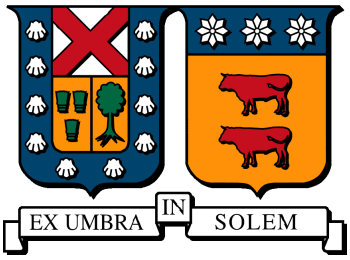
UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INGENIERÍA MECÁNICA
SANTIAGO-CHILE



APLICACIÓN Y ANÁLISIS DE MODELOS BASADOS EN INTELIGENCIA ARTIFICIAL EN EL MANTENIMIENTO PREDICTIVO MEDIANTE LA ESTIMACIÓN DE LA VIDA ÚTIL REMANENTE DE UN ACTIVO CRÍTICO SIMULADO

PABLO IGNACIO ARRIAGADA CASTRO

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL MECÁNICO

PROFESOR GUÍA : PhD. CHRISTOPHER COOPER
PROFESOR CORREFERENTE : DR. ING. JAIME NÚÑEZ SEGOVIA

octubre 2024

Agradecimientos

En primer lugar, agradezco profundamente a mi familia, quienes siempre creyeron en mí y me brindaron su amor incondicional y constante apoyo en los momentos más desafiantes durante este largo camino. Agradezco en especial a mis padres, por enseñarme que con esfuerzo y dedicación todo es posible, y por estar a mi lado en los momentos más difíciles. También quiero expresar mi gratitud a mis amigos, quienes hicieron más llevadero este proceso. Finalmente, agradezco a mis profesores guía y correferente, cuya paciencia y orientación fueron claves para culminar este trabajo de manera satisfactoria. Sin su ayuda y disposición, esta meta no habría sido alcanzable.

Resumen

El mantenimiento predictivo se ha convertido en una estrategia esencial para la gestión eficiente de activos críticos en la industria, permitiendo maximizar el tiempo de operación y minimizar los costos asociados a fallas inesperadas. Una de las áreas clave dentro del mantenimiento predictivo es la estimación de la vida útil remanente (RUL) de los equipos, la cual proporciona información crucial para la planificación de mantenimientos preventivos. En este contexto, el aprendizaje profundo ha emergido como una herramienta poderosa debido a su capacidad para modelar comportamientos complejos y no lineales en los datos.

Este estudio se enfoca en la evaluación comparativa de cuatro arquitecturas de aprendizaje profundo para la predicción del RUL en bombas tríplex, utilizando datos sintéticos generados mediante simulación. Para generar las trayectorias de degradación hasta el fallo, se desarrolló un modelo de simulación basado en ecuaciones diferenciales, que incorporó tres modos de falla principales: WornBearing, LeakSeal y BlockedIn, así como sus combinaciones, resultando en siete escenarios de falla distintos. Para modelar el deterioro progresivo causado por los modos de falla, se utilizó un modelo de degradación exponencial.

Se evaluaron cuatro arquitecturas de redes neuronales profundas: Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), CNN-LSTM híbrida, y CNN-LSTM con preprocesamiento (CNN-LSTM-p). Estas arquitecturas fueron analizadas en dos escenarios: uno que simula condiciones realistas con datos de entrenamiento desbalanceados, y otro que incorpora conocimiento experto para mejorar la calidad de los conjuntos de datos.

Los resultados revelaron diferencias significativas en el desempeño de los modelos. En el primer escenario, el modelo CNN demostró el mejor rendimiento general en los conjuntos de datos de entrenamiento y prueba, mientras que el modelo CNN-LSTM exhibió la menor degradación de rendimiento entre ambos conjuntos. En el segundo escenario, el modelo

CNN-LSTM-p obtuvo el mejor rendimiento general, mientras que el modelo LSTM mostró la menor degradación de rendimiento entre los conjuntos de entrenamiento y prueba. Se observaron variaciones en la precisión predictiva de los modelos en distintas fases de vida de las unidades, con fluctuaciones pronunciadas hacia el final de la vida útil. A pesar de las variaciones, todos los modelos demostraron una elevada capacidad para seguir la tendencia general del RUL en ambos escenarios, incluso en presencia de modos de falla no vistos durante el entrenamiento. Los hallazgos sugieren que los modelos son capaces de capturar patrones sutiles indicativos del inicio del proceso de degradación, lo cual es particularmente relevante para la detección temprana de fallas. La variabilidad en el desempeño entre las diferentes fases de vida de las bombas indica la presencia de patrones distintivos en cada etapa de degradación.

Este estudio proporciona una evaluación comparativa detallada de diferentes arquitecturas de aprendizaje profundo en el contexto específico de bombas tríplex, subrayando la importancia de considerar múltiples arquitecturas y escenarios al desarrollar sistemas de predicción de RUL. Se recomiendan futuras investigaciones para abordar las limitaciones identificadas, incluyendo la expansión del conjunto de datos, el análisis de una mayor variedad de modos de falla, y la validación con datos operativos reales, con el fin de potenciar la robustez y aplicabilidad de los modelos de predicción de RUL en entornos industriales.

Índice general

Agradecimientos I

Resumen II

Índice general VI

Índice de figuras IX

Índice de tablas X

1. Introducción **1**

1.1. Introducción 1

1.2. Objetivo general 4

1.3. Objetivos específicos 4

2. Marco Teórico **5**

2.1. Fundamentos del Mantenimiento Predictivo 5

2.1.1. Métodos de Gestión del Mantenimiento 5

2.1.2. Fallas y Modos de falla 7

2.1.3. Mantenimiento Predictivo 10

2.1.4. Ventajas del Mantenimiento Predictivo 12

2.2. Fundamentos del Machine Learning 13

2.2.1. ¿Qué es el Machine Learning? 13

2.2.2. Tipos de Machine Learning 14

2.2.3. Tareas de Machine Learning 16

2.2.4. Etapas en el desarrollo de modelos de Machine Learning 21

2.3. Machine Learning en el Mantenimiento Predictivo 22

2.4. Principales desafíos en Machine Learning 23

2.4.1. Calidad de los datos . . . 23
2.4.2. Desajustes en el entrenamiento . . . 24
2.4.3. Falta de datos en el entrenamiento . . . 25
2.5. Redes Neuronales . . . 25
2.5.1. Perceptrón . . . 26
2.5.2. Modelo de Red Neuronal . . . 27
2.5.3. Función de Activación . . . 29
2.5.4. Función de Coste o Pérdida . . . 33
2.5.5. Teorema Universal de Aproximación . . . 34
2.5.6. Backpropagation . . . 35
2.5.7. Algoritmos de optimización . . . 36
2.6. Redes Neuronales Convolucionales (CNNs) . . . 39
2.6.1. Pooling Layers . . . 41
2.7. Redes Neuronales Recurrentes (RNNs) . . . 42
2.7.1. Long Short-Term Memory (LSTM) . . . 43

