UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA MECÁNICA

DETECCIÓN DE FALLAS EN EQUIPOS UTILIZANDO MODELOS EN BASE A DEEP LEARNING

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL MECÁNICO

SEBASTIÁN MONTAGNA PUGA

PROFESOR GUÍA:
ENRIQUE LÓPEZ DROGUETT

MIEMBROS DE LA COMISIÓN:
VIVIANA MERUANE NARANJO
JOSÉ CARDEMIL IGLESIAS

SANTIAGO DE CHILE
2018

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE INGENIERO CIVIL MECÁNICO
POR: SEBASTIÁN MONTAGNA PUGA
FECHA: 2018
PROF. GUÍA: ENRIQUE LÓPEZ DROGUETT

DETECCIÓN DE FALLAS EN EQUIPOS UTILIZANDO MODELOS EN BASE A DEEP LEARNING

Los equipos mecánicos están sujetos a daño durante la operación, lo que deteriora su estructura y funcionamiento produciendo fallas. La detección preventiva de fallas y el pronóstico de vida remanente son herramientas muy útiles en el ámbito práctico, permitiendo evitar tiempos inesperados de parada del equipo, además de permitir agendar la mantención en un momento propicio según la condición en la que se encuentre el equipo en operación.

Se propone implementar un modelo novedoso para el análisis de registros de series temporales en base a Deep Learning, redes neuronales convolucionales causales, que ha presentado muy buenos resultados realizando tareas de generación de secuencias con dependencias de largo alcance [1]. Los objetivos del trabajo propuesto en el presente informe son los siguientes:

Objetivo General:

- *Determinar la vida remanente en equipos mecánicos mediante la implementación de un modelo en base a CNNs causales.*

Objetivos Específicos:

- *Analizar, indexar y clasificar los registros de señales de sensores de los equipos pertinentes.*
- *Generar un modelo en base a redes neuronales convolucionales causales para el pronóstico y estimación de vida remanente.*
- *Verificar y corroborar resultados obtenidos comparando con métodos actuales y particularmente métodos en base a Long Short-Term Memory.*

Teniendo la base de datos del registro de los equipos, se procede a definir la arquitectura del modelo en base a Deep Learning y a realizar el entrenamiento e implementación del modelo. Luego, se analizan y verifican los resultados. En caso de que los resultados no sean satisfactorios se procede a cambiar los hiper-parámetros de la arquitectura del modelo y se repite el procedimiento.

Los resultados obtenidos validan la implementación del modelo propuesto por medio de métodos comparativos entre modelos con y sin los métodos que se busca implementar.

Los valores obtenidos para las predicciones de la base de datos en la que se implementa el modelo responden a lo esperado y al comparar con el estado del arte, se puede notar que el modelo realiza buenas predicciones, no ajustándose con tanta precisión, pero obteniendo mejores resultados en las bases de datos con más parámetros de operación debido a la capacidad de aprendizaje más general.

A mis padres, los conocidos que marcaron mi rumbo y las personas que permitieron que se pudiera realizar este trabajo.

Tabla de Contenido

1 Introducción

Equipos utilizados en ingeniería sufren varios tipos de daño durante la operación, los diferentes mecanismos de daño producen efectos diversos en la estructura del equipo, alterando sus características y desempeño [2], lo que puede desencadenar potenciales fallas. La capacidad de realizar pronósticos sobre el estado de equipos mecánicos es una herramienta relevante en la gestión de los activos físicos ya que permite evitar imprevistos durante la operación, resguardar la seguridad de operadores y reducir los costos de operación y mantención.

En el presente informe se exponen algunos de los métodos que se utilizan para realizar el monitoreo de equipos y se identifican los desafíos que plantea la implementación de nuevos métodos. Además, se expone y desarrolla un método que actualmente presenta buenos resultados en estudios aplicados a tareas similares y que se propone como alternativa frente a los métodos tradicionales.

La intención del informe es sentar las bases de contenido necesario para la implementación de un modelo en base a redes neuronales convolucionales (CNNs) causales para realizar un pronóstico de la vida remanente en equipos.

Se define la metodología que se utilizará para cumplir los objetivos propuestos, se plantean los desafíos que podrían aparecer durante el desarrollo del modelo y se presenta el trabajo realizado y los resultados obtenidos.

1.1 Motivación

Las fallas en equipos conllevan un tiempo de parada no deseado que significa tiempo perdido de operación y un riesgo para la seguridad de los operarios en caso de que la falla ocurra de forma catastrófica. La detección temprana de fallas en equipos mecánicos tiene mucha relevancia en el ámbito práctico, permitiendo a operadores gestionar el tiempo de parada necesario para realizar la mantención de una falla detectada, en un momento escogido.

Para evitar la ocurrencia de fallas, los equipos en operación deben ser inspeccionados periódicamente por un experto para garantizar que se mantengan operativos, mejorar su ciclo de vida y respetar los estándares de seguridad.

La inspección periódica es exhaustiva, costosa y requiere de acceso al equipo. Por esta razón, se estudia la posibilidad de implementar alternativas remotas más económicas que garanticen resultados comparables a los que entregan métodos actuales y que reduzcan los costos de operación [3].

1.2 Objetivo General

- Determinar la vida remanente en equipos mecánicos mediante la implementación de un modelo en base a redes neuronales convolucionales causales.

1.3 Objetivos Específicos

- Analizar, indexar y clasificar los registros de señales de sensores de los equipos pertinentes.
- Generar un modelo en base a redes neuronales convolucionales causales para el pronóstico y estimación de vida remanente.
- Verificar y corroborar resultados obtenidos comparando con métodos actuales y particularmente métodos en base a Long Short-Term Memory.

1.4 Alcances

Los alcances definidos para el trabajo corresponden al tratamiento y la indexación de los registros de operación de los equipos que serán monitoreados, correspondientes a la base de datos pública "*NASA Turbofan Engine Degradation Simulation Data Set"* [4].

La implementación de un modelo en base a CNNs causales con una estructura acorde que obtenga buenos resultados en la prevención de fallas y pronóstico de vida remanente; el consecuente análisis de los resultados y la posterior validación comparando resultados con métodos de monitoreo alternativos de regresión y en base a Deep Learning, utilizando Long Short-term Memory (LSTM).

2 Antecedentes Generales

2.1 Monitoreo de equipos

Los métodos que se utilizan se distinguen por utilizar el registro de algunas de las variables del equipo en operación, tal como las vibraciones, el sonido, imágenes térmicas o el análisis de las partículas que se encuentran en el aceite de lubricación del equipo.

Los registros de las variables son analizados con distintos métodos para identificar el estado en que se encuentra el equipo y cómo progresa el deterioro con la operación.

Los métodos para analizar los registros de los equipos son, actualmente, modelos en base a Shallow Learning, como redes neuronales o Support Vector Machine. Estos modelos permiten identificar características de los datos y realizar predicciones, pero sus aplicaciones son limitadas debido a que no son capaces de modelar interacciones más complejas en bases de datos masivas con datos multidimensionales, conocidas como Big Data.

De las variables que se pueden analizar, el monitoreo de vibraciones es simple de realizar y registrar. Los métodos para el tratamiento de vibraciones se encuentran ampliamente desarrollados y permiten realizar pronósticos de forma satisfactoria haciendo que, de los métodos tradicionales, prevención de fallas por medio del monitoreo de vibraciones del equipo sea el más utilizado.

2.1.1 Monitoreo por Termografía

El aumento de la temperatura en las diferentes regiones de un equipo es un claro indicador del daño estructural o de componentes. Fallas como conexiones eléctricas expuestas, componentes

materiales dañados y piezas motrices con mala lubricación producen un aumento considerable en la distribución de temperatura local.

Actualmente, técnicas de termografía infrarroja han madurado hasta ser lo suficientemente aceptadas como para el monitoreo de equipos basado en el registro on-line de la temperatura en las diferentes zonas del equipo que se encuentra en operación.

Por medio de cámaras infrarrojas se puede distinguir visualmente la presencia de algún aumento de temperatura que no sea normal en la operación del equipo, además, por la naturaleza del aumento de temperatura, se puede distinguir la zona y los componentes que se verían afectados en caso de presentarse la falla.

2.1.2 Monitoreo por Análisis de Aceite

El monitoreo por análisis de aceite corresponde a la realización de análisis espectrográfico a muestras de aceite que se retiran periódicamente de los equipos en operación.

El análisis espectrográfico permite identificar material particulado que se encuentre suspendido en el aceite y el material particulado corresponde al producto del desgaste de las piezas que se encuentran en movimiento y que cuentan con aceite de lubricación para disminuir el roce y disipar calor.

De esta forma, por medio de la identificación del material particulado en el aceite, se puede identificar el daño presente en los componentes del equipo, antes de que el daño sea significativo y perjudicial para la operación del equipo.

2.1.3 Monitoreo por Vibraciones

Se considera que todo equipo tiene un espectro normal de vibración, hasta que se manifiesta una falla que altera el funcionamiento y el espectro cambia. El monitoreo por vibraciones ha probado ser muy efectivo para identificar los estados de funcionamiento, particularmente en equipos mecánicos rotatorios.

El registro de las vibraciones se realiza por medio de acelerómetros que monitorean los equipos en puntos relevantes de la estructura para obtener secuencias de vibraciones que representen el funcionamiento de los equipos.

De los métodos que se estudian actualmente para analizar registros de vibraciones destaca la implementación de modelos en base a Deep Learning (DL) [5], particularmente en base a redes neuronales recursivas y unidades de memoria LSTM. Estos modelos obtienen excelentes resultados que superan a los de los métodos tradicionales, además de permitir realizar el monitoreo y pronóstico sin tener que dedicar a una persona a la interpretación y prevención de la falla en el equipo.

2.1.4 Big Data

Big Data es el término con el que se conocen las estrategias no tradicionales y tecnologías necesarias para recolectar, organizar, procesar y analizar data sets de gran tamaño. A pesar de que el problema de tratar con bases de datos que superan las capacidades de procesamiento de un equipo

no es nuevo, en los últimos años, debido a la facilitación en la adquisición de información y datos, se ha tenido un incremento exponencial en la información disponible en data sets, por lo que la cantidad de data sets de gran tamaño, sus aplicaciones y funcionalidad ha crecido dramáticamente.

Las principales características de un data set catalogado como Big Data radican en un enorme volumen de datos, una enorme variedad dentro de los datos, además de gran variabilidad entre los datos de una misma clasificación. Por lo anterior, uno de los parámetros más relevantes en el análisis de Big Data es la velocidad de procesamiento de datos, imposibilitando la implementación de varios métodos utilizados en data set de menor tamaño e invitando a métodos de procesamiento más generales que permitan realizar el procesamiento de forma más rápida aprovechando el valor de los datos recolectados.

2.1.5 Monitoreo por Machine Learning

El monitoreo por Machine Learning (ML) es una de las soluciones más probadas y verificadas para el monitoreo en casos donde se ve involucrado el Big Data. La aplicación de algoritmos de ML permite tratar los datos multivariables multidimensionales de forma óptima, pudiendo extraer gran cantidad de información que con métodos tradicionales tomaría tiempos extremadamente largos.

La implementación de algoritmos en base a ML se da en todo el espectro de los tipos de monitoreo, por ejemplo, para el monitoreo por análisis de aceite, los resultados obtenidos a partir del espectrograma del aceite pueden ser clasificados con algún algoritmo para identificar las condiciones de operación de los equipos y definir el estado de operación del equipo en base a los datos históricos registrados.

Una particularidad de la aplicación con ML es que los algoritmos probados y ratificados hasta el momento consisten en mayor parte en algoritmos más básicos y desactualizados conocidos como algoritmos en base a Shallow Learning. Actualmente se estudia la implementación de algoritmos de mayor complejidad, conocidos como DL, buscando mejorar el monitoreo en base a los algoritmos del estado del arte en ML que ya han probado ser funcionales en la aplicación para la prevención y detección de fallas.

Las CNNs causales han entregado mejores resultados que las LSTM en tareas de generación a partir de secuencias, por lo que se implementarán en la realización de pronósticos a partir del registro de la operación de los equipos o de datos secuenciales.

2.2 Deep Learning

Los métodos de DL, particularmente las redes neuronales, son una clase de algoritmos para Machine Learning basados en múltiples capas de neuronas que actúan como unidades de procesamiento no-lineal para extraer características representativas a partir de una base de datos y un proceso de aprendizaje.

En la Figura.1 se muestra un esquema de la estructura básica de los modelos basados en DL. Se aprecian las diferentes capas con sus respectivas neuronas, además de las conexiones entre las neuronas de las diferentes capas. Es importante notar que algunas capas del modelo cumplen

diferentes funciones, particularmente se distingue la primera capa que contiene los nodos de entrada, la última capa que contiene los nodos de salida y la capa intermedia, o las capas intermedias, que conectan los nodos de entrada con los nodos de salida.

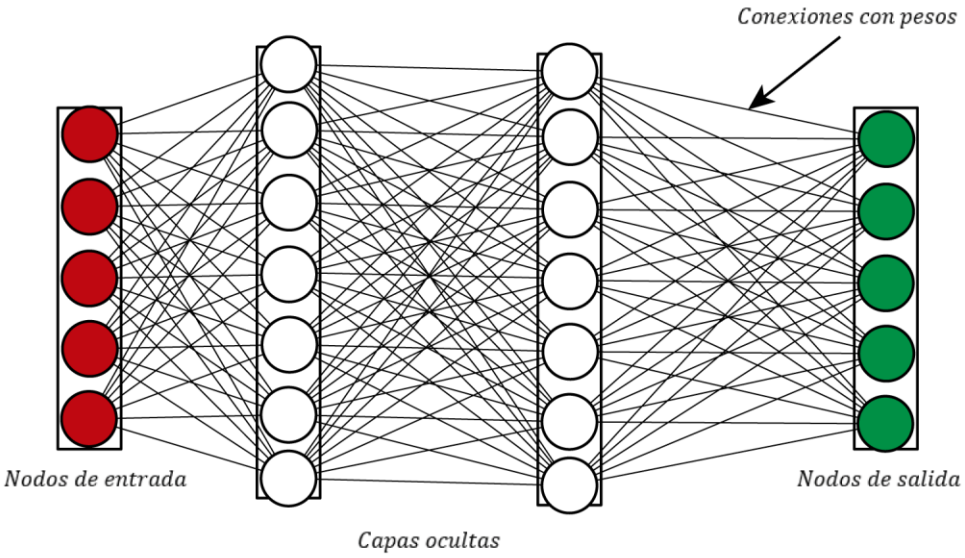

Figura 1: Esquema modelo básico de Deep Learning. Fuente: Elaboración propia.

La característica principal de estos modelos es que aprenden a identificar características relevantes de la base de datos con la que se realiza el entrenamiento. El aprendizaje se logra por medio de la conceptualización jerárquica de los diferentes niveles de abstracción de la base de datos, utilizando algún método de optimización para minimizar problemas no convexos que ajusten la estructura de la red durante el método de backpropagation que se utiliza en el aprendizaje y que será detallado en secciones siguientes.

2.2.1 Neuronas

Las neuronas son la unidad básica de los modelos en base a DL. La neurona de una capa se conecta con por lo menos una neurona de la capa anterior y por lo menos una neurona de la capa siguiente.

Cada conexión entre neuronas tiene asignado un parámetro que se conoce como peso y que indica la importancia relativa de esa conexión con respecto a las demás conexiones que van entrando a cada neurona. En la Figura.2 se aprecia una representación gráfica de una neurona y sus parámetros.

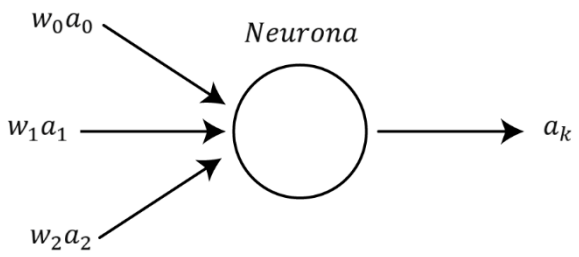

Figura 2: Neurona con sus valores de entrada y de salida. Fuente: Elaboración propia.

Cada neurona computa los valores de las conexiones que entran y entrega un valor para las conexiones que salen y que posteriormente se propagarán a las neuronas siguientes. El cómputo de la neurona consiste en una combinación lineal de los pesos de cada conexión que entra a la neurona, multiplicado por el valor de la neurona con la que se realiza la conexión en la capa anterior a la de la neurona que computa. Al valor calculado se le suma otro parámetro para evitar los casos en que la combinación lineal de las conexiones de entrada suma cero, tal y como se muestra en la ecuación (1).

$$\mathrm{a_k} = \mathrm{w_0 a_0} + \mathrm{w_1 a_1} + \mathrm{w_2 a_2} + \cdots + \mathrm{w_n a_n} + \mathrm{b_k} \tag{1}$$

Donde a_k es el valor de la neurona k, w_i es el peso i de las n conexiones de entrada que ingresan a la neurona k, a_i es el valor de la neurona con la que la neurona k está conectada en la conexión i de las n conexiones de entrada y b_k es el valor de la constante de la neurona k que se conoce como bias de la neurona.

2.2.2 Respuesta Neuronas

Los valores obtenidos por los cómputos realizados en las neuronas podrían ser prácticamente cualquier valor. Para evitar esto, los resultados se filtran utilizando una función activación que regula la respuesta de la neurona.

Actualmente se implementan funciones complejas para cada tarea en específico, por lo que la respuesta de la neurona se definirá como la función genérica que se muestra en la ecuación (2). En secciones siguientes se define la función que se implementará en el modelo de CNNs causales.

$$Respuestaneurona = f(a_k) \tag{2}$$

Donde a_k es el valor del cómputo con la ecuación (1) de la neurona k.

De esa forma, la respuesta de cada neurona a los datos ingresados por las conexiones de entrada queda según se muestra en la ecuación (3).

$$a_k^{(l+1)} = f\left(w_{k,0}a_0^{(l)} + w_{k,1}a_1^{(l)} + w_{k,2}a_2^{(l)} + \cdots + w_{k,n}a_n^{(l)} + b_k^{(l+1)}\right) \tag{3}$$

Donde $a_k^{(l+1)}$ es el valor correspondiente a la neurona k de la capa $l+1$, $w_{k,i}$ corresponde al peso de la conexión i de las n conexiones de la neurona k, $a_i^{(l)}$ es el valor correspondiente a la

neurona de la capa l que tiene la conexión i con la neurona k de la capa $l+1$ y $b_k^{(l+1)}$ es el valor del bias de la neurona k de la capa $l+1$.

2.2.3 ReLu

La función ReLu es la función activación más ampliamente utilizada para modular la respuesta de las neuronas.

Su utilización se debe a la simpleza del cómputo, el rápido entrenamiento del modelo y los buenos resultados que ha mostrado en los diferentes tipos de modelos en base a DL.

La función ReLu se puede apreciar en la ecuación (4) y en la Figura.3, donde se muestra su formulación matemática que permite notar la simpleza del cálculo a realizar y su representación gráfica.

$$ReLu(x) = \max(x, 0) \tag{4}$$

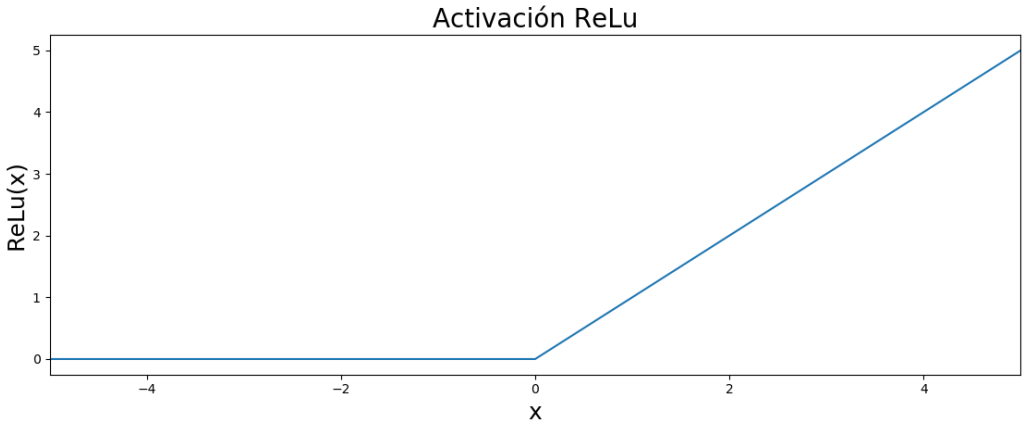

Figura 3: Gráfico de la función activación ReLu. Fuente: Elaboración propia.

2.2.4 Modelo

El modelo se construye expresando los valores de las neuronas de cada capa como sistemas de ecuaciones lineales que dependen de los valores de la capa anterior, de esta forma se llega a un sistema de ecuaciones global que resuelve la salida del modelo a partir de los datos de entrada. En la ecuación (5) se aprecia el sistema de ecuaciones para el cómputo de todas las neuronas de una de las capas.

$$f\left(\begin{bmatrix} {w_{0,0}}^{(l)} & {w_{0,1}}^{(l)} & \dots & {w_{0,n}}^{(l)} \\ {w_{1,0}}^{(l)} & {w_{1,1}}^{(l)} & \dots & {w_{1,n}}^{(l)} \\ \vdots & \vdots & \ddots & \vdots \\ {w_{m,0}}^{(l)} & {w_{m,1}}^{(l)} & \dots & {w_{m,n}}^{(l)} \end{bmatrix} \begin{bmatrix} a_0^{(l)} \\ a_1^{(l)} \\ \vdots \\ a_n^{(l)} \end{bmatrix} + \begin{bmatrix} b_0^{(l+1)} \\ b_1^{(l+1)} \\ \vdots \\ b_n^{(l+1)} \end{bmatrix}\right) = \begin{bmatrix} a_0^{(l+1)} \\ a_1^{(l+1)} \\ \vdots \\ a_n^{(l+1)} \end{bmatrix} \tag{5}$$

Donde ${w_{m,n}}^{(l)}$ es el peso de la conexión m de entrada a la neurona n que pertenece a la capa l, $a_n^{(l)}$ es el valor de la neurona n que pertenece a la capa l, $b_n^{(l+1)}$ es el valor del bias de la

neurona n que pertenece a la capa $l+1$ y $a_n^{(l+1)}$ es el valor de la neurona n que pertenece a la capa $l+1$.

