y minimizar la función de error.

La función de error ( $E$ ) puede ser, por ejemplo, la suma de las distancias de cada punto a la frontera de su clase. Volviendo a la Figura ??, los puntos azules que se

encuentran en la región naranja aumentarán mucho el valor de la función de error, mientras que los que están en la región azul no aumentarán casi su valor. La variación de la función de error respecto a un pequeño cambio los parámetros se representa matemáticamente como el gradiente gradiente:  $\frac{\partial E}{\partial w}$ .

Como se ve en la Figura ??, el peso asociado a una de las primeras conexiones afecta a prácticamente todas las conexiones posteriores y por lo tanto, también influirán en su gradiente. Esto hace que el aprendizaje de este primer peso sea especialmente complicado y convierte en esencial la utilización del *backpropagation* para el proceso de optimización.

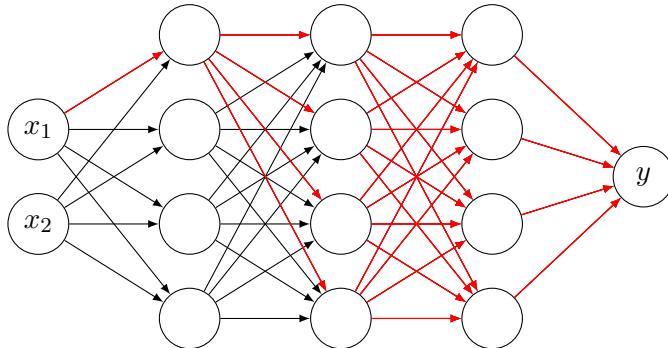


Figura 2.14: Influencia de un parámetro en la red

Cuando la red clasifica erróneamente un *input* genera un valor alto de la función de error. En una NN, el error de las últimas capas depende directamente de las capas anteriores, es decir, si se detecta que una neurona no tiene influencia sobre el error, las capas posteriores a ella tampoco lo tendrán. En esto consiste el *backpropagation*, en la retropropagación de errores, lo que hace muy eficiente el proceso. En la Figura ?? se puede ver el inicio del algoritmo partiendo de la última capa.

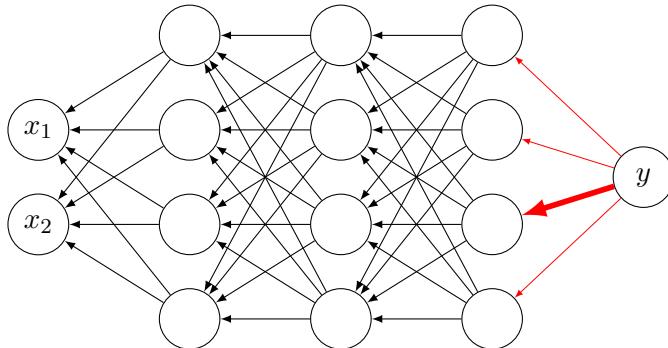


Figura 2.15: primer pase de backpropagation

Esta distribución de errores se utiliza para actualizar los pesos y bias de cada una de las neuronas.

Ahora que se han imputado los errores a las neuronas de la última capa se puede proceder a repetir el mismo proceso de antes como si este fuera el error final de la red, como si esta fuera ahora la última capa. Se va avanzando capa tras capa moviendo el error hacia atrás.

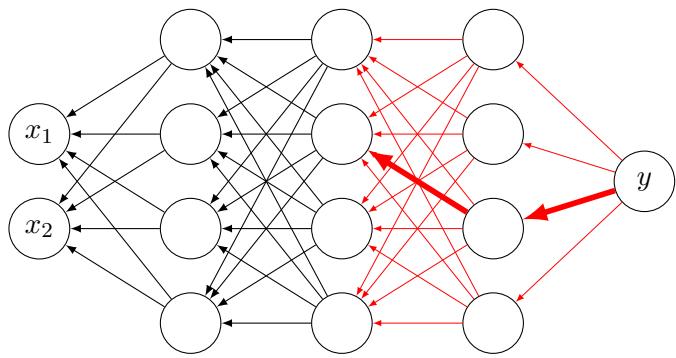


Figura 2.16: Segundo pase

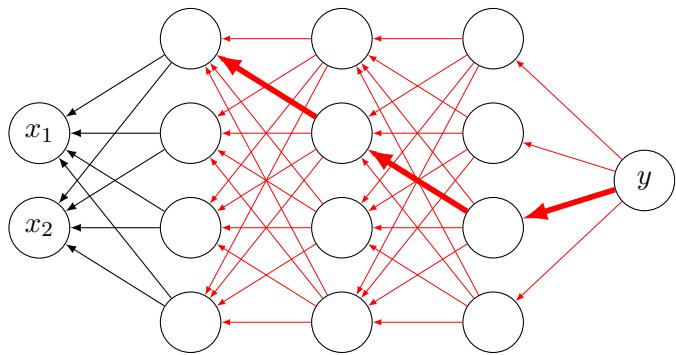


Figura 2.17: Tercer pase

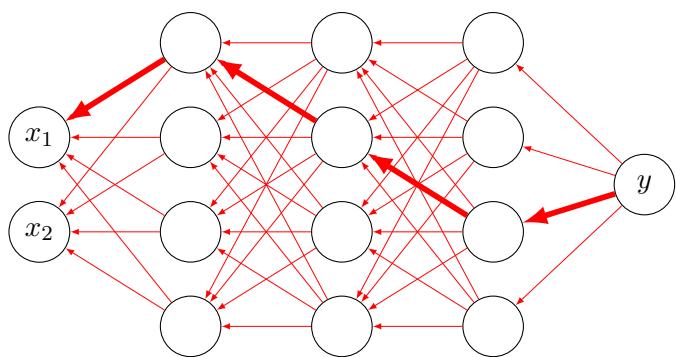


Figura 2.18: Cuarto pase

## **2.4. IA aplicado a la monitorización estructural**

TODOS los sectores menos aeronáutica - entre 5 y 10 pags

Poner un estado actuar de como se esta aplicando la IA a la monitorización estructuras en general

## **2.5. IA aplicada a la monitorización de estructuras aeronáuticas**

INESASSE (5 pags VER TFM Jaime García SOLO INTRODUCIR)

Dar una vuelta al apartado anterior centrándolo sobre todo en estructuras aeronáuticas. Si se puede, poner ejemplos de estructuras de material compuesto e inferir en la complejidad que dan frente a las metálicas por su direccionalidad de propiedades.

## **2.6. RNN LSTM**

Esplicación completa y detallada

## Capítulo 3

# Aplicación de Deep Learning en SHM con sensores de deformación

### 3.1. Proyecto INESASSE

El Proyecto INESASSE (Integración y Explotación de Sistemas de Autodiagnóstico y Supervisión de Salud Estructural en aviones no tripulados) tiene como objetivo principal investigar la capacidad de implantación, en condiciones operativas, de sistemas de supervisión estructural (SSE), empleando redes de sensores de fibra óptica.

Este proyecto de investigación fue realizado por el INTA (Instituto Nacional de Técnica Aeronáutica), en conjunto con la ETSIAE (Escuela Técnica Superior de Ingeniería Aeronáutica y del Espacio), y consta de 3 fases:

1. Elaboración de un modelo de elementos finitos (FEM) del sistema estructural, a través del cual se establece la morfología de la red de sensores. Sobre este FEM se simularán distintos daños producidos sobre la estructura, a fin de analizarlos mediante distintas técnicas SHM y determinar la posición óptima de la red sensorial.
2. Fabricación de dos estructuras idénticas de CFRP que se instrumentaran con la red de sensores definida en la fase 1, integrables en el UAV en cuestión. Tras esto, a una de estas estructuras se le realizará una campaña de ensayos en tierra a fin de validar los FEM, además de comparar los resultados obtenidos mediante herramientas SHM sobre el modelo teórico y el real. Al igual que en el FEM, se realizarán los mismos daños estructurales controlados.
3. En la última fase del proyecto, la estructura fabricada anteriormente que no fue ensayada, se integrará en un UAV en operaciones de vuelo reales, procesando datos de los sensores integrados en la fase 2, e investigando las capacidades del sistema SHM seleccionado para el vuelo.

El vehículo aéreo no tripulado que protagoniza este proyecto es el UAV Milano, con una masa en servicio de 900 kg y una autonomía de hasta 20 horas. En la Figura ?? se puede apreciar una imagen del mismo.

#### 3.1.1. Estructura y red de sensores

La estructura de la aeronave sobre la que se ha realizado el FEM y se ha fabricado después es el fuselaje posterior del UAV Milano. Esta estructura está compuesta por:



Figura 3.1: UAV INTA Milano

- Revestimiento inferior con un larguero longitudinal en T dividido en dos y refuerzos sobre las pestañas. Fabricado con cinta UD fuera de autoclave.
- Revestimiento superior con dos largueros en L y refuerzos sobre las pestañas. Fabricado con cinta UD fuera de autoclave.
- Cuatro cuadernas con refuerzos sobre las mismas. Fabricadas con cinta UD fuera de autoclave.
- Dimensiones aproximadas de 2600x850x890 mm.

En la Figura ?? se puede ver el diseño explosionado de los componentes descritos.

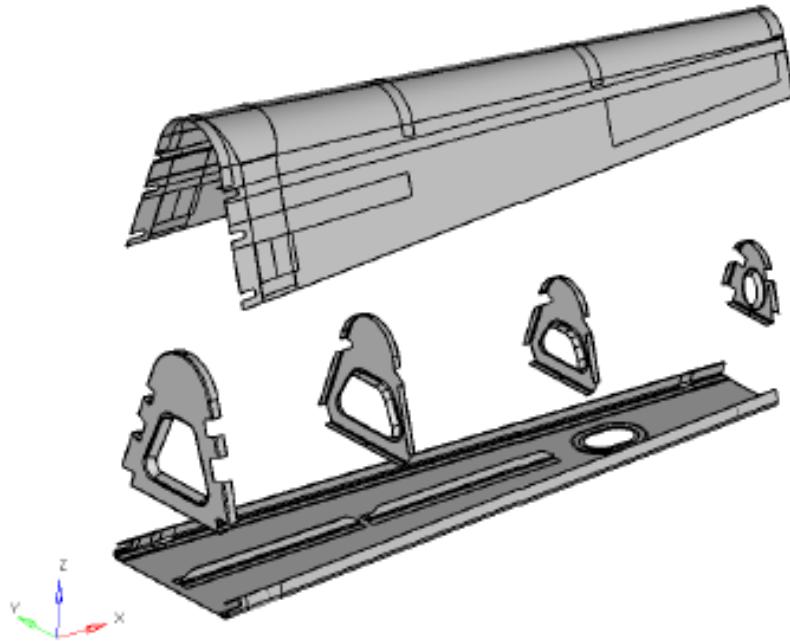


Figura 3.2: Fuselaje posterior

Sobre esta estructura se van a distribuir dos redes de sensores de fibra óptica, uno formado por sensores FBG y otro por OBR. Con ellos se recogerá el campo de deformaciones

de la estructura que se procesará para obtener la clasificación de los daños. En las Figuras ?? y ?? se tiene una representación gráfica de la posición de estas redes en la estructura.

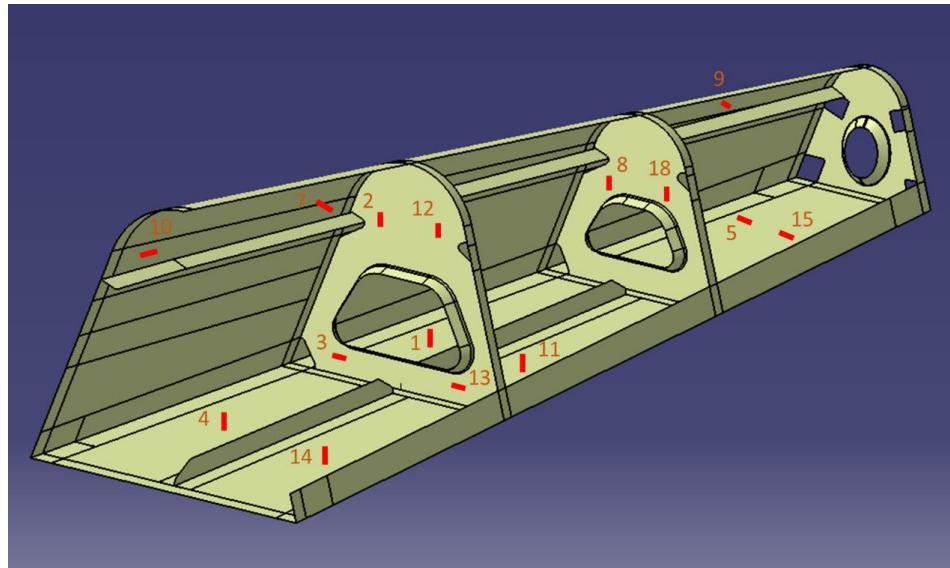


Figura 3.3: Red de sensores FBG

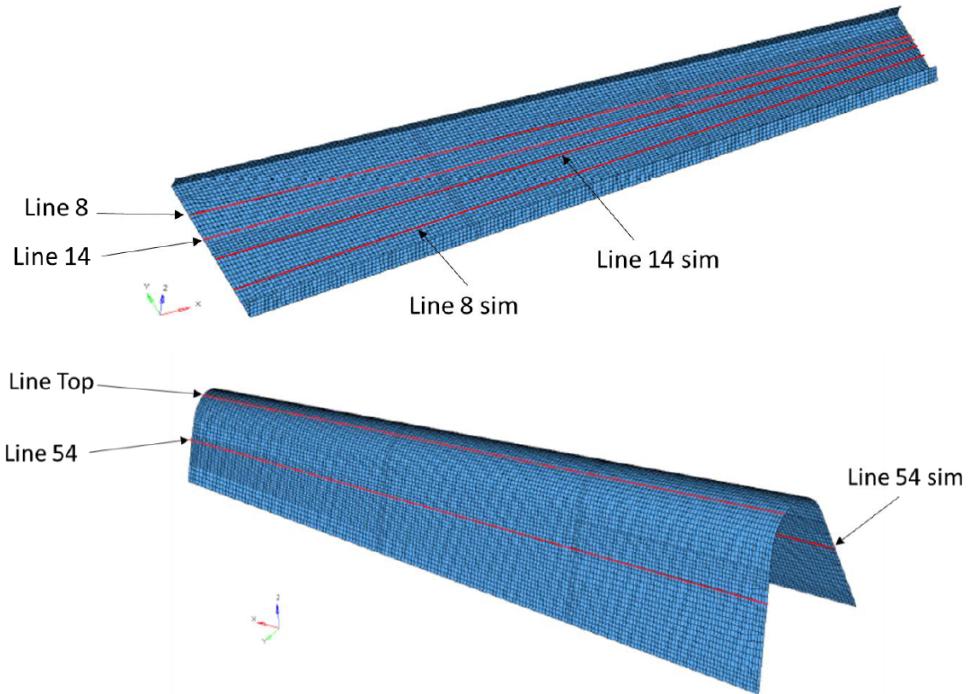


Figura 3.4: Red de sensores OBR

Es importante indicar que, puesto que los sensores de medida distribuida son unidireccionales, se han obtenido las deformaciones en la dirección longitudinal de la fibra (tracción/compresión).

### 3.1.2. Planteamiento de daños y descripción de ensayos

El objetivo de este proyecto es tener un sistema SHM para detectar daños en un fuselaje de material compuesto durante su operación, por lo tanto, se necesita definir la localización de estos daños y sus tamaños.

Estos daños corresponden con desencolados parciales o totales en las zonas que se definirán a continuación. En la Figura ?? se puede ver la localización de cada daño en la estructura y en la Tabla ?? se pueden ver sus características.

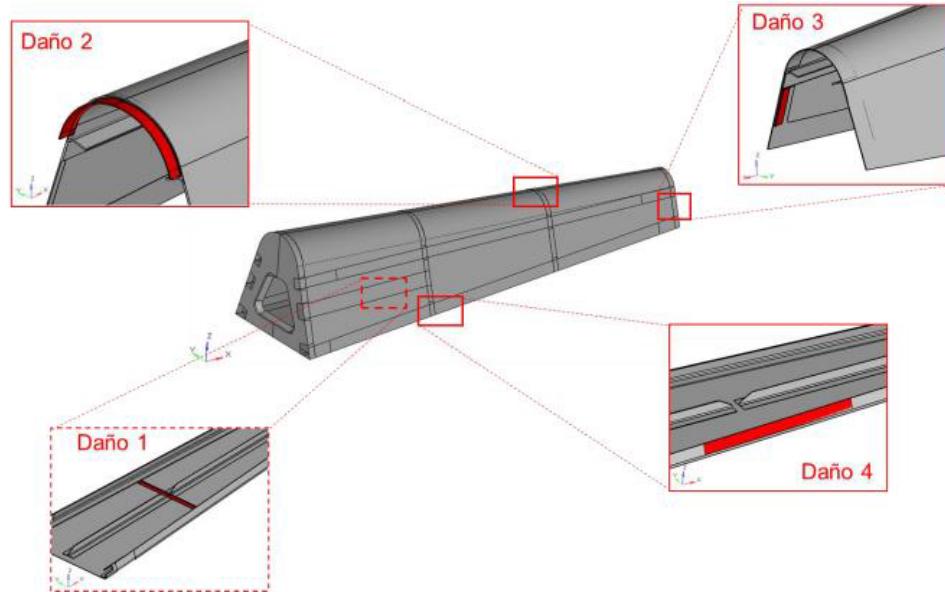


Figura 3.5: Esquema de los daños realizados a la estructura

DAÑO	TIPO DE DAÑO	ELEMENTOS AFECTADOS	LONGITUD MÁXIMA	INCREMENTO
D1	Desencolado parcial del pie de la cuaderna	Cuaderna 2, revestimiento inferior	450 mm	30 mm
D2	Desencolado parcial de la pestaña superior de la cuaderna	Cuaderna 3, revestimiento superior	350 mm	25 mm
D3	Desencolado parcial de la pestaña lateral de la cuaderna	Cuaderna 4, revestimiento superior	180 mm	30 mm
D4	Desencolado parcial entre el revestimiento inferior y el revestimiento superior	Revestimiento inferior, revestimiento superior	900 mm	30 mm

Tabla 3.1: Definición de daños en el fuselaje

Es importante resaltar que, aunque el ensayo estructural y las simulaciones se realizaron para los cuatro daños, se llegó a la conclusión de que el daño 3 era irrelevante tras ver los resultados obtenidos. Por lo tanto, dicho daño fue descartado.