3. Estado del Arte **45**
3.1. Enfoques en la estimación del RUL . . . 46
3.1.1. Redes Neuronales Convolucionales (CNN) . . . 46
3.1.2. Redes de Memoria a Corto y Largo Plazo (LSTM) . . . 47
3.1.3. Redes Híbridas . . . 48

4. Metodología **51**
4.1. Modelado de la Bomba tríplex . . . 51
4.1.1. Definición del Sistema . . . 52
4.1.2. Modos de Falla . . . 68
4.1.3. Modelo de Degradación . . . 69
4.2. Generación de Datos Sintéticos Simulados . . . 75
4.2.1. Ruido de Sensores . . . 75
4.2.2. Discretización de Señales . . . 76
4.2.3. Construcción del Conjunto de Datos . . . 77
4.3. Preprocesamiento de los Datos . . . 77
4.3.1. Suavizado de las Señales de Entrada . . . 78
4.3.2. Generación de Características . . . 79
4.3.3. Selección de Características . . . 82

4.3.4. Normalización . . . 82
4.3.5. Remuestreo con Ventana Deslizante . . . 83
4.3.6. Definición de Función Objetivo . . . 84
4.4. División de Datos y Definición de Escenarios Experimentales . . . 85
4.4.1. Justificación de la Estrategia de División de los Datos . . . 86
4.4.2. Definición de Escenarios Experimentales . . . 86
4.5. Desarrollo de los Modelos Predictivos . . . 88
4.5.1. Modelo CNN . . . 88
4.5.2. Modelo LSTM . . . 89
4.5.3. Modelo CNN-LSTM . . . 90
4.5.4. Modelo CNN-LSTM-p . . . 92
4.5.5. Dropout . . . 93
4.5.6. Optimizadores . . . 94
4.5.7. Funciones de Pérdida . . . 94
4.5.8. Learning Rate Scheduling . . . 94
4.6. Entrenamiento de los Modelos Predictivos . . . 95
4.6.1. Tamaño de Lote . . . 95
4.6.2. Número de Épocas . . . 95
4.6.3. Callbacks . . . 95
4.7. Evaluación de los Modelos Predictivos . . . 96
4.7.1. Métricas de Evaluación . . . 96
4.7.2. Tablas y Figuras . . . 98

5. Resultados y Discusión **99**
5.1. Resultados de la Simulación . . . 99
5.2. Desempeño de los Modelos Predictivos . . . 105
5.3. Discusión de Resultados . . . 114

6. Conclusiones **119**

7. Anexos **133**

Índice de figuras

2.1. Vista general de los tipos de mantenimiento. 7
2.2. Taxonomía de los enfoques en el pronóstico. 9
2.3. Enfoques en el pronóstico híbrido. 10
2.4. Curva de estado paramétrico e intervalo de vida residual. 12
2.5. Esquema Modelo de Degradación - RUL. 12
2.6. Flujo de trabajo de un sistema de aprendizaje automático supervisado. 16
2.7. Diagrama resumen tipos de machine learning. 20
2.8. Pasos principales para el diseño de un modelo de machine learning. 21
2.9. Modelo Subajustado, Óptimo y Sobreajustado en aprendizaje automático. . . 24
2.10. Representación gráfica de un perceptrón o neurona con n datos de entrada n pesos y una función de activación f. (Fuente: Elaboración propia a partir de "Neural Networks and Learning Machines" - Simon S. Haykin [32]) 27
2.11. Ejemplo de representación de una red neuronal multicapa (fully-connected) con 3 entradas, 3 capas ocultas y una capa de salida. (Fuente: Elaboración propia a partir de "Neural Networks and Learning Machines" - Simon S. Haykin [32]) 29
2.12. Función de activación Sigmoide σ. 30
2.13. Función de activación Tanh. 31
2.14. Función de activación ReLU. 32
2.15. Función de activación ELU. 32
2.16. Función de activación Softmax. 33
2.17. Interpretación de grafos computacionales del Backpropagation. 35
2.18. Representación gráfica del descenso de gradiente. 37
2.19. Representación gráfica del proceso de convolución entre una matriz de entrada (imágen) y un kernel. 40
2.20. Representación gráfica de los procesos de Max-pooling y Average-pooling . . 41
2.21. Representación gráfica de una Fully Recurrent Neural Network. 42

2.22. Representación gráfica de una actualización de la capa oculta $h^{[t]}$ en una arquitectura LSTM. 44

3.1. DAG Framework. 50

4.1. Esquema que representa los distintos módulos y conexiones presentes en el modelo simulado de la bomba tríplex. 52
4.2. Esquema del controlador PI. 54
4.3. Esquema del sistema del émbolo. 58
4.4. Representación de la fuerza de fricción como suma de las fuerzas de fricción de Stribeck, de Coulomb y los componentes viscosos. 61
4.5. Eje cigüeñal de la bomba CAT 1051. 63
4.6. Unidad B002 del conjunto de datos simulado: (a) Datos de los 6 sensores normalizados, y (b)Función lineal por partes del RUL. 85
4.7. Diagrama esquemático de la arquitectura del modelo CNN. 89
4.8. Diagrama esquemático de la arquitectura del modelo LSTM. 90
4.9. Diagrama esquemático de la arquitectura del modelo CNN-LSTM. 91
4.10. Diagrama esquemático de la arquitectura del modelo CNN-LSTM-p. 93