La ecuación (5) puede expresarse de forma más simple, escribiendo las matrices como vectores que representan cada capa, tal como se muestra en la ecuación (6).

$$a^{(l+1)} = f\left(W_l a^{(l)} + {b_l}^{(l+1)}\right) \tag{6}$$

Donde $a^{(l+1)}$ es el vector de todas las neuronas de la capa $l+1$, W_l es el vector con todos los pesos de las conexiones entrantes a la capa $l+1$, $a^{(l)}$ es el vector de todas las neuronas de la capa l y ${b_l}^{(l+1)}$ es el vector con los bias de cada neurona de la capa $l+1$.

De esta forma, el problema de ajustar el modelo se reduce a elegir de buena forma los valores de los pesos de cada conexión y los valores del bias de cada neurona de una capa, tal que los valores obtenidos a la salida correspondan a lo esperado según los valores de entrada a la red.

El proceso de ajustar los valores del modelo se conoce como el aprendizaje de la red.

2.3 Aprendizaje

El aprendizaje de la red neuronal consiste en el ajuste de los valores de la estructura de la red para encontrar una función que mapee un grupo de entradas a un grupo de salidas identificadas como correctas para cada imagen de entrada.

El aprendizaje se da a través del entrenamiento de la red, que consiste en alimentar el modelo con datos que están identificados y para los que se conoce el output que la red neuronal debería entregar.

Se inicia generando de forma aleatoria valores para los pesos y bias del modelo, luego se ingresan los valores de los datos identificados y se computa el valor de la salida de la red, que se compara con el valor identificado para los datos ingresados. Se calcula el error del resultado obtenido por el modelo y el resultado esperado para los datos según una función de costos.

Finalmente se utiliza el algoritmo de backpropagation para propagar el error de las neuronas de salida a todas las neuronas del modelo y mediante un algoritmo de optimización se ajustan los pesos para minimizar el error obtenido por la función de costos y así, continuar con el entrenamiento.

Es importante notar que el aprendizaje generalmente se realiza en paquetes o *batches* y que el ajuste de los pesos, mediante el algoritmo de optimización, se realiza para minimizar el error promedio obtenido para la función de costos de todos los datos del paquete, lo que propicia un aprendizaje más generalizado del modelo.

2.3.1 Inicialización de Variables

La inicialización de variables corresponde al proceso por el cual se asignan valores a todos los parámetros del modelo de forma inicial, antes de realizar cualquier modificación. La

inicialización de variables permite realizar cómputos con el modelo antes de los procesos de optimización y entrega la semilla para la generación de los cálculos del modelo construido.

El método de inicialización de variables utilizado en este modelo es el '*Xavier Initializer*' que busca resolver dos problemas que se presentan con la inicialización de variables:

1. Si los valores de inicialización de un modelo comienzan muy pequeños, la señal disminuye por cada capa que pasa hasta que se pierde la información.
2. Si los valores de inicialización de un modelo comienzan muy grandes, la señal crece a medida que pasa por cada capa hasta que es muy grande para ser útil.

La solución a estos problemas se logra por medio de la generación de los valores de los pesos a partir de una distribución Gaussiana con promedio 0 y una varianza específica en función de la cantidad de neuronas que alimentan cada neurona.

2.3.2 Descenso por Gradiente

El descenso por gradiente es un algoritmo de primer orden para la optimización iterativa y búsqueda del mínimo de una función cualquiera.

Para encontrar el mínimo local utilizando el descenso por gradiente, se toma un punto inicial de la función, se calcula el gradiente de la función en el punto inicial y luego se avanza una distancia proporcional al negativo del gradiente. Luego, en el punto siguiente se vuelve a calcular el gradiente y se repiten los pasos siguientes hasta llegar a un punto donde el gradiente de la función sea igual a 0, indicando que se ha encontrado el mínimo local de la función.

El descenso por gradiente se basa en la observación de que la función multivariable $F(x)$ es diferenciable en la vecindad del punto en el que se ubica cada iteración del algoritmo. Por lo tanto, $F(x)$ decrece de forma más rápida si se va desde el punto inicial hacia la dirección negativa del gradiente.

De esta forma, el algoritmo del descenso por gradiente se puede apreciar en la secuencia siguiente.

Optimización con descenso por gradiente de $F(x)$:	
1. ***Se toma un punto inicial***:	a_n
2. ***Se calcula el gradiente en el punto***:	$\nabla F(a_n)$
3. ***Se avanza según el gradiente***:	$a_{n+1} = a_n - \gamma \nabla F(a_n)$
4. ***Si γ es suficientemente pequeño***:	$F(a_n) > F(a_{n+1})$
5. ***Se repite el proceso con punto inicial***:	a_{n+1}
6. ***Se encuentra el mínimo local cuando***:	$\nabla F(a_{n+1}) = 0$

En la Figura.4 se puede apreciar una representación del descenso por gradiente recorriendo las curvas de nivel de una función.

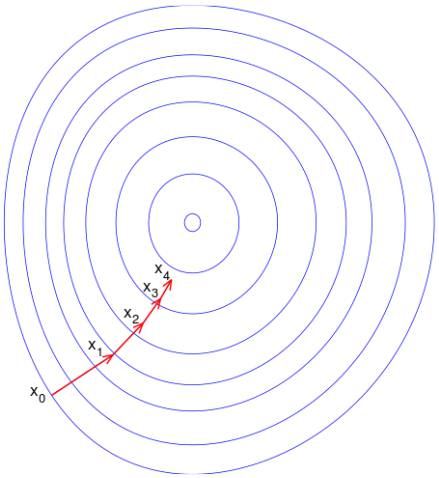

Figura 4: Descenso por gradiente en las curvas de nivel de una función. Fuente: Wikipedia.

2.3.3 Función de Costos

En la optimización matemática la función de costos es la función que mapea los valores obtenidos por un modelo a una representación en números reales que permite representar un costo asociado a los valores obtenidos, en relación a cierto evento. El problema de optimización busca minimizar la función de costos para que los valores obtenidos a partir del modelo tengan el menor costo posible.

El costo en modelos de DL se entiende como la diferencia entre los valores obtenidos durante el entrenamiento y los valores esperados según los labels de los datos de entrenamiento. La optimización de la función de costos, que es la función objetivo del problema de optimización, busca ajustar los valores obtenidos durante el entrenamiento a los valores reales definidos para los datos de entrenamiento y así mejorar iterativamente las predicciones realizadas.

Para la función de costos se utiliza generalmente el error cuadrado de los valores obtenidos en comparación con los valores esperados. Esta elección corresponde al método estándar utilizado en el área y se explica por el reducido costo computacional que tiene el cálculo frente a otras medidas de error, además de alcanzar resultados comparables a otros métodos más complejos para la optimización de los parámetros de las CNNs.

Para un problema multi clase con c el número de clases y N ejemplos de entrenamiento, el error cuadrado está dado por la ecuación (7).

$$E^N = \frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{c}(t_k^n - y_k^n)^2 \tag{7}$$

Donde t_k^n es la dimensión k del patrón n que corresponde a la salida objetivo de una capa e y_k^n es el valor de la neurona k de una capa de salida en respuesta al patrón n de entrada.

Con problemas multi clase de clasificación, la salida objetiva corresponderá a la asignación de cada objetivo a una única clase c, donde el k elemento de t^n es positivo si el patrón pertenece a la clase k. El resto de los valores de t^n serán cero o negativo dependiendo de la función de activación definida para las neuronas.

Por lo anterior, la ecuación (7) corresponde a la suma de los errores individuales en cada patrón, se considera una única secuencia n para la propagación y el cálculo del error queda según la ecuación(8).

$$E^N = \frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{c}(t_k^n - y_k^n)^2 = \frac{1}{N}\|t^n - y^n\|_2^2 \tag{8}$$

Como se mencionó en la sección de aprendizaje, el error también puede considerarse como el promedio del error calculado para N patrones de un paquete con los que se está alimentando la red y que se expresa según la ecuación (9).

$$E = \frac{1}{N}\sum_{x=1}^{n}\left\|\left(y(x) - y'^{(x)}\right)\right\|_2^2 \tag{9}$$

Donde $y(x)$ corresponde al vector de características o *labels* definidos para los datos de entrada y $y'^{(x)}$ corresponde al vector con los valores de la predicción realizada por el modelo para cada dato de entrada.

2.3.4 Backpropagation

La propagación hacia atrás o *backpropagation* es el algoritmo utilizado para realizar el entrenamiento de la red basándose en el cálculo del error a la salida de la red y la posterior propagación del error hacia los pesos de las neuronas de las capas interiores.

El algoritmo consiste en calcular el gradiente de la función de costos a la salida del modelo con respecto a los pesos de las conexiones entre las últimas capas del modelo, y luego repetir la derivación con respecto a los pesos de las conexiones con la capa que precede y así sucesivamente.

Cuando se conoce el término del gradiente correspondiente a cada neurona se puede proceder a minimizar el error de la función de costos por medio de algún algoritmo de optimización que utilice el gradiente de la función objetivo. El algoritmo escogido preferentemente es alguna versión del descenso por gradiente.

El algoritmo para una iteración del método de descenso por gradiente aplicado a backpropagation se muestra a continuación:

1. ***Ingresar un paquete de datos de entrenamiento***.

2. ***Para cada dato de entrenamiento D***: $Fijar\ el\ valor\ de\ las\ neuronas\ a^{(0)}$

2.1. ***Feedforward***: $Para\ cada\ capa\ l = 0{,}1 \dots, L-1\ computar\ a^{(l+1)} = f\left(W^{(l)}a^{(l)} + b^{(l)}\right)$

2.2. ***Error de la salida***: $Computar\ el\ vector\ \delta_D^L\ con\ \delta_D^L = \nabla_a E^D \odot a'^{(l)}$

2.3. ***Backpropagation del error***: $Para\ cada\ l = L-1, L-2, \dots, 1\ computar\ \delta_D^l = \left((W^{l+1})^T \delta_D^{l+1}\right) \odot a'^{(l)}$

3. ***Descenso por Gradiente***: $Para\ cada\ \ l = L, L-1, \dots, 1$

3.1. ***Ajustar pesos***: $W^{(l)} \to W^{(l)} - \frac{\mu}{N}\sum_x \delta_D^l \left(a^{(l-1)}\right)^T$

3.2. ***Ajustar biases***: $b^{(l)} \to b^{(l)} - \frac{\mu}{N}\sum_D \delta_D^l$

Donde L corresponde a la cantidad de capas del modelo, E^D es el valor de la función de costos obtenida a partir del dato de entrenamiento D, ∇_a es el operador para el gradiente en función del valor de las neuronas, δ_D^L es el vector que indica como varía la función de costos en relación al valor de los pesos y bias, N es el número de datos de entrenamiento por paquete y μ es el *ratio de aprendizaje* del modelo que define la distancia que se traslada el punto que se está optimizando en cada iteración del descenso por gradiente.

2.4 Regularización

La regularización consiste en la implementación de diferentes métodos que permiten facilitar la resolución de problemas de optimización complejos no convexos. Los diferentes métodos se aplican para aumentar la cantidad de información que procesa la función de costos o

para disminuir la complejidad de las restricciones del problema, permitiendo evitar enormes tiempos de optimización y overfitting.

2.4.1 Drop-Out

Cuando los modelos en base a DL son muy profundos, contando con muchas neuronas, se procede a implementar drop-out para regularizar el modelo y evitar que se sobreajuste a los datos de entrenamiento.

El método de drop-out se aplica en el entrenamiento cuando se realiza el cómputo de los valores de una capa. En el momento de realizar el cómputo, se define una probabilidad constante, llamada *keep probability*, que indica la probabilidad de que cada neurona en particular se active durante una iteración del entrenamiento o sea desechada y su valor considerado como nulo.

Desechar neuronas de forma aleatoria permite evitar que se generen sobreajustes por dependencias muy fuertes de una neurona en particular. Generalmente logra mejorar la certeza del modelo por medio de un aprendizaje más generalizado.

2.4.2 L2 Regularization

Un método de regularización corresponde a introducir más información a la función de costos, por medio de una penalización, con tal de plantear un problema de optimización que involucre más variables relevantes en el desarrollo del problema.

La regularización implementada es conocida como "*L2 Regularization"* o *"Ridge Regression"* y consiste en agregar a la función de costos el cuadrado de los valores de todos los pesos del problema, multiplicados por un *ratio de regularización* tal y como se muestra en la ecuación (10).

$$E = Función\ de\ Costos\ + \gamma \sum_{j=1}^{P} w_j^2 \tag{10}$$

Donde γ es el ratio de regularización y w_j es el peso j de todos los pesos P del modelo. El resto de la ecuación es la función de costos de la ecuación (9) mostrada en la sección correspondiente.

2.5 Redes Neuronales Convolucionales

Como típicas redes neuronales, las CNNs cuentan con una capa de entrada, una de salida y varias capas intermedias. Las capas intermedias que conforman las CNNs pueden ser de varios tipos, según el tipo de neurona que tengan, la cantidad de neuronas a las que la neurona respectiva está conectada y cómo se realiza la conexión. De los diferentes tipos de capas que conforman una CNN destacan las capas de convolución y pooling. Las capas de convolución serán explicadas más adelante, las de pooling se omitirán por no ser pertinentes a las CNNs causales.

La existencia de capas de convolución permite extraer características de datos que tienen una ubicación espacial en un volumen definido, como la imagen en blanco y negro que se muestra en la Figura.5, donde la figura corresponde a la representación gráfica y a los valores de cada píxel en la imagen de tamaño 12x12.

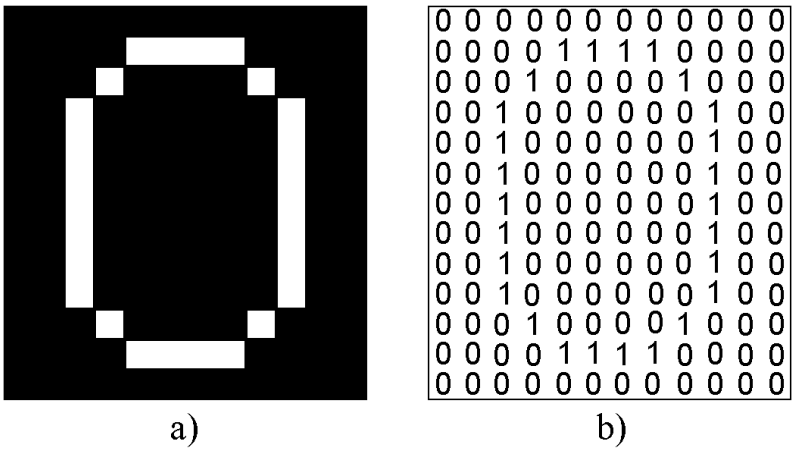

Figura 5: Imagen de un '0' de 12x12 píxeles. a) Representación gráfica de la imagen. b) Representación en bits de la imagen. Fuente: Elaboración propia.

La implementación de modelos en base a CNNs permite identificar las características presentes en las secuencias de vibraciones a lo largo de un eje temporal, siendo la razón por la que las CNNs y modelos con estructuras similares son actualmente la alternativa predominante para la implementación de nuevos mecanismos para el diagnóstico, pronóstico y prevención de fallas en equipos mecánicos.

2.5.1 Capas de Convolución

Los parámetros de las neuronas que forman las capas de convolución corresponden a matrices con los pesos de las conexiones, que se conocen como *kernels*. Los kernels son de dimensiones pequeñas en relación con el volumen que ingresa a la capa, pero mantienen el mismo grosor. Los kernels actúan como filtros y se utilizan para realizar convolución con las matrices de las imágenes que ingresan a las neuronas de la capa de convolución.

Por medio del entrenamiento de la red, los valores de los kernels obtenidos permiten identificar características en regiones de las imágenes que son ingresadas a la capa de convolución respectiva. Durante el paso hacia adelante de la imagen introducida a la red, el kernel de cada neurona se convoluciona a lo largo de toda la extensión de las matrices que ingresan a la neurona.

De esta forma cada neurona de la capa de convolución genera un volumen de menor o igual tamaño al entrante pero que incluye las características que el filtro aplicado logra detectar por medio de la convolución. El volumen generado se conoce como un mapa de características o *features*. La cantidad de pasadas que realiza el kernel en la matriz imagen corresponde a la cantidad de features de una capa de convolución.

En la Figura.6 se presenta un dibujo esquemático de una convolución con un filtro 2x2 sobre la superficie de una imagen. Luego de realizar la convolución y asignar el valor al píxel de destino indicado en la imagen, el filtro se desplaza sobre la superficie de la imagen, como se muestra en la Figura.7, y va calculando progresivamente los valores de cada píxel restante en la

imagen resultante. Luego de haber recorrido completamente la superficie de la imagen se tiene una convolución completa.

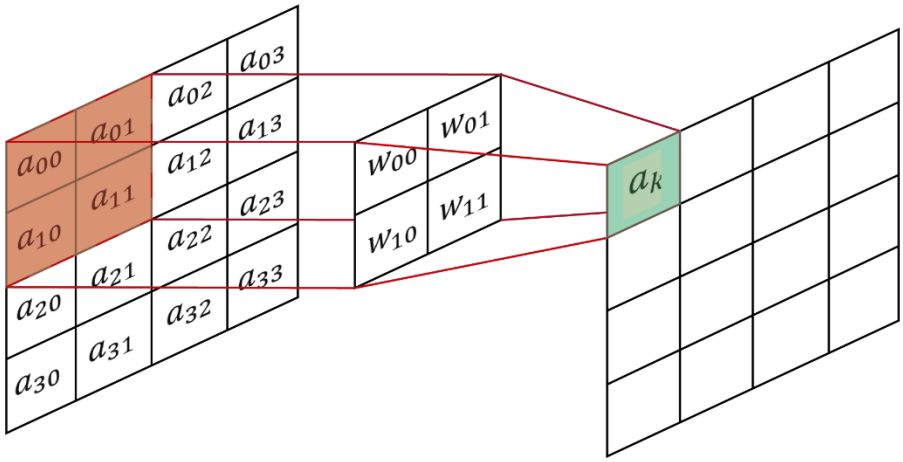

Figura 6: Esquema del proceso de convolución en el momento que el filtro se encuentra sobre la esquina superior izquierda de la imagen o capa anterior. Fuente: Elaboración propia.

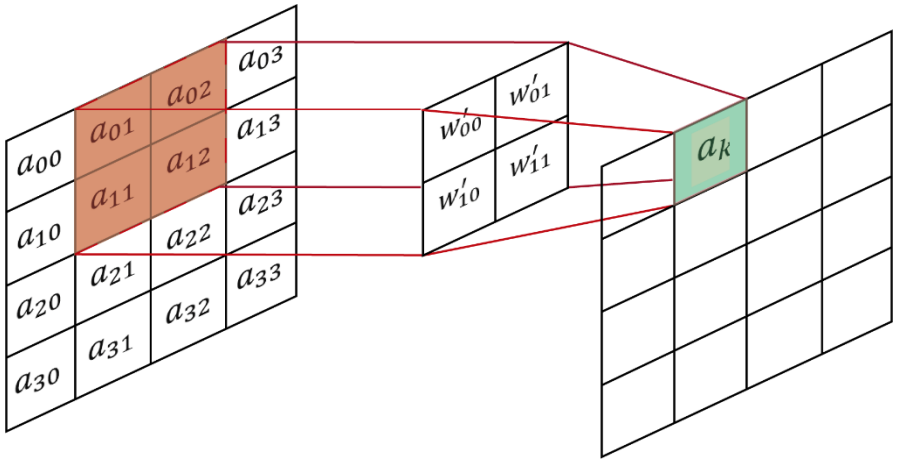

Figura 7: Siguiente iteración de la convolución, donde el filtro se ha desplazado por la imagen para generar los píxeles resultantes. Fuente: Elaboración propia.

En la ecuación (XX), se muestra el cómputo de cada neurona según el filtro que pasa por la imagen de la capa anterior, en relación con las figuras anteriormente mostradas.

$$\begin{aligned} a_{k0} &= w_{00}a_{00} + w_{01}a_{01} + w_{10}a_{10} + w_{11}a_{11} \\ a_{k1} &= w'_{00}a_{01} + w'_{01}a_{02} + w'_{10}a_{11} + w'_{11}a_{12} \\ &\vdots \end{aligned} \tag{9}$$

Es importante notar que la convolución a lo largo de toda la extensión de la matriz puede incluir superposición en el cálculo de las convoluciones, lo que se define según el *stride y padding* de la convolución. La implementación de superposición con los distintos kernels puede ser importante porque permite relacionar las características de regiones aledañas en la matriz inicial, complejizando los pronósticos de la red.

En una CNN el stride es una matriz que indica la dirección y los saltos que realiza el kernel de la convolución a medida que recorre el espacio de la imagen con el que convoluciona.

El padding corresponde al método escogido para tratar las convoluciones que involucran los límites del espacio de la imagen donde ya no hay información. Para mantener las dimensiones de la imagen se rodea la imagen con un espacio de ceros, que permite realizar las convoluciones sin disminuir el tamaño de la imagen resultante.