Para facilitar el trabajo a los operarios que realizaron los ensayos del fuselaje, se sustituyó la unión adhesiva por uniones remachadas. Esto permitió controlar mejor la progresión de los daños y también que fueran reparados tal y como se ve en la Figura ??.

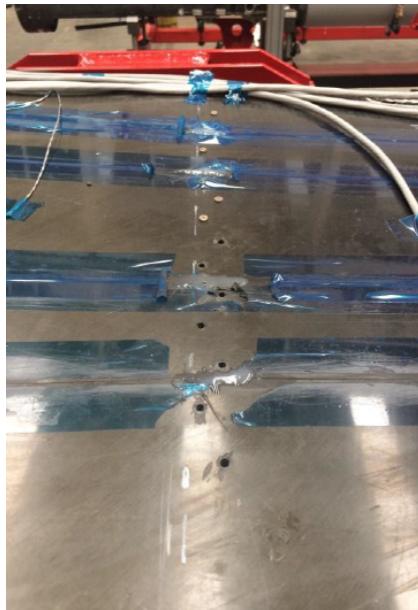


Figura 3.6: Realización del daño por extracción de remaches

Los ensayos sobre la estructura real se simularon también siguiendo la misma configuración en el FEM. Las condiciones de contorno a las que está sometido el fuselaje son las siguientes:

- Empotramiento en la sección de unión con el fuselaje anterior
- Carga vertical en la sección de unión con la cola

El ensayo comienza con la estructura descargada y se aplica la carga de forma escalonada. La carga va aumentando 400 N en 10 escalones hasta llegar a un máximo de 4 kN.

La configuración del banco de ensayos se puede ver en la Figura ??

### 3.1.3. Modelo de elementos finitos

Se va a hacer un breve resumen de las características del FEM utilizado. Este modelo consta de 62802 grados de libertad, el tamaño medio de sus elementos es de 15 mm, y está formado por:

- 26181 nodos
- 34080 elementos quad4 (Ratio máximo de 4.768)
- 25 elementos tria3 (Ratio máximo de 6.219)

Los elementos tria3 del modelo se encuentran principalmente en las cuatro cuadernas debido a que su geometría es más complicada de mallar que la del resto del fuselaje.

En la Figura ?? se pueden ver los distintos elementos del modelo y en cuales de ellos se introducen las condiciones de contorno.

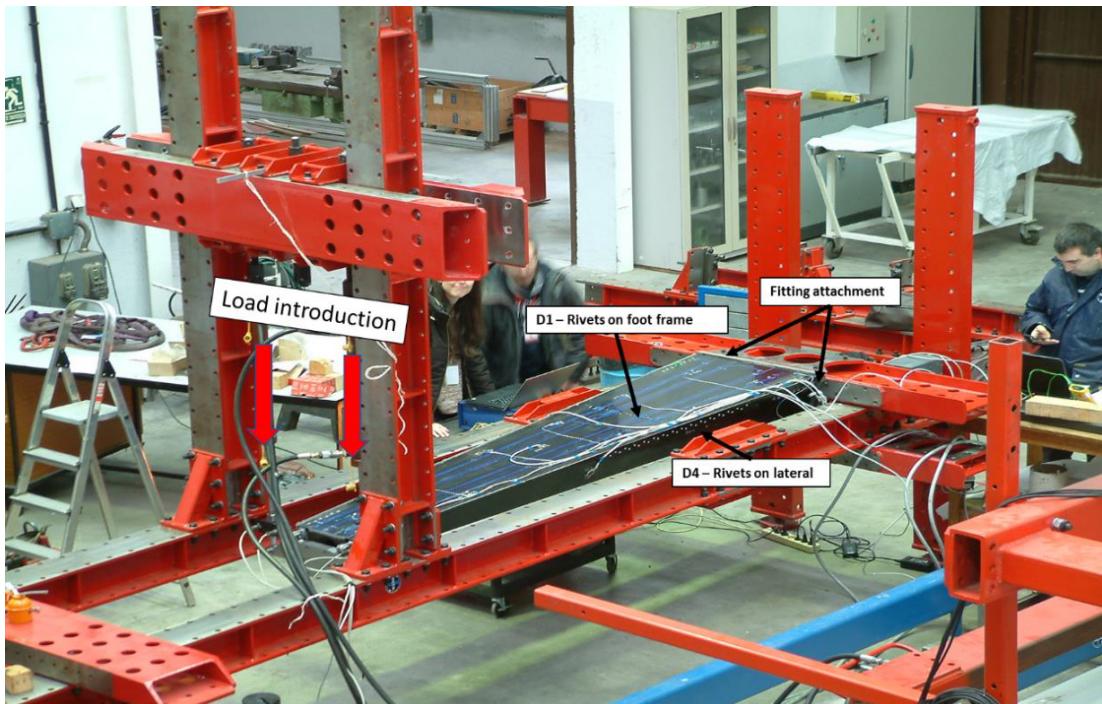


Figura 3.7: Vista general de la configuración de los ensayos

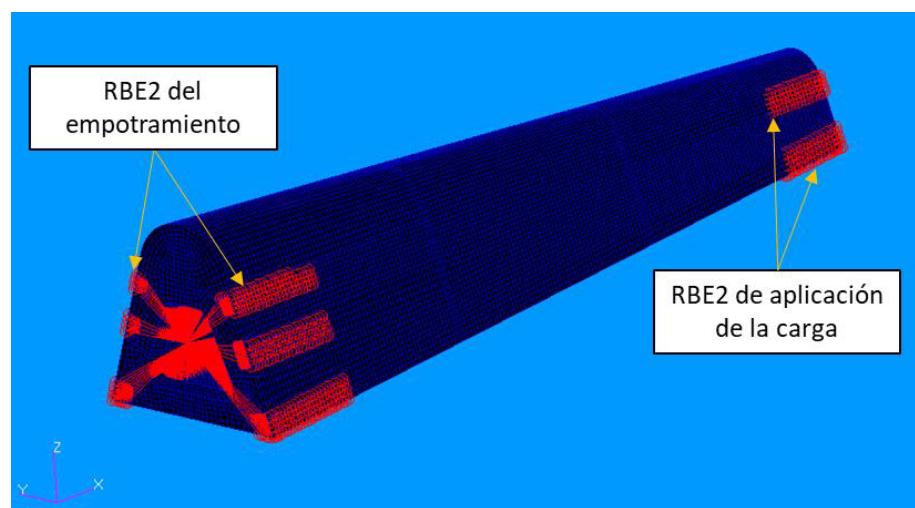


Figura 3.8: Modelo de elementos finitos del fuselaje posterior del UAV Milano

### 3.2. Pre-procesado de señales

Una vez que se han realizado los ensayos y simulaciones, es necesario pre-procesar los datos recogidos por las redes de sensores para ajustarlos en contenido y forma para que las redes neuronales puedan realizar una clasificación correcta.

Los datos que se obtienen de los ensayos reales se van a tratar de forma diferente a los obtenidos a través del FEM, por ello, este apartado está separado en dos grupos, *Ensayo real* y *FEM*.

#### 3.2.1. Ensayo real

La red de sensores FBG registra las deformaciones de forma ininterrumpida durante el ensayo completo, dando así una curva de deformación frente al tiempo para cada uno de los sensores distribuidos en la estructura, como se puede ver a continuación en la Figura ??.

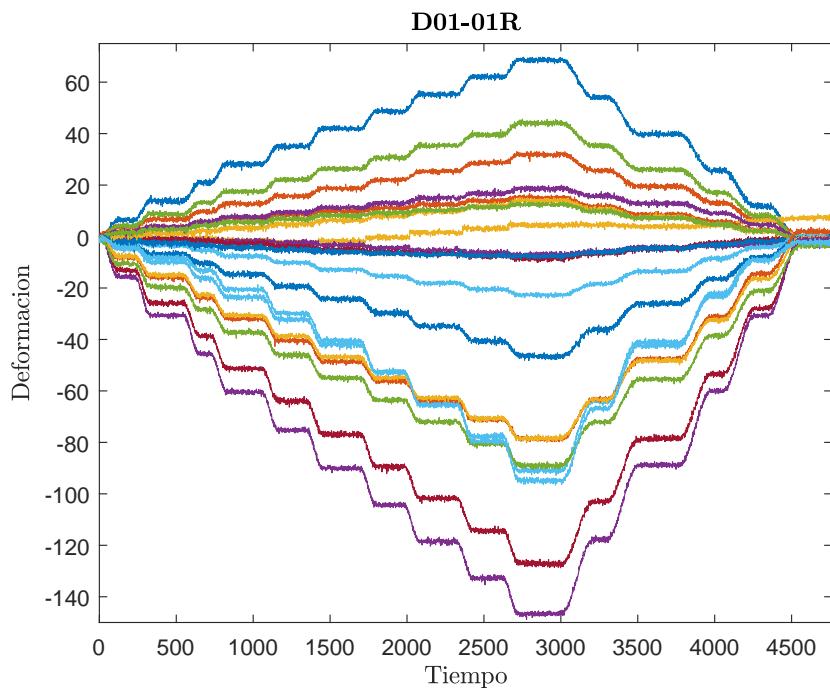


Figura 3.9: Medida de todos los sensores durante el ensayo del Daño 1 con un remache extraído

Cuando los actuadores llegan a un nivel de carga paran durante un tiempo antes de aumentar o disminuir la carga aplicada con el fin de estabilizar la carga. Por lo tanto, para separar las medidas en diferentes niveles de carga hay que detectar cuando el actuador comienza a moverse. La detección de el comienzo de aplicación de carga está representada en las Figuras ?? y ??.

Finalmente se eliminan las medidas obtenidas durante la transición entre escalones usando como referencia las líneas naranjas de la Figura ?? y se fija la carga mínima que se va a usar durante el estudio en un 30 % de la carga máxima.

Una vez que se han aislado los escalones se dibuja un histograma para ver la variabilidad de los datos obtenidos. Como se ve en la Figura ??, los valores de las medidas

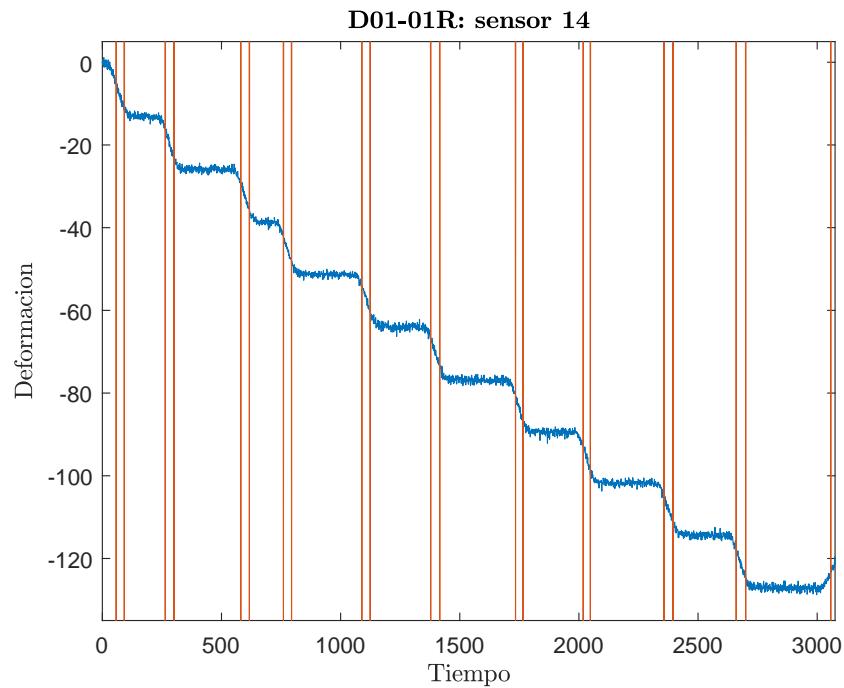


Figura 3.10: Detección de los escalones producidos por las carga vertical

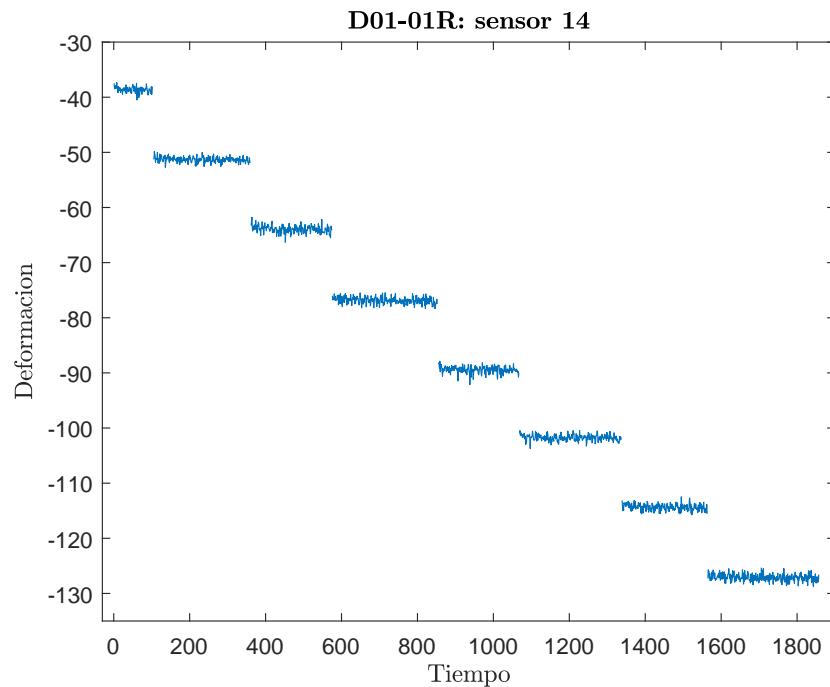


Figura 3.11: Medidas de los sensores FBG en el ensayo preprocesadas

se asemejan a una distribución normal con mida -127.75 deformaciones. Esto quiere decir que el ruido introducido por los sensores FBG hace que cada medida que se realiza en un tiempo diferente pueda ser considerada como un ensayo independiente. De un único ciclo de carga se obtienen una gran cantidad de datos para entrenar a la red.

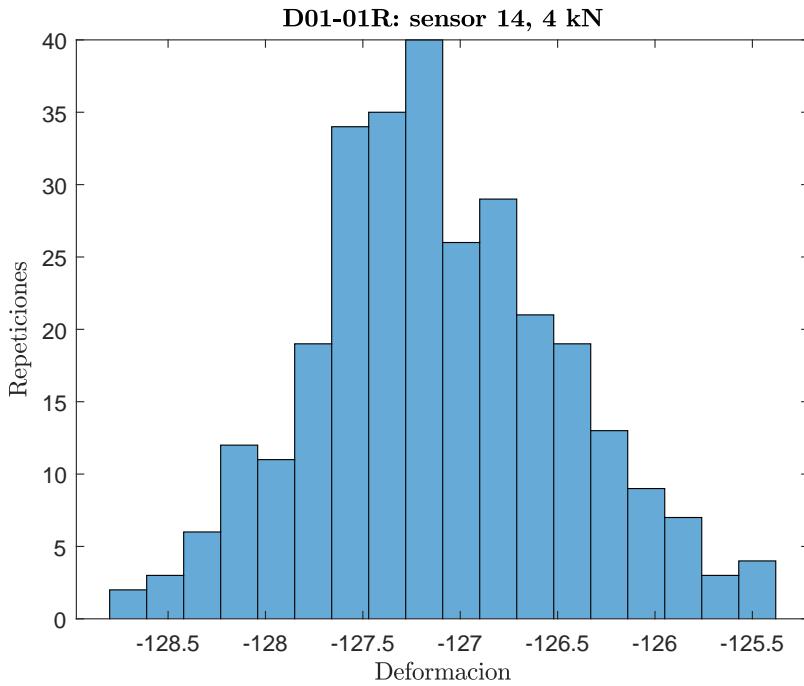


Figura 3.12: Histograma del escalón 4 kN, 100 % de carga

Tras pre-procesar las señales se puede visualizar el campo de deformaciones medido por los sensores FBG para cada estado de daño y el estado sin daño o de referencia en la Figura ???. Al estar cada daño localizado en una zona diferente de la estructura, los cambios en el campo de deformaciones son locales y en la Figura ?? se aprecia que sensores están más cerca de la zona dañada ya que sufren una mayor variación respecto a las medidas obtenidas de la estructura sin daño.