5.1. Caudal volumétrico a la salida de cada émbolo y caudal total de la bomba. No se han habilitado fallas. 99
5.2. Caudal y la velocidad angular de la bomba cuando el fallo de fuga de fluido por desgaste en los sellos está activado en el émbolo 1. 100
5.3. Caudal y la velocidad angular de la bomba cuando el fallo de entrada bloqueada está activado en la válvula de retención correspondiente al émbolo 1. 100
5.4. Efecto en el caudal de la bomba y en la corriente demandada debido al incremento del área de fuga de fluido en la junta debido al desgaste de los sellos en el émbolo 1. 101
5.5. Efecto en el caudal de la bomba y en la corriente demandada debido al incremento en el bloqueo en la válvula de retención de entrada del émbolo 1. . . . 102
5.6. Efecto en el caudal de la bomba y en la corriente demandada debido al incremento en el desgaste de los rodamientos del eje cigüeñal de la bomba. 102
5.7. cambio de estado. 103
5.8. Comparación de la señal continua simulada de la corriente eléctrica del motor vs. la señal con ruido y la señal discretizada mediante ZOH. 104

5.9. Comportamiento normal sin fallas de una bomba tríplex, con señales de sensores discretas y con ruido. 104
5.10. Resultados de la predicción en las unidades de prueba en el escenario 1: (a) Modelo CNN, (b) Modelo LSTM, (c) Modelo CNN-LSTM, (d) Modelo CNN-LSTM-p. 109
5.11. Resultados de la predicción en las unidades de prueba en el escenario 2: (a) Modelo CNN, (b) Modelo LSTM, (c) Modelo CNN-LSTM, (d) Modelo CNN-LSTM-p. 113

Índice de tablas

4.1. Variables medibles del sistema simulado de la bomba tríplex. 78
4.2. Características del dominio del tiempo. 80
4.3. Características del dominio de la frecuencia. 81
4.4. Configuración de unidades en los conjuntos de entrenamiento, validación y prueba en los escenarios 1 y 2. 87

5.1. Tabla resumen de unidades simuladas. 105
5.2. Comparación de precisión de la predicción de los modelos en el set de entrenamiento en el escenario 1. 106
5.3. Comparación de precisión de la predicción de los modelos en el set de prueba en el escenario 1. 107
5.4. Degradación de la precisión de predicción de los modelos entre los conjuntos de entrenamiento y prueba en el escenario 1. 108
5.5. Comparación de precisión de la predicción de los modelos en el set de entrenamiento en el escenario 2. 110
5.6. Comparación de precisión de la predicción de los modelos en el set de prueba en el escenario 2. 111
5.7. Degradación de la precisión de predicción de los modelos entre los conjuntos de entrenamiento y prueba en el escenario 2. 112

Capítulo 1

Introducción

1.1. Introducción

En las últimas décadas, la industria ha experimentado una transformación significativa debido a la convergencia de avances tecnológicos en áreas como la inteligencia artificial, la robótica, el Internet de las cosas (IoT) y los sistemas ciberfísicos. Esta confluencia de innovaciones ha dado lugar a lo que recientemente se conoce como la Cuarta Revolución Industrial o Industria 4.0 [88]. Esta nueva era industrial se caracteriza por la digitalización de los procesos productivos, la interconexión de sistemas y la automatización inteligente, lo que está redefiniendo la forma en que se conciben y operan las instalaciones industriales. En este contexto de rápido avance tecnológico, el mantenimiento de activos industriales ha adquirido una importancia crucial. Las fallas en los equipos pueden resultar en consecuencias graves, tanto en términos económicos como de seguridad operacional. A lo largo del tiempo, las estrategias de mantenimiento han evolucionado significativamente, adaptándose a las nuevas tecnologías y necesidades de la industria [72].

El mantenimiento industrial ha pasado por varias etapas de desarrollo. Inicialmente, se basaba en un enfoque correctivo o "run-to-failure", donde los sistemas o componentes se operaban hasta el momento de la falla. Este método se aplicaba debido a las limitaciones en las capacidades de monitoreo y la naturaleza estocástica de las fallas. La preparación para estas situaciones consistía principalmente en mantener un inventario de repuestos y personal listo para realizar las reparaciones necesarias [64]. Posteriormente, se desarrolló el mantenimiento preventivo, que introdujo la programación rutinaria de actividades de mantenimiento. Sin embargo, buscando superar las limitaciones de los enfoques anteriores, surgió el man-

tenimiento predictivo. Este método busca realizar acciones de mantenimiento antes de que ocurra una falla, basándose en la condición o estado de salud específico de cada máquina, con el objetivo de maximizar la vida útil de los activos [95]. El mantenimiento predictivo ha ganado prominencia en los últimos años, no por ser un concepto nuevo, sino debido a los avances tecnológicos que lo han hecho más accesible y efectivo. La reducción en los costos de sensores y el desarrollo de tecnologías de análisis y procesamiento de datos han permitido la detección de patrones que serían difíciles de identificar mediante métodos tradicionales [51]. Estos avances se alinean con los principios fundamentales de la Industria 4.0, que incluyen la interconexión de dispositivos, la recolección masiva de datos y el análisis en tiempo real [106]. La integración de estas tecnologías en los procesos de mantenimiento está permitiendo una gestión más eficiente y proactiva de los activos industriales. Actualmente, la tendencia en el campo del mantenimiento predictivo se centra en la aplicación de métodos de inteligencia artificial para estimar la Vida Útil Restante (RUL) de los activos y detectar anomalías en su funcionamiento [42]. Estas técnicas, que abarcan el aprendizaje profundo y el análisis de big data, son componentes esenciales de la transformación digital en curso en la industria [50].

Este estudio se enfoca en el modelado de una bomba tríplex en el contexto del mantenimiento predictivo, con el objetivo principal de explorar el potencial de las redes neuronales profundas para la predicción de la vida útil remanente (RUL). El trabajo abarca las siguientes etapas:

- Formulación de ecuaciones que describen el comportamiento y degradación de la bomba tríplex
- Generación de datos sintéticos que simulan la operación y degradación del activo
- Aplicación de técnicas de preprocesamiento a estos datos
- Desarrollo y evaluación de modelos predictivos basados en redes neuronales convolucionales (CNNs) y redes recurrentes de memoria a corto y largo plazo (LSTM) para la estimación del RUL

La decisión de utilizar datos sintéticos se fundamenta en las limitaciones de acceso a datos reales en entornos industriales, donde la información operativa a menudo es considerada confidencial [44]. Además, los datos industriales suelen estar desbalanceados, con una

mayoría de muestras correspondientes a operaciones normales y pocas instancias de fallas, frecuentemente mal etiquetadas. Este enfoque no solo permite el desarrollo y evaluación de nuevos algoritmos de predicción de RUL y detección de anomalías, sino que también ofrece la posibilidad de generar conjuntos de datos balanceados para mejorar los modelos existentes [114].