2.5.2 Capas Completamente Conectadas

Las capas completamente conectadas son capas donde cada neurona de la capa está conectada a todas las neuronas de la capa anterior.

De esta forma, las capas completamente conectadas son capaces de reducir la multidimensionalidad de las features de alta complejidad que pudieran extraer las capas de convolución y así generan una respuesta simple a partir de las features detectadas.

Por esto, son utilizadas para simplificar la información y generar el output del modelo después de una serie de capas de convolución.

2.6 Redes Neuronales Convolucionales Causales

Las CNNs causales son un modelo estadístico autoregresivo de probabilidad condicional que procesa secuencias de datos con dependencias de largo alcance. El modelo es capaz de realizar predicciones secuenciales, donde la predicción que se realiza está condicionada en los valores temporales previos, pudiendo modelar las dependencias de las secuencias de datos por medio de la construcción de una distribución de probabilidad explícita. La distribución de probabilidad para una secuencia de datos $X = x_1, \dots, x_T$ se define como se muestra en la ecuación (11).

$$p(x) = \prod_{t=1}^{T} p(x_t | x_1, \dots, x_{t-1}) \tag{11}$$

Donde T es el instante en que se realizó la última predicción, t corresponde a los valores de los tiempos previos y $p(x)$ es la distribución de probabilidad de x condicionada en los valores previos.

En la definición se puede apreciar que cada muestra x_t está condicionada en las muestras de todos los saltos de tiempo previos.

El modelo se construye a partir de apilar una gran cantidad de CNNs cuyas capas intermedias son de convolución dilatada y de convolución causal. Estas capas corresponden a modificaciones a la forma de aplicar la convolución típica y serán expuestas con mayor profundidad en secciones siguientes.

En la Figura.8 se puede apreciar la convolución sobre una imagen que se genera a partir de los valores de los píxeles precedentes y el filtro de convolución causal utilizado para definir la convolución en función de los píxeles relevantes.

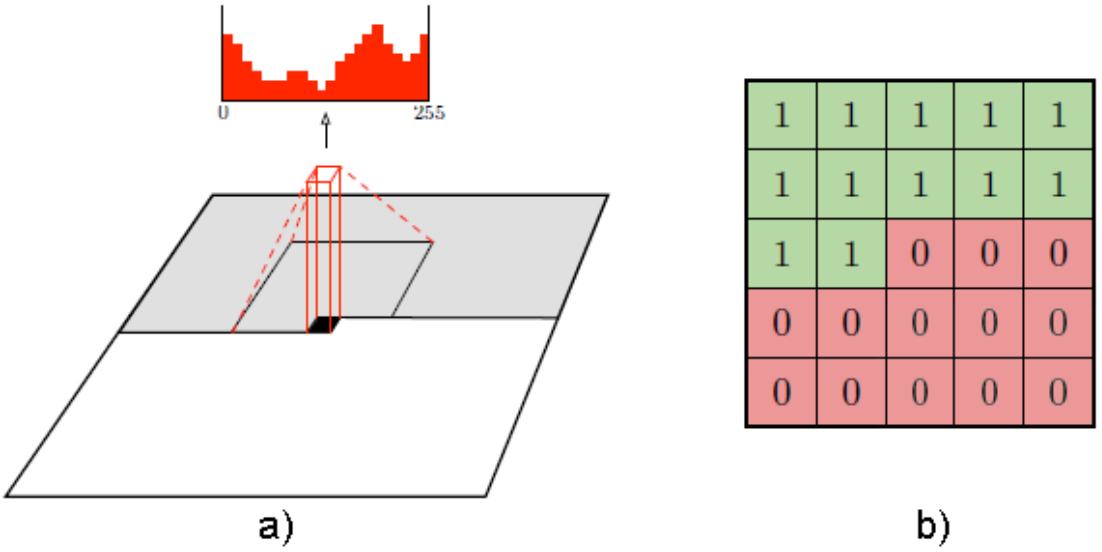

Figura 8:Esquema gráfico de una convolución causal. a) Convolución causal aplicada a una imagen con la consecuente predicción. b) Filtro causal para la convolución de a). Adaptado desde "Conditional Image Generation with PixelCNN Decoders" por A. van den Oord et al., Google DeepMind 2016.

El modelo promete ser una buena alternativa en la realización del pronóstico de vida remanente en equipos porque ha superado el desempeño de las LSTM en tareas similares, siendo las LSTM el estado del arte actual en la realización de pronósticos; sacrificando mínimamente la certeza del pronóstico en busca de favorecer una implementación mucho más fácil y rápida [6].

Las principales ventajas de las CNNs causales son que tienen baja latencia, el entrenamiento se puede realizar en paralelo, tienen enormes campos receptivos ajustables y el tiempo de computación es lineal con respecto al largo de la secuencia. El modelo además es capaz de modelar trenes de onda sin tratamiento previo y puede ser aplicado en Big Data.

La posibilidad de implementar el modelo en bases de datos de gran tamaño es gracias a que el trabajo bruto requiere de recursos computacionales, por lo que el manejo de grandes cantidades de datos requiere de mínimas horas hombre.

2.6.1 Convolución Dilatada

La convolución dilatada es una convolución donde el filtro de cada neurona que realiza la convolución es expandido para abarcar un área mayor y permitir que el campo receptivo de la convolución involucre información a gran escala del contexto de los datos.

La expansión del campo receptivo puede darse de forma exponencial a medida que se profundiza la red, manteniendo la resolución, la cobertura y permitiendo que los recorridos de la señal sean cortos considerando la profundidad de la red.

En la estructura del modelo, la convolución dilatada corresponde a la inclusión de ceros entre los valores de los pesos de un filtro de una capa de convolución. De esta forma el filtro crece, pero la cantidad de parámetros que deben ser ajustados y computados se mantiene constante.

La convolución d-dilatada corresponde a la convolución dilatada donde la distancia a la que quedan los pesos que no son cero se define según el parámetro d. Para una secuencia de entrada de una dimensión, la formulación matemática de la convolución d-dilatada se puede apreciar en la ecuación (12).

$\mathbf{1D - Input}$: $x = [x_1, x_2, \dots, x_n]$ *de tamaño* n

$\boldsymbol{Filtro\ kernel}$: $k = [k_{-l}, k_{-l+1}, \dots, k_l]$ *de tamaño* $2l + 1$

$$(k * x)_t = \sum_{i=-l}^{l} k_i \cdot x_{t-d \cdot i} \ \ para\ t \in \{1,2, \dots, n\} \tag{12}$$

Donde x es la secuencia de datos que se ingresa, k es el filtro de la convolución y $(k * x)_t$ es el resultado del cómputo de la convolución. Los diferentes valores que puede tomar d definen cuantas neuronas desconectadas hay entre cada neurona conectada. En el caso de que $d = 1$ se tiene la convolución normal con un filtro sin expandir.

En la Figura.9 se puede apreciar la expansión del filtro a medida que se aumenta el parámetro d y en la Figura.10 se puede apreciar un esquema representativo de como el filtro dilatado define las conexiones entre las neuronas de capas consecutivas.

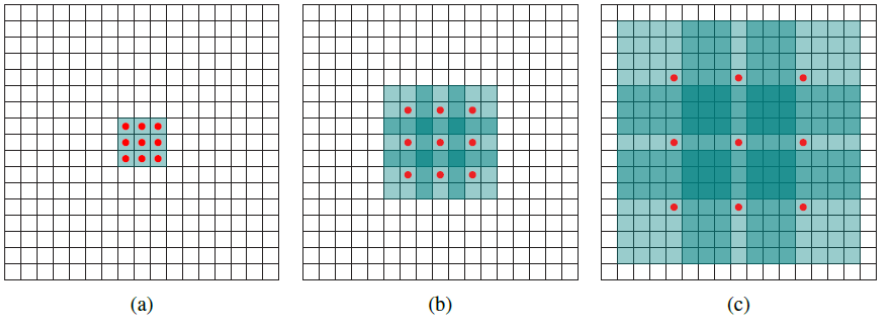

Figura 9: Dilatación de un filtro de convolución. a) Filtro sin dilatar. b) Filtro con convolución 1-dilatada. c) Filtro con convolución 3-dilatada. Adaptado desde "Multi-scale context aggregation by dilated convolutions" por F. Yu y V. Koltun, ICLR 2016 submission.

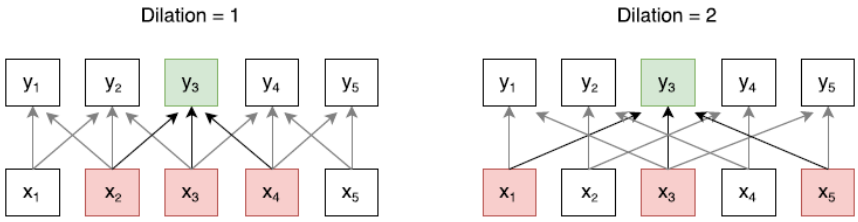

Figura 10: Conexiones entre neuronas de diferentes capas con convolución dilatada. Adaptado desde "Fast Reading comprehension with convnets" por F. Wu, N. Lao y G. Yang, ICLR 2018 submission.

2.6.2 Convolución Causal

La convolución causal es la que define las dependencias de las variables que participan de los cómputos que se realizan en cada instante y que permiten condicionar las predicciones a los valores anteriores en el eje temporal.

Definir las dependencias se logra por medio de una certera definición del filtro de convolución para que actúe como máscara y restrinja las conexiones entre neuronas, manteniendo el orden temporal o espacial de los datos y de las predicciones.

En el caso de secuencias de una dimensión, la definición de la máscara consiste en asignarle el valor cero a los pesos de las conexiones de las neuronas que aún no se han computado o que no se quieren computar.

En la ecuación (13) se puede apreciar cómo la inclusión de los ceros permite que la predicción corresponda a la multiplicación de las probabilidades condicionales de los valores hasta el instante T.

$$\textbf{1D} - \textbf{Input}: \quad x = [x_1, x_2, \dots, x_n] \text{ de tamaño } n$$

$$\textbf{Filtro kernel}: \quad k = [k_1, k_2, \dots, k_T, 0, \dots, 0] \text{ de tamaño } m \leq n$$

$$(k * x)_{T+1} = \sum_{i=1}^{n} k_i \cdot x_i = \sum_{i=1}^{T} k_i \cdot x_i \ \ para\ T < n \tag{13}$$

Donde $(k * x)_{T+1}$ es la próxima predicción condicionada y T es el tiempo de la última predicción.

En la Figura.11 se aprecia el modelo de DL aplicado a la convolución de una serie temporal, donde se identifica que las entradas del modelo incluyen solo los valores precedentes a la predicción con tal de asegurar la causalidad del modelo.

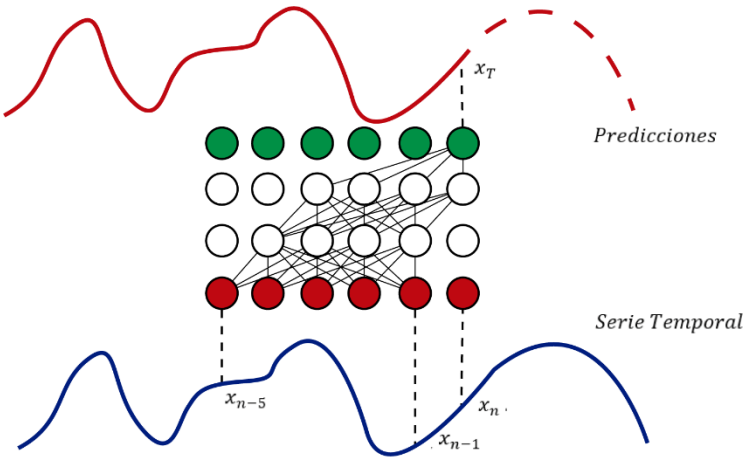

Figura 11: Convolución causal en una serie temporal para generar una predicción. Fuente: Elaboración propia.

2.6.3 Gated Units

Las CNNs causales han mostrado mejores resultados en la generación de secuencias cuando la modulación de la respuesta de las capas de convolución se realiza con Gated Units (GU) [1]. La formulación matemática de GU se puede apreciar en la ecuación (14).

$$a^{(l+1)} = \tanh\left(W_{k,f} * x\right) \odot \sigma\left(W_{k,g} * x\right) \tag{14}$$

Donde $a^{(l+1)}$ es la respuesta de las neuronas de la capa $l+1$, $W_{k,f}$ es la matriz de los pesos de las k conexiones del filtro de convolución f, x es la secuencia de valores, $\odot$ es el producto multiplicativo por elemento, σ es la función sigmoide y $W_{k,g}$ es la matriz de los pesos de las k conexiones de la puerta g.

Esta función permite agregar la unidad multiplicativa que debiese permitir modelar interacciones más complejas de la base de datos. Además, permite acelerar y facilitar el entrenamiento de modelos muy profundos [1].

En la Figura.12 se muestra un esquema de un stack de GU utilizadas en Wavenet. Cada celda extrae características según la ecuación (14).

El mapa de características de cada celda pasa por una capa completamente conectada y luego se suma a la respuesta que procesó la celda para tratar de emular la capacidad de memoria que tienen las celdas LSTM. Luego, la respuesta con memoria pasa a la siguiente celda y finalmente todas las características de las celdas se suman para generar la respuesta del stack de celdas GU tal y como se muestra en la Figura.12.

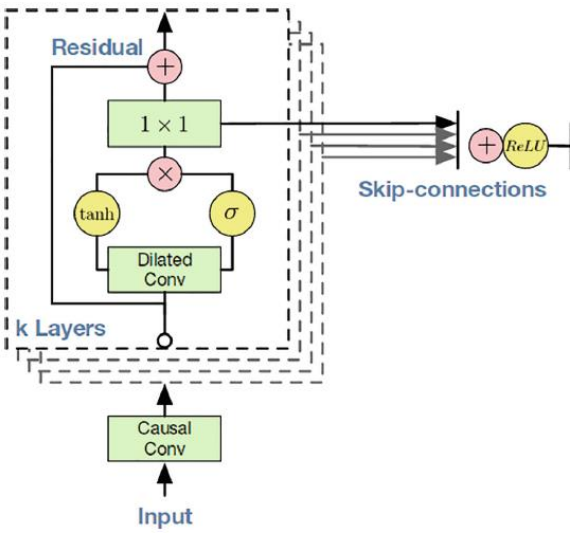

Figura 12: Esquema de un stack de GU en WaveNet. Adaptado desde "WaveNet: A generative model for raw audio" por A. van den Oord et al., Google DeepMind 2016.

Los bloques verdes de la Figura.12 corresponden a convoluciones con un filtro específico, las circunferencias amarillas corresponden a las funciones activación de los resultados de la convolución y las circunferencias rojas corresponden a las operaciones elemento a elemento entre mapas de características.

3 Metodología

La metodología implementada es estándar y corresponde a los hitos necesarios que se deben ir cumpliendo para poder generar, entrenar e implementar algoritmos de DL en bases de datos consistentes, correspondientes al registro de sensores en equipos durante la operación [7].

El proceso se inicia con la base de datos que se quiere analizar. Las bases de datos corresponden a las señales registradas por los sensores durante la operación del equipo y corresponden a datos multidimensonales relativos a la frecuencia de sampleo del sensor.

En la Figura.13 se presenta el diagrama de flujo con las etapas de la metodología que serán enunciadas más adelante.

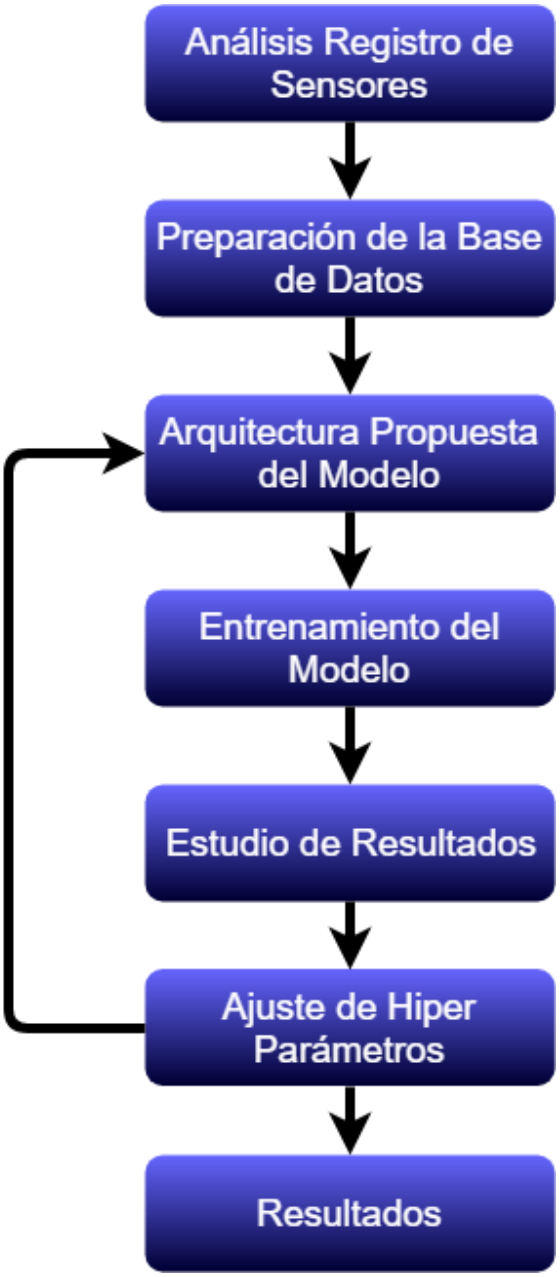

Figura 13: Diagrama de flujos de la metodología.

3.1 Análisis Registro de Sensores

El registro de la operación se da en la dimensión del tiempo y está sujeto a las condiciones de operación en el momento del registro, por lo que, para poder realizar el pronóstico, se deben analizar los registros y limpiar la base de datos para facilitar la extracción de las características importantes de cada registro por medio de la CNN causal.

Los métodos utilizados para el tratamiento consisten principalmente en la reducción del ruido o en desechar alguno de los datos que pertenecen a la base de datos.

3.2 Preparación de la Base de Datos

Posterior al análisis de la base de datos, que define el tipo de información que se le entregará al modelo en base a DL, se realiza el preprocesamiento de los datos que los identifica y clasifica para realizar el entrenamiento y posterior testeo del modelo.

Los datos preparados se distribuyen aleatoriamente en dos grupos que serán dedicados al entrenamiento de la red y a la posterior validación del modelo que fue entrenado.

3.3 Arquitectura Propuesta del Modelo

La arquitectura del modelo está definida por las estructuras internas que se implementan según el tipo de datos con los que se entrenará el modelo, considerando el objetivo del modelo.

En este caso, teniendo un modelo predictivo a partir de secuencias temporales, se propone una arquitectura compuesta por capas de convolución, un stack de GU y capas completamente conectadas para la extracción de características y realización de predicciones.

A continuación, se muestran las estructuras del modelo y los parámetros más importantes que definen la arquitectura.

- Capas de Convolución:
  - Cantidad de capas
  - Features de cada capa
  - Kernel de convolución de cada capa
  - Dilatación del kernel de cada capa
  - Causalidad de la convolución

- Stack de GU:
  - Cantidad de celdas del stack
  - Kernel de convolución de la celda
  - Dilatación del kernel de la celda
  - Causalidad de la convolución

- Capas completamente conectadas:
  - Cantidad de capas
  - Cantidad de neuronas de cada capa

3.4 Entrenamiento del Modelo

Luego de que se define la arquitectura se procede a realizar el entrenamiento que permite ajustar los valores del modelo para que sea capaz de identificar los datos que fueron preparados y predecir la vida remanente del equipo.

En el entrenamiento también se deben definir los parámetros del aprendizaje que se exponen a continuación.

- Keep probability en cada capa
- Tamaño de los batches
- Ratio de aprendizaje
- Ratio de regularización
- Épocas de entrenamiento
- Padding

3.5 Estudio de Resultados

Los resultados obtenidos en la implementación del modelo se estudian y se define si son satisfactorios o no.

3.6 Ajuste de Hiper-parámetros

En caso de que los resultados no sean satisfactorios se repite el paso expuesto en la sección 3.3 y se varían los parámetros de la arquitectura y del entrenamiento hasta que el modelo sea capaz de pronosticar el daño de los equipos por medio de la interpretación de la base de datos de los sensores.

4 Desarrollo

El modelo se desarrolla e implementa en el data set "*NASA Turbofan Engine Degradation Simulation Data Set - CMAPSS"*. La metodología utilizada es la desarrollada en la sección anterior y se aplica siguiendo cada paso.

4.1 CMAPSS Data set

Los datos de *CMAPSS* corresponden a datos de operación de turbinas áreas con 90.000 $[lb]$ de empuje que se generan a partir de un programa de simulación basado en modelos físicos termodinámicos y que son liberados al público por la *NASA* para servir de validación en la implementación de diferentes métodos de predicción y estimación del daño durante la operación.

El data set se divide en cuatro sub sets, denominados FD001, FD002, FD003 y FD004, que se diferencian entre ellos en la cantidad de modos de falla y las condiciones de operación.

Las condiciones de operación varían en la altitud, entre el nivel del mar y 12,2 $[km]$, en el Mach de operación, entre 0 y 0,9 y las temperaturas a nivel del mar, entre -51 a 39 ℃.

Cada sub set cuenta con registros para el entrenamiento y registros de testeo para la validación del modelo. En la Tabla.1 se muestran las especificaciones de cada uno de los cuatro sub sets que componen el data set *CMAPSS*.

Tabla 1: Especificaciones y características de cada sub set de CMAPSS.