Por otro lado, los sensores OBR realizan una captura instantánea del campo de deformaciones. Esto quiere decir que se tiene una única muestra para cada tipo de daño y nivel de carga (ejemplo en la Figura ??), lo cual es insuficiente para conseguir unos buenos resultados con la red.

Siguiendo el mismo razonamiento usado con los sensores FBGs, si se añade una distribución de ruido gausiano a un valor del campo de deformaciones, se pueden generar ensayos independientes con los que entrenar a la red. Sin embargo, el campo de deformaciones de la Figura ?? ya contiene ruido asociado a la medida.

Para conseguir un campo de deformaciones que usar como referencia, al que añadir posteriormente ruido, se va a utilizar el filtro *Savitzky-Golay*. Este filtro digital es comúnmente usado para suavizar series de datos equiespaciados [**Savitzky-Golay**].

La diferencia entre la medida sin filtrar y filtrada se aprecia claramente en la Figura ??.

Ahora que se tiene una medida de referencia, ya se puede añadir la distribución de ruido de amplitud 1.5 deformaciones (Figura ??) y generar más muestras.

En la Figura ?? se puede ver un detalle del caso D01-R03 en el que se ha ampliado las deformaciones medidas por una línea de sensor. Aquí se aprecia más claramente que cada una de estas muestras se pueda considerar como un ensayo independiente.

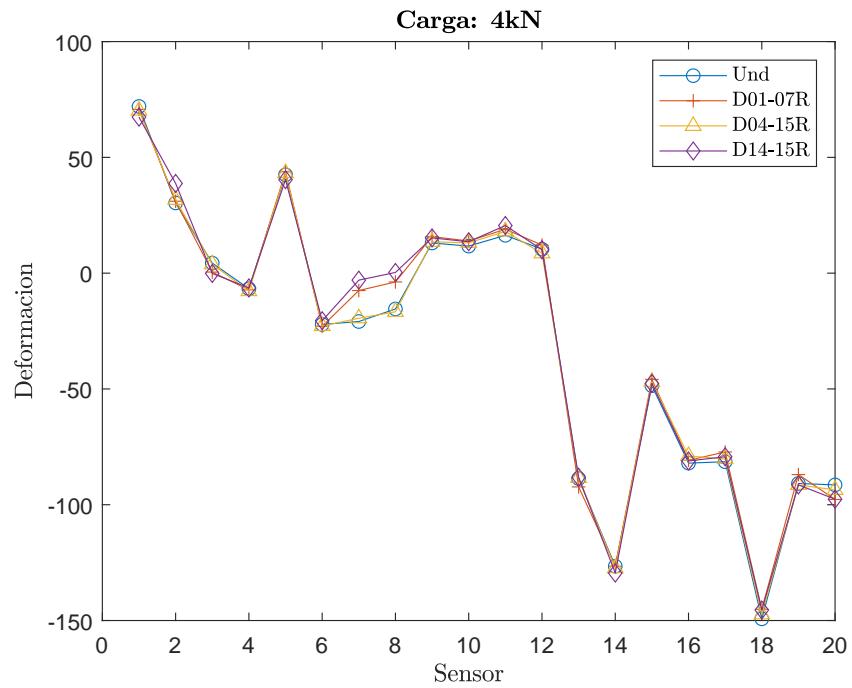


Figura 3.13: Campo de deformaciones medido por los sensores FBG para todos los daños

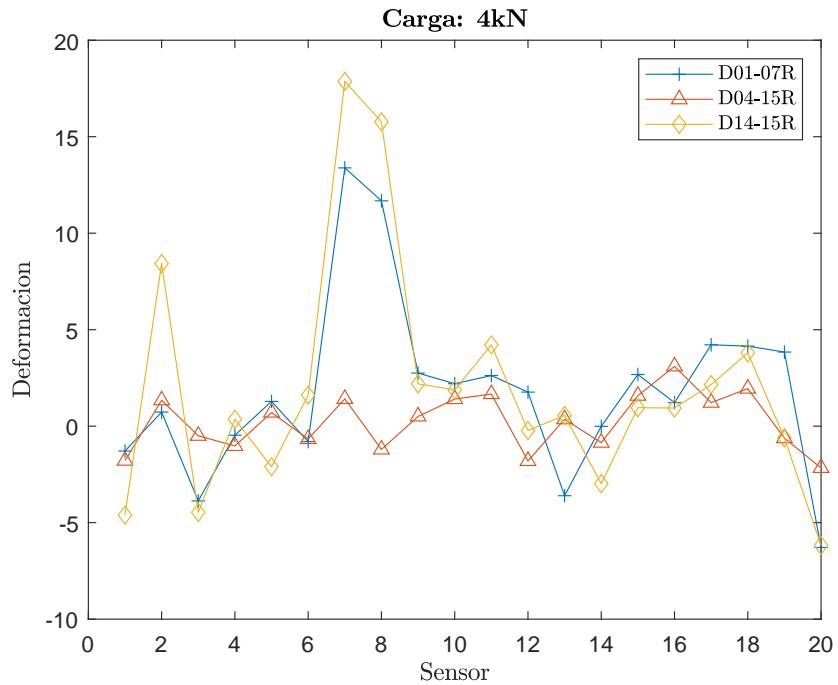


Figura 3.14: Diferencia de los campos de deformaciones referenciado con el estado sin daño (FBG)

Igual que se ha hecho con los FBGs, en las Figuras ?? y ?? se pueden ver los campos de deformaciones provocados por los distintos daños con la carga máxima y su diferencia

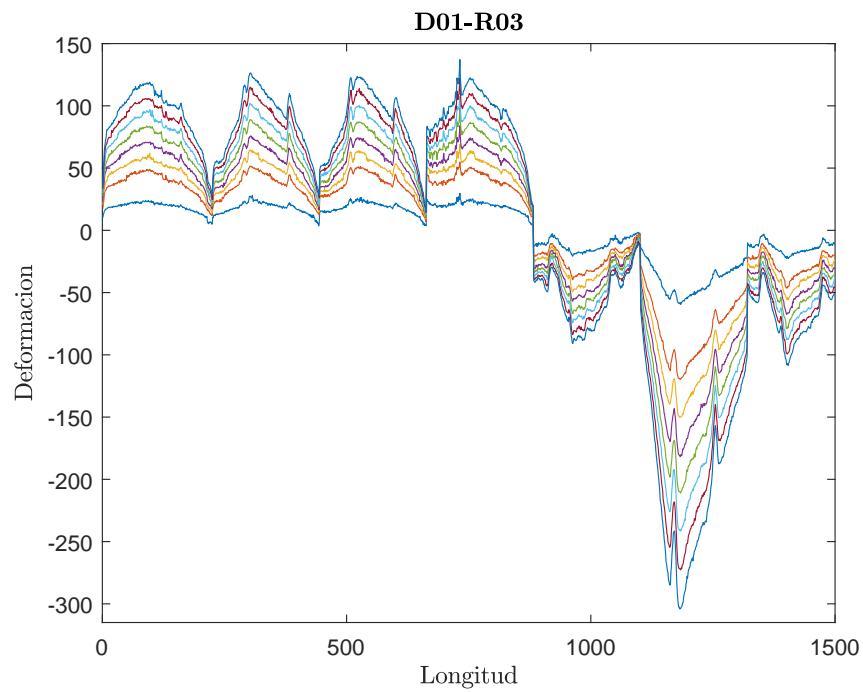


Figura 3.15: Campo de deformaciones provocado por el daño 1 con 3 remaches extraídos para todos los casos de carga

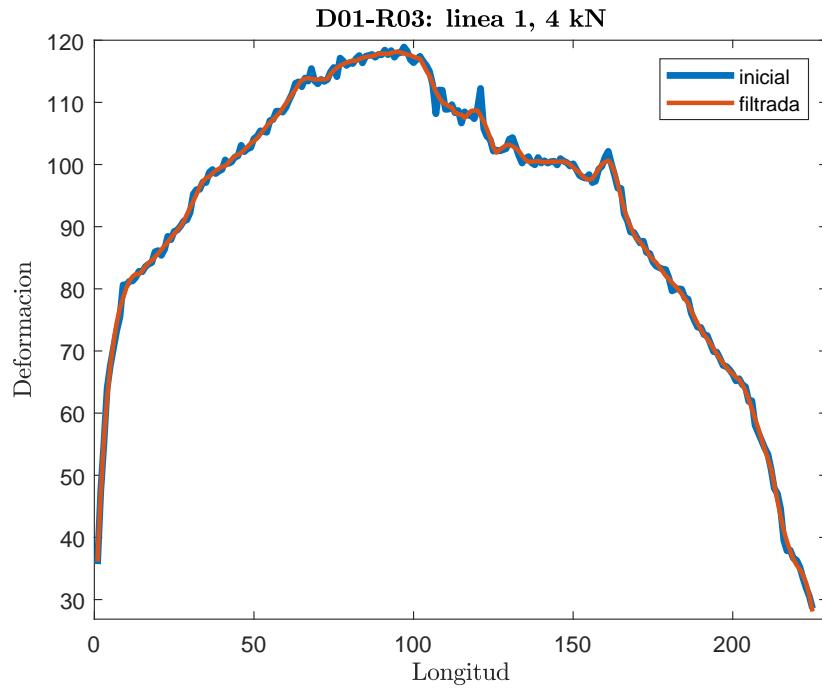


Figura 3.16: Aplicación del filtro Savitzky-Golay a una linea de sensor OBR

respecto al estado sin daño.

<https://towardsdatascience.com/hands-on-graph-neural-networks-with-pytorch-pytorch->

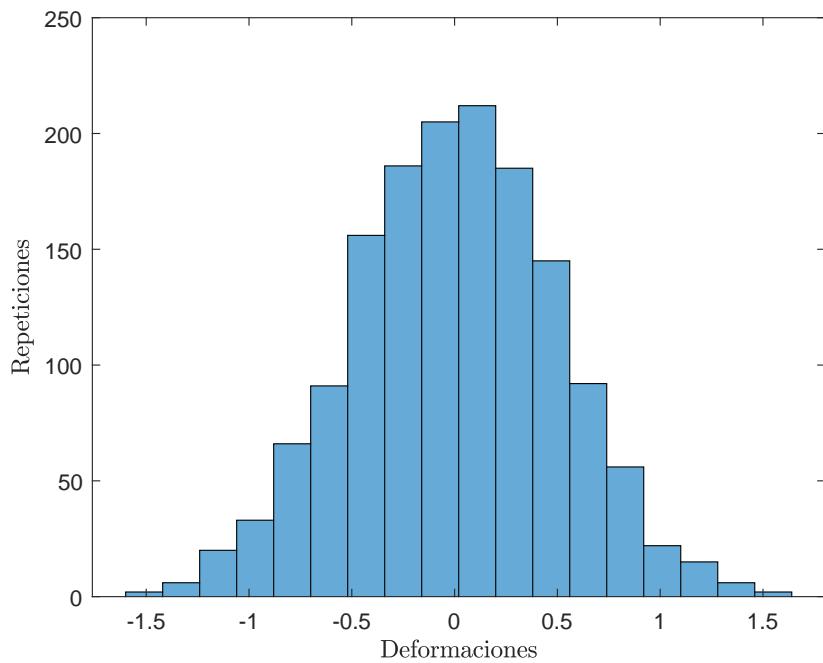


Figura 3.17: Distribución de ruido gausiano con amplitud 1.5 deformaciones

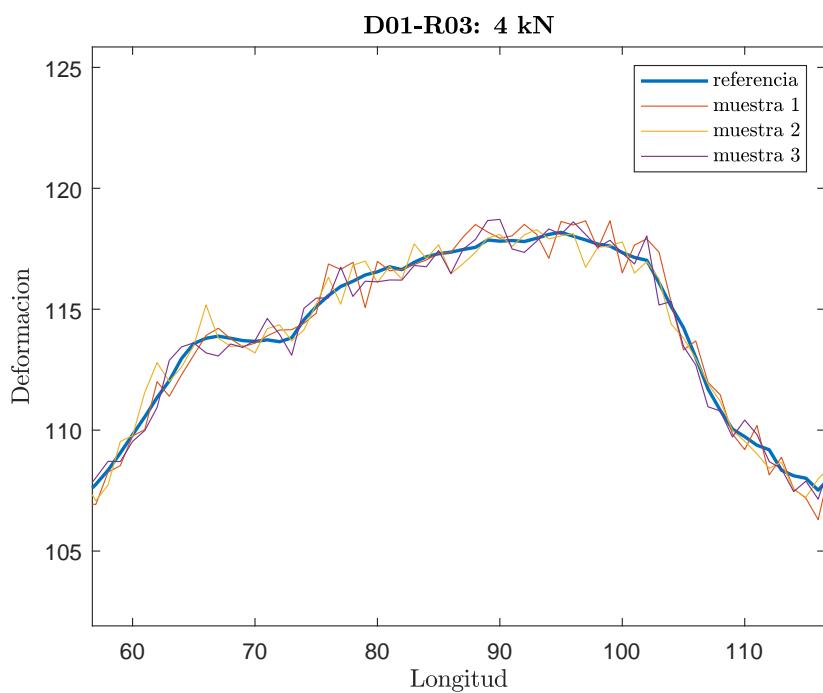


Figura 3.18: Detalle de las variaciones provocadas por el ruido gausiano

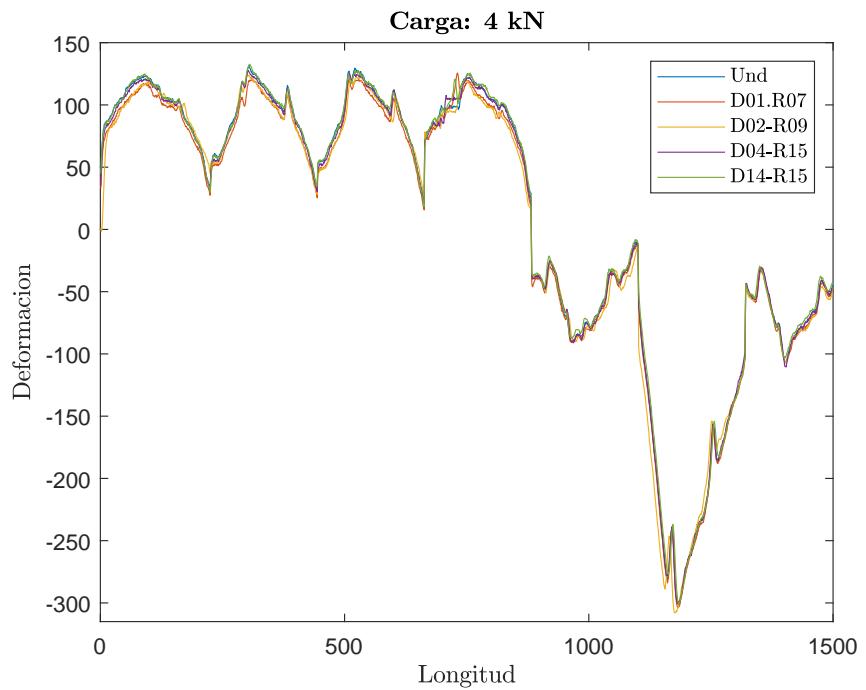


Figura 3.19: Campo de deformaciones medido por el sensor OBR para todos los daños

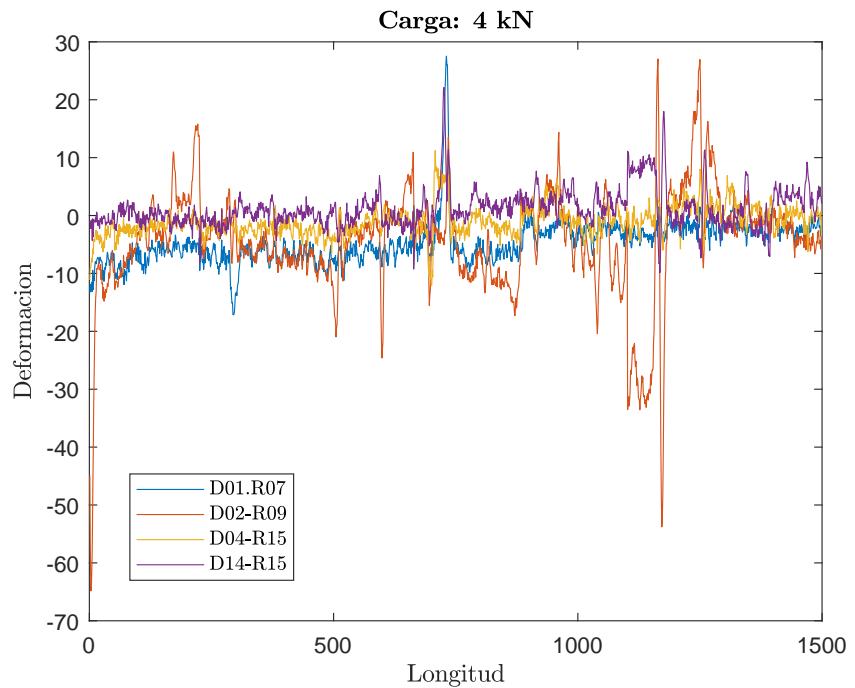


Figura 3.20: Diferencia de los campos de deformaciones referenciado con el estado sin daño (OBR)

### 3.3. Aplicación de DL al a la detección de daños mediante sensores de deformaciones

#### 3.3.1. Visualización del campo de deformaciones

Una vez que se han procesado las deformaciones se tiene un *Dataset* para los sensores FBG en el que cada muestra es de dimensión 20 (20 sensores) mientras que en el dataset de los sensores OBR las muestras son de dimensión 1499 (equivalente a tener 1499 sensores).