La relevancia de este trabajo se enmarca en el contexto de la Industria 4.0, donde la calidad y cantidad de los datos son fundamentales para el desarrollo de sistemas inteligentes y autónomos [74]. La integración efectiva de estas técnicas avanzadas de mantenimiento podría resultar en una mejora significativa de la eficiencia operativa, la reducción de costos y el aumento de la seguridad en las instalaciones industriales modernas. Se espera que los resultados de este estudio contribuyan al avance de las metodologías de mantenimiento predictivo en la era de la Industria 4.0, proporcionando perspectivas valiosas sobre la aplicación de técnicas de aprendizaje profundo en este campo. Además, se anticipa que los hallazgos y las limitaciones identificadas abrirán caminos para futuras investigaciones, estableciendo una base para la aplicación práctica de estas tecnologías en entornos industriales reales [79].

1.2. Objetivo general

El objetivo general de este trabajo de titulación es evaluar diferentes arquitecturas de redes neuronales profundas para la estimación de la vida útil remanente (RUL) en el contexto del mantenimiento predictivo, aplicadas a datos sintéticos generados mediante un modelo de ecuaciones diferenciales que simule trayectorias de degradación hasta el fallo de activos críticos.

1.3. Objetivos específicos

- Modelar una bomba tríplex como activo crítico mediante la formulación de las ecuaciones que describen su comportamiento, la caracterización de sus modos de falla y la definición del modelo de degradación que simula la evolución de su deterioro con el tiempo.
- Generar datos sintéticos realistas simulando la operación y degradación del activo, añadiendo ruido blanco y discretizando las señales generadas.
- Aplicar técnicas de preprocesamiento a los datos generados, incluyendo el remuestreo, la normalización y la selección de características en los dominios del tiempo y la frecuencia.
- Desarrollar modelos predictivos basados en redes neuronales convolucionales (CNNs) y Long Short-Term Memory (LSTM), para la estimación de la vida útil remanente (RUL) y definir los distintos escenarios para su entrenamiento.
- Evaluar el rendimiento de los modelos predictivos entrenados, definiendo métricas apropiadas para medir la precisión en las estimaciones, y determinar la eficacia de los modelos en el contexto del mantenimiento predictivo.

Capítulo 2

Marco Teórico

2.1. Fundamentos del Mantenimiento Predictivo

2.1.1. Métodos de Gestión del Mantenimiento

Para entender mejor el mantenimiento predictivo, es necesario considerar en primer lugar otras técnicas de gestión del mantenimiento cuyos enfoques son más tradicionales. En entornos y plantas industriales típicamente se emplean dos tipos de gestión del mantenimiento: el enfoque *"correctivo"* y el *"preventivo"*.

Mantenimiento Correctivo (Run-to-Failure)

Antes de la década de los 60, la misión principal de los servicios de mantenimiento era la intervención de equipos rotos con el objetivo de repararlos lo antes posible. Este enfoque en el mantenimiento es conocido como ***mantenimiento correctivo***. La lógica detrás de la gestión correctiva de activos es simple y directa: cuando una máquina se rompe, se arregla. El estándar EN 13306 (2001) define al mantenimiento correctivo como ***"un mantenimiento llevado a cabo después de ocurrida una falla y su intención es la de volver a poner a una activo en un estado en el cual éste pueda llevar a cabo sus funciones requeridas"***. El mantenimiento correctivo es una técnica reactiva que espera hasta que ocurra la falla funcional del equipo antes de aplicar o ejecutar cualquier acción de mantenimiento. Por otro lado, si la intervención del mantenimiento es provisional, se habla de "mantenimiento paliativo", mientras que si los trabajos son definitivos, se habla de "mantenimiento curativo".

Mantenimiento Preventivo

El mantenimiento correctivo se ha ido complementando gradualmente con un enfoque que busca anticiparse a las fallas mediante actividades de mantenimiento llevadas a cabo antes de la ocurrencia de éstas. Este enfoque es conocido como ***mantenimiento preventivo***, y se define como ***"un mantenimiento llevado a cabo en intervalos predeterminados o de acuerdo a criterios prescritos cuya intención es la de reducir la probabilidad de falla o la degradación de la operación de un activo"***. Cuando las intervenciones de mantenimiento se realizan en intervalos fijos de tiempo, se suele usar el término "mantenimiento predeterminado". Este tipo de mantenimiento se realiza siguiendo un horario planificado (horas de trabajo, kilómetros completados, etc.), en donde las piezas y componentes son reemplazados sin previa inspección del estado de degradación de los activos. Este tipo de mantenimiento suele conducir a un exceso de intervenciones y por lo tanto a desperdicios financieros para las empresas.

Mantenimiento Predictivo

El mantenimiento predictivo se define como ***"un mantenimiento basado en la condición que se lleva a cabo siguiendo una previsión derivada del análisis y la evaluación de los parámetros significativos de la degradación de un elemento"*** [61]. La idea subyacente es la de proyectar al futuro el estado actual de un activo, para así estimar el tiempo de operación antes de que ocurra una falla, por lo que, el ***mantenimiento predictivo*** representa un enfoque más dinámico que toma en consideración las condiciones actuales de los equipos e intenta predecir la evolución del estado de éstos a lo largo del tiempo.

La Figura 2.1 ejemplifica los tipos de mantenimiento. Cada uno de ellos tiene su rol, pero al optar por un enfoque ***correctivo***, las industrias posponen las actividades de mantenimiento y asumen el riesgo de la pérdida de disponibilidad de sus activos; por otra parte, el mantenimiento ***preventivo*** se anticipa a las intervenciones, lo cual usualmente resulta en un recambio de partes y componentes que ni siquiera han cumplido la mitad de su ciclo de vida útil o de operación [14]. Por lo tanto, una buena estrategia de mantenimiento debiese mejorar la condición de los equipos, reducir las tasas de falla en los equipos y minimizar los costos de mantenimiento mientras se maximiza la vida útil de los activos. Debido a lo anterior, el mantenimiento ***predictivo*** es el que se alza entre los demás enfoques.