Data Set	FD001	FD002	FD003	FD004
Unidades de motores para el entrenamiento	100	260	100	249
Unidades de motores para el testeo	100	259	100	248
Condiciones de operación	1	6	1	6
Modos de falla	1	1	2	2
Muestras de entrenamiento	17.731	48.819	21.820	57.522
Muestras de testeo	100	259	100	248

4.1.1 Training set

El set de entrenamiento corresponde al registro de los ciclos de operación hasta la falla de los sensores de varias turbinas aéreas que operan en diferentes condiciones según cada sub set del data set.

Cada turbina inicia su vida operativa con diferentes grados de fallas de manufactura y deterioro inicial que se asumen como desconocidos, por lo que se considera que cada turbina inicia su operación en estado saludable.

El registro de los sensores se realiza hasta el momento en que la turbina es declarada como inoperativa.

En la Figura.14 se muestra el registro de los 21 sensores de una unidad desde el comienzo en operación hasta la falla.

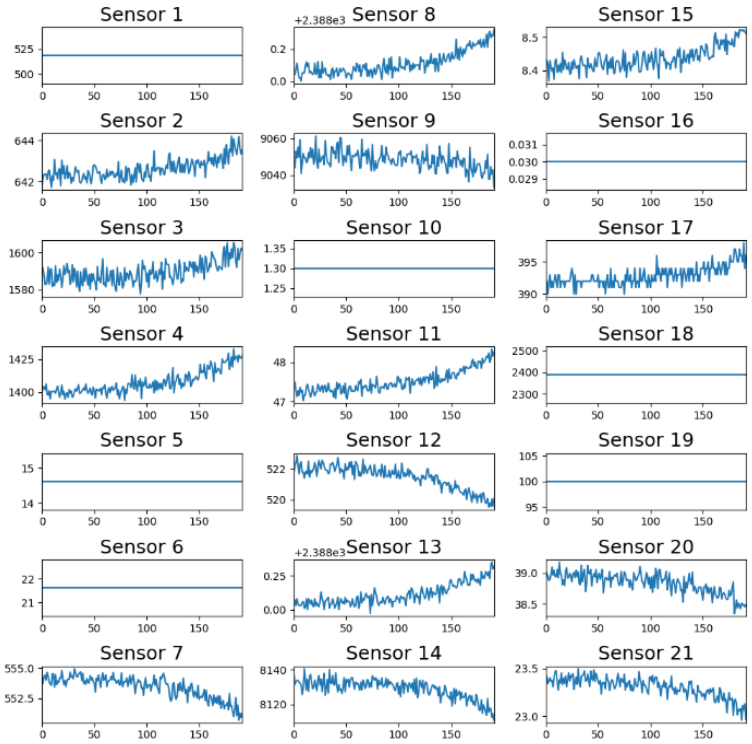

Figura 14: Registro de los 21 sensores durante la operación de una unidad de CMAPSS. Fuente: Elaboración propia.

4.1.2 Test set

En este set, el registro de sensores de los ciclos de operación de las unidades se encuentra truncado en algún momento antes de que la unidad se declare como inoperativa. El objetivo es estimar la vida útil remanente de cada unidad de este set en base a la información entregada por las unidades del training set.

Para verificar las estimaciones, los valores de la vida útil remanente de cada unidad del test set son entregados con el data set.

4.2 Medidas de Desempeño en CMAPSS

Para estimar el desempeño del modelo en el data set CMAPSS se utilizan dos medidas para cuantificar el acierto de las predicciones realizadas.

La primera consiste en la función de costos que guía la optimización en función de la raíz de la distancia promedio cuadrada entre las predicciones y los labels de los registros.

La segunda consiste en la asignación de puntaje según una función de S*core* que se expondrá a continuación.

4.2.1 Root Mean Square Error (RMSE)

Esta medida corresponde a la función de costos del problema de optimización del modelo planteado y representa la medida de del error cuadrado. La utilización de esta medida permite cuantificar la diferencia negativa y positiva además de considerar que a medida que la predicción

se aleja del objetivo, el valor del error aumenta al cuadrado, por lo que los valores pequeños indican una muy buena aproximación.

En la ecuación (15) se muestra la función del error cuadrado calculada como la raíz del error promedio cuadrado de la cantidad de datos perteneciente al batch que se está cuantificando.

$$E = \sqrt{\frac{\sum(y(x) - y'(x))^2}{N}} \tag{15}$$

Donde $y(x)$ corresponde a los *labels* definidos para los datos de entrada y $y'(x)$ corresponde al vector con los valores de la predicción realizada por el modelo para cada dato de entrada.

En la Figura.15 se muestra la representación gráfica de esta medida y se aprecia como los valores son simétricos en torno a 0, además de su crecimiento según el cuadrado del error.

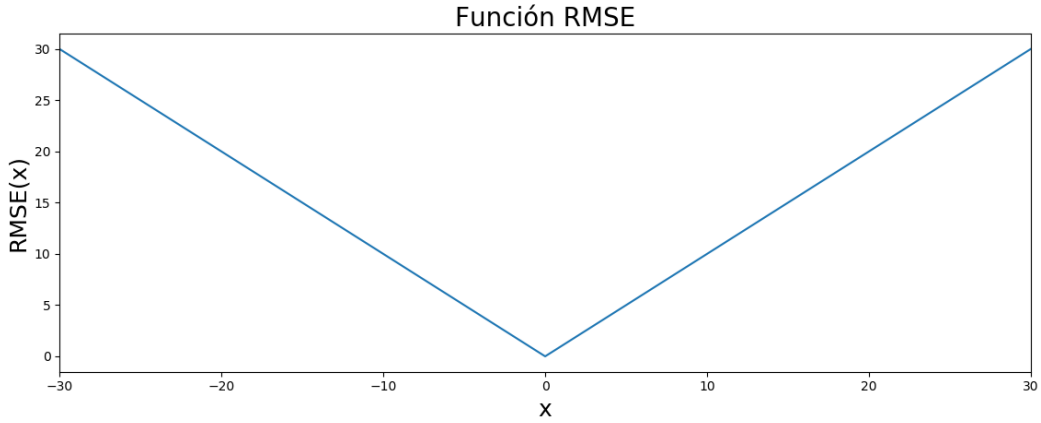

Figura 15:Gráfico con los valores del RMSE según el valor del error en la predicción. Fuente: Elaboración propia.

4.2.2 Score

Esta medida corresponde a un valor asignado a la certeza de la predicción en función de su cercanía y si corresponde a una subestimación o sobreestimación de la vida útil remanente. El cálculo del valor se realiza según la ecuación (16).

$$Score = \sum_{t=1}^{N} s_t$$

$$s_t = \begin{cases} e^{\frac{-d_t}{13}} - 1 \ \ para \ \ d_t < 0 \\ e^{\frac{-d_t}{10}} - 1 \ \ para \ \ d_t \geq 0 \end{cases} \tag{16}$$

$$d_t = RUL_{predicho} - RUL_{label}$$

Donde s_t es la Score de una predicción, $RUL_{predicho}$ es el valor de la vida remanente obtenido en base al modelo y RUL_{label} es el valor real de la vida remanente según los labels de los registros.

En la Figura.16 se muestra el gráfico de la función scoring, mostrando la diferencia entre las Scores asignadas a las predicciones tardías y las predicciones tempranas.

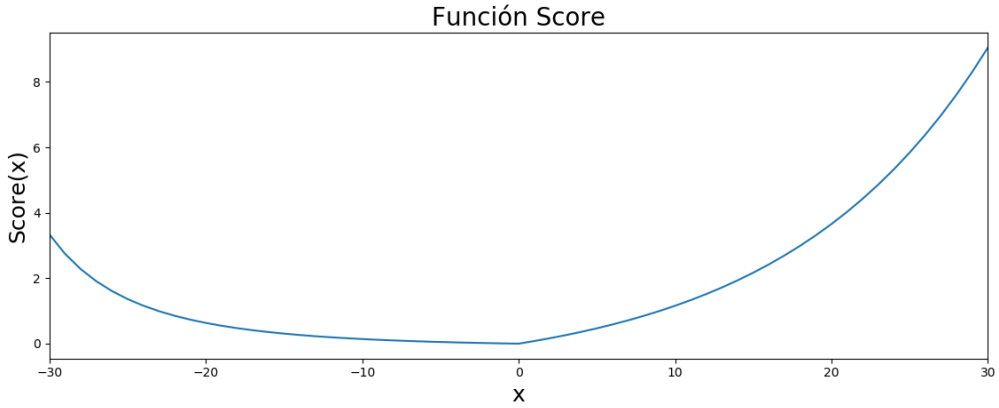

Figura 16: Función Score para CMAPSS. Fuente: Elaboración propia.

4.3 Preparación de la Base de Datos

De todos los sensores que registran la operación de las turbinas, se descartan algunos sensores, con la intención de evitar sobreajustes o biases en el entrenamiento del modelo que no sean representativos de la operación de la turbina.

Los registros descartados son los sensores 1, 5, 6, 10, 16, 18 y 19, porque, como se puede apreciar en la Figura.14, sus valores son constantes durante todos los ciclos de operación de las turbinas.

De esta forma, los sensores que se utilizarán para el entrenamiento y testeo del modelo son los 14 restantes.

4.3.1 Labels Registros

Los *labels* de los registros de las turbinas en CMAPSS corresponden al valor de la vida remanente en cada ciclo de tiempo del registro de operación de una unidad.

Para realizar el entrenamiento del modelo, los labels del training set han de ser definidos ya que son los valores que se utilizan para calcular el error de la función de costos y son los que guían y definen la optimización del modelo.

Los labels no son parte del data set, por lo que se deben definir a partir del conocimiento de cuando la turbina es declarada inoperativa.

Para definir los labels en cada ciclo de tiempo de los registros se utilizan un modelo de degradación no lineal de la turbina.

4.3.1.1 Degradación no lineal

El modelo de degradación no lineal considera que el deterioro en la turbina comienza a partir de un valor R_{early}, donde previo a ese valor, la turbina no sufre degradación y luego, cuando

queden esa cantidad de ciclos de operación, la turbina comenzará a deteriorarse linealmente hasta que es declarada inoperativa.

En la Figura.17 se muestra un gráfico con la degradación no lineal de una turbina con 200 ciclos de operación y un R_{early} igual a 125 ciclos, valor similar al que se utiliza para generar los labels en función de los estudiado por referencia [8].

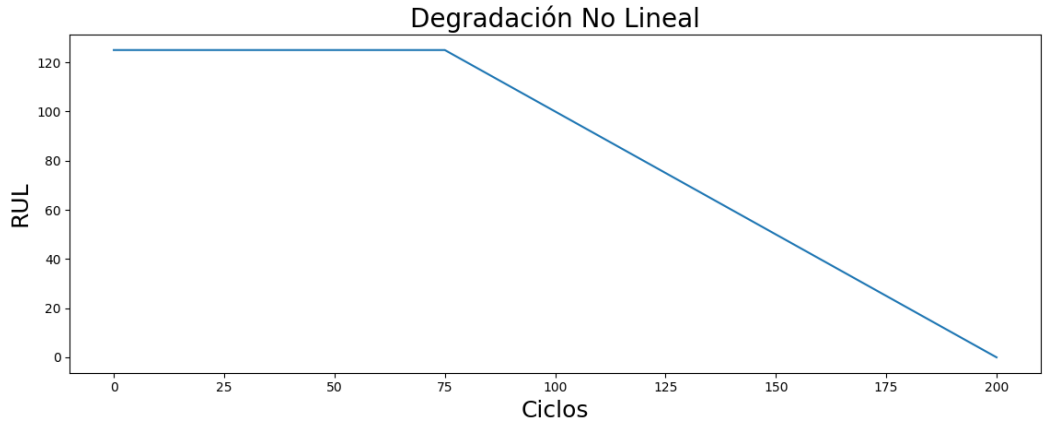

Figura 17: Gráfico con la vida remanente para una turbina según degradación no lineal. Fuente: Elaboración propia.

4.3.2 Normalización Registros

Como los sensores miden diferentes variables, las magnitudes de los valores de cada serie de datos difieren considerablemente. Para evitar problemas por sobreajustes en el entrenamiento del modelo, cada serie de registros se normaliza en función del valor máximo y mínimo de cada secuencia.

Los valores registrados de cada sensor se proyectan en el intervalo [0,1] según la ecuación (17).

$$\bar{x}_{i,j} = \frac{x_{i,j} - x_{i,mín}}{x_{i,máx} - x_{i,mín}} \tag{17}$$

Donde $\bar{x}_{i,j}$ corresponde al valor proyectado j del sensor i, $x_{i,j}$ corresponde al valor j del sensor i, $x_{i,máx}$ corresponde al valor máximo del sensor i y $x_{i,mín}$ corresponde al valor mínimo del sensor i.

4.3.3 Ventana Temporal Registros

Para generar más datos de entrenamiento a partir de los registros del training set se procede a utilizar una segmentación de los datos según una ventana temporal.

La ventana temporal escogida corresponde al largo mínimo del registro de una unidad de turbina en cada sub set del data set. De esta forma se evitan problemas por registros de operación más cortos que la ventana de tiempo y se maximiza la cantidad de datos en los que se podrá entrenar el modelo.

En la Tabla.2 se muestran los valores de ventana de tiempo utilizados para cada sub set de CMAPSS. El valor de la ventana de tiempo define el tamaño de la imagen que ingresa a la red neuronal, considerando los 14 sensores que se utilizan para el entrenamiento y testeo.

Tabla 2: Valores de la ventana de tiempo y la imagen utilizada en la segmentación de los registros.

Sub set	Ventana de Tiempo	Tamaño Imagen
FD001	30	30x14
FD002	21	21x14
FD003	30	30x14
FD004	19	19x14

4.3.4 Input Modelo

Finalmente, la ventana de tiempo con la información de los 14 sensores que se utilizarán y que han sido pre-procesados definen una imagen con la que se alimenta la CNN causal.

En la Figura.18 se muestra un ejemplo de unas imágenes con los 14 sensores para una ventana de tiempo de la turbina perteneciente al data set FD001 y que se muestra en la Figura.6.

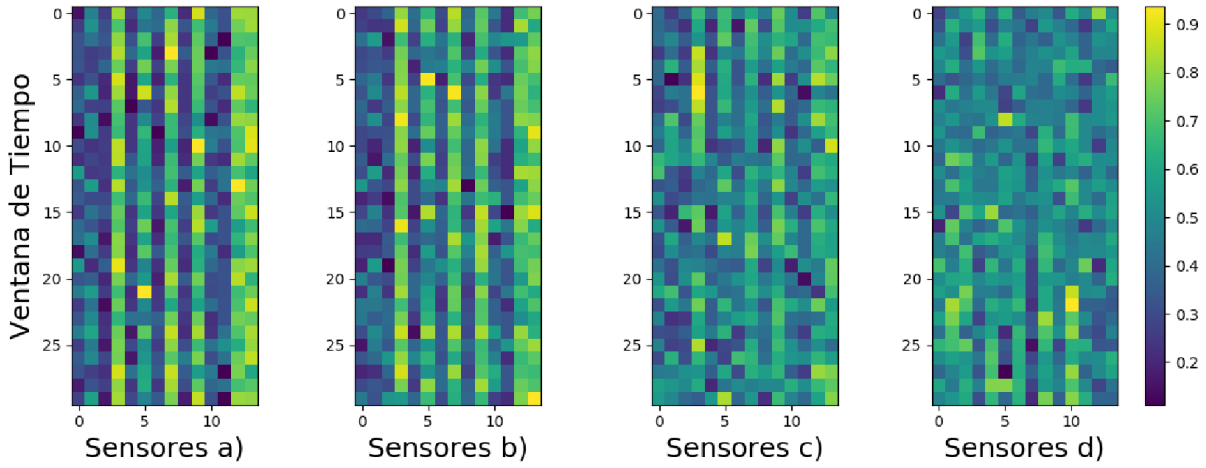

Figura 18: Imágenes de entrada a la CNN, donde a) es el registro de los sensores al inicio del ciclo de operación y d) al final de la operación. Las imágenes b) y c) corresponden a las etapas intermedias. Fuente: Elaboración propia.

5 Resultados

5.1 Grid Searchs Relevantes

El método *Grid Search* corresponde al método tradicional para el ajuste de los hiper-parámetros de la arquitectura y del entrenamiento.

El método es simplemente una búsqueda exhaustiva variando los valores y combinaciones de los hiper-parámetros en búsqueda de alguna combinación de hiper-parámetros óptima que minimice la función de costos.

A continuación, se muestra una porción de los diferentes grid search que fueron ejecutados, con los parámetros utilizados y los resultados obtenidos. Los grid searchs mostrados corresponden a la validación de la implementación de una CNN causal.

5.1.1 Grid Search Causalidad

Para estudiar la funcionalidad en la aplicación de la causalidad en la generación de predicciones y entrenamiento del modelo se propone la realización de un grid search con varios modelos donde se compara la misma arquitectura con la implementación de causalidad y sin ésta.

En la Tabla.3 y la Tabla.4 se presentan los parámetros que se utilizaron para realizar el grid search, considerando las variables relevantes y los parámetros respectivos de la arquitectura y del entrenamiento.

Tabla 3: Parámetros arquitectura del grid search de causalidad.

Estructura	Hiper-Parámetro	Valores probados
Capas de Convolución	Cantidad de capas de convolución	3, 4, 5
	Features de cada capa	8, 16, 32
	Kernels de convolución	10x1
	Dilatación del kernel de convolución	1, 2, 3
	Causalidad de la convolución	Si y No
Stacks de GU	Cantidad de celdas	0
	Kernel de convolución de la celda	-
	Dilatación kernel de la celda	-
	Causalidad de la covolución	-
Capas Completamente Conectadas	Cantidad de capas	2, 3, 4
	Neuronas de cada capa	32, 128, 512

Tabla 4: Parámetros entrenamiento del grid search de causalidad.

Hiper-Parámetros Entrenamiento	Valores Probados
Keep probability	0.6
Tamaño batches de entrenamiento	256
Ratio de regularización	0, 0.001
Épocas de entrenamiento	300

Para cada combinación de parámetros el modelo se aplica en la base de datos FD001 del data set y se calcula el RMSE y el Score respectivo.

A partir de los valores obtenidos con el modelo se genera una serie de gráficos que se muestran en la Figura.19. Los gráficos mostrados corresponden al cálculo del valor promedio de una porción del total de los valores ordenados de mayor a menor según el valor del Score causal. De esta forma, a la izquierda de los gráficos se muestran los valores asociados con el menor Score causal y a la derecha se muestra el promedio de los resultados obtenidos por el grid search, donde el cálculo de los valores se hizo para la porción completa de los datos.

De esta forma, con los gráficos presentados, se puede extraer información sobre cómo varían los parámetros de los mejores valores obtenidos con respecto a los valores promedio obtenidos a partir del grid search.

En los gráficos, el índice *'m'* en la nomenclatura indica que el valor fue calculado incluyendo causalidad. Los valores se ordenan según el valor del Score obtenido con causalidad, porque se espera que, dado que la causalidad disminuye la cantidad de información disponible para la predicción, los valores de RMSE sean un poco peores, mientras que, por la causalidad, las predicciones quedan supeditadas a los valores anteriores, incidiendo en la reducción del Score al permitir que las predicciones de falla se realicen con anterioridad.

En este caso, el gráfico más relevante corresponde a la comparación entre '*Score con y sin máscara'*. Este gráfico presenta la diferencia entre las Scores obtenidas con y sin causalidad.

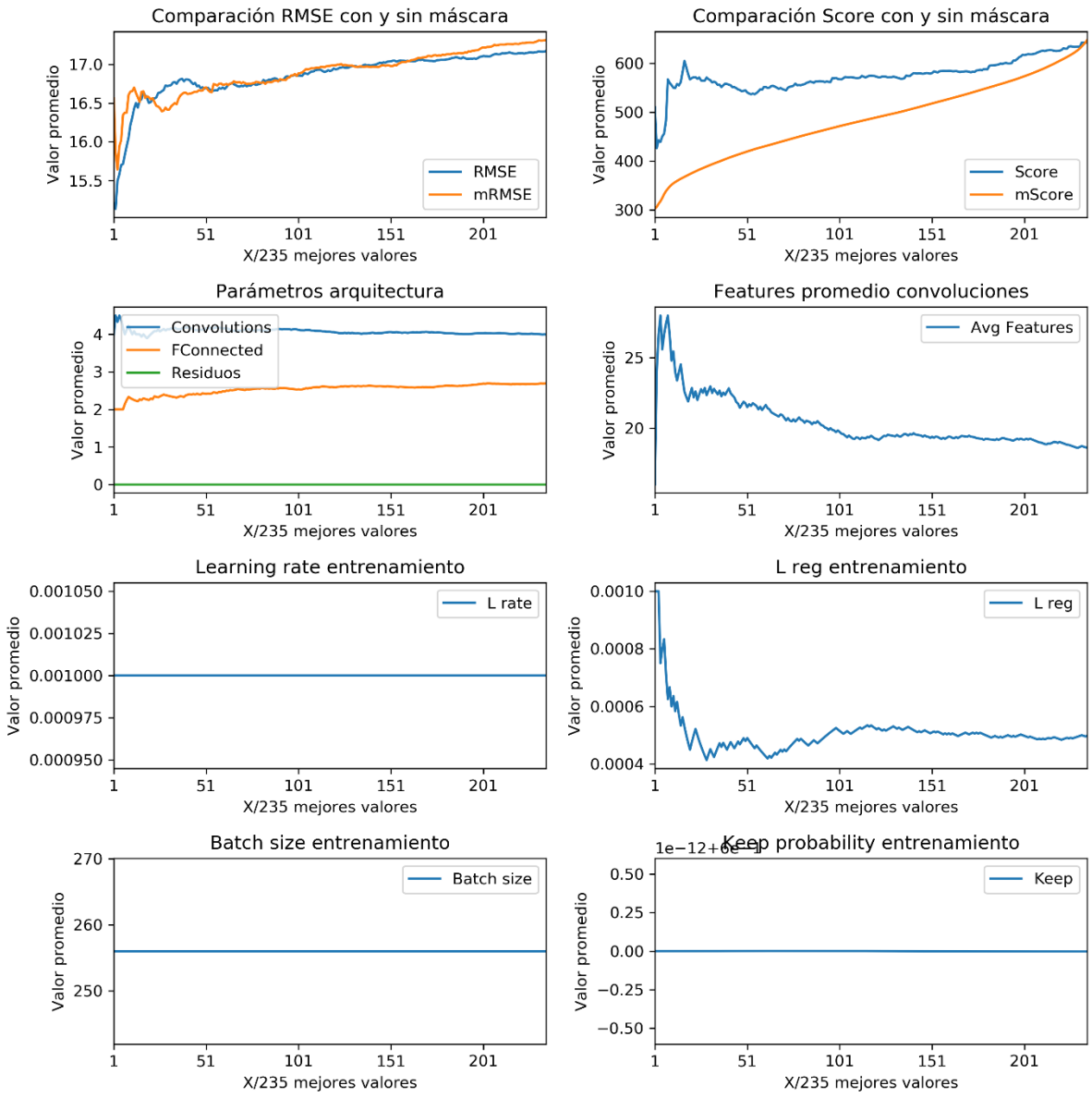

Figura 19: Gráficos con los resultados del grid search de causalidad. Fuente: Elaboración propia.