Un ser humano no puede visualizar un espacio con tantas dimensiones. Esto hace complicado tener una idea de la distribución espacial de las muestras y mucho menos imaginar las fronteras que genera la NN entre las distintas clases de datos (como se ha explicado en *Representación gráfica del funcionamiento una NN*).

Para reducir la dimensionalidad de los datasets y tener una representación gráfica bidimensional de ellos se va a utilizar un algoritmo de Machine Learning muy popular para la visualización de datos multidimensionales: **t-SNE (t-distributed Stochastic Neighbor Embedding)**. A continuación se va a hacer una breve introducción al funcionamiento de este algoritmo extraído de [[t-sne](#)].

*Stochastic Neighbor Embedding (SNE)* comienza convirtiendo las distancias euclídeas entre puntos del dataset de alta dimensionalidad en probabilidades condicionales que representan su similitud. La similitud del punto  $x_j$  con el punto  $x_i$  es la probabilidad condicional  $p_{j|i}$  y su proximidad se calcula usando una distribución de densidad de probabilidad gaussiana centrada en  $x_i$ . Para los puntos cercanos, el  $p_{j|i}$  es relativamente alto, mientras que para los puntos lejanos, el  $p_{j|i}$  será casi infinitesimal. Matemáticamente, el la probabilidad condicional  $p_{j|i}$  viene dada por la Ecuación ??.

$$p_{j|i} = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma_i^2)}{\sum_{k \neq i} (\exp(-\|x_i - x_k\|^2 / 2\sigma_i^2))} \quad (3.1)$$

En el espacio de baja dimensión los puntos  $y_i$  y  $y_j$  son la transformación de  $x_i$  y  $x_j$  del espacio con alta dimensionalidad. Es posible calcular una probabilidad condicional similar,  $q_{j|i}$  a la anterior. Para ello, fijamos la varianza del gaussiano que se emplea en el cálculo de las probabilidades condicionales  $q_{j|i}$  en  $\frac{1}{\sqrt{2}}$ . Por lo tanto, modelamos la similitud del punto  $y_j$  con  $y_i$  con la Ecuación ??

$$q_{j|i} = \frac{\exp(-\|y_i - y_j\|^2)}{\sum_{k \neq i} (\exp(-\|y_i - y_k\|^2))} \quad (3.2)$$

Si los puntos  $y_i$  y  $y_j$  del mapa de baja dimensionalidad modelan bien la similaridad entre los puntos  $x_i$  y  $x_j$  del espacio con alta dimensionalidad, las probabilidades  $p_{j|i}$  y  $q_{j|i}$  serán iguales. Por lo tanto, el objetivo de SNE es minimizar las discordancias entre  $p_{j|i}$  y  $q_{j|i}$ .

Igual que para las NN, es necesario tener una función de costes que represente el error que está cometiendo el algoritmo y así optimizarla por medio del descenso del gradiente. La función de costes elegida es la divergencia Kullback-Leibler individual entre las

distribuciones de probabilidad conjunta, P y Q (Equación ??).

$$C = KL(P||Q) = \sum_i \sum_j p_{ji} \log \frac{p_{ji}}{q_{ji}}$$
 (3.3)

Para ambos sensores se va a visualizar la similitud entre los distintos tipos de daño, su tamaño y la la carga aplicada.

- **Sensores FBG**

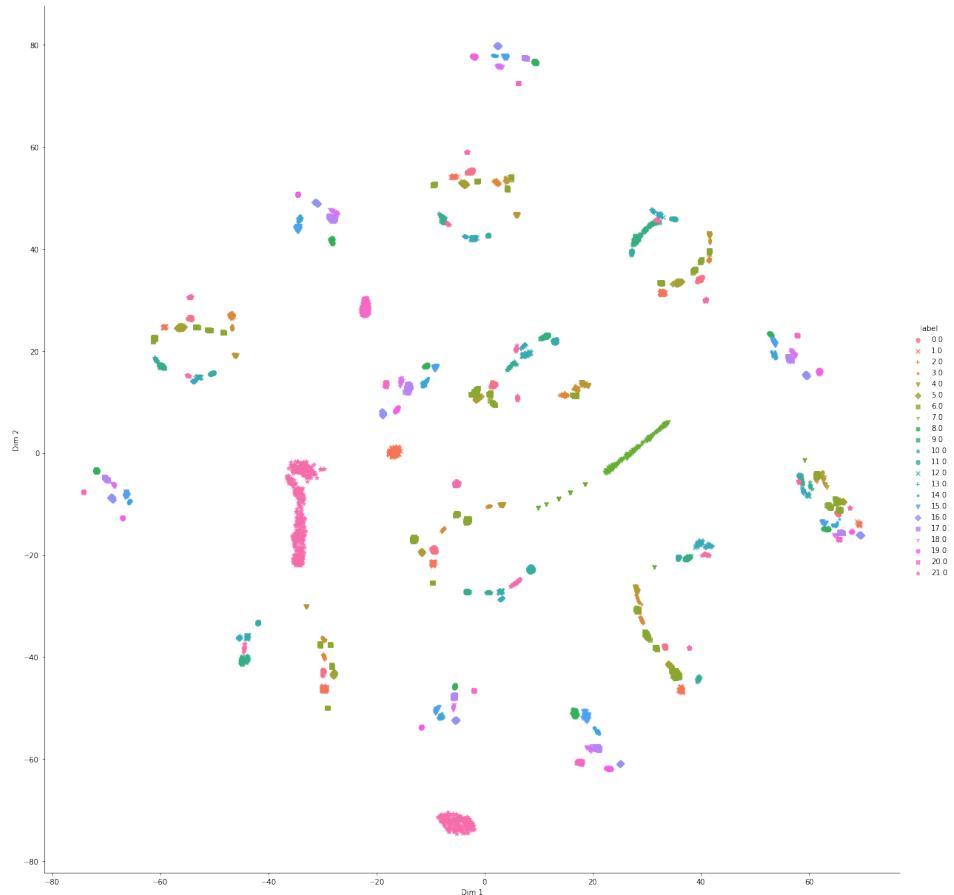


Figura 3.21: Visualización de los distintos tamaños de daño

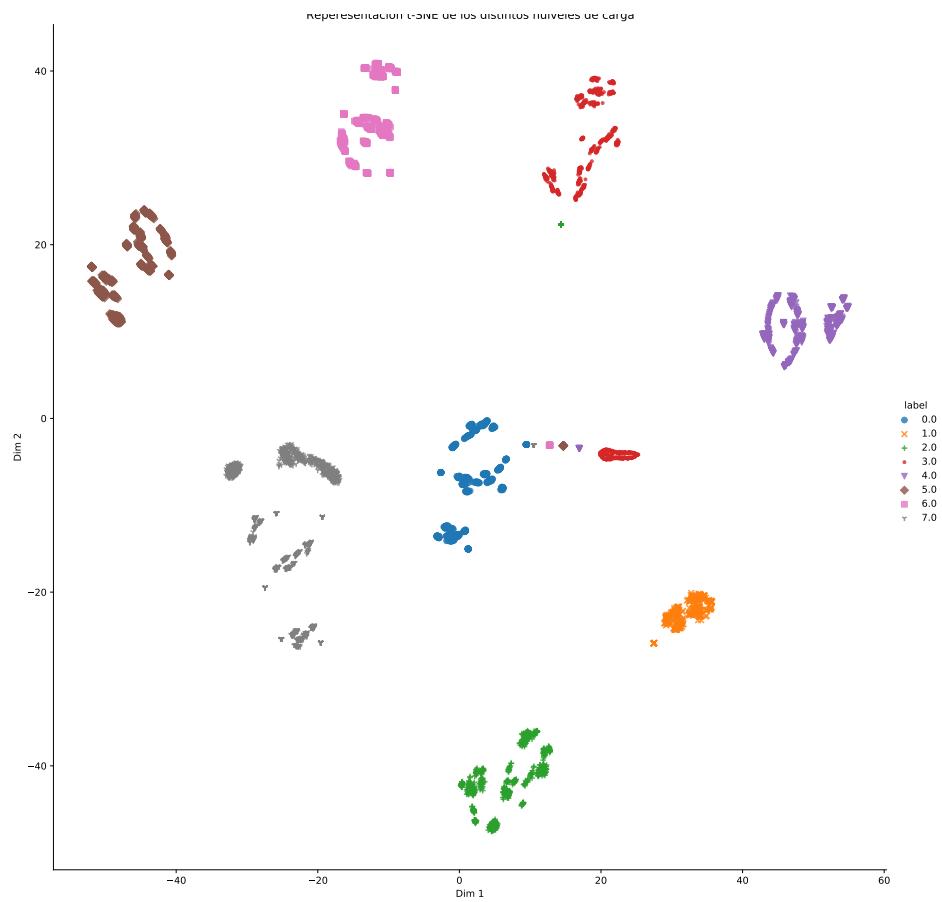


Figura 3.22: Visualización de los distintos niveles de carga

- **Sensores OBR**

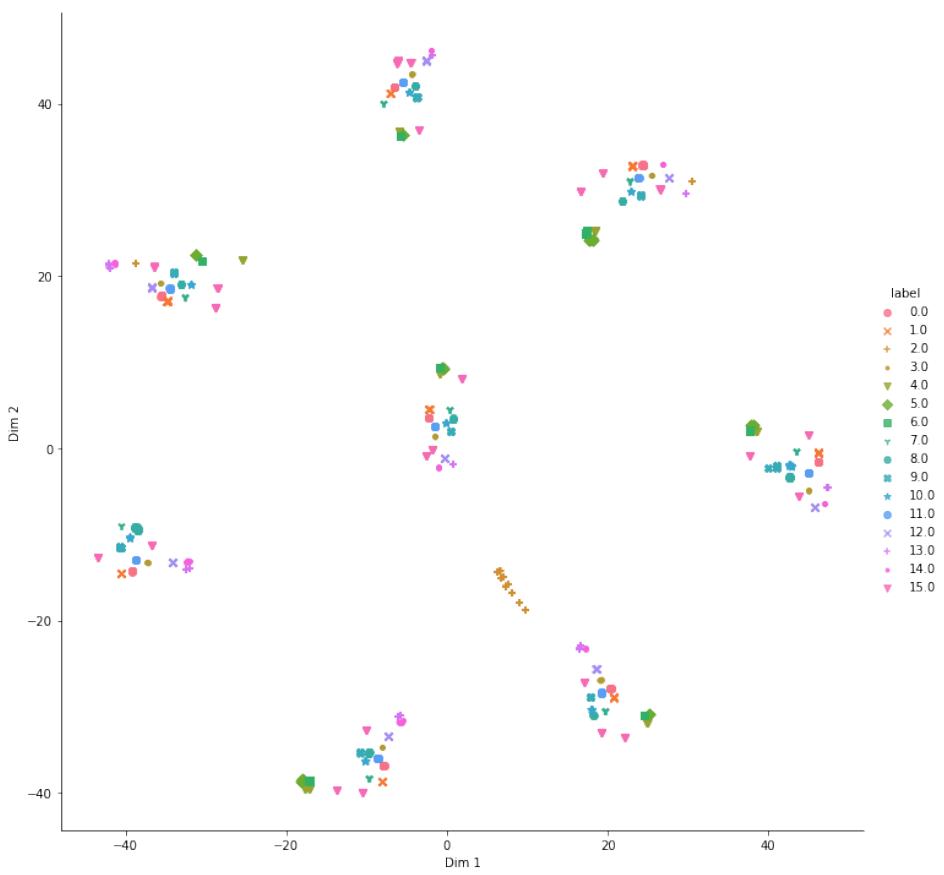


Figura 3.23: Visualización de los distintos tamaños de daño

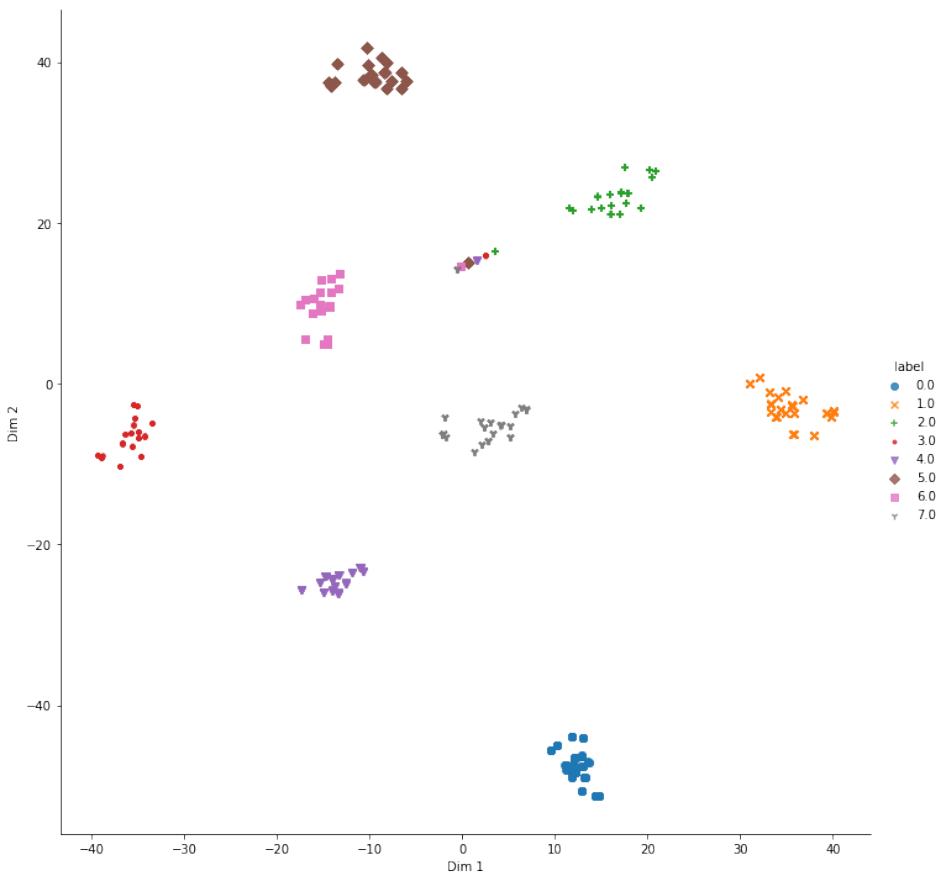


Figura 3.24: Visualización de los distintos niveles de carga

Aquí se puede ver claramente en las Figuras ?? y ?? que las muestras que están sometidas a un mismo nivel de carga están próximas en el espacio y separadas de los otros *clusters* que forman el resto de cargas. En las Figuras ?? y ?? también se ven pequeños clusters de cada tipo de daño repartidos en grupos más grandes correspondientes a los distintos niveles de carga. Por lo tanto, las muestras se separan en clusters de niveles de carga (claramente diferenciados) y dentro de cada carga, se agrupan por tipo y tamaño de daño.

### 3.3.2. Arquitectura de la Red Neuronal

En este apartado se tiene que explicar la arquitectura escogida. Tengo que encontrar una forma buena de pintarlo.

### 3.3.3. Resultados de la clasificación

Una buena forma de representar el funcionamiento de una NN en la tarea de clasificación es mediante el uso de Matrices de Confusión (CM).

En estas matrices, cada columna pertenece a una única clase. Al introducir un campo de deformaciones etiquetado como una clase en concreto (D01-R03), la red, después de realizar todas las operaciones definidas por su arquitectura, sacará una clase como output. Entonces, si se ha realizado una clasificación exitosa, la CM sumará en la columna D01-R03 y fila D01-R03 una unidad. Sin embargo, si clasifica erróneamente, sumará uno en una fila diferente.

Por lo tanto, cuanto mayor sea el valor de las celdas de la diagonal principal de la CM, mejor realizará la red la tarea de clasificación.