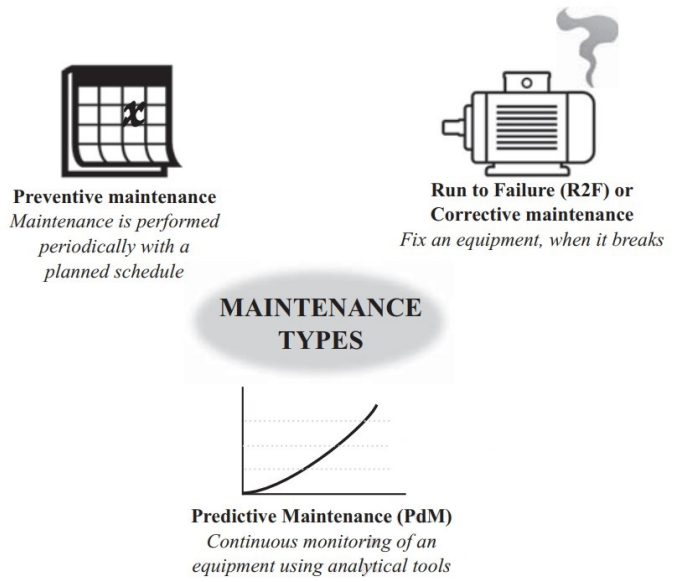


Figura 2.1: Vista general de los tipos de mantenimiento.
(Fuente: *A systematic literature review of machine learning methods applied to predictive maintenance* - Carvalho, T. P., Soares, F. A. [14])

2.1.2. Fallas y Modos de falla

El concepto de ***falla*** juega un rol protagónico en la gestión del mantenimiento, ya que las distintas estrategias siempre buscan corregirlas o bien evitar que ocurran. A la hora de calificar un evento como "falla", suele existir discrepancia en opiniones. Sin embargo, la definición más precisa se relaciona con el estado deseado de funcionamiento del equipo bajo análisis. De acuerdo con la norma SAE JA 1011, una falla funcional se define como ***un estado en el cual un recurso o sistema físico no es capaz de ejecutar una función específica a un nivel de desempeño deseado*** [93]. Una falla está siempre relacionada con una función requerida y ocurre cuando esta no puede ser realizada o su desempeño cae fuera de los requerimientos esperados [57]. Por lo tanto, si un equipo presenta una pérdida de capacidad total o parcial para mantener su nivel de operación establecido por la función que debe realizar dentro de su contexto operativo, estamos en presencia de una falla. Las fallas se suelen agrupar conforme al siguiente criterio:

Falla Funcional u Operacional

Ocurre una falla funcional cuando el componente deja de satisfacer completamente la función requerida como resultado de manifestarse uno o más modos de falla.

Falla Sintomática o Potencial

Una falla sintomática es una condición de estado verificable (defecto o desviación incipiente) en los componentes o sus propiedades, que indica que una falla funcional está en proceso de ocurrencia, aunque el componente no haya dejado de cumplir completamente sus funciones. Si no se toma acción alguna, la falla sintomática conducirá irremediablemente a una falla funcional.

Falla Recurrente

Categoría de fallas funcionales repetitivas fuera de la frecuencia de ocurrencia establecida como aceptable. También se denominan *crónicas* y pueden ser causadas por un modo de falla o por diferentes actuando simultáneamente.

Por otro lado, el concepto de ***modo de falla*** es central en la gestión del mantenimiento, ya que describe la manera en que un elemento o sistema puede fallar. Según la norma SAE JA 1011, un modo de falla se define como el evento que origina la incapacidad de un equipo o componente para cumplir su función [93]. Un modo de falla es, por lo tanto, una descripción del estado observable del equipo una vez que ha ocurrido la falla, o la forma en que esta se manifiesta. Este concepto no debe confundirse con las causas del fallo ni con los mecanismos de degradación.

La afirmación de que el manejo adecuado de los modos de falla es clave en la gestión de activos cobra relevancia cuando consideramos que estos modos representan la forma específica en que un equipo puede perder su funcionalidad, ya sea por un evento repentino o progresivo. Estos eventos pueden tener un impacto significativo en el rendimiento del equipo, dependiendo de cómo se manifieste la falla [57].

Por otro lado, el concepto de falla implica que el pronóstico de las mismas se debe basar en criterios de evaluación cuyos limites dependen de los sistemas de monitoreo y en el desempeño esperado de los activos. En otras palabras, el ***pronóstico*** que se busca con el mantenimiento predictivo no sólo implica la proyección del comportamiento de los sistemas en el futuro, sino que también la habilidad de identificar el estado de salud de los equipos en cada instante. La métrica principal en el pronóstico de fallas es el ***Tiempo para la Falla (TTF)***, también denominado como ***Vida Útil Remanente (RUL)***.

Durante las últimas décadas, varias herramientas y métodos de pronóstico de fallas se han propuesto, donde por lo general éstos difieren según el tipo de aplicación considerada y de la naturaleza de los datos disponibles. La siguiente clasificación usualmente logra un buen consenso dentro de la comunidad del PHM. (Figura 2.2):

- Pronóstico de fallas basado en un modelo físico.
- Pronóstico de fallas guiado por datos.
- Pronóstico de fallas híbrido.

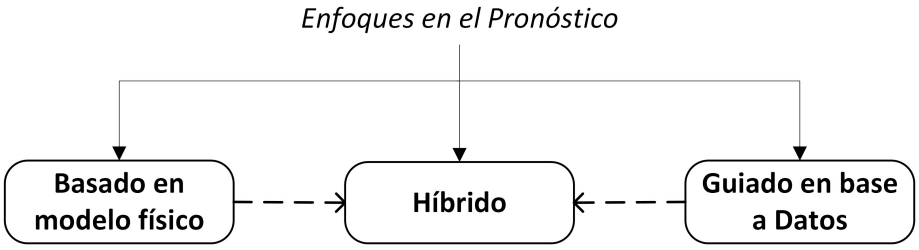


Figura 2.2: Taxonomía de los enfoques en el pronóstico.
(Fuente: Elaboración propia a partir de: *From Prognostics and Health Systems Management to Predictive Maintenance 1 - R. Gouriveau, K. Medjaher, N. Zerhouni [61])*

Los métodos basados en un modelo físico requieren la construcción de un modelo dinámico que represente el comportamiento del sistema e integre el mecanismo de degradación (modelos de fatiga, desgaste, corrosión, etc.), cuya evolución se modela mediante una ley determinista o mediante un proceso estocástico. Estos métodos suelen ofrecer resultados más precisos que los de los otros dos enfoques y además tienen la ventaja de ser interpretables, ya que los parámetros del modelo están relacionados con las cantidades físicas del sistema, y una degradación induce sistemáticamente desviaciones paramétricas. Sin embargo, el mayor inconveniente de este tipo de enfoque reside en el hecho de que para los sistemas reales es difícil, o incluso imposible, obtener un modelo dinámico que integre los mecanismos de degradación de forma analítica debido a la complejidad y la diversidad de los fenómenos físicos.