5.1.2 Grid Search Residuos

Para estudiar la funcionalidad en la aplicación de las celdas de residuos se propone la realización de un grid search donde se estudia el comportamiento de los resultados con y sin stack de celdas, además de variar la cantidad de celdas con las que se implementa el modelo.

En la Tabla.5 se presentan los parámetros de la arquitectura con los que se realizó el grid search y en la Tabla.6 se muestran los parámetros del entrenamiento con los que se realizó el entrenamiento en el grid search.

Tabla 5:Parámetros arquitectura del grid search de residuos.

Estructura	Hiper-Parámetro	Valores probados
Capas de Convolución	Cantidad de capas de convolución	5, 6
	Features de cada capa	32, 64
	Kernels de convolución	5x1
	Dilatación del kernel de convolución	1, 2, 3
	Causalidad de la convolución	Si y No
Stacks de GU	Cantidad de celdas	0, 3, 5
	Kernel de convolución de la celda	5x1
	Dilatación kernel de la celda	1
	Causalidad de la covolución	Si y No
Capas Completamente Conectadas	Cantidad de capas	2, 3
	Neuronas de cada capa	64, 256, 512

Tabla 6:Parámetros entrenamiento del grid search de residuos.

Hiper-Parámetros Entrenamiento	Valores Probados
Keep probability	0.6
Tamaño batches de entrenamiento	256
Ratio de regularización	0, 0.001
Épocas de entrenamiento	300

Para cada combinación de los parámetros del modelo, se aplica el modelo a las cuatro bases de datos del data set *CMAPSS.*

Los resultados se presentan de la misma forma que en la sección anterior, donde los gráficos de la Figura.20 presentan los resultados obtenidos como el valor promedio de los parámetros para una porción del total de los datos, ordenados de menor a mayor según la Score causal de la base de datos FD001.

En este caso, el gráfico más relevante corresponde al de '*Parámetros arquitectura según FD001 mScore'*, donde se puede apreciar cómo evoluciona la cantidad de residuos según la calidad de los resultados.

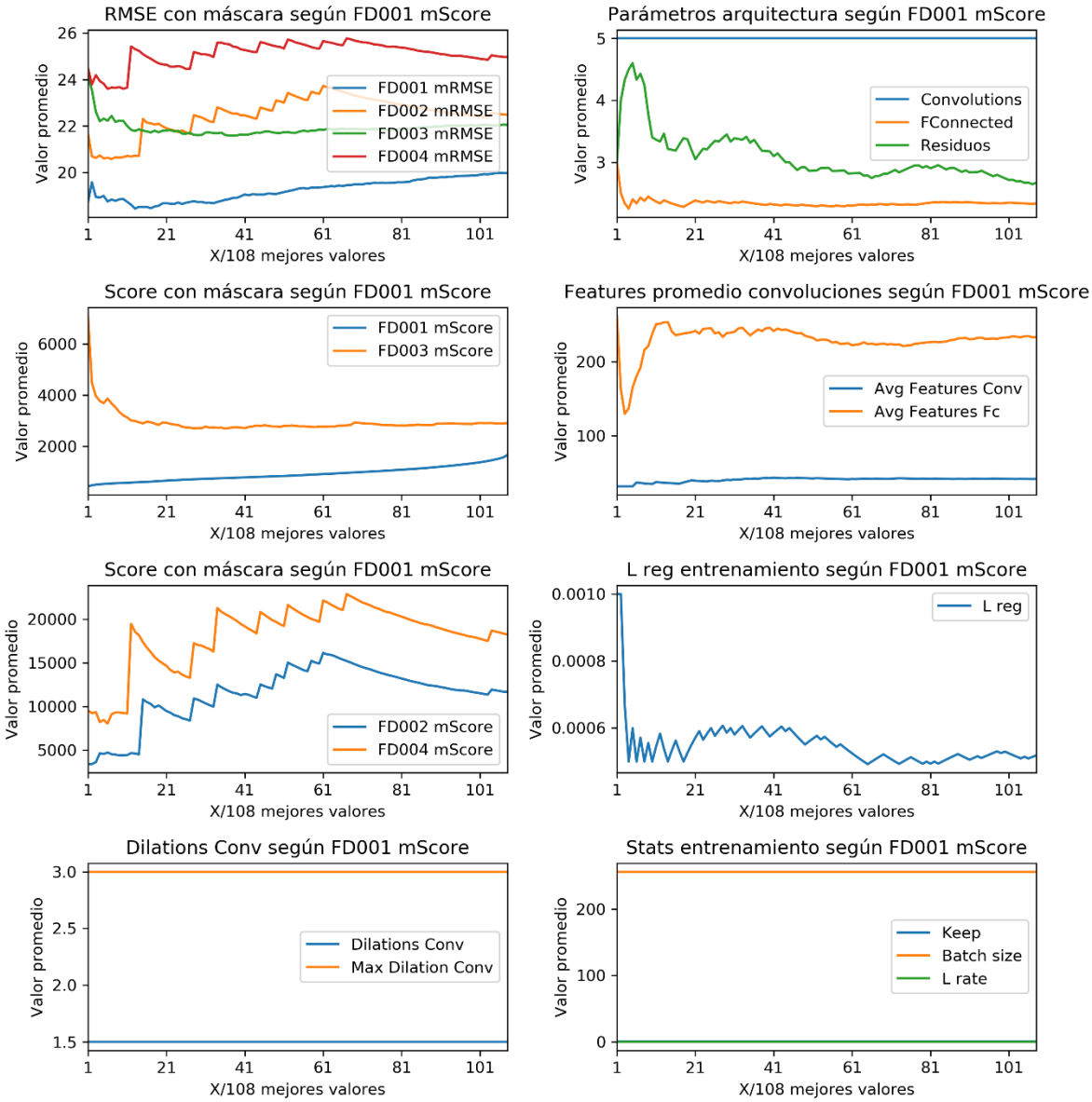

Figura 20: Gráficos con los resultados del grid search de residuos. Fuente: Elaboración propia.

5.1.3 Grid Search Dilatación

Para estudiar la funcionalidad en la aplicación de la dilatación en la convolución se realiza un grid search que compare los resultados por un mismo modelo con y sin dilatación.

En la Tabla.7 se presentan los parámetros de la arquitectura con los que se realiza el grid search y en la Tabla.8 se presentan los parámetros de entrenamiento.

Tabla 7: Parámetros arquitectura del grid search de dilatación.

Estructura	Hiper-Parámetro	Valores probados
Capas de Convolución	Cantidad de capas de convolución	3, 4, 5, 6
	Features de cada capa	16
	Kernels de convolución	9x1
	Dilatación del kernel de convolución	1, 2, 3
	Causalidad de la convolución	Si y No
Stacks de GU	Cantidad de celdas	3
	Kernel de convolución de la celda	7x1
	Dilatación kernel de la celda	1
	Causalidad de la covolución	Si y No
Capas Completamente Conectadas	Cantidad de capas	3
	Neuronas de cada capa	32, 64, 256, 512

Tabla 8: Parámetros entrenamiento del grid search de dilatación.

Hiper-Parámetros Entrenamiento	Valores Probados
Keep probability	0.6
Tamaño batches de entrenamiento	256, 512
Ratio de regularización	0, 0.001
Épocas de entrenamiento	300

Para cada combinación de los parámetros del modelo, se aplica el modelo a las cuatro bases de datos del data set *CMAPSS.*

Los resultados se presentan de la misma forma que en las secciones anteriores. Los gráficos de la Figura.21 presentan los resultados obtenidos como el valor promedio de los parámetros para una porción del total de los datos, ordenados de menor a mayor según la Score causal de la base de datos FD001.

En esta sección, el gráfico más relevante corresponde al de '*Dilations Conv según FD001 mScore'*, donde se puede apreciar cómo evoluciona la aplicación de la dilatación según la calidad de los resultados.

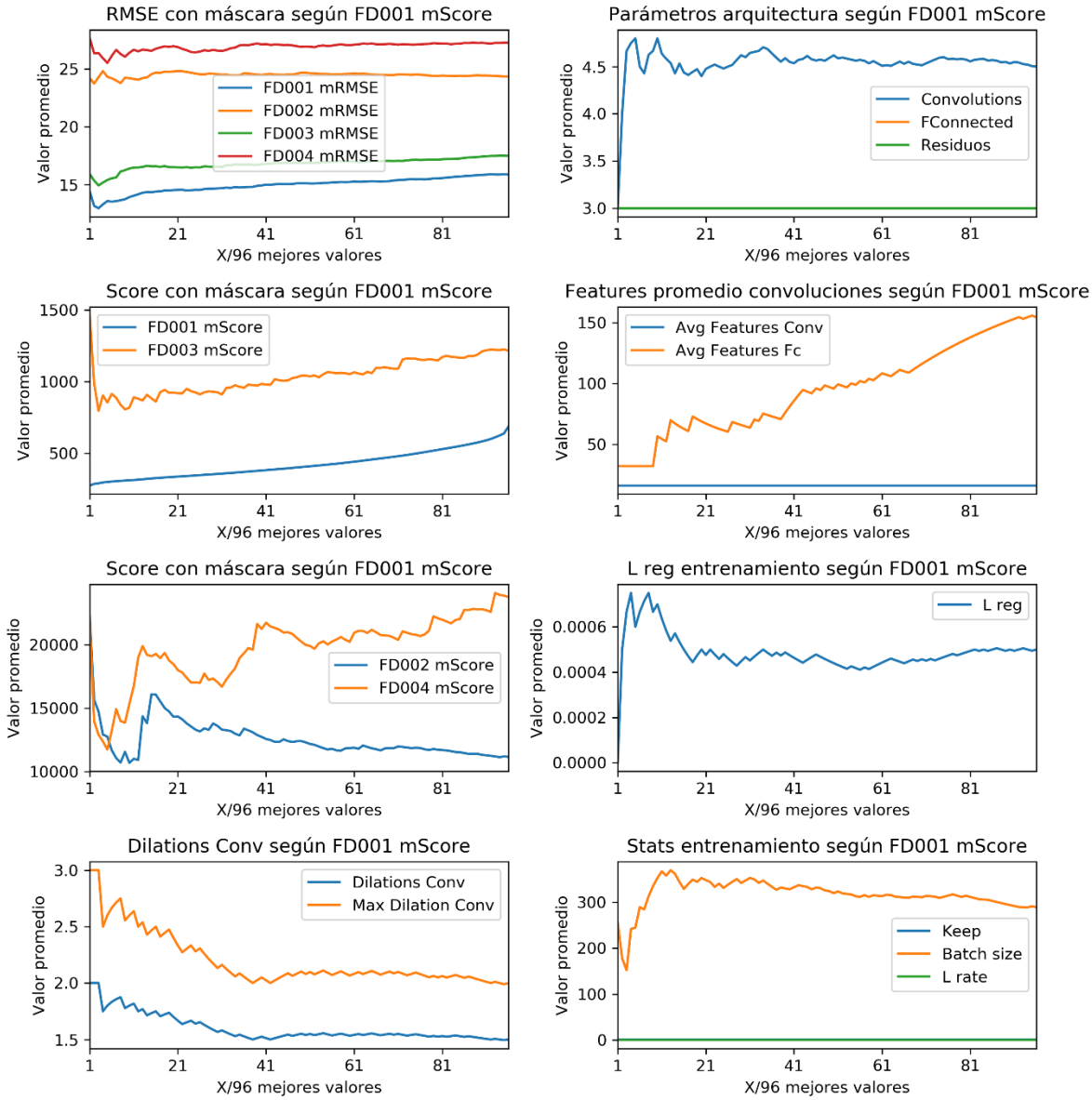

Figura 21: Gráficos con los resultados del grid search de dilatación. Fuente: Elaboración propia.

5.2 Arquitectura Modelo

A partir de los grid searchs anteriores y otros varios realizados en los demás parámetros que no se muestran por no ser relevantes, se llega a definir la arquitectura del modelo para cada una de las bases de datos del data set CMAPSS.

Los valores definen la arquitectura que se implementó para obtener los resultados finales de RMSE y Score implementando causalidad, dilatación y residuos.

5.2.1 Arquitectura FD001

Los parámetros de la arquitectura definida para FD001 se presentan en la Tabla.9 y los parámetros para el entrenamiento en la Tabla.10.

Tabla 9: Parámetros arquitectura FD001.

Estructura	Hiper-Parámetro	Valores probados
Capas de Convolución	Cantidad de capas de convolución	4
	Features de cada capa	8
	Kernels de convolución	9x1
	Dilatación del kernel de convolución	1, 2, 3
	Causalidad de la convolución	Si
Stacks de GU	Cantidad de celdas	5
	Kernel de convolución de la celda	7x1
	Dilatación kernel de la celda	1
	Causalidad de la covolución	Si
Capas Completamente Conectadas	Cantidad de capas	3
	Neuronas de cada capa	32

Tabla 10: Parámetros entrenamiento FD001.

Hiper-Parámetros Entrenamiento	Valores Probados
Keep probability	0.6
Tamaño batches de entrenamiento	100
Ratio de regularización	0.001
Épocas de entrenamiento	300

Los parámetros de la arquitectura definen la estructura del modelo que se implementará. En la Figura.22 se aprecia un esquema simple incluyendo las capas de convolución las celdas residuales y las capas completamente conectadas que finalmente llevan al output en la predicción del modelo a partir de la imagen inicial que se mostró en la sección respectiva.

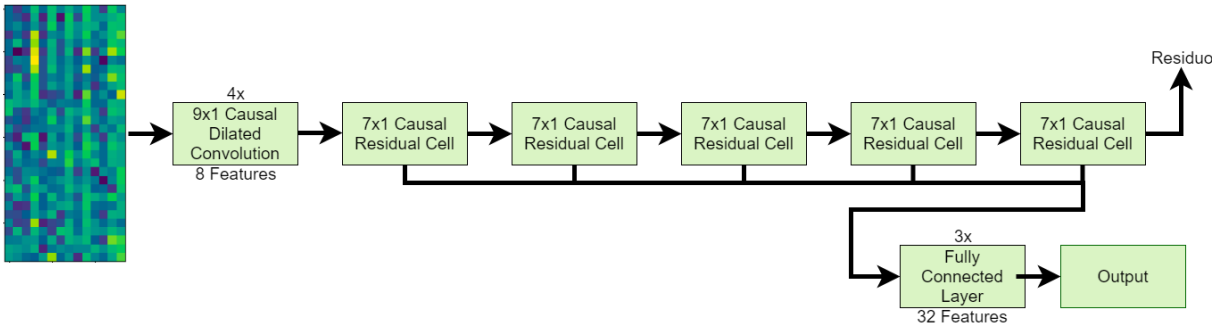

Figura 22: Arquitectura FD001. Fuente: Elaboración propia.

Luego, en la Figura.23 se muestra el filtro con máscara que se utiliza para la convolución, donde la dilatación crece a medida que se profundiza el modelo. Este filtro corresponde al filtro con mejores resultados en los grid searchs realizados, además de contar con referencias para la utilización de filtro de 10x1 [8], siendo el de 9x1 la misma cantidad de pesos cuando se aplica la causalidad.

El filtro tiene ancho unitario porque se asume que no hay una relación espacial entre sensores que permita extraer información valiosa.Los filtros con dilatación que se muestran en la figura no incluyen la dilatación entre los valores que se hacen 0 gracias a la máscara que aplica causalidad.

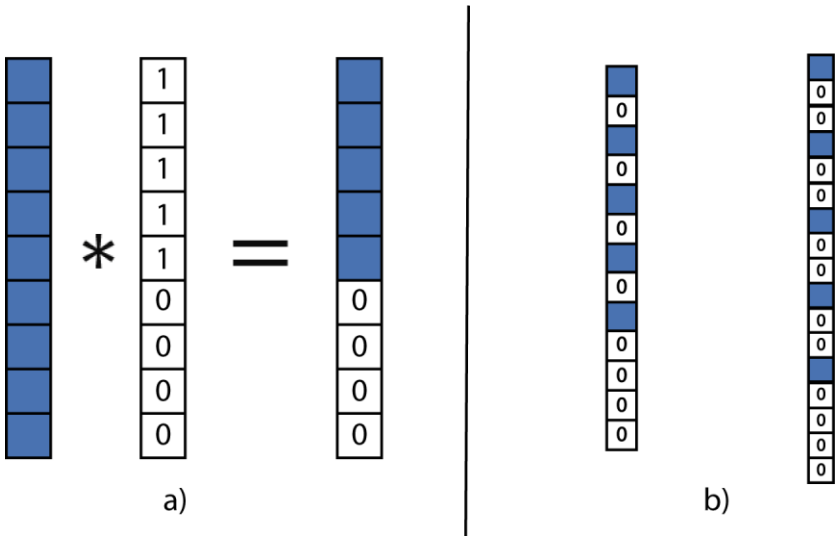

Figura 23: Kernels utilizados. a) Filtro con máscara para aplicar causalidad. b) Filtro con dilatación 2 a la izquierda y filtro con dilatación 3 a la derecha. Fuente: Elaboración propia.

5.2.2 Arquitectura FD002

Los parámetros de la arquitectura definida para FD002 se presentan en la Tabla.11 y los parámetros para el entrenamiento en la Tabla.12.

Tabla 11: Parámetros arquitectura FD002.

Estructura	Hiper-Parámetro	Valores probados
Capas de Convolución	Cantidad de capas de convolución	4
	Features de cada capa	16
	Kernels de convolución	9x1
	Dilatación del kernel de convolución	1, 2, 3
	Causalidad de la convolución	Si
Stacks de GU	Cantidad de celdas	5
	Kernel de convolución de la celda	7x1
	Dilatación kernel de la celda	1
	Causalidad de la covolución	Si
Capas Completamente Conectadas	Cantidad de capas	3
	Neuronas de cada capa	512,256,128

Tabla 12: Parámetros entrenamiento FD002.

Hiper-Parámetros Entrenamiento	Valores Probados
Keep probability	0.6
Tamaño batches de entrenamiento	100
Ratio de regularización	0
Épocas de entrenamiento	300

En la Figura.24 se presenta un diagrama simplificado de la arquitectura, similar al mostrado en arquitectura FD001, pero en este caso cambian las features de las capas de convolución y de las capas completamente conectadas. El aumento en las features conlleva una mayor capacidad para modelar comportamientos más complejos.

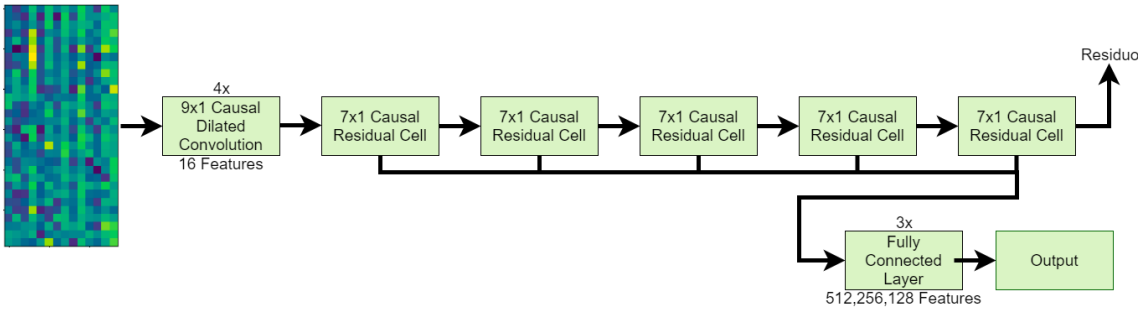

Figura 24:Arquitectura FD002. Fuente: Elaboración propia.

Los filtros de esta y de las demás arquitecturas son idénticos a los mostrados en la Figura.23 de la sección arquitectura FD001.

5.2.3 Arquitectura FD003

Los parámetros de la arquitectura definida para FD003 se presentan en la Tabla.13 y los parámetros para el entrenamiento en la Tabla.14.

Tabla 13: Parámetros arquitectura FD003.