- Sensores FBG

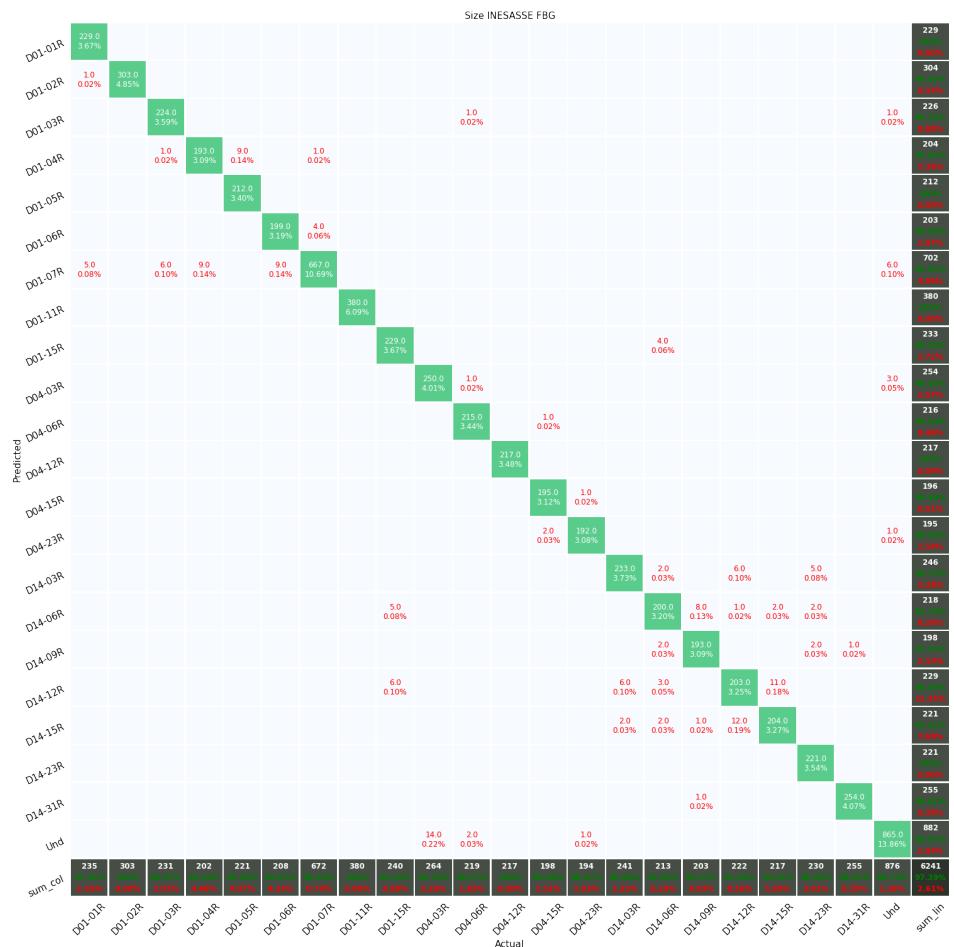


Figura 3.25: Matriz de confusión: Tamaños de daño



Figura 3.26: Matriz de confusión: Nivel de carga

- Sensores OBR

		Confusion matrix																	
		Actual																	
		D01-R03	D01-R07	D01-R11	D01-R15	D02-R03	D02-R09	D02-R11	D04-R06	D04-R15	D04-R23	D04-R31	D14-R06	D14-R15	D14-R23	D14-R31	Und	sum_col	
Predicted		85.0 4.76%	7.0 0.39%															92 92.39%	
D01-R03		1.0 0.06%	91.0 5.10%															7.61%	
D01-R07				97.0 5.43%														92 88.31%	
D01-R11					110.0 6.16%													1.09%	
D01-R15						83.0 4.65%	13.0 0.73%											97 100%	
D02-R03						14.0 0.78%	51.0 2.86%	24.0 1.34%										110 100%	
D02-R09									97.0 5.43%									96 96.41%	
D02-R11									79.0 4.43%									89 87.54%	
D04-R06										97.0 5.43%								79 100%	
D04-R15										97.0 5.43%	9.0 0.50%							97 100%	
D04-R23										109.0 6.11%								106 101.4%	
D04-R31										108.0 6.05%								109 100%	
D14-R06											97.0 5.43%							108 100%	
D14-R15											98.0 5.49%							97 100%	
D14-R23											101.0 5.66%							98 100%	
D14-R31												115.0 6.44%						104 97.11%	
Und													291.0 16.30%					120 95.83%	
																		291 4.17%	
																		1785 4.26%	

Figura 3.27: Matriz de confusión: Tamaños de daño

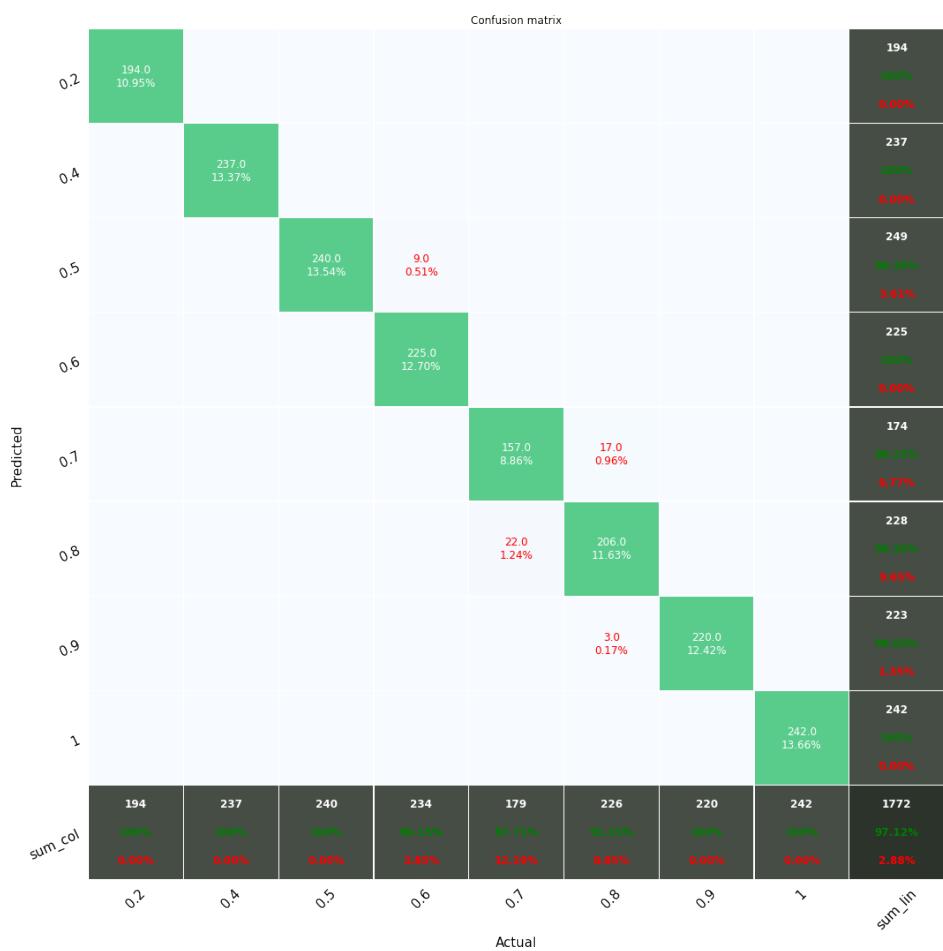


Figura 3.28: Matriz de confusión: Nivel de carga

### 3.4. Aplicación de DL al a la detección de cargas térmicas

En una estructura aeronáutica no solo los daños provocan cambios en el campo de deformaciones, la temperatura también es un factor importante a tener en cuenta. Una aeronave puede estar sometida 50°C en tierra y a -50°C durante un vuelo. Este amplio rango de temperaturas genera cargas y deformaciones de un valor equivalente a las aerodinámicas.

Cuando la temperatura no es uniforme puede generar concentración de tensiones equivalentes a las que provoca un daño localizado, por lo que su estudio es apropiado en SHM.

Ya que este trabajo consiste en una primera aproximación al uso de DL para la detección de cargas térmicas, solo se va a estudiar el caso de varias temperaturas uniformemente distribuidas.

#### 3.4.1. Estructura bajo estudio

Muchas de las estructuras aeronáuticas se pueden considerar vigas. Dependiendo de las cargas que vayan a soportar durante la operación, estas vigas estarán formadas por unos componentes u otros.

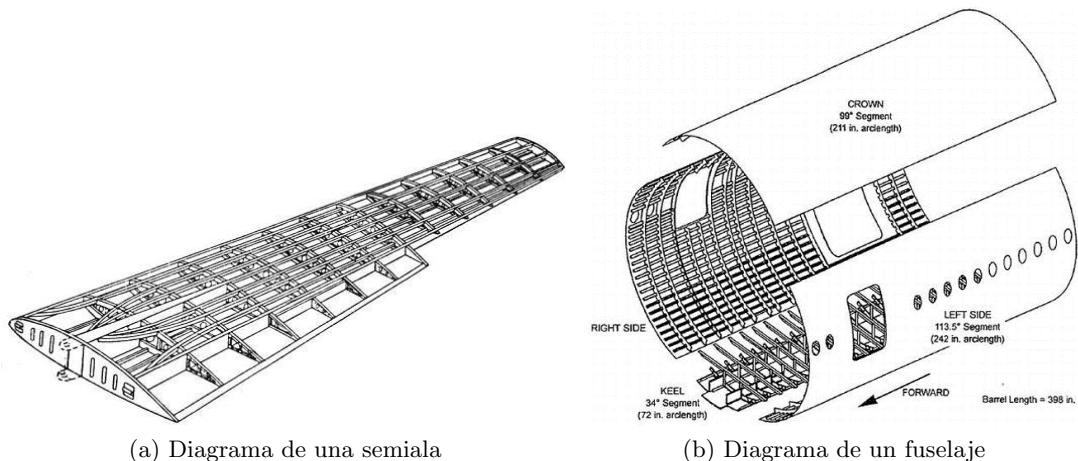


Figura 3.29: Diagramas de estructuras aeronáuticas

En la Figura ?? se puede comparar una semiala con un fuselaje. Las semialas están sometidas a flexión y torsión, por ello estás formadas por larguerillos y costillas, sin embargo el fuselaje no trabaja a torsión, por lo que no tiene costillas.

Este estudio se ha centrado en la influencia de diferentes temperaturas sobre una viga con forma de omega y 4 costillas equiespaciadas, igual a la que se ve en la Figura ??.

Para obtener las deformaciones necesarias con las que se alimentará a la red se ha creado un modelo de elementos finitos (FEM). Las partes que forman el conjunto son los siguientes:

- Tapa superior e inferior fabricada con laminado de 10mm
- Lateral izquierdo y derecho fabricada con laminado de 4mm
- Costillas fabricadas con laminado de 4mm



Figura 3.30: Imagen de la viga que se va a estudiar

Las características de los laminados se pueden ver en la Tabla ??

Espesor laminado	Esperor capa	Número de capas	Orientación
10	0.2	50	$[(\pm 45, 90, 0, 90), 0_{15}]_S$
4	0.2	20	$[(\pm 45, 90, 0_7]_S$

Tabla 3.2: Laminados

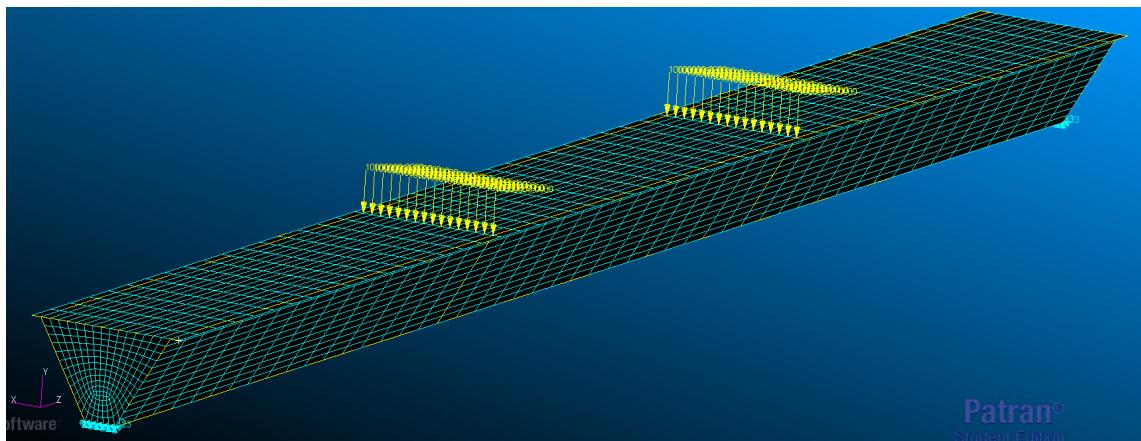


Figura 3.31: Viga con condiciones de contorno aplicadas

En la Figura ?? se puede ver la viga ya en Patran después de haber creado las superficies y elementos y haber aplicado las condiciones de contorno.

Las condiciones de contorno son las siguientes:

- Carga vertical sobre distribuida sobre la parte superior de las costillas centrales
- Empotramiento de los nodos correspondientes a la parte inferior de los laterales.

### 3.4.2. Definición de daños y sensores

En esta sección no solo se va a buscar clasificar la temperatura de una estructura, sino que también se le van a añadir dos tipos de daños típicos en estructuras de material compuesto.

Estos daños van a ser los siguientes:

- Despegue parcial de la tapa superior y uno de los laterales. Este daño se va a llamar D01
- Empotramiento de los nodos correspondientes a la parte inferior de los laterales.

Para obtener las deformaciones, se van a utilizar varias filas de elementos orientados en una misma dirección global. Con esto se consigue simular un sensor OBR similar al que se usó en el proyecto INESASSE. Se pueden ver los elementos que se han seleccionada para hacer de sensor OBR en la Figura ??.

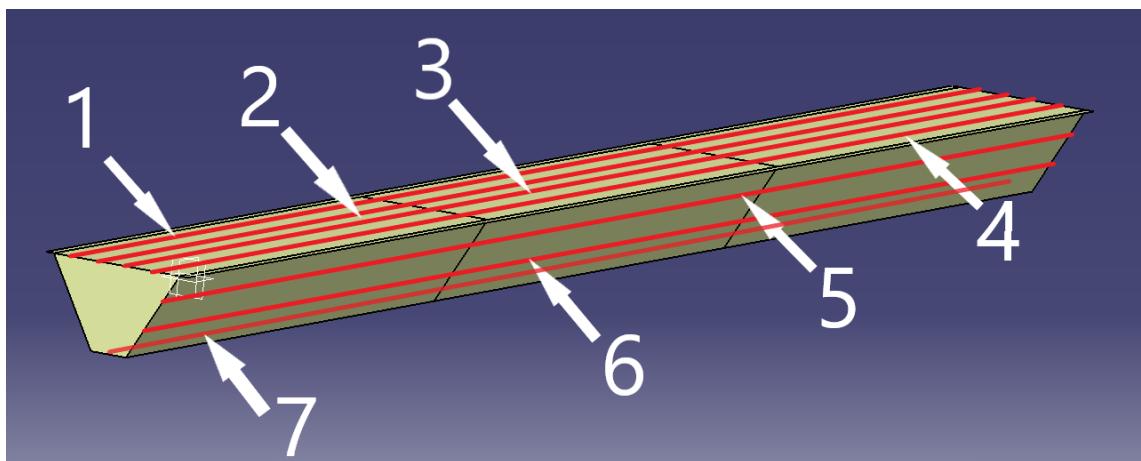


Figura 3.32: Tiras de sensores de los que se van a extraer las deformaciones

## Capítulo 4

# Caracterización de impactos con Deep Learning

Como se ha mencionado en la introducción, el objetivo de la segunda herramienta es la localización de impactos.

C. Aguilar en [Christian'TFM] utilizó algoritmos de triangulación para localizar impactos en placas de material compuesto. Para ello, recogió los diferentes tiempos de llegada de las ondas de Lamb provocadas por el impacto con cada uno de los piezoelectricos integrados en la estructura. Utilizando el campo de velocidades sobre la estructura y el tiempo de llegada a los diferentes piezoelectricos, se puede obtener la posición del impacto.

Los materiales compuestos tienen propiedades anisótropas, por lo que la velocidad de propagación de las ondas por el material no es igual en todas las direcciones. Esto hizo que en [Christian'TFM] se tuviera que calcular el campo de velocidades sobre la placa rigidizada.

La herramienta que se quiere desarrollar en este estudio busca localizar impactos en estructuras complejas fabricadas en material compuesto. En la Figura ?? se puede ver un ejemplo de este tipo de estructuras. Esta complejidad hace inviable la utilización de algoritmos de triangulación por la dificultad de calcular campo de velocidades, entonces se abre la puerta a la utilización de Deep Learning para este fin.

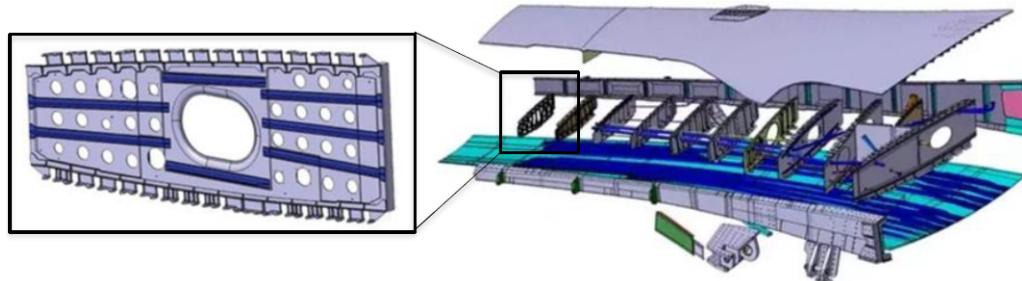


Figura 4.1: Costillas en una semiala del Airbus A380

La estructura que se va a utilizar en este estudio es un cuarto de costilla del Airbus A380. En la Figura ?? se puede apreciar la complejidad de la estructura con múltiples vaciados y rigidizadores. También se ven los piezoelectricos encargados de recoger las ondas de Lamb provocadas por los impactos.