El pronóstico basado en datos se basa en la explotación de los datos de monitorización, que se procesan para extraer las características que reflejan el comportamiento del sistema y su degradación. Estas cantidades se emplean además para entrenar modelos predictivos de los estados actuales y futuros del sistema, y así proporcionar una estimación de la ***Vida Útil Remanente***. El desarrollo de sensores y sistemas de monitorización, combinado con las cre-

cientes prestaciones informáticas, ofrecen notables capacidades de procesamiento, análisis y aprendizaje, por lo que facilitan la aplicación de este enfoque. Por otro lado, el pronóstico basado en datos pierde precisión cuando los modelos aprendidos se desvían del comportamiento real del sistema, por lo que, constituye un compromiso entre aplicabilidad y precisión.

Un método de pronóstico híbrido es una integración de un modelo de comportamiento físico y un enfoque basado en datos. Por lo general, se distinguen dos clases de pronósticos híbridos (Figura 2.3). Cuando se puede establecer un modelo físico se utiliza un enfoque basado en datos para estimar y predecir los parámetros no observables del modelo. En este caso, se habla de "enfoques en serie". Un enfoque llamado "en paralelo" consiste en combinar la salida del modelo físico con la salida de una herramienta basada en datos para reconstruir una salida global. En estos casos, la herramienta basada en datos suele utilizarse para estimar y predecir los fenómenos que no se explican y que, por tanto, no se han modelizado.

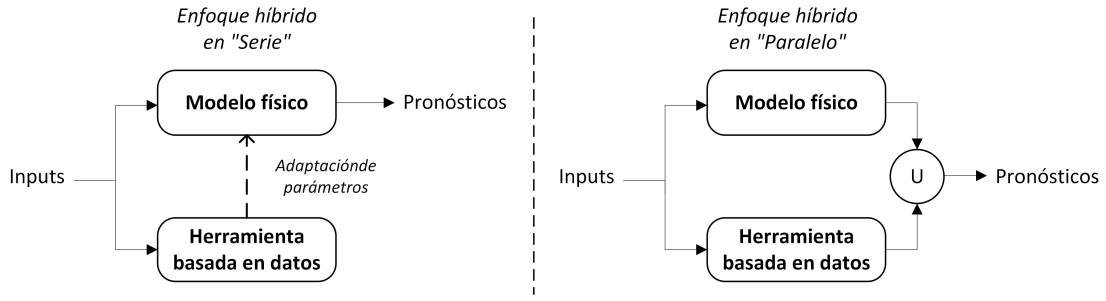


Figura 2.3: Enfoques en el pronóstico híbrido.
(Fuente: Elaboración propia a partir de:*From Prognostics and Health Systems Management to Predictive Maintenance 1 - R. Gouriveau, K. Medjaher, N. Zerhouni [61])*

2.1.3. Mantenimiento Predictivo

Con el mantenimiento predictivo se mide la degradación o deterioro progresivo de cada equipo por separado (aunque también se suelen medir los síntomas prematuros de sus partes constituyentes) sin desarmes mayores, tiempo antes que se produzca la detención súbita correspondiente a la avería total del componente (falla funcional). Por lo tanto, el mantenimiento predictivo procura llevar al límite de su utilización la vida útil remanente de los componentes del activo [72].

Algunas de las herramientas utilizadas por el Mantenimiento Predictivo son:

- Los sentidos humanos a través de inspecciones.
- Análisis de vibraciones.
- Termografía infrarroja.
- Inspección por ultrasonido.
- Tribología
- Emisión acústica.
- Análisis de partículas de desgaste, etc.

En general los parámetros que se miden en los activos permiten poner de manifiesto sus fallas sintomáticas provocadas por desgaste, defectos o desviaciones de alguna variable física.

Para aplicar ***mantenimiento predictivo*** debe ser posible detectar alguna variable relacionada con el grado de deterioro del elemento. En la Figura 2.4 se aprecia cómo el nivel de buen funcionamiento de cierta variable comienza a decaer en el punto E. A partir de E, el deterioro se hace progresivo hasta culminar en la falla funcional (punto F). No obstante, en raras ocasiones es posible notar el síntoma justo en E. Cuando el deterioro del activo alcanza cierto valor, se dice que existe una falla sintomática. Cuanto antes pueda detectarse el síntoma (punto S), más largo será el espacio de tiempo entre éste y la falla total de la función, (Δt_{SF} en la figura 2.4). Δt_{SF} corresponde a la ***Vida Útil Remanente*** del equipo, siendo éste el intervalo de tiempo durante el cual la falla advierte de su existencia.

La curva de la Figura 2.4 muestra el grado de deterioro de una condición física o variable. La magnitud medida, intrínseca al elemento, debe demostrar una relación directa entre los valores que adopta y el grado de deterioro. El nivel vibratorio de una máquina rotante, la temperatura de un motor eléctrico o la concentración de ácido disuelto en un aceite, son ejemplos de variables mensurables y relacionadas con la degradación del componente o del equipo al que pertenece. A partir del punto E se produce un quiebre en el estado normal y comienza a decaer; de todas maneras, no es seguro que el síntoma pueda identificarse sino hasta S. Por esta razón S se llama falla sintomática y su empeoramiento total hasta allí es ΔE_S. Si bien el empeoramiento aparece como decreciente a partir de E, éste puede obedecer al crecimiento de alguna magnitud física (temperatura, vibración, ruido, etc.). El deterioro siempre se considera decreciente, independientemente del comportamiento de la magnitud que se evalúa.