Estructura	Hiper-Parámetro	Valores probados
Capas de Convolución	Cantidad de capas de convolución	4
	Features de cada capa	8
	Kernels de convolución	9x1
	Dilatación del kernel de convolución	1, 2, 3
	Causalidad de la convolución	Si
Stacks de GU	Cantidad de celdas	3
	Kernel de convolución de la celda	7x1
	Dilatación kernel de la celda	1
	Causalidad de la covolución	Si
Capas Completamente Conectadas	Cantidad de capas	3
	Neuronas de cada capa	32

Tabla 14: Parámetros entrenamiento FD003.

Hiper-Parámetros Entrenamiento	Valores Probados
Keep probability	0.6
Tamaño batches de entrenamiento	100
Ratio de regularización	0
Épocas de entrenamiento	300

En la Figura.25 se presenta un diagrama con la arquitectura implementada en FD003, donde se aprecia que, a diferencia de las arquitecturas anteriores, la cantidad de celdas de residuo se redujo a tres, mientras que los parámetros de las capas de convolución y de las capas completamente conectadas son similares a las de la arquitectura FD001.

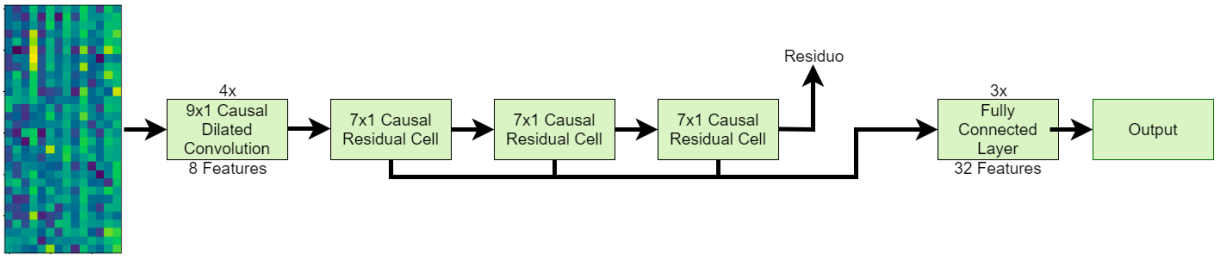

Figura 25: Arquitectura FD003. Fuente: Elaboración propia.

5.2.4 Arquitectura FD004

Los parámetros de la arquitectura definida para FD004 se presentan en la Tabla.15 y los parámetros para el entrenamiento en la Tabla.16.

Tabla 15: Parámetros arquitectura FD004.

Estructura	Hiper-Parámetro	Valores probados
Capas de Convolución	Cantidad de capas de convolución	4
	Features de cada capa	16
	Kernels de convolución	9x1
	Dilatación del kernel de convolución	1, 2, 3
	Causalidad de la convolución	Si
Stacks de GU	Cantidad de celdas	3
	Kernel de convolución de la celda	7x1
	Dilatación kernel de la celda	1
	Causalidad de la covolución	Si
Capas Completamente Conectadas	Cantidad de capas	3
	Neuronas de cada capa	512, 256, 128

Tabla 16: Parámetros arquitectura FD004.

Hiper-Parámetros Entrenamiento	Valores Probados
Keep probability	0.6
Tamaño batches de entrenamiento	100
Ratio de regularización	0
Épocas de entrenamiento	300

En la Figura.26 se presenta el diagrama con la arquitectura implementada en la base de datos FD004, donde se aprecia que la estructura de las capas de convolución y capas completamente conectadas es similar a la arquitectura implementada en FD002, pero la cantidad de las celdas de residuo se reduce a tres.

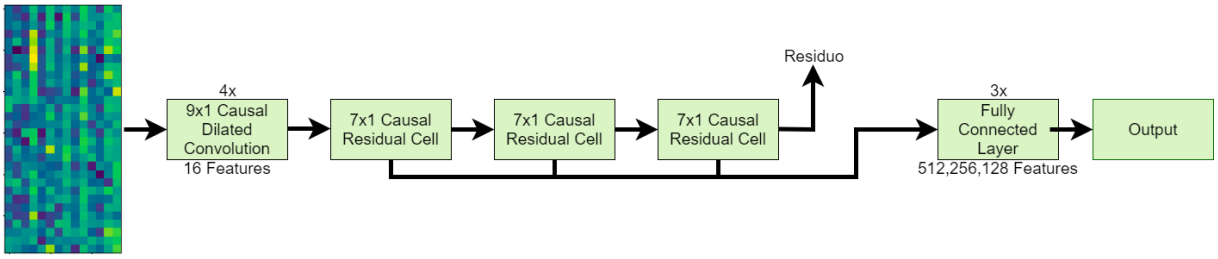

Figura 26: Arquitectura FD004. Fuente: Elaboración propia.

5.3 Predicciones Modelo

En las secciones siguientes se presentan los resultados en cuanto al RMSE y al Score obtenidos por la aplicación de las arquitecturas del modelo definidas para cada base de datos del data set CMAPSS.

A pesar de que la arquitectura se define para una base de datos, el modelo se implementa en las cuatro para estudiar el comportamiento y definir si existe la posibilidad de implementar un modelo que realice las predicciones de forma general.

Los resultados mostrados corresponden a cinco iteraciones del modelo en cada base de datos en el caso con causalidad y sin causalidad, por lo que también se presentan los resultados promedio obtenidos.

A partir de las cinco iteraciones de los modelos se escoge la mejor iteración con causalidad para la base de datos que se definió la arquitectura y se presenta un gráfico con las estimaciones de la vida útil remanente en el test set del data set CMAPSS.

5.3.1 FD001

El data set FD001 corresponde a turbinas que operan en una única condición de operación y que presentan un único modo de falla. Siendo la base de datos más simple de las cuatro estudiadas los resultados obtenidos son los mejores en comparación a las otras bases de datos y las predicciones del modelo se ajustan de buena forma a los valores del test set.

Primero, en la Tabla.17 se presentan los valores promedio obtenidos para las cuatro bases de datos en el caso de que las convoluciones se realizan con causalidad. A modo de comparación, en la Tabla.18 se presentan los valores promedio obtenidos para el caso sin causalidad. Finalmente, en la Tabla.19 se presentan los valores obtenidos para las cinco iteraciones del modelo en cada data set considerando únicamente el caso con causalidad.

Tabla 17: Valores de los resultados promedio obtenidos con causalidad para FD001.

	FD001	FD002	FD003	FD004
RMSE	**14,8**	25,8	16,7	25,6
Score	**401,8**	12331,8	953,6	10393,0
Tiempo [s]	**450,3**	1138,7	470,1	1152,4

Tabla 18: Valores de los resultados promedio obtenidos sin causalidad para FD001.

	FD001	FD002	FD003	FD004
RMSE	**14,3**	25,7	16,7	29,9
Score	**434,8**	14969,1	1263,8	45814,2
Tiempo [s]	**353,1**	759,602	393,653	1010,9

Tabla 19: Valores de los resultados obtenidos para cada iteración con causalidad para FD001.

FD001	**Iteración 1**	**Iteración 2**	**Iteración 3**	**Iteración 4**	**Iteración 5**
RMSE	**13,7**	**15,5**	**14,7**	**15,9**	**14,3**
Score	**334,8**	**616,3**	**273,7**	**441,9**	**342,1**
FD002	**Iteración 1**	**Iteración 2**	**Iteración 3**	**Iteración 4**	**Iteración 5**
RMSE	26,8	23,8	27,6	24,9	25,8
Score	10274,0	6133,3	24157,9	8280,0	12813,7
FD003	**Iteración 1**	**Iteración 2**	**Iteración 3**	**Iteración 4**	**Iteración 5**
RMSE	16,2	16,7	17,1	17,3	16,5
Score	1023,3	703,7	1151,3	745,0	1144,7
FD004	**Iteración 1**	**Iteración 2**	**Iteración 3**	**Iteración 4**	**Iteración 5**
RMSE	24,3	25,6	28,4	26,3	23,4
Score	6673,8	10372,6	12593,6	16463,4	5861,4

A partir de los valores expuestos se escoge el mejor valor obtenido en relación con el Score y se presenta en la Tabla.20.

Tabla 20: Mejor iteración obtenida para el data set FD001.

	FD001
RMSE	14,7
Score	273,7
Tiempo [s]	445,6

Finalmente, en la Figura.27 se presenta un gráfico con las predicciones realizadas por el modelo de la Tabla.20, en comparación con los valores del test set. A partir de la imagen se puede apreciar el error y el comportamiento de las predicciones.

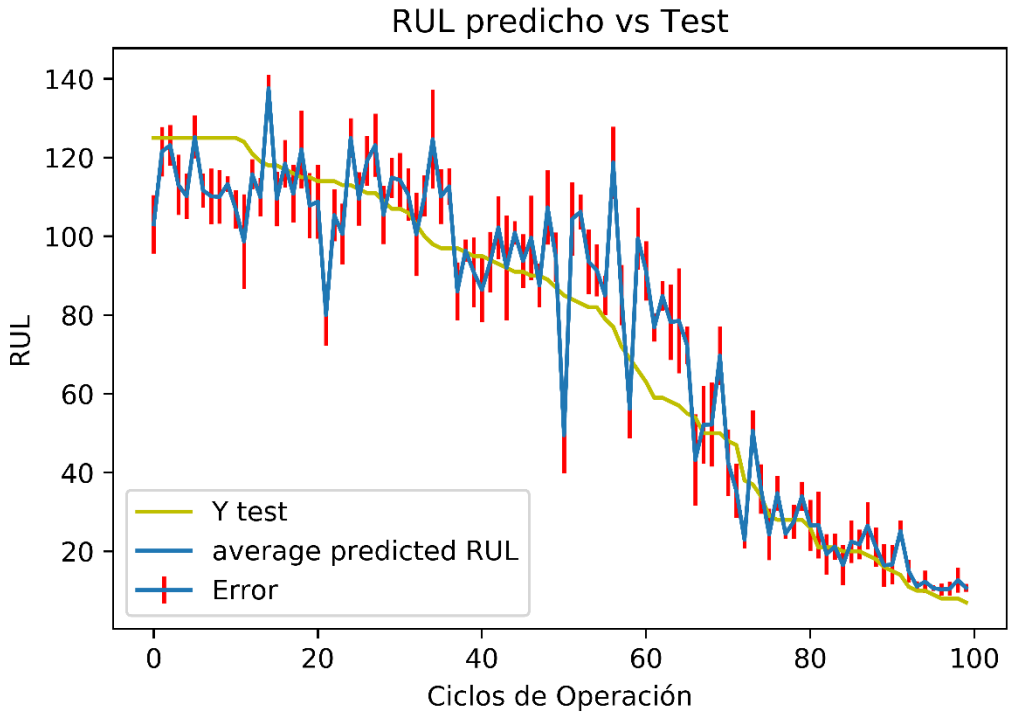

Figura 27: Predicciones del RUL vs test set en FD001. Fuente: Elaboración propia.

5.3.2 FD002

El data set FD002 corresponde a turbinas que operan en seis diferentes condiciones de operación y que presentan un único modo de falla. Esta base de datos es la segunda más compleja, obteniendo predicciones que no se ajustan de muy buena forma a los datos del data set de testeo.

Inicialmente, en la Tabla.21 se presentan los valores promedio obtenidos para las cuatro bases de datos utilizando la arquitectura que mejores resultados obtuvo en FD002. Los resultados consideran los valores obtenidos con convoluciones con causalidad. Luego, en la Tabla.22 se presentan los resultados obtenidos para las convoluciones sin causalidad y finalmente en la Tabla.23 se presentan los valores obtenidos para las cinco iteraciones del modelo en cada data set considerando únicamente el caso con causalidad.

Tabla 21: Valores de los resultados promedio obtenidos con causalidad para FD002.

	FD001	FD002	FD003	FD004
RMSE	17,4	**24,1**	19,9	27,5
Score	695,1	**6907,0**	2018,8	11005,0
Tiempo [s]	571,5	**1528,0**	731,7	1772,4

Tabla 22: Valores de los resultados promedio obtenidos sin causalidad para FD002.

	FD001	FD002	FD003	FD004
RMSE	16,5	**22,7**	18,4	26,5
Score	600,0	**8680,7**	1067,7	17781,1
Tiempo [s]	523,9	**1214,9**	651,0	1412,4

Tabla 23: Valores de los resultados obtenidos para cada iteración con causalidad para FD002.

FD001	**Iteración 1**	**Iteración 2**	**Iteración 3**	**Iteración 4**	**Iteración 5**
RMSE	16,3	17,4	18,7	18,0	16,4
Score	546,0	836,0	610,1	680,3	803,0
FD002	**Iteración 1**	**Iteración 2**	**Iteración 3**	**Iteración 4**	**Iteración 5**
RMSE	**23,0**	**24,7**	**26,1**	**23,7**	**23,1**
Score	**3322,1**	**10038,0**	**4847,9**	**8325,2**	**8001,7**
FD003	**Iteración 1**	**Iteración 2**	**Iteración 3**	**Iteración 4**	**Iteración 5**
RMSE	20,1	19,7	21,0	20,3	18,3
Score	1785,8	952,7	2760,1	2854,7	1740,5
FD004	**Iteración 1**	**Iteración 2**	**Iteración 3**	**Iteración 4**	**Iteración 5**
RMSE	31,0	25,5	29,9	26,2	25,2
Score	11472,2	11454,7	9428,3	12248,6	10421,3

Desde los valores expuestos se escoge la iteración con mejor Score en el data set FD002 y se muestra en la Tabla.24.

Tabla 24: Mejor iteración obtenida para el data set FD002.

	FD002
RMSE	**23,0**
Score	**3322,1**
Tiempo [s]	**1536,2**

En la Figura.28 se muestra un gráfico que compara la vida útil remanente predicha por el modelo propuesto, en la mejor iteración obtenida, en relación con los valores esperados para cada dato de testeo. En este caso no se agregan las barras de error debido a que la magnitud del error es muy grande y no se logra apreciar en la imagen.

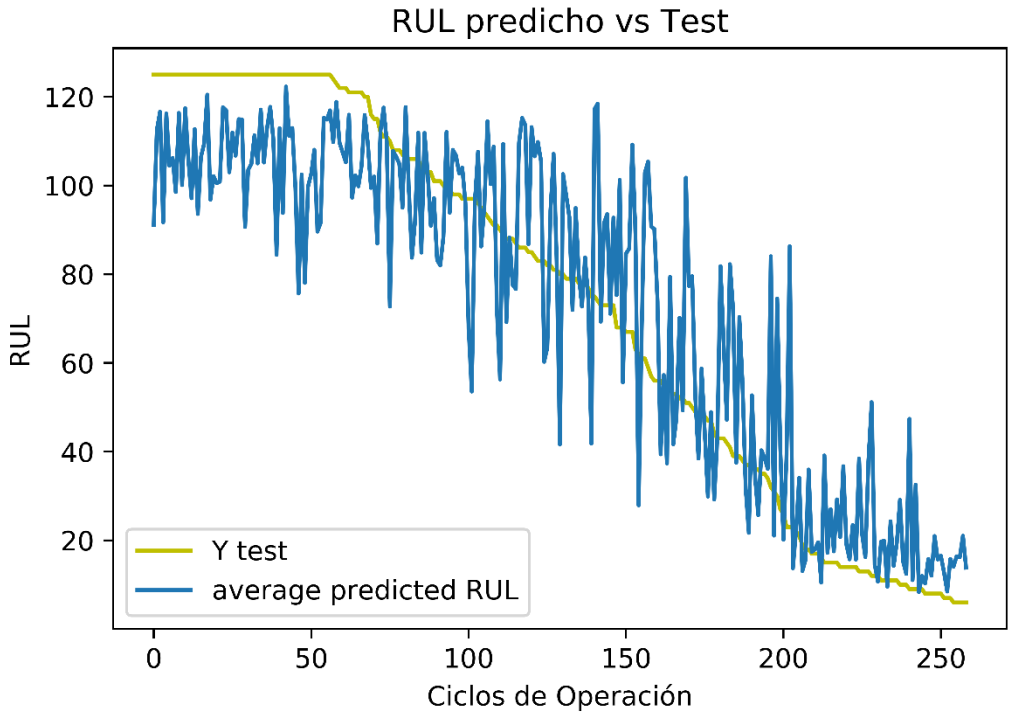

Figura 28: Predicciones del RUL vs test set en FD002. Fuente: Elaboración propia.

5.3.3 FD003

El data set FD003 corresponde a turbinas que operan en una condición de operación y que presentan dos modos de falla. Esta base de datos es la segunda más simple de predecir en base a los resultados, obteniendo predicciones que se ajustan de buena forma a los datos del data set de testeo.

Inicialmente, en la Tabla.25 se presentan los valores promedio obtenidos para las cuatro bases de datos utilizando la arquitectura que mejores resultados obtuvo en FD003. Los resultados consideran los valores obtenidos con convoluciones con causalidad. Luego, en la Tabla.26 se presentan los resultados obtenidos para las convoluciones sin causalidad y finalmente en la

Tabla.27 se presentan los valores obtenidos para las cinco iteraciones del modelo en cada data set considerando únicamente el caso con causalidad.

Tabla 25: Valores de los resultados promedio obtenidos con causalidad para FD003.

	FD001	FD002	FD003	FD004
RMSE	14,7	28,3	**17,4**	32,0
Score	420,8	23113,9	**876,3**	63178,3
Tiempo [s]	407,3	1167,9	**510,0**	1357,8

Tabla 26: Valores de los resultados promedio obtenidos sin causalidad para FD003.

	FD001	FD002	FD003	FD004
RMSE	16,2	28,1	**17,4**	30,7
Score	447,5	17747,9	**1241,8**	52422,9
Tiempo [s]	323,7	990,9	**416,9**	1046,3

Tabla 27: Valores de los resultados obtenidos para cada iteración con causalidad para FD003.

FD001	**Iteración 1**	**Iteración 2**	**Iteración 3**	**Iteración 4**	**Iteración 5**
RMSE	14,7	15,0	15,2	14,1	14,5
Score	542,5	306,4	475,4	397,5	382,2
FD002	**Iteración 1**	**Iteración 2**	**Iteración 3**	**Iteración 4**	**Iteración 5**
RMSE	27,3	30,7	29,5	26,3	27,9
Score	13593,5	17045,2	38916,6	23209,2	22805,2
FD003	**Iteración 1**	**Iteración 2**	**Iteración 3**	**Iteración 4**	**Iteración 5**
RMSE	**16,4**	**17,0**	**18,5**	**17,4**	**17,5**
Score	**764,2**	**1137,4**	**532,3**	**1182,8**	**764,9**
FD004	**Iteración 1**	**Iteración 2**	**Iteración 3**	**Iteración 4**	**Iteración 5**
RMSE	29,4	37,4	30,1	36,2	26,9
Score	28601,5	117024,9	57699,3	85971,3	26594,4

En la Tabla.28 se presenta el mejor resultado obtenido con la arquitectura en la base de datos FD003.

Tabla 28: Mejor iteración obtenida para el data set FD003.

	FD003
RMSE	18,5
Score	532,3
Tiempo [s]	510,3

En la Figura.29 se presenta un gráfico con las predicciones realizadas por el modelo de la Tabla.28 incluyendo las barras de error en las predicciones.

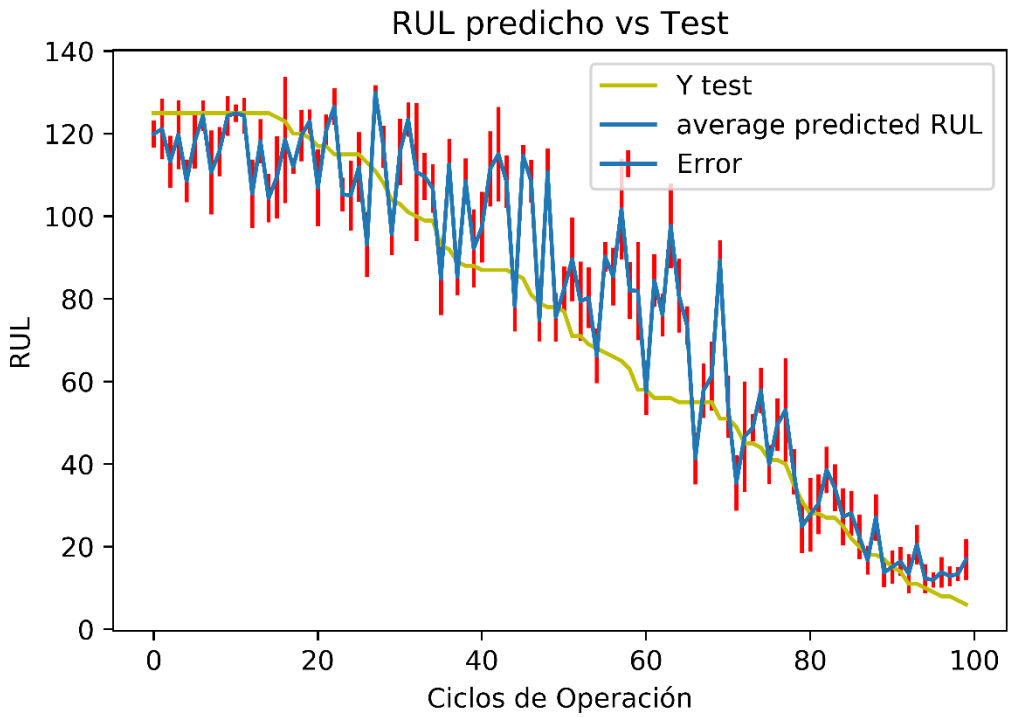

Figura 29: Predicciones del RUL vs test set en FD003. Fuente: Elaboración propia.