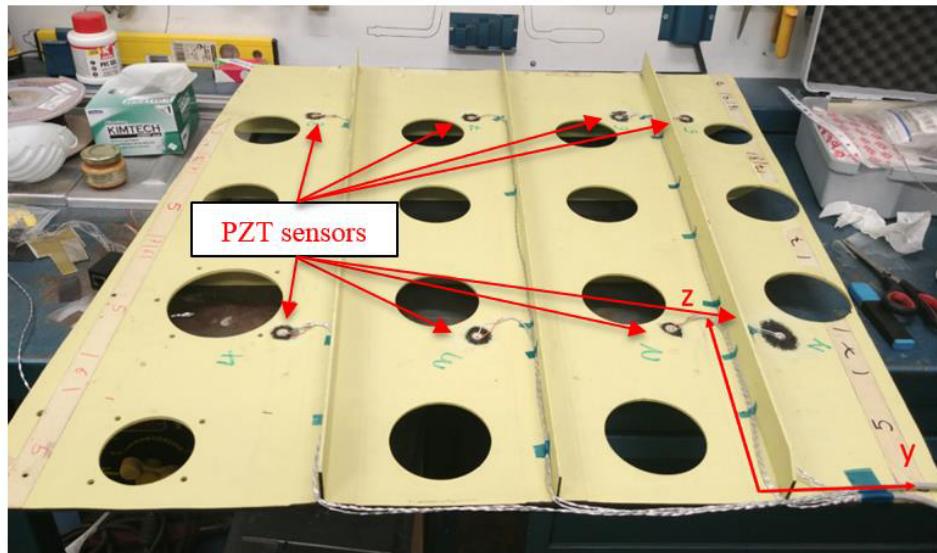


Figura 4.2: Estructura en estudio instrumentada

Como se ha visto anteriormente, las NN clasificadoras dan un resultado discreto, en forma de clases. Para que la red pueda clasificar, se ha discretizado la superficie de la costilla en diferentes celdas, cuyos índices serán considerados las clases. Esta discretización se puede ver en la Figura ?? junto con la posición de los piezoelectricos.

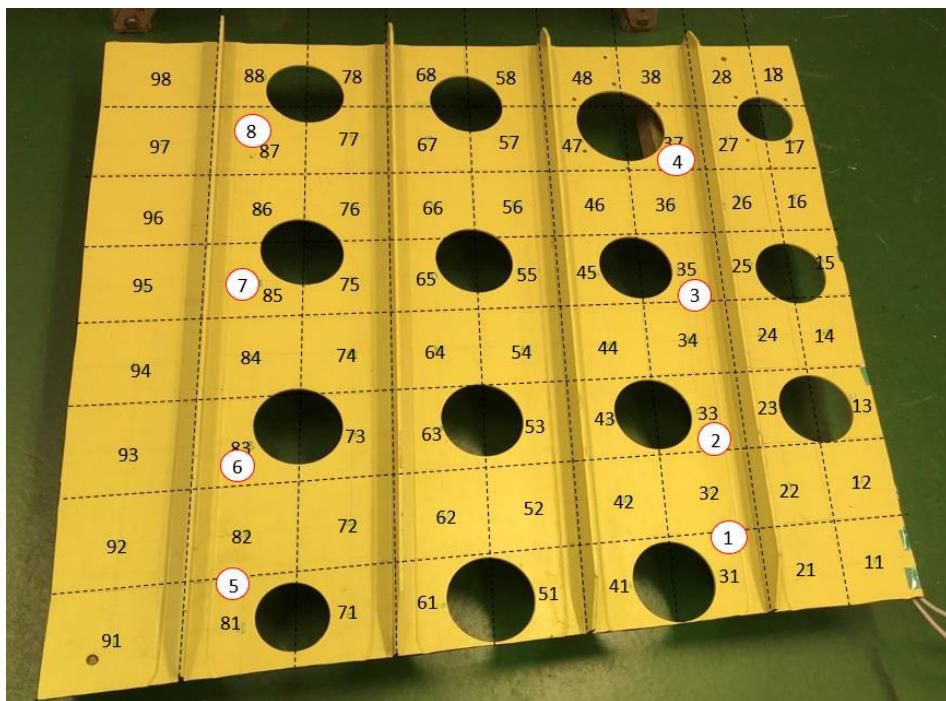


Figura 4.3: Discretización de la superficie de la costilla

Al igual que para las redes de detección de daño, se necesita un Data Set con una cantidad de muestras del orden de  $10^3$  por clase. El realizar una cantidad tan elevada de impactos de forma manual y precisa no es viable, lo que ha llevado a plantear dos métodos diferentes de obtención de impactos que se desarrollarán en este punto:

- Impactador automático
- Generación de impactos con redes tipo Generative Adversarial Networks (GANs)

Una vez se hayan conseguido los impactos, se creará una nueva red que tendrá como objetivo clasificar un impacto en las celdas de la costilla.

## **4.1. Impactador automático (SI TE PARECE BIEN)**

Una forma muy eficiente de realizar una tarea repetitiva es mediante su automatización, sobre todo si se requiere un nivel alto de precisión durante todo el proceso. Para entrenar a las redes neuronales es necesario una gran cantidad de impactos para cada celda, cuanta mayor cantidad de impactos e incluso, cuanta mayor dispersión dentro cada celda, mejor clasificará ante nuevos impactos que no han formado parte del proceso de entrenamiento de la red.

Para llevar a cabo el diseño se necesitan definir una serie de requisitos que el modelo final deberá cumplir.

### **4.1.1. Impactador estructura**

### **4.1.2. Electrónica - Software y hardware**

### **4.1.3. Adquisición y procesado de datos**

## 4.2. Generative Adversarial Networks

Muchos algoritmos de Machine Learning tienen como entrada datos complejos (una imagen, por ejemplo) y generan una salida simple (la clase avión). Sin embargo, el objetivo de un modelo generativo es completamente opuesto. A partir de una entrada simple, como puede ser un vector de números aleatorios, generar como salida una imagen realista de un avión.

Las Generative Adversarial Networks (GANs) son un tipo de modelo generativo muy efectivo, creado en el 2014 por Ian J. Goodfellow [GANs], que ha suscitado un gran interés en la comunidad de ML.

Un concepto importante de las GANs es que utiliza la aleatoriedad como instrumento creador. Esto hace que cuando introduzcas una entrada aleatoria nunca genere una imagen repetida, ya que hay muy pocas probabilidades de generar dos grupos de datos aleatorios iguales.

Pero igual de importante es que pensar en términos de probabilidades también nos ayuda a traducir el problema de la generación de imágenes en un marco matemático natural. Obviamente no se quieren elegir imágenes uniformemente al azar, ya que eso sólo produciría ruido. En su lugar, queremos que el sistema aprenda qué imágenes tienen probabilidades de ser caras, y cuáles no. Matemáticamente, esto implica modelar una distribución de probabilidad en las imágenes, es decir, una función que diga qué imágenes son probables de ser caras y cuáles no. Este tipo de problema se puede entender como modelar una función en un espacio de altas dimensiones y es exactamente el tipo de cosas para las que están hechas las redes neuronales.

La gran idea que define a una GAN es establecer este problema de modelización como una especie de concurso. De aquí viene la parte „adversaria” del nombre. La idea clave es construir no una, sino dos redes enfrentadas: un generador y un discriminador. El generador trata de crear salidas sintéticas aleatorias (por ejemplo, imágenes de rostros), mientras que el discriminador trata de diferenciarlas de las salidas reales (por ejemplo, una base de datos de famosos). El objetivo es que a medida que las dos redes se enfrenten, ambas mejoren cada vez más, con el resultado final de una red generadora que produzca salidas realistas.

En resumen: Las redes generadoras adversarias son redes neuronales que aprenden a elegir muestras de una distribución especial (la parte „generativa” del nombre), y lo hacen estableciendo una competencia (por lo tanto „adversaria”).

Esta idea de dos redes enfrentadas se puede ver en la Figura ??

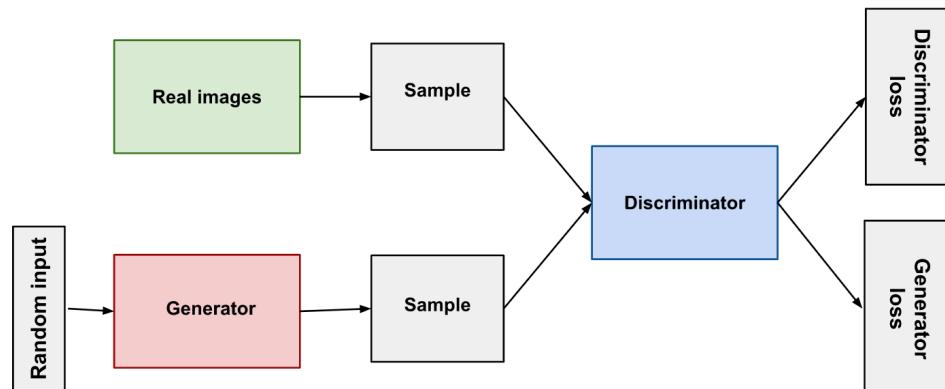


Figura 4.4: Representación simplificada de la arquitectura de una red GAN

## Descripción extraída de [GAN\*lab]

En este estudio no se tiene como objetivo crear imágenes realistas, sino impactos que sean similares a los que se recogieron en [Christian\*TFM]. Para entender mejor la forma de los datos que se van a utilizar se han representado varias muestras en la Figura ???. Cada muestra consta de las medidas de tensión (voltios) recogidas por los ocho piezoelectricos en una ventana de tiempo de 0.0125 segundos. Este tipo de datos se puede clasificar como series de datos temporales multivariable.

Si se presta atención a las subfiguras ?? y ?? puede apreciarse que, aun siendo un impacto en la misma celda, las mediciones son muy dispares. Recuperando la idea de que una parte esencial en las GANs es que se generan los nuevos impactos a partir de datos aleatorios, el que haya variación dentro de impactos en una misma celda facilita a la red el aprendizaje y, por lo tanto, el generar impactos más similares a los reales.

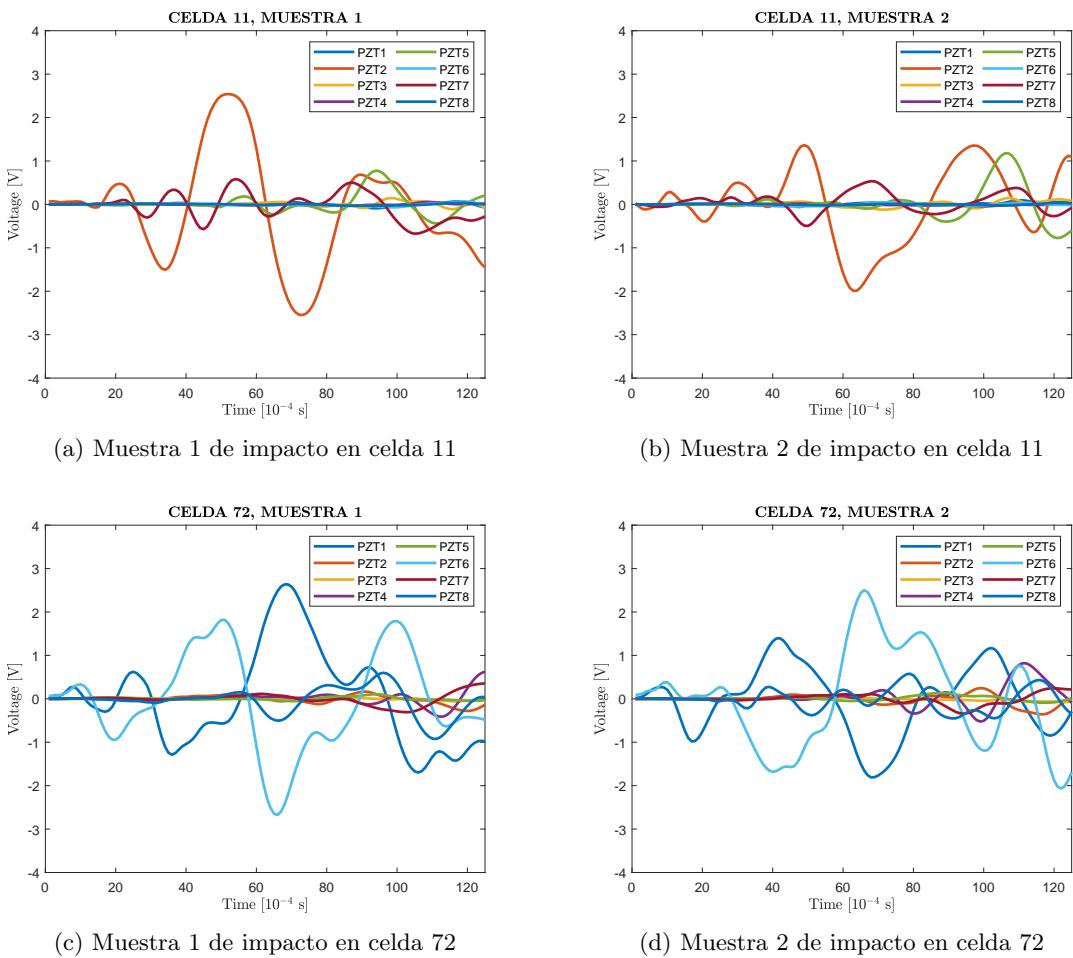


Figura 4.5: Diferentes ejemplos de impactos

### 4.2.1. Modelo generativo de secuencias temporales

¿Qué es un buen modelo generativo para datos de series temporales? El entorno temporal plantea un desafío único para el modelado generativo. Un modelo no sólo tiene la tarea de capturar las distribuciones de los rasgos dentro de cada punto temporal, sino

que también debe capturar la dinámica potencialmente compleja de esas variables a lo largo del tiempo. Específicamente, en el modelado de datos secuenciales multivariable  $x_{1:T} = (x_1; \dots; x_T)$ , también deseamos capturar con precisión la distribución condicional  $p(x_t|x_{1:t_1})$  de las transiciones temporales.

Desde el 2014, gran parte del trabajo se ha centrado en mejorar la dinámica temporal de los modelos autorregresivos para la predicción de secuencias. Éstos abordan principalmente el problema de los errores de composición durante el muestreo de múltiples etapas, introduciendo diversas modificaciones en el momento de su formación para reflejar con mayor precisión las condiciones del tiempo de validación. Los modelos autorregresivos factorizan explícitamente la distribución de las secuencias en un producto de los condicionales  $\Pi_{tp}(x_t|x_{1:t_1})$ . Sin embargo, aunque es útil en el contexto de la predicción, este enfoque es fundamentalmente determinista y no es verdaderamente generativo en el sentido de que se pueden tomar muestras aleatorias de nuevas secuencias a partir de ellas sin condicionamientos externos. Por otra parte, otra línea de trabajo se ha centrado en la aplicación directa del marco de las GANs a los datos secuenciales, principalmente mediante la instanciación de redes recurrentes para las funciones de generador y discriminador. Aunque es sencillo, el objetivo adversario busca modelar  $p(x_{1:t_1})$  directamente, sin apalancar el previo autorregresivo. Es importante, simplemente sumar la función de costes (loss) estándar de la GAN sobre las secuencias de vectores puede no ser suficiente para asegurar que la dinámica de la red capte eficientemente las dependencias escalonadas presentes en los datos de entrenamiento.

Las Redes Generativas Adversas de Series Temporales (TimeGAN) proponen un novedoso mecanismo para unir ambos hilos de la investigación, dando lugar a un modelo generativo explícitamente entrenado para preservar la dinámica temporal. TimeGAN crea, un marco natural para generar datos de series temporales realistas en varios dominios.

#### 4.2.2. Arquitectura TimeGAN

Como ya se ha mencionado al inicio de este capítulo, las GANs están constituidas principalmente por un generador y un discriminador. Sin embargo, la arquitectura de TimeGAN varía ligeramente de esta generalización con el objetivo de poder retener la dinámica de los datos temporales.

TimeGAN está formada por cuatro componentes: una función de integración, función de recuperación, generador de secuencia y discriminador de secuencia.

La idea clave es que los componentes de autocodificación (los dos primeros) se entrena conjuntamente con los componentes adversarios (los dos últimos), de modo que TimeGAN aprende simultáneamente a codificar características, generar representaciones e iterar a través del tiempo. La red de integración proporciona el espacio latente, la red adversaria opera dentro de este espacio y la dinámica latente de los datos reales y sintéticos se sincronizan a través de una pérdida supervisada (función de costes). Describimos cada uno a su vez.

- **Funciones de integración y recuperación**

Las funciones de integración y recuperación proporcionan asignaciones entre la característica y el espacio latente, lo que permite a la red adversaria aprender la dinámica temporal subyacente de los datos a través de representaciones de dimensiones más bajas.

Sean  $H_S$  y  $H_X$  los espacios vectoriales latentes correspondientes a los espacios de las características  $S$  y  $X$ . Entonces la función de integración  $e : S \times \Pi_t X \rightarrow H_S \times \Pi_t H_X$

lleva las características estáticas y temporales a su estado latente  $h_S, h_{1:T} = e(s, x_{1:T})$ .  $e$  se ha implementado mediante una RNN.

$$\mathbf{h}_S = e_S(s), \quad \mathbf{h}_t = e_X(\mathbf{h}_S, \mathbf{h}_{t-1}, \mathbf{x}_t) \quad (4.1)$$

donde  $e_S : S \rightarrow H_S$  es una red integradora para variables estáticas, y  $e_X : H_S \times H_X \times X \rightarrow H_X$  una red recurrente integradora para características temporales. En la dirección opuesta, la función recuperadora  $r : H_S \times \Pi_t H_X \rightarrow S \times \Pi_t X$  coge las características latentes y las devuelve al espacio de características  $\tilde{s}, \tilde{x}_{1:T} = r(\mathbf{h}_S, \mathbf{h}_{1:T})$ . Aquí se implementa  $r$  usando una red *feedforward* en cada paso

$$\tilde{s} = r_S(\mathbf{h}_S), \quad \tilde{x}_t = r_X(\mathbf{h}_t) \quad (4.2)$$

donde  $r_S : H_S \rightarrow S$  y  $r_X : H_X \rightarrow X$  son redes de recuperación para las integraciones estáticas y temporales.