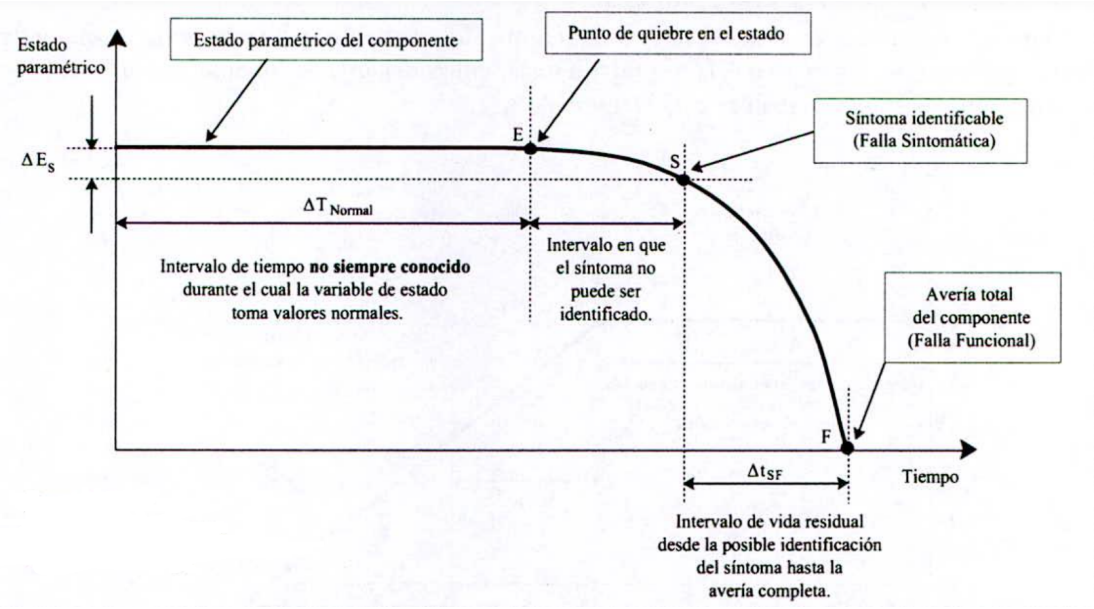


Figura 2.4: Curva de estado paramétrico e intervalo de vida residual.
(Fuente: *Manual de Mantenimiento, Ingeniería, Gestión y Organización* - Alejandro J. Pistarelli. [71])

En resumen, el ***mantenimiento predictivo*** consiste en un método de gestión del mantenimiento que se basa en el análisis y evaluación de una serie de características o variables recogidas del sistema (principalmente mediante la implementación de sensores) dado un modelo de ***degradación*** obtenido previamente (ver Figura 2.5). El resultado más conocido del análisis y evaluación de las características del sistema en el modelo de degradación corresponde a la ***Vida Útil Remanente (RUL)***.

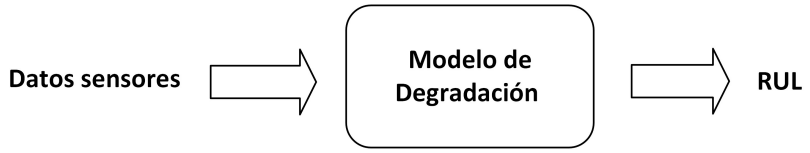


Figura 2.5: Esquema Modelo de Degradación - RUL.
(Fuente: Elaboración propia.)

2.1.4. Ventajas del Mantenimiento Predictivo

Dentro de sus ventajas, se destacan:

- ✓ Reducir los tiempos de parada no programada pronosticando la falla imprevista con la mínima interrupción del equipo.

- ✓ Seguir la evolución de una falla sintomática a lo largo del tiempo evitando llegar a la avería catastrófica o disminuyendo su consecuencia.
- ✓ Optimizar los recursos (mano de obra y materiales) programando las tareas correctivas o de restauración. Suelen disminuirse las horas extra por reparación de imprevistos.
- ✓ Ofrece un panorama del estado de los componentes para decidir el momento más oportuno de su reemplazo o reparación
- ✓ Realizar diagnósticos discrecionales para analizar causas raíz de la falla.
- ✓ Aprovechar más *cantidad de vida* en equipos con periodos de vida útil prolongados o cuyo empeoramiento hacia la falla es progresivo.
- ✓ Ahorrar energía detectando fugas de calor en sistemas de calefacción o aislamiento.
- ✓ Eliminar daños emergentes en piezas contiguas del equipo o en otras máquinas.

2.2. Fundamentos del Machine Learning

Desde su evolución, el ser humano ha utilizado muchos tipos de herramientas para realizar diversas tareas de forma más sencilla. La creatividad del cerebro humano llevó a la invención de diferentes máquinas, las cuales han facilitado la vida del ser humano al permitirle satisfacer diversas necesidades vitales, como los viajes, las industrias y la informática. Y el ***Machine Learning*** es una de ellas. Cuando la mayoría de la gente oye hablar de machine learning, se imagina alguna especie de robot inteligente con pensamiento autónomo similar a un Terminator, pero el machine learning no es precisamente una fantasía futurista salida de la ciencia ficción.

2.2.1. ¿Qué es el Machine Learning?

En 1959, **Arthur Samuel**, un profesional de la informática, quién fue pionero en el estudio del campo de la inteligencia artificial, describió al machine learning como ***"el campo de estudio que da a las computadoras la habilidad de aprender sin ser explícitamente programadas"*** [84].

Una definición más técnica e ingenieril es dada por **Tom M. Mitchell** en 1997: ***"Se dice que un programa de ordenador es capaz de aprender de la experiencia E con respecto a alguna tarea T y alguna medida de rendimiento P, si su rendimiento en T medido por P mejora con la experiencia E"*** [ver 63, p. 2].

Para entender mejor la definición anterior, se procede a definir los siguientes conceptos:

- **Tarea T:** Se refiere a los tipos de tareas que pueden ser resueltas mediante los modelos y algoritmos del aprendizaje automático. Sea $\mathbf{x} = \{x_1, x_2, x_3, ..., x_n\} \in \mathbb{R}^n$ una colección de características o variables, diremos que **x** es una muestra o entrada del modelo de aprendizaje automático donde cada valor de x_i es una característica o variable. Por ejemplo, los datos entregados por sensores de un sistema en un instante dado de tiempo son una posible muestra o entrada de un modelo de aprendizaje automático. Algunas de las tareas más comunes que abordan los algoritmos de aprendizaje automático son las de clasificación, regresión, clustering, detección de anomalías, entre otras.