5.3.4 FD004

El data set FD004 corresponde a turbinas que operan en seis diferentes condiciones de operación y que presentan dos modos de falla. Esta base de datos es la más compleja de predecir, obteniendo predicciones que se ajustan de buena forma a los datos del data set de testeo.

Inicialmente, en la Tabla.29 se presentan los valores promedio obtenidos para las cuatro bases de datos utilizando la arquitectura que mejores resultados obtuvo en FD003. Los resultados consideran los valores obtenidos con convoluciones con causalidad. Luego, en la Tabla.30 se presentan los resultados obtenidos para las convoluciones sin causalidad y finalmente en la

Tabla.31 se presentan los valores obtenidos para las cinco iteraciones del modelo en cada data set considerando únicamente el caso con causalidad.

Tabla 29: Valores de los resultados promedio obtenidos con causalidad para FD004.

	FD001	FD002	FD003	FD004
RMSE	17,2	23,8	20,0	**26,4**
Score	906,0	8972,6	2021,7	**9416,9**
Tiempo [s]	445,7	1123,3	557,9	**1272,4**

Tabla 30: Valores de los resultados promedio obtenidos sin causalidad para FD004.

	FD001	FD002	FD003	FD004
RMSE	16,8	25,9	19,2	**26,5**
Score	675,4	14895,3	1501,1	**15927,2**
Tiempo [s]	416,4	951,6	519,8	**1122,5**

Tabla 31: Valores de los resultados obtenidos para cada iteración con causalidad para FD004.

FD001	**Iteración 1**	**Iteración 2**	**Iteración 3**	**Iteración 4**	**Iteración 5**
RMSE	17,4	17,5	18,5	16,2	16,3
Score	850,5	670,6	1484,5	723,2	801,6
FD002	**Iteración 1**	**Iteración 2**	**Iteración 3**	**Iteración 4**	**Iteración 5**
RMSE	24,6	23,4	25,4	22,1	23,7
Score	8099,1	8127,2	12169,6	9511,9	6955,2
FD003	**Iteración 1**	**Iteración 2**	**Iteración 3**	**Iteración 4**	**Iteración 5**
RMSE	19,0	20,0	22,2	20,0	18,6
Score	1452,2	1708,7	2320,4	3409,7	1217,2
FD004	**Iteración 1**	**Iteración 2**	**Iteración 3**	**Iteración 4**	**Iteración 5**
RMSE	**30,0**	**25,0**	**24,1**	**25,8**	**27,2**
Score	**9840,4**	**8736,7**	**6908,7**	**11528,9**	**10069,9**

En la Tabla.32 se presenta el mejor valor obtenido para las iteraciones realizadas en FD004 con la arquitectura expuesta en la sección correspondiente.

Tabla 32: Mejor iteración obtenida para el data set FD004.

	FD004
RMSE	24,1
Score	6908,7
Tiempo [s]	1279,0

En la Figura.30 se presentan las predicciones obtenidas por el modelo de la Tabla.32, comparándolo con los valores del data set de testeo. En este caso tampoco se agregan las barras de error por ser de gran magnitud y no poder apreciarse bien en la imagen.

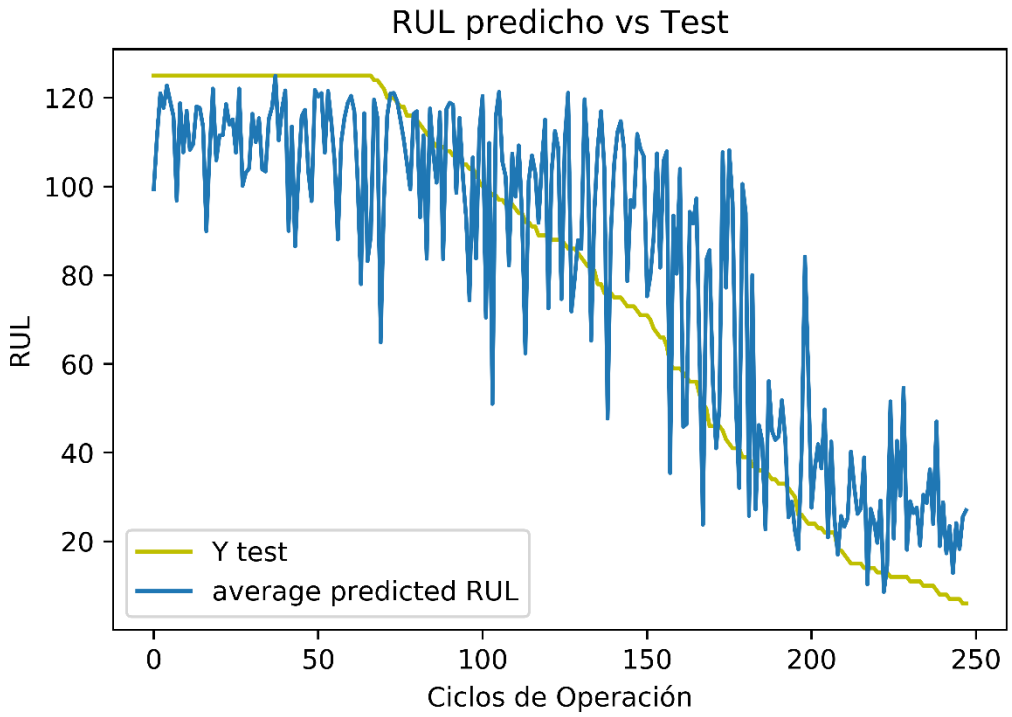

Figura 30: Predicciones del RUL vs test set en FD004. Fuente: Elaboración propia.

6 Análisis de Resultados

Luego de presentar los resultados se procede a realizar un análisis cuantitativo y cualitativo que permita extraer información y conclusiones a partir del trabajo producido.

Cada sección de resultados se analizará en forma particular para luego realizar una comparación de los resultados obtenidos por el modelo y los resultados obtenidos por otros modelos de DL aplicados a la misma base de datos.

6.1 Grid Search Causalidad

Este grid search se realizó con la intención de verificar la utilidad de la implementación de las máscaras de causalidad a los filtros de convolución.

A partir del modo de operación de las máscaras, se esperaba que el RMSE de las predicciones aumentara un pequeño porcentaje debido a la reducción en la información disponible. Por otro lado, se esperaba que el Score se redujera con respecto al caso sin causalidad, debido a que las máscaras condicionan las predicciones a valores temporalmente anteriores, permitiendo que las predicciones sean condicionadas por la información de tiempos anteriores al caso sin causalidad, haciendo que el modelo subestime la cantidad de vida útil remanente, incidiendo en la reducción del Score a causa de la forma no simétrica de la función.

En la Figura.31 se muestra el gráfico comparativo entre los mejores resultados obtenidos con y sin máscara para el Score en FD001.

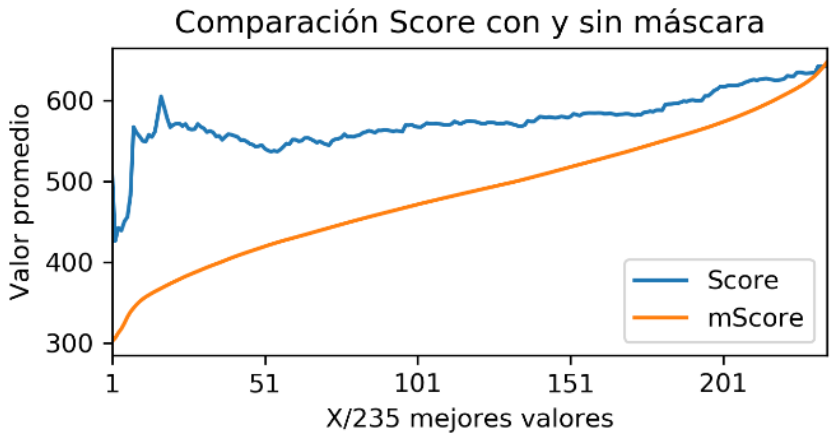

Figura 31: Gráfico comparativo entre Score con y sin causalidad. Fuente: Elaboración propia.

A partir del gráfico se aprecia que hay una considerable reducción en el Score obtenido al aplicar la causalidad por medio de la implementación de máscaras.

En la Figura.32 se muestra el gráfico comparativo entre los mejores resultados obtenidos para RMSE ordenados según los valores de Score obtenidos de FD001.

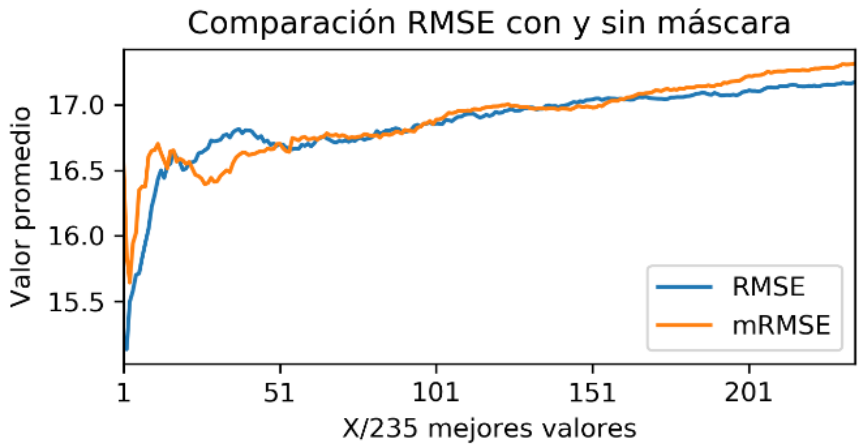

Figura 32: Gráfico comparativo entre RMSE con y sin causalidad. Fuente: Elaboración propia.

En el gráfico se puede apreciar que no hay diferencia significativa entre el comportamiento de las dos curvas y que el valor del RMSE con máscara tiende a ser un poco mayor, particularmente en el valor de más a la derecha donde se comparan los valores promedio obtenidos a partir del grid search.

6.2 Grid Search Residuos

Este grid search se realizó con la intención de validar y estudiar la aplicación del stack de residuos y la cantidad de celdas que lo forman.

En la Figura.33 se muestra el gráfico con los valores de la cantidad de celdas residuales de cada modelo que se probó en el grid search, según los parámetros de la arquitectura que se mostraron en la sección respectiva.

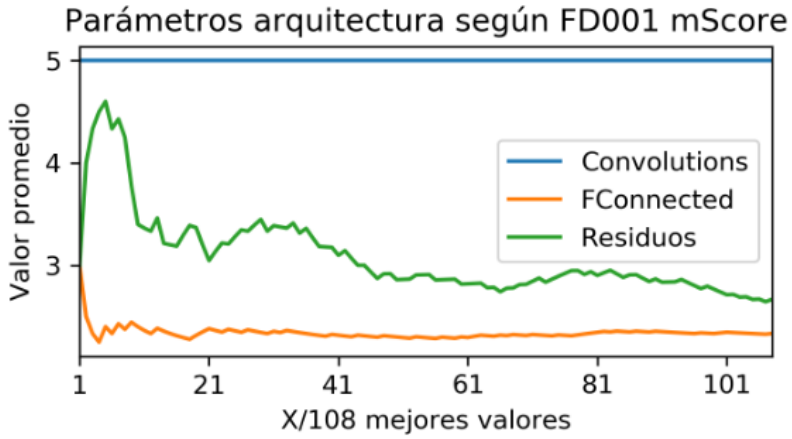

Figura 33: Gráfico comparativo de la cantidad de celdas residuales según la calidad de los resultados. Fuente: Elaboración propia.

A partir del gráfico se puede verificar que la implementación de celdas residuales mejora considerablemente los resultados, siendo la mayoría de los mejores resultados los obtenidos a partir de modelos que contenían celdas de residuos.

También se puede apreciar que los valores óptimos para la aplicación de una cantidad determinada de celdas es cercano a 3, aunque dentro de los diez mejores valores hay una gran porción de modelos con hasta cinco celdas de residuos.

6.3 Grid Search Dilatación

Este grid search se realizó con la intención de verificar y validar la implementación de dilataciones en las capas de convolución.

Se espera que la implementación de dilataciones mejore los resultados en cuanto al RMSE y al Score, debido a que permite integrar información que se encuentra a mayor distancia en la imagen y permite realizar predicciones basadas en tiempos temporales previos que sin dilatación serían desconocidos.

En la Figura.34 se presenta el gráfico con los valores de las dilataciones de los modelos ordenados según los mejores valores del Score para FD001.

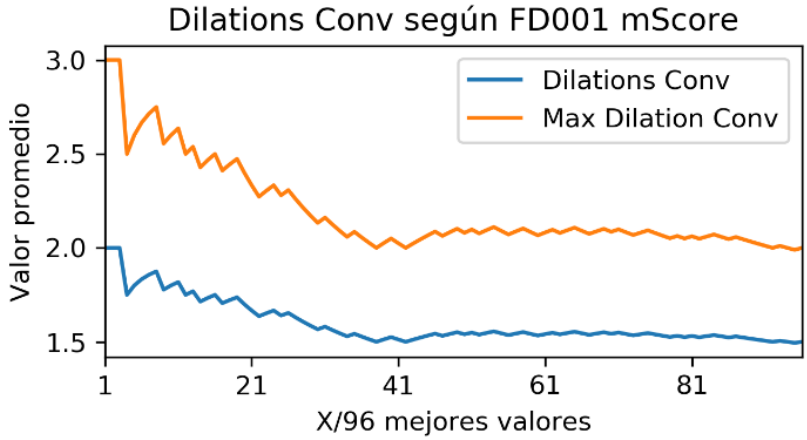

Figura 34: Gráfico comparativo de la cantidad de dilatación según la calidad de los resultados. Fuente: Elaboración propia.

Siendo el valor uno el caso sin dilatación y tres el valor máximo de dilatación implementada, se puede notar que hay una clara diferencia entre los valores obtenidos utilizando o no utilizando dilatación. La gran mayoría de los buenos resultados incluye hasta la dilatación tres y la curva indica claramente la tendencia de que los mejores valores obtenidos cuentan con dilatación.

6.4 Resultados FD001

A partir de las cinco iteraciones expuestas en los resultados de FD001 se calcula la desviación estándar con respecto al promedio para ver cómo se comportan los resultados.

En la Tabla.33 se muestran los valores promedio expuestos en la sección de resultados y los valores calculados para la desviación estándar del RMSE y del Score.

Tabla 33: Valores promedios obtenidos y desviación estándar FD001.

FD001	Valor promedio	Desviación estándar
RMSE	14,8	0,9
Score	401,8	134,2

Con los valores calculados se nota que hay una baja variabilidad en los resultados del RMSE ya que la desviación estándar representa un 6% del valor promedio. Por otro lado, hay una alta variabilidad en los resultados del Score, donde la desviación estándar de los datos representa un valor cercano al 33% del valor promedio.

Luego se estudia cuánto varía el valor del RMSE y el Score obtenido en el caso de aplicar causalidad.

En la Tabla.34 se muestra la variación porcentual de los valores de los resultados obtenidos al aplicar causalidad.

Tabla 34: Comparación valores obtenidos sin y con causalidad FD001.

FD001	Sin causalidad	Con causalidad	Variación porcentual al aplicar causalidad
RMSE	14,3	14,8	**3,5 %**
Score	434,8	401,8	**-7,6 %**

De la tabla se puede apreciar que los valores de la variación porcentual se corresponden a lo esperado. Los valores porcentuales son pequeños, pero significativos y se concluye que en la base de datos FD001 la aplicación de causalidad aumenta el RMSE y disminuye el Score de los resultados obtenidos.

6.5 Resultados FD002

A partir de las cinco iteraciones realizadas en la base de datos FD002 se calcula la desviación estándar de los datos con respecto al promedio mostrado en la sección de resultados.

En la Tabla.35 se muestran los valores de la desviación estándar para los valores obtenidos del RMSE y el Score por el modelo que se implementa en FD002.

Tabla 35: Valores promedios obtenidos y desviación estándar FD002.

FD002	Valor promedio	Desviación estándar
RMSE	24,1	1,3
Score	6907,0	2743,4

Con los valores calculados se nota que hay una baja variabilidad en los resultados del RMSE ya que la desviación estándar representa un 5% del valor promedio. Por otro lado, hay una alta

variabilidad en los resultados del Score, donde la desviación estándar de los datos representa un valor cercano al 40% del valor promedio.

Luego se estudia cuánto varía el valor del RMSE y el Score obtenido en el caso de aplicar causalidad. En la Tabla.36 se presenta la variación porcentual calculada.

Tabla 36: Comparación valores obtenidos sin y con causalidad FD002.

FD002	Sin causalidad	Con causalidad	Variación porcentual al aplicar causalidad
RMSE	22,7	24,1	**6,2 %**
Score	8680,7	6907,0	**-20,4 %**

De la tabla se aprecia que los valores del cambio porcentual al aplicar causalidad se comportan según lo esperado. Los valores porcentuales son significativos y se concluye que en la base de datos FD002 la aplicación de causalidad aumenta el RMSE y disminuye considerablemente el Score de los resultados obtenidos.

6.6 Resultados FD003

Con las cinco iteraciones realizadas en la base de datos FD003 se repite el mismo proceso que para las otras bases de datos y se calcula la desviación estándar de los datos con respecto al promedio mostrado en la sección de resultados.

En la Tabla.37 se muestran los valores de la desviación estándar para los valores obtenidos del RMSE y el Score por el modelo que se implementa en FD003.

Tabla 37: Valores promedios obtenidos y desviación estándar FD003.

FD003	Valor promedio	Desviación estándar
RMSE	17,4	0,8
Score	876,3	276,3

Al igual que para los casos anteriores, se nota que hay una baja variabilidad en los resultados del RMSE ya que la desviación estándar representa un 4% del valor promedio. Mientras que para el Score hay una alta variabilidad en los resultados, donde la desviación estándar de los datos representa un valor cercano al 32% del valor promedio.

Luego se estudia cuánto varía el valor del RMSE y el Score obtenido en el caso de aplicar causalidad. En la Tabla.38 se presenta la variación porcentual calculada.

Tabla 38: Comparación valores obtenidos sin y con causalidad FD003.

FD003	Sin causalidad	Con causalidad	Variación porcentual al aplicar causalidad
RMSE	17,4	17,4	**0,0 %**
Score	1241,8	876,3	**-29,4 %**

De la tabla se aprecia que los valores del cambio porcentual al aplicar causalidad se comportan dentro lo esperado. El valor porcentual obtenido para la variación del RMSE, a pesar de ser cero, se encuentra dentro de la pequeña variación esperada. El valor del cambio porcentual obtenido para el Score actúa según lo esperado y se condice con los valores obtenidos para FD001 y FD002, con la causalidad disminuyendo considerablemente el Score.

6.7 Resultados FD004

Finalmente, con las cinco iteraciones realizadas en la base de datos FD004 se repite el mismo proceso que para las otras bases de datos y se calcula la desviación estándar de los datos con respecto al promedio mostrado en la sección de resultados.

En la Tabla.39 se muestran los valores de la desviación estándar para los valores obtenidos del RMSE y el Score por el modelo que se implementa en FD004.

Tabla 39: Valores promedios obtenidos y desviación estándar FD004.

FD004	Valor promedio	Desviación estándar
RMSE	26,4	2,3
Score	9416,9	1719,0

Como en los casos anteriores, se nota que hay una baja variabilidad en los resultados del RMSE ya que la desviación estándar representa un 8% del valor promedio. Mientras que para el Score hay una alta variabilidad en los resultados, aunque menor a las otras bases de datos, donde la desviación estándar de los datos representa un valor cercano al 20% del valor promedio.

Luego se estudia cuánto varía el valor del RMSE y el Score obtenido en el caso de aplicar causalidad. En la Tabla.40 se presenta la variación porcentual calculada.

Tabla 40: Comparación valores obtenidos sin y con causalidad FD004.

FD004	Sin causalidad	Con causalidad	Variación porcentual al aplicar causalidad
RMSE	26,5	26,4	**-0,4 %**
Score	15927,2	9416,9	**-40,9 %**

De la tabla se aprecia que los valores del cambio porcentual al aplicar causalidad se comportan según lo esperado. El valor porcentual obtenido para la variación del RMSE, a pesar de

ser negativo, es muy cercano a cero y se encuentra dentro de la pequeña variación esperada. El valor del cambio porcentual obtenido para el Score actúa según lo esperado y al igual que para las otras bases de datos, la aplicación de causalidad disminuye considerablemente el Score obtenido por los modelos.

6.8 Predicciones FD001

Luego, para cada base de datos se analiza cómo se comportan las predicciones de los modelos en función de la presencia de daño en las turbinas.

Para esto, los gráficos de predicciones mostrados en la sección de resultados se dividen en cinco secciones temporales a lo largo de los diferentes labels de deterioro de las turbinas de testeo y se estudia el comportamiento de las predicciones en las cinco secciones con la intención de analizar cómo se comportan las predicciones del modelo en función del daño de la turbina y de la vida útil remanente en cada ciclo de operación.

Para cada sección se calcula el RMSE de las predicciones con respecto a los labels de la base de datos de testeo y se calcula el valor de la desviación estándar de las predicciones para tener una noción de la certeza de las predicciones y de su variabilidad según la sección del gráfico que define el estado de las turbinas de testeo.

En la Figura.35 se muestra el gráfico con las predicciones de FD001 y las secciones temporales en las que se divide el gráfico para realizar los cálculos del RMSE y de la desviación estándar.