#### • Generador y discriminador de secuencias

En vez de generar datos sintéticos directamente en el espacio de características, el generador primero crea su salida en el espacio de integración. Sean  $Z_S$  y  $Z_X$  los espacios vectoriales sobre los que sabemos que las distribuciones están definidas y de donde se extraen los vectores aleatorios como entrada para generar  $H_S$  y  $H_X$ . Entonces la función de generación  $g : Z_S \times \Pi_t Z_X \rightarrow H_S \times \Pi_t H_X$  lleva una pareja de vectores aleatorios estáticos y temporales al espacio latente sintético  $\hat{\mathbf{h}}_S, \hat{\mathbf{h}}_{1:T} = g(\mathbf{z}_S, \mathbf{z}_{1:T})$ . Se implementa  $g$  con una RNN.

$$\hat{\mathbf{h}}_S = g_S(\mathbf{z}_S), \quad \hat{\mathbf{h}}_t = g_X(\hat{\mathbf{h}}_S, \hat{\mathbf{h}}_{t-1}, \mathbf{z}_t) \quad (4.3)$$

donde  $g_S : Z_S \rightarrow H_S$  es una red generadora de características estáticas y  $g_X : H_S \times H_X \times Z_X \rightarrow H_X$  es un generador recurrente para características temporales. Vectores aleatorios  $\mathbf{z}_S$  pueden ser muestreados desde una distribución cualquiera y  $\mathbf{z}_t$  sigue un proceso estocástico. Aquí se usa una distribución y un proceso Winer respectivamente. Finalmente el discriminador también opera en el espacio integrado. La función discriminatoria  $d : H_S \times \Pi_t H_X \rightarrow [0, 1] \times \Pi_t [0, 1]$  recibe las características temporales y estáticas, devolviendo la clasificación  $\tilde{y}_S, \tilde{y}_{1:T} = d(\tilde{\mathbf{h}}_S, \tilde{\mathbf{h}}_{1:T})$ . La notación  $\tilde{\mathbf{h}}$  refiere a datos reales ( $h$ ) como sintéticos ( $\hat{h}$ ) en el espacio vectorial de integración, al igual que  $\tilde{y}$  indica clasificación real ( $y$ ) como sintética ( $\hat{y}$ ). Aquí se implementa  $d$  a partir de una RNN bidimensional con una capa *feedforward* en la salida

$$\tilde{\mathbf{y}}_S = d_S(\tilde{\mathbf{h}}_S), \quad \tilde{\mathbf{y}}_t = d_X(\tilde{\mathbf{u}}_t, \tilde{\mathbf{u}}_t) \quad (4.4)$$

donde  $\tilde{\mathbf{u}}_t = \tilde{c}_X(\tilde{\mathbf{h}}_S, \tilde{\mathbf{h}}_t, \tilde{\mathbf{u}}_{t-1})$  y  $\tilde{\mathbf{u}}_t = \tilde{c}_X(\tilde{\mathbf{h}}_S, \tilde{\mathbf{h}}_t, \tilde{\mathbf{u}}_{t+1})$  respectivamente denotan las secuencias hacia adelante y atrás de los *hidden states*,  $\tilde{c}_X$  y  $\tilde{c}_X$  son funciones recurrentes y  $d_S, d_X$  son las funciones de clasificación de la última capa.

El diagrama de bloques del funcionamiento de TimeGAN se puede ver en la Figura ??

Todo lo comentado sobre TimeGAN en este trabajo ha sido extraído de **[TimeGAN]**. Aquí se puede encontrar tanto el paper de su publicación como el código necesario para la implementación de esta red.

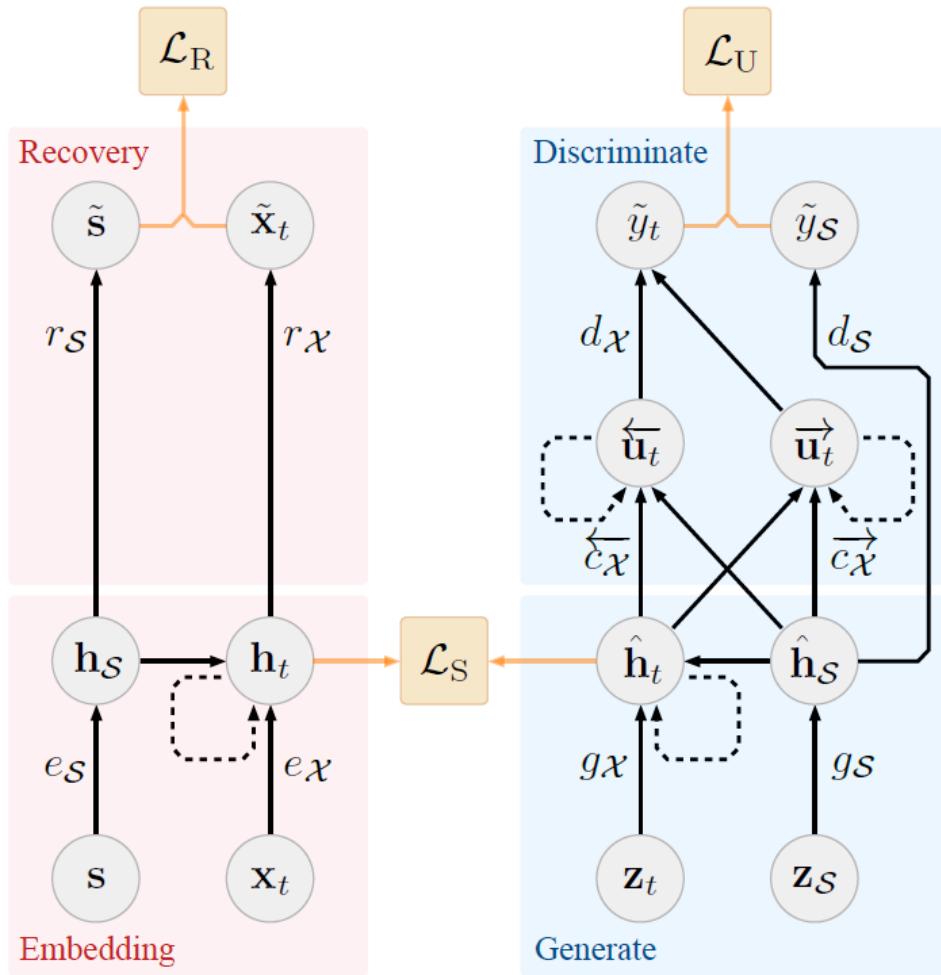


Figura 4.6: Diagrama de bloques del funcionamiento de TimeGAN

#### 4.2.3. Resultados de la generación de impactos con TimeGAN

Antes de comenzar con el proceso de generación de impactos es necesario definir una serie de parámetros de arquitectura y del proceso de entrenamiento.

- Tipo de capa recurrente: GRU
- Número de neuronas en las capas ocultas: 64
- Número de épocas en el entrenamiento: 400

Con los parámetros ya definidos se puede comenzar el proceso. Es importante tener en cuenta que se van a generar los impactos de forma individual e independiente. Esto quiere decir que se introducirán a la red todos los impactos reales de, por ejemplo, la celda 11 y una vez que se haya entrenado la TimeGAN se generarán 1500 impactos sintéticos. Después se borrará todo el aprendizaje que la red haya generado y se introducirán los impactos de la siguiente celda y volverá a repetirse el ciclo.

En la Figura ?? se puede ver varios impactos sintéticos que coinciden con las celdas elegidas en la Figura ???. Si se comparan visualmente se podrían confundir con un impacto real sin ninguna duda.

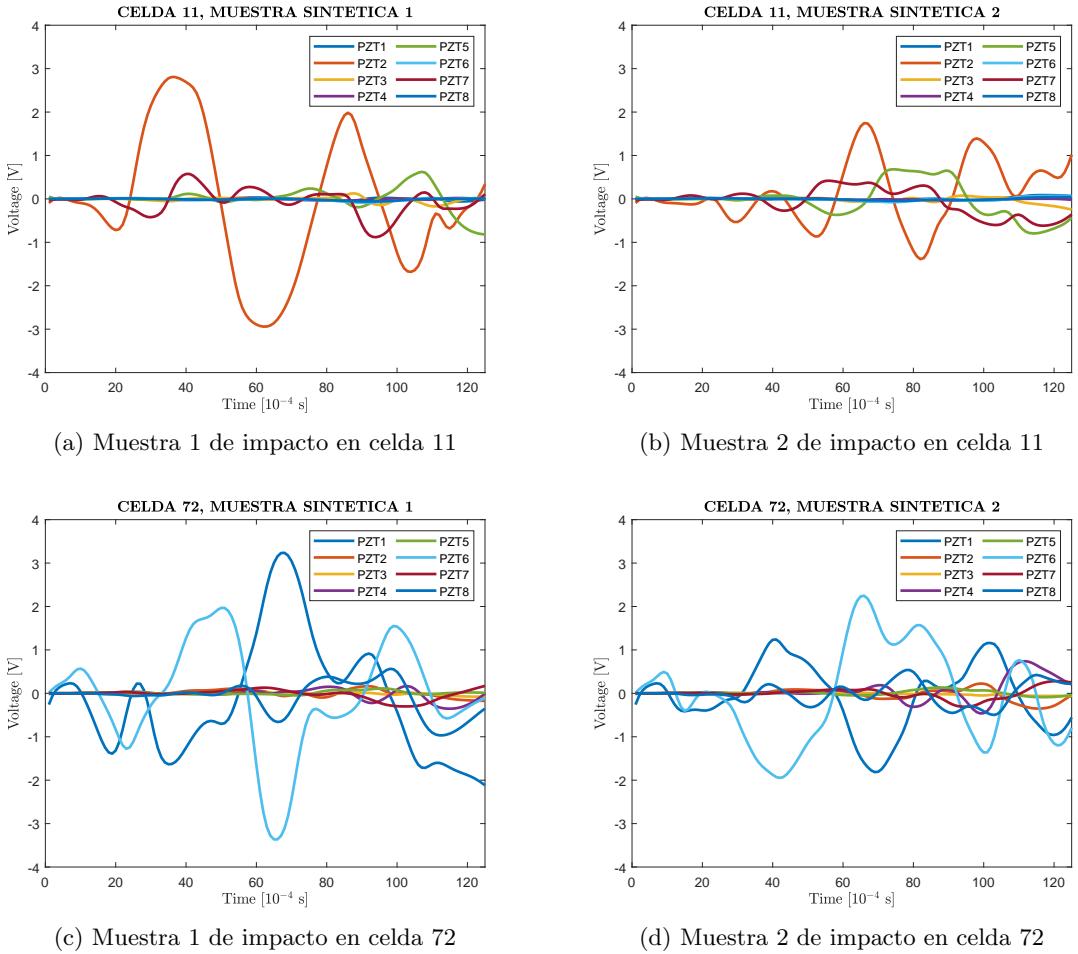


Figura 4.7: Diferentes ejemplos de impactos generados sintéticamente con TimeGAN

Pero para tener un nivel alto de confianza en que los nuevos datos que se han generado sintéticamente son ciertamente equivalentes a los impactos reales se ha decidido utilizar el algoritmo t-SNE de visualización.

En el capítulo 3, este algoritmo demostró que los daños similares se agrupaban en grupos o *clusters* independientes, por lo tanto, para poder considerar que los impactos reales y generados son similares dentro de una celda y diferentes al resto de celdas lo que se espera al ejecutar el algoritmo es lo siguiente:

- cada celda esté en un cluster independiente
- los impactos reales y sintéticos de una misma celda estén distribuidos de forma uniforme en el cluster correspondiente