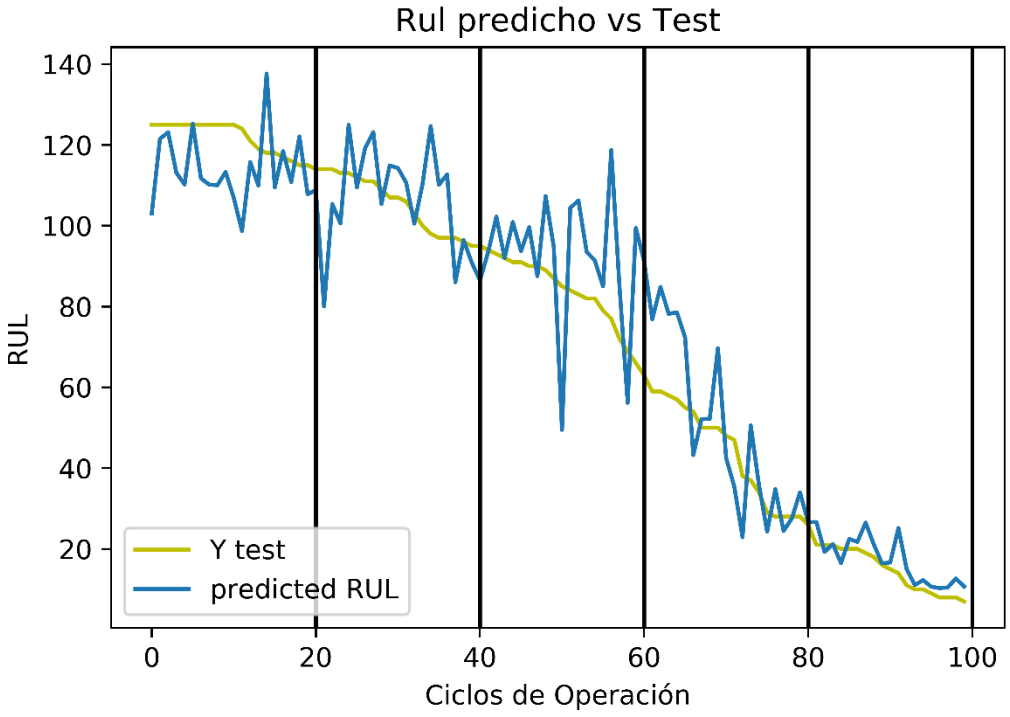

Figura 35: Gráfico seccionado de las predicciones realizadas por el modelo implementado en FD001. Fuente: Elaboración propia.

En la Tabla.41 se presentan los valores calculados del RMSE y la desviación estándar de las predicciones según las secciones del gráfico de izquierda a derecha.

Tabla 41: Análisis predicciones del modelo en FD001 según sección del gráfico.

FD001	1	2	3	4	5
RMSE	17,8	11,7	17,5	13,3	4,8
Desviación predicciones	12,3	11,8	17,4	20,3	7,0

A partir de los valores del RMSE calculado se puede notar que el modelo es capaz de predecir de buena forma cuando las turbinas presentan casi la totalidad del daño y les queda poca vida útil remanente. Los demás valores muestran que el modelo es malo identificando los estados saludables de la turbina, además de presentar una mala capacidad para definir cuándo inicia el daño, hasta que este ya está muy desarrollado.

Luego, estudiando los valores de la desviación estándar de las predicciones del modelo, se puede verificar que el comportamiento del modelo es similar a lo indicado según el RMSE, mostrando que el modelo no es tan bueno para identificar el daño hasta que éste ya está muy desarrollado y que tiene dificultades para identificar el daño de la turbina a lo largo de la porción lineal de la degradación.

6.9 Predicciones FD002

Para la base de datos FD002 se repite el proceso de la sección anterior para estudiar las predicciones del modelo.

En la Figura.36 se muestra el gráfico con las predicciones de FD002 y las secciones temporales en las que se divide el gráfico.

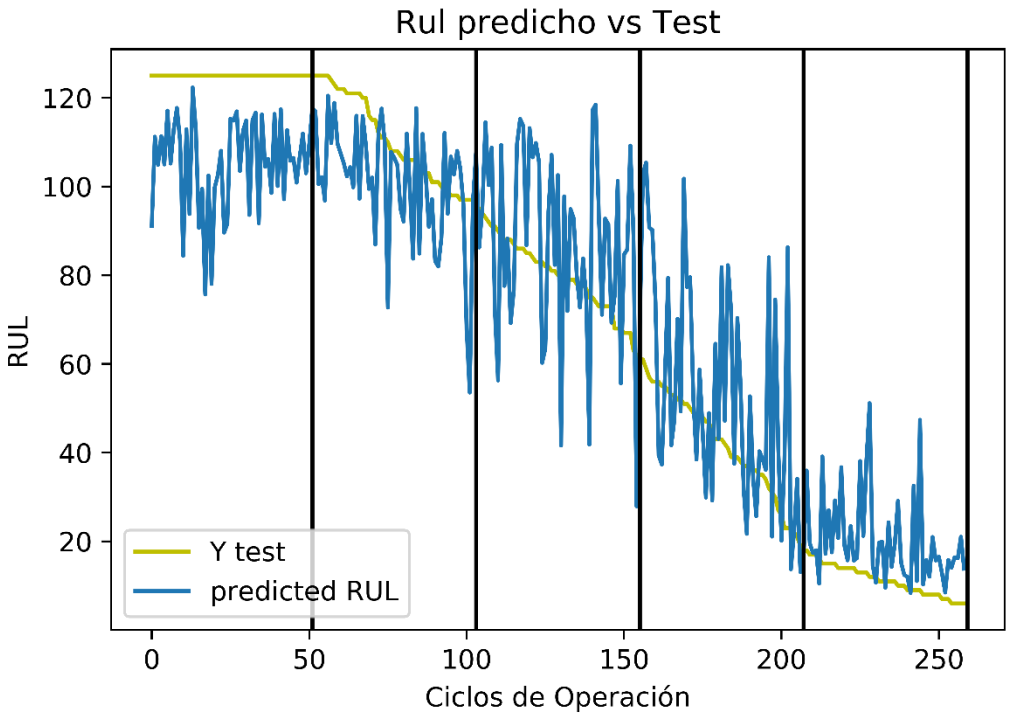

Figura 36: Gráfico seccionado con las predicciones realizadas por el modelo implementado en FD002. Fuente: Elaboración propia.

En la Tabla.42 se presentan los valores calculados del RMSE y la desviación estándar de las predicciones según las secciones del gráfico de izquierda a derecha.

Tabla 42: Análisis predicciones del modelo en FD002 según sección del gráfico.

FD002	1	2	3	4	5
RMSE	30,1	22,1	24,6	25,3	14,1
Desviación predicciones	15,6	15,3	24,0	26,1	12,4

A partir de los valores del RMSE calculado se puede notar como el modelo tiene dificultad para identificar el daño de la turbina hasta que éste está muy desarrollado y queda poca vida útil remanente. Los valores en general presentan un alto error en las predicciones, lo que se condice con el alto valor de RMSE y Score obtenidos en resultados.

Estudiando los valores de la desviación estándar de las predicciones del modelo, se puede verificar que el comportamiento del modelo es similar a lo indicado según el RMSE, mostrando que el modelo es malo para identificar el daño hasta que este ya está muy desarrollado y que tiene dificultades para identificar el daño de la turbina a lo largo de la porción lineal de la degradación.

6.10 Predicciones FD003

Repitiendo el mismo proceso que para FD001 y FD002 se divide el gráfico de predicciones, mostrado en resultados, en secciones temporales y se procede a analizar el comportamiento de las predicciones.

En la Figura.37 se muestra el gráfico con la comparación entre las predicciones y los valores de testeo y las secciones de división.

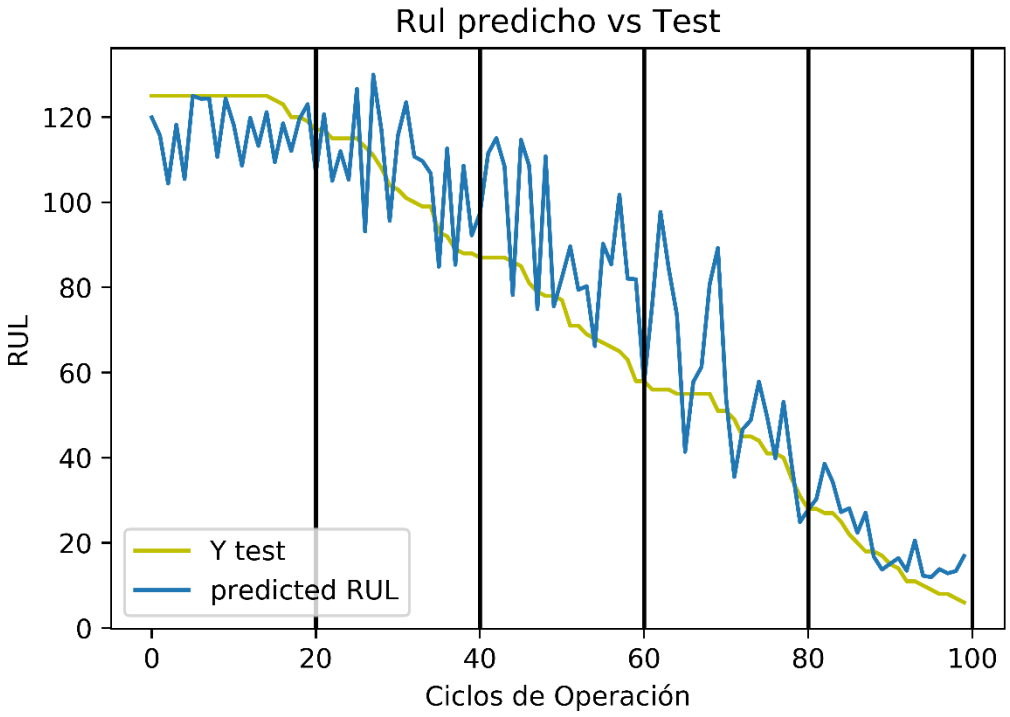

Figura 37: Gráfico seccionado con las predicciones realizadas por el modelo implementado en FD003. Fuente: Elaboración propia.

En la Tabla.43 se presentan los valores calculados del RMSE y la desviación estándar de las predicciones según las secciones del gráfico de izquierda a derecha.

Tabla 43: Análisis predicciones del modelo en FD003 según sección del gráfico.

FD003	1	2	3	4	5
RMSE	9,4	14,7	21,0	20,3	7,7
Desviación predicciones	7,8	15,3	16,4	22,1	10,14

A partir de los valores del RMSE calculado se puede notar como el modelo es bueno identificando los estados saludables de las turbinas y los estados próximos al daño que las deja inoperativas. También se nota que la mayor dificultad está en la predicción del daño a lo largo de la porción lineal de la degradación de las turbinas.

Estudiando los valores de la desviación estándar de las predicciones del modelo, se puede verificar que el comportamiento del modelo es similar a lo indicado según el RMSE, mostrando que el modelo es bueno para identificar el estado de las turbinas cuando se encuentran saludables y cuando se encuentran próximas al daño que las deja inoperativas.

6.11 Predicciones FD004

Finalmente, para la base de datos FD004 se realiza el mismo proceso.

En la Figura.38 se muestra el gráfico con la comparación entre los valores predichos y los valores de testeo, además de incluir las secciones temporales en las cuales se realizará el análisis.

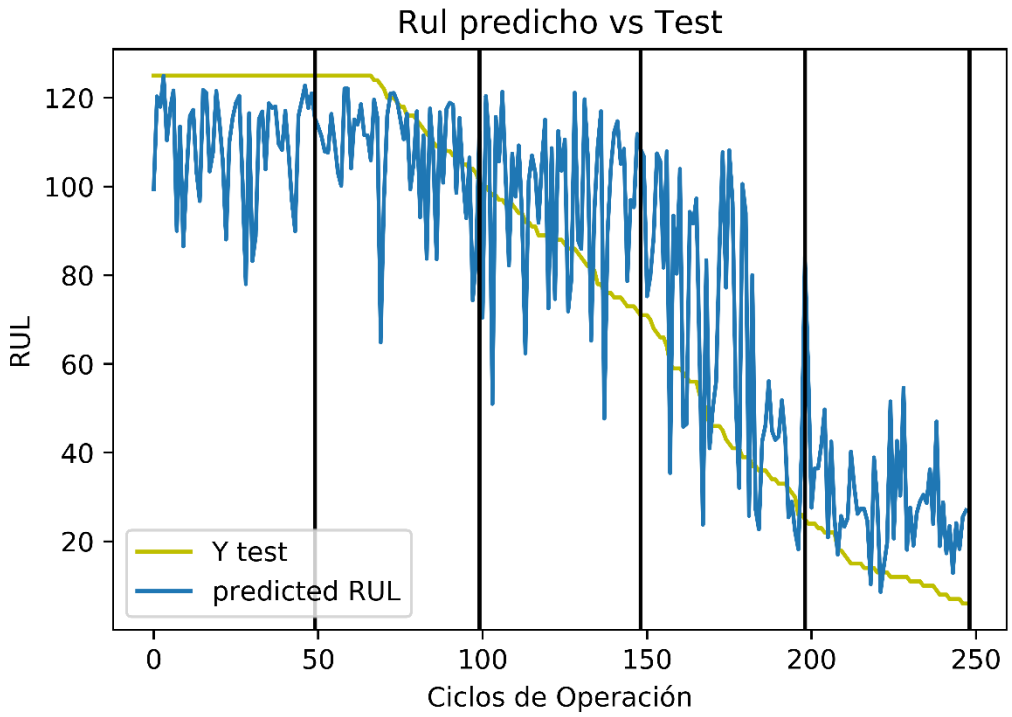

Figura 38: Gráfico seccionado con las predicciones realizadas por el modelo implementado en FD004. Fuente: Elaboración propia.

En la Tabla.44 se presentan los valores calculados del RMSE y la desviación estándar de las predicciones según las secciones del gráfico de izquierda a derecha.

Tabla 44: Análisis predicciones del modelo en FD004 según sección del gráfico.

FD004	1	2	3	4	5
RMSE	22,9	23,3	25,4	31,2	16,2
Desviación predicciones	15,4	19,5	23,1	33,1	13,1

A partir de los valores del RMSE calculado se puede notar como el modelo tiene dificultad para identificar el daño de la turbina en la totalidad de la porción de los datos, mejorando cuando la turbina tiene poca vida útil remanente, pero sin realizar buenas predicciones. Los valores en general presentan un alto error en las predicciones, lo que se condice con el alto valor de RMSE y Score obtenidos en resultados.

Estudiando los valores de la desviación estándar de las predicciones del modelo, se puede verificar que el comportamiento del modelo es similar a lo indicado según el RMSE, mostrando

que el modelo es malo para identificar el daño hasta que este ya está muy desarrollado y que tiene dificultades para identificar el daño de la turbina a lo largo de la porción lineal de la degradación, mejorando cuando las turbinas están en estado saludable.

6.12 Comparación Resultados

Para estudiar la calidad de los resultados con la arquitectura propuesta del modelo es necesario comparar los valores obtenido con otros modelos que hayan sido implementados en la misma base de datos.

Para esto, se compararán los resultados obtenidos por las redes neuronales causales residuales con los valores obtenidos en la referencia [9] en base a redes neuronales convolucionales y unidades LSTM y en la referencia [8] en base a redes neuronales convolucionales.

En la Tabla.44 y la Tabla.45 se presentan los valores obtenidos por los dos modelos anteriormente expuestos y los obtenidos en el desarrollo de este trabajo por las redes neuronales convolucionales causales residuales.

Tabla 45: Comparación resultados obtenidos por LSTM y residual causal CNN.

	CNNBILSTM				CNN residual causal			
	RMSE		Score		RMSE		Score	
	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD
FD001	10,1	0,6	339,9	72,8	14,8	0,9	401,8	134,2
FD002	21,0	0,3	10830,1	1788,5	24,1	1,3	6907,0	2743,4
FD003	11,34	0,7	1189,5	167,8	17,4	0,8	876,3	276,3
FD004	22,7	0,4	9849,1	954,2	26,4	2,3	9416,9	1719,0

Tabla 46: Comparación resultados obtenidos por CNN y residual causal CNN.

	DCNN				CNN residual causal			
	RMSE		Score		RMSE		Score	
	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD
FD001	12,6	0,2	273,7	24,1	14,8	0,9	401,8	134,2
FD002	22,4	0,3	10412,0	544,0	24,1	1,3	6907,0	2743,4
FD003	12,6	0,1	284,1	26,5	17,4	0,8	876,3	276,3
FD004	23,31	0,4	12466,0	853,0	26,4	2,3	9416,9	1719,0

De las tablas anteriores se puede notar que en general, los valores del RMSE son peores a Scores comparables, lo que se entiende por la reducción en la cantidad de información disponible que se ingresa al realizar las predicciones implementando convoluciones causales.

Los valores de Score obtenidos son muy cercanos a los valores obtenidos por las referencias, mejorando considerablemente en las bases de datos más complejas, FD002 y FD004, lo que se debe a la capacidad de extraer características abstractas y generales por parte de las CNNs causales residuales.

Es importante notar que los valores de las desviaciones estándar obtenidas por el modelo son considerablemente mayores a los obtenidos por los otros modelos. La gran desviación se entiende por la mayor imprecisión del modelo propuesto debido a la disminución en la información que se utiliza por las convoluciones al aplicar la causalidad y la asimetría de la función Score que para predicciones tardías entrega valores considerablemente mayores que para predicciones con antelación.

7 Conclusiones

A partir del trabajo realizado, los resultados mostrados y el análisis de resultados se procede a enunciar las conclusiones del trabajo.

Primero, se validó la implementación de la causalidad por medio de los resultados obtenidos para la serie de modelos del grid search respectivo. Se pudo notar que la implementación de la causalidad aumenta mínimamente el RMSE obtenido y reduce considerablemente el Score de las cuatro bases de datos.

La reducción en el Score al aplicar causalidad es de un 7,6% en la base de datos FD001, 20,4% en la base de datos FD002, 29,4% en la base de datos FD003 y 40,4% en la base de datos FD004.

Luego se validó la implementación de las celdas de residuos gracias a los resultados obtenidos en el grid search respectivo. Los residuos implementados ayudan considerablemente en la reducción del Score obtenido en resultados, lo que se debe a su capacidad de modelar interacciones más complejas de la base de datos por medio de la inclusión de la variable multiplicativa.

Estudiando las predicciones de los modelos en cada base de datos se tiene que el modelo implementado en FD001 logra muy buenos resultados en las predicciones, siendo capaz de predecir de buena forma cuando las turbinas presentan casi la totalidad del daño y les queda poca vida útil remanente. El modelo es peor identificado los estados saludables y presenta una mala capacidad para identificar el daño en la porción lineal de la degradación de las turbinas.

En FD002, el modelo implementado presenta malos resultados en la predicción de los estados de las turbinas y de la vida útil remanente. El modelo tiene dificultades para identificar el daño hasta que éste está muy desarrollado y queda poca vida útil remanente. Los resultados obtenidos presentan un alto error en las predicciones, particularmente en la porción lineal de la degradación de las turbinas.

En FD003, el modelo implementado es bueno identificando los estados saludables de las turbinas y los estados próximos al daño que las deja inoperativas. La mayor dificultad en las predicciones está en la predicción del daño a lo largo de la porción lineal de la degradación de las turbinas, de forma similar a los modelos anteriormente expuestos.

Para FD004, el modelo implementado tiene dificultad para identificar el daño de las turbinas en la totalidad de la porción de los datos, mejorando cuando la turbina tiene poca vida útil remanente, pero sin realizar buenas predicciones.

Finalmente, con los resultados obtenidos para cada base de datos por parte de las CNNs causales, se puede enunciar, a partir de la comparación con otros modelos implementados en CMAPSS, que las CNNs causales logran resultados comparables pero con una mayor imprecisión en la predicción y modelación de series temporales multivariables, realizando un aprendizaje más general que condiciona la capacidad de predecir de forma precisa.

8 Bibliografía

[1] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, «WaveNet: A generative model for raw audio»,2016.

[2] O. Abdeljaber, O. Avci, S. Kiranyaz, M. Gabbouj y D. Inman, «Real-Time based structural damage detection using one-dimensional convolutional neural networks»,2016.

[3] P. D. McFadden, «Examination of a technique for the early detection of failure in gears by signal processing of the time domain average of the meshing vibration», 1987.

[4] A. Saxena and K. Goebel (2008). "Turbofan Engine Degradation Simulation Dat a Set", NASA Ames Prognostics Data Repository (http://ti.arc.nasa.gov/project/prognosti c-data-repository), NASA Ames Research Center, Moffett Field, CA

[5] Z. Chen, C. Li y R. Sanchez, «Gearbox fault identification and classification with convolutional neural networks», 2015.

[6] Y. Gu y Z. Ling, «Waveform modeling using stacked dilated convolutional neural networks for speech bandwidth extension», 2017.

[7] O. Janseens, V. Slavkovikj, B. Vervisch, K. Stockman, M. Loccufier, S. Verstockt, R. Van de Walle y S. Van Hoecke, «Convolutional neural network based fault detection for rotating machinery», 2015.

[8] G. Babu, P. Zhao y X. Li, «Deep convolutional neural network based regression approach for estimation of remaining life», 2016.

[9] S. Cofré, E. López y V. Meruane, «Deep remaining useful life estimation framework for big machinery data», 2018

[10] A. Lrizhevsky, I. Sutskever y G. Hinton, «Imagenet classification with deep convolutional neural networks», 2014.

[11] J.P. Cassar y M. Staroswiecki, «A structural approach for the design of failure detection and identification systems», 1997.

[12] J. Bouvrie, «Notes on convolutional neural networks», 2006.

[13] A. van der Oord, N. Kalchbrenner, O.Vinyals, L. Espeholt, A. Graves, K. Kavukcuoglu, «Conditional image generation with PixelCNN decoders», 2015.