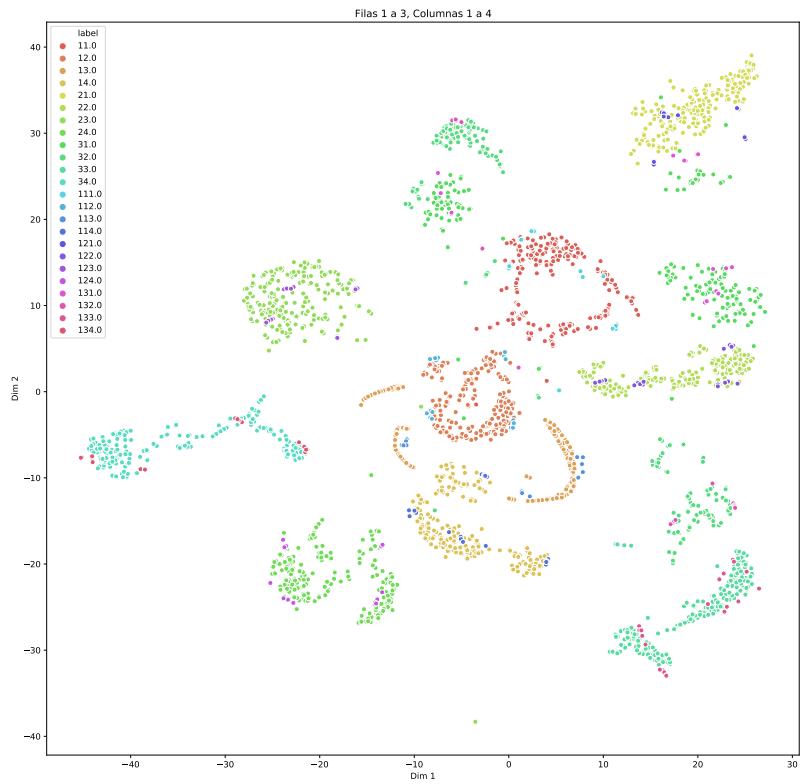


Figura 4.8: t-SNE de las celdas 11, 12, 13, 14, 21, 22, 23, 24, 31, 32, 33, 34

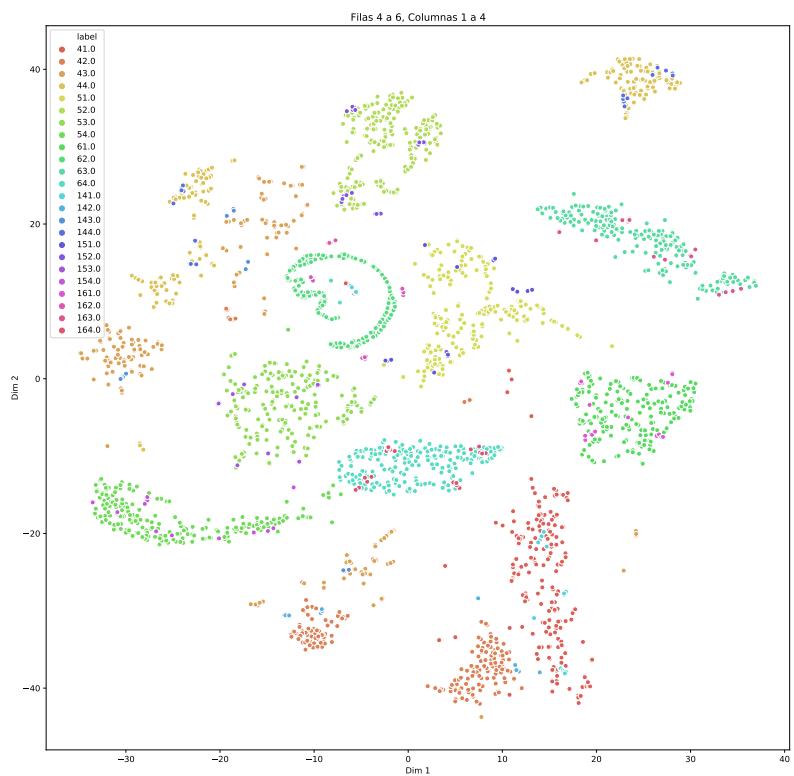


Figura 4.9: t-SNE de las celdas 41, 42, 43, 44, 51, 52, 53, 54, 61, 62, 63, 64

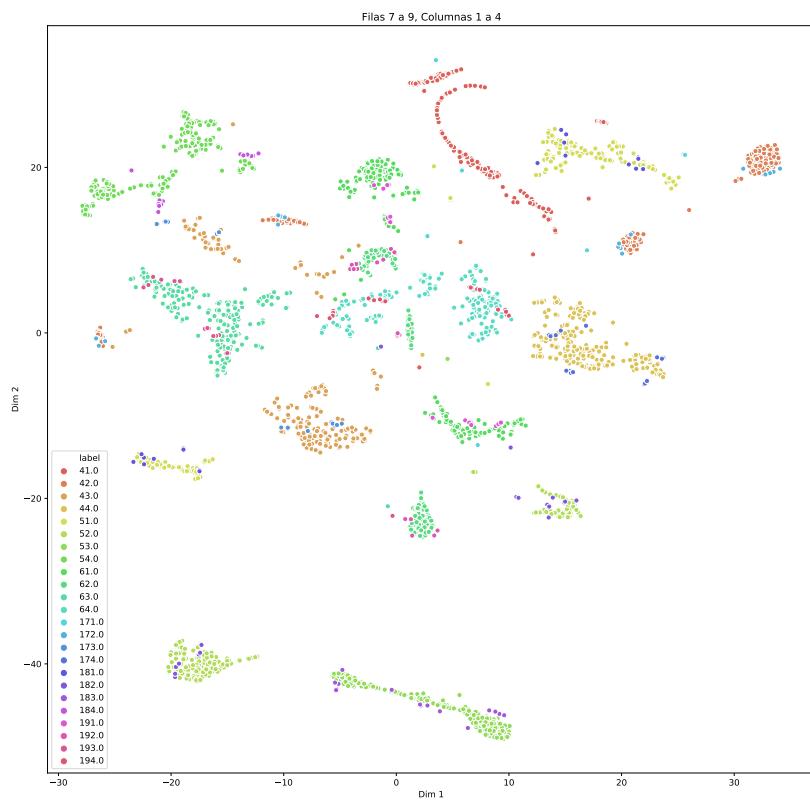


Figura 4.10: t-SNE de las celdas 71, 72, 73, 74, 81, 82, 83, 84, 91, 92, 93, 94

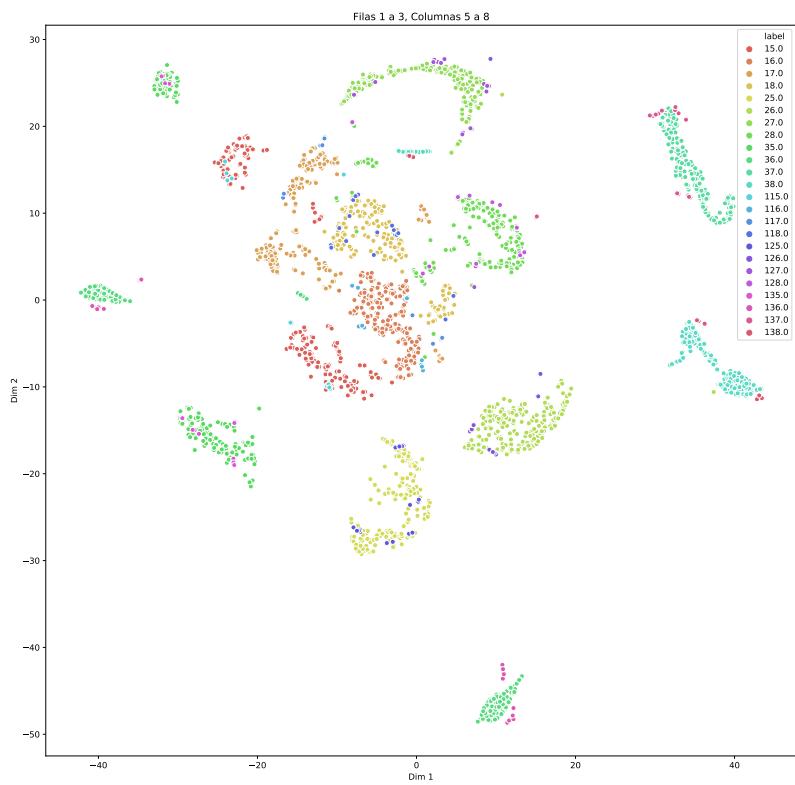


Figura 4.11: t-SNE de las celdas 15, 16, 17, 18, 25, 26, 27, 28, 35, 36, 37, 38

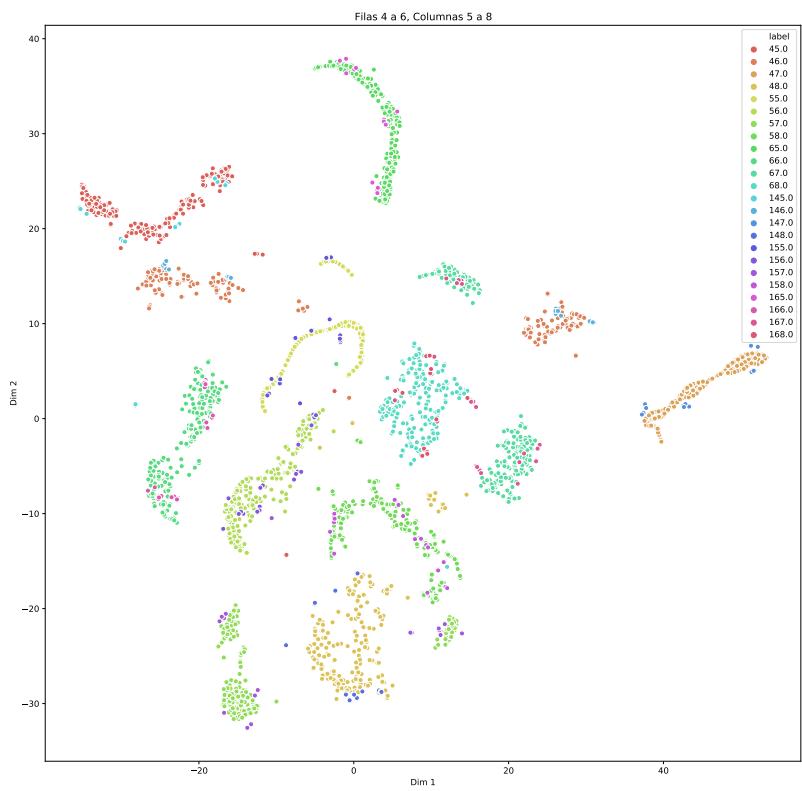


Figura 4.12: t-SNE de las celdas 45, 46, 47, 48, 55, 56, 57, 58, 65, 66, 67, 68

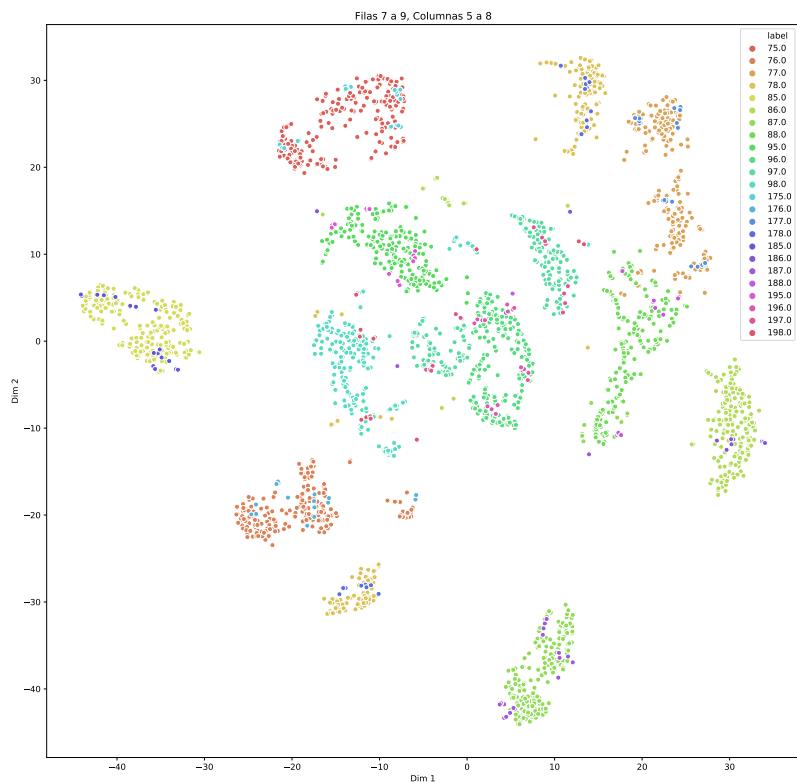


Figura 4.13: t-SNE de las celdas 75, 76, 77, 78, 85, 86, 87, 88, 95, 96, 97, 98

De la Figura ?? a ?? se puede ver la representación de los impactos en las diferentes celdas. Al haber 72 celdas en total, se ha decidido realizar la visualización en grupos de 3x4. De este modo se puede entender de forma más sencilla las relaciones entre datos sintéticos y reales al haber menos puntos por figura.

En la leyendas hay dos tipos de datos diferentes, xx y 1xx. Los xx corresponden a los impactos sintéticos y los 1xx son sus homólogos reales.

En la Figura ?? se puede ver con claridad que los puntos correspondientes a la celda 34 están distribuidos entre dos agrupaciones pequeñas de los puntos 134. Esto quiere decir que los impactos sintéticos son similares a los reales y, a su vez, diferentes al resto de impactos.

Con este resultado se puede considerar que TimeGAN ha generado unos impactos similares a los reales y que se pueden utilizar para alimentar a una red clasificadora.

## 4.3. Caracterización de la posición

### 4.3.1. Arquitectura del clasificador

Falta preparar el modelo de arquitectura para que se vea bien.

### 4.3.2. Resultados

Como la cantidad de impactos reales que se tienen es muy reducida, se ha decidido entrenar al clasificador únicamente con los impactos generados sintéticamente. Esto permite comprobar también el funcionamiento de un clasificador cuando los datos de entrenamiento y validación tienen cierta dispersión.

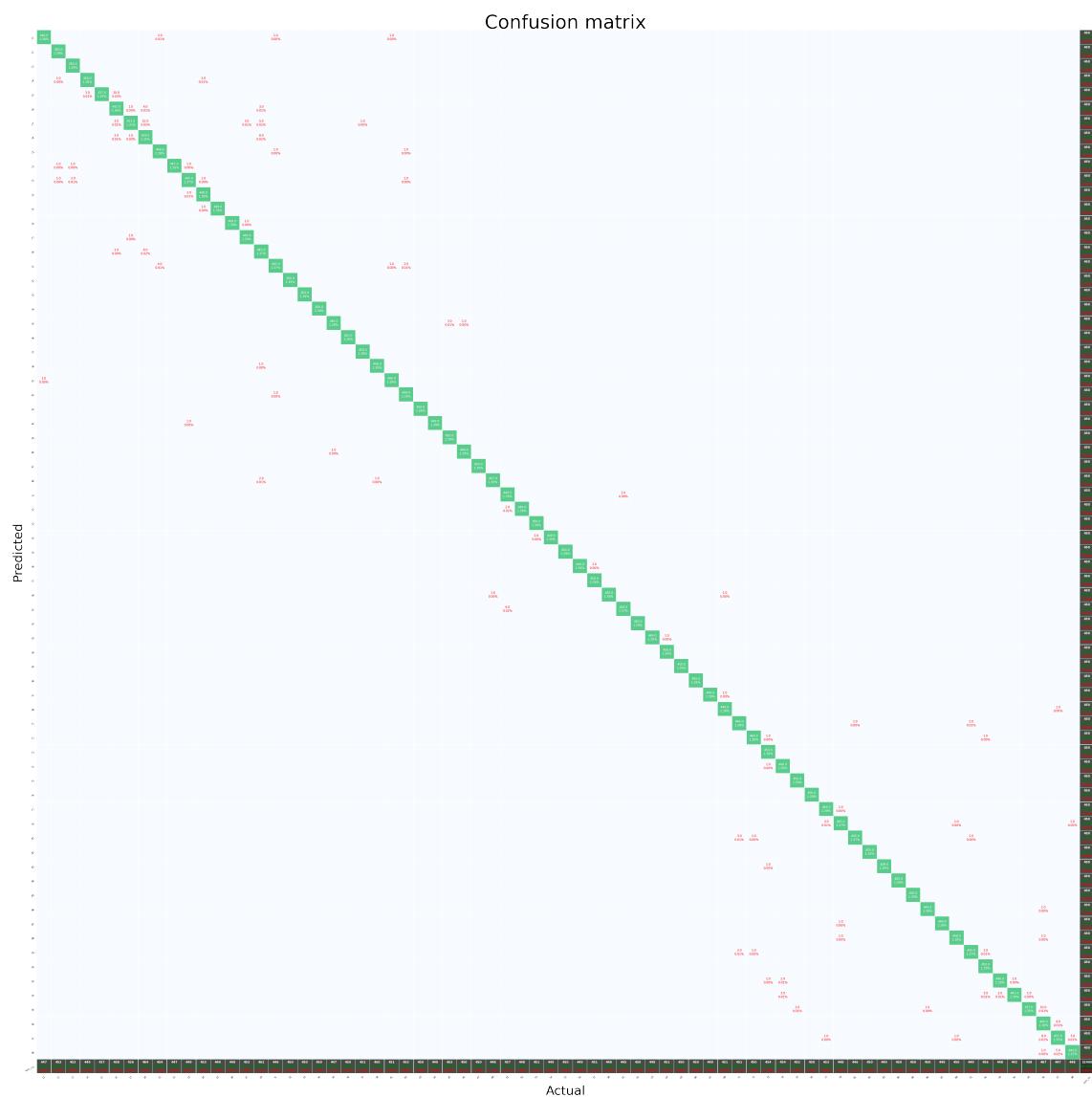


Figura 4.14: Matriz de confusión con los impactos sintéticos:  
**PRECISIÓN 99,38 %**

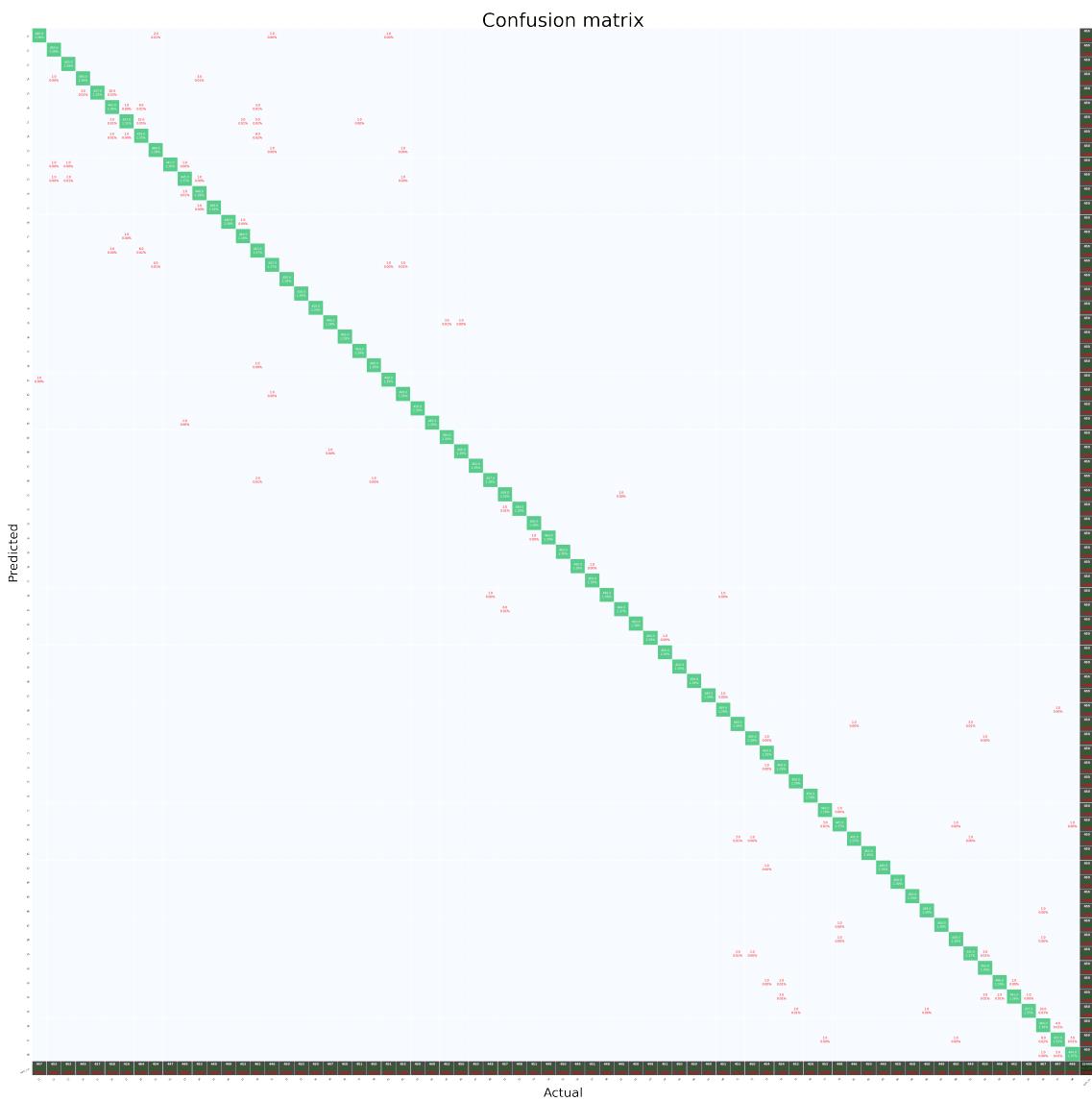


Figura 4.15: Matriz de confusión con los impactos reales: **PRECISIÓN  
86,86 %**

En la Figura ?? se puede comprobar que cuando entrenas una red con un determinado grupo de datos y lo validas con datos extraídos de la misma fuente, su precisión puede llegar a ser tan alta como la que se ha conseguido aquí.

Por otro lado, cuando se valida con un grupo de datos diferente al que se ha entrenado, su precisión cae. Sin embargo, en la Figura ?? se muestra que la red entrenada con los datos sintéticos es capaz de clasificar con una precisión alta los datos reales.

#### **4.4. Caracterización de la energía**

CON DATOS DE CHRISTIAN Y AIRBUS AL MENOS  
Red que clasifica la energía de impacto.

#### **4.5. Caracterización de la velocidad**

Si se termina por realizar el impactador, también se buscará una clasificación de velocidad da impacto para diferentes masas y misma altura de suelta.

## **4.6. Caracterización completa de un impacto**

Combinación de los anteriores clasificadores para comprobar la precisión completa de un impacto.

## **Capítulo 5**

# **Conclusiones y trabajos futuros**

### **5.1. Conclusiones**

al menos 3-4 págs

## **5.2. Trabajos futuros**

1 pag qewrg weqwewe [**einstein**]