- **Medida de Rendimiento P:** Para evaluar que tan preciso es un modelo generado por un algoritmo de aprendizaje automático, se debe medir con un valor cuantitativo el rendimiento del modelo. Esta medida de rendimiento P generalmente está asociada al tipo de tarea T a ser llevada a cabo.

- **Experiencia E**: La experiencia se entiende como el dataset con el cual el modelo de aprendizaje automático se entrena para que, a partir de éste se lleve a cabo la tarea T solicitada. Un dataset no es más que un conjunto de datos de entrada o muestras, es decir:

$$\mathbf{X} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, ..., \mathbf{x}^{(m)}\} \tag{2.1}$$

  Donde X es un dataset compuesto de m entradas con n características o variables cada una.

El machine learning o ***aprendizaje automático*** es una aplicación dentro del campo de la inteligencia artificial, en donde una computadora/máquina es capaz de aprender de las experiencias pasadas para realizar predicciones futuras.

2.2.2. Tipos de Machine Learning

En el campo del aprendizaje automático, existen diversas formas de clasificar los sistemas. Una de las más frecuentes y ampliamente estudiadas en la literatura especializada es la que categoriza estos sistemas según el tipo de supervisión utilizada durante el entrenamiento de los modelos [63] [6]. Sin embargo, los sistemas de aprendizaje automático también pue-

den clasificarse atendiendo a otros criterios. Por ejemplo, se pueden diferenciar en función de su capacidad para aprender de forma incremental durante su funcionamiento, o si operan mediante la comparación directa de nuevos datos con información previamente conocida. Alternativamente, algunos sistemas se caracterizan por su habilidad para detectar patrones en los datos de entrenamiento con el fin de construir modelos predictivos [89] [28].

Generalmente, los sistemas de aprendizaje automático se pueden clasificar en tres grandes categorías: Aprendizaje Supervisado, Aprendizaje no Supervisado y Aprendizaje por Reforzamiento.

Aprendizaje Supervisado

En el aprendizaje supervisado, las máquinas se entrenan utilizando datos de entrenamiento "etiquetados", donde en base a dichos datos, las máquinas son capaces de predecir las salidas. La presencia de estos datos etiquetados implica que al momento de entrenar al modelo de aprendizaje automático, algunos de los datos de entrada ya deben estar correctamente clasificados con su respectiva salida. Por lo tanto, el aprendizaje supervisado es un proceso que consiste en proporcionar datos de entrada y de salida correctos al modelo, teniendo como objetivo encontrar una función de mapeo para asignar el vector de entrada X con el vector de salida Y. Lo anterior implica que el dataset de entrada se divide en dos grupos: dataset de entrenamiento y dataset de prueba o validación. La siguiente Figura 2.6 muestra el flujo de trabajo de un sistema de aprendizaje automático supervisado.

Aprendizaje no Supervisado

A diferencia de los algoritmos de aprendizaje supervisado, los no supervisados no reciben datos de entrada etiquetados con su correspondiente salida, por lo que tienen que encontrar por sí mismos la estructura presente en los datos que ingresan al modelo. El aprendizaje automático no supervisado puede ser un objetivo en sí mismo (descubrir patrones ocultos en los datos), o también un medio para alcanzar un fin (aprender de características en los datos).

Aprendizaje por Reforzamiento

El aprendizaje por reforzamiento es un campo que se basa en el modelado de un entorno para que un modelo, llamado "agente", aprenda a interactuar con el entorno resolviendo una función de optimización basada en recompensas por las buenas acciones y castigos por las malas, es decir, el agente aprende en un entorno interactivo por ensayo y error utilizando la retroalimentación de sus propias acciones y experiencias para llegar hacia el objetivo final.

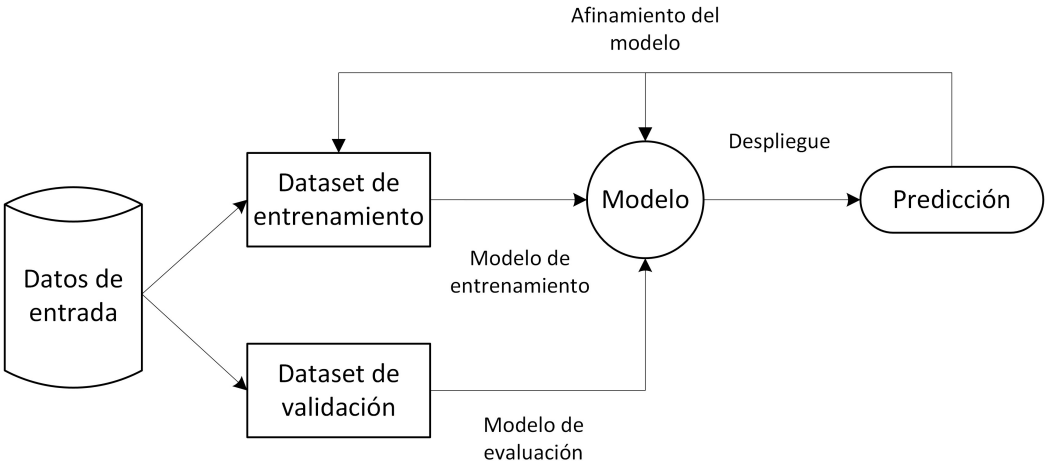


Figura 2.6: Flujo de trabajo de un sistema de aprendizaje automático supervisado.
(Fuente: Elaboración propia a partir de "Machine Learning: A Review" - Batta Mahesh [58])

2.2.3. Tareas de Machine Learning

En el campo del aprendizaje automático, el enfoque hacia los diferentes problemas puede ser categorizado en una serie de ***tareas***, independiente de la naturaleza supervisada o no supervisada del método de machine learning empleado. A continuación se presentan las tareas más comunes y relevantes que se pueden realizar utilizando métodos de aprendizaje automático.

Regresión

Las tareas de regresión se basan en la estimación del valor de una variable continua en función de las características disponibles, es decir, se le pide al algoritmo de aprendizaje automático que prediga un valor numérico dada una entrada. Para resolver esta tarea se utiliza una función $f : \mathbb{R} \rightarrow \mathbb{R}^n$. Las tareas de regresión representan una de las áreas principales en el aprendizaje automático, especialmente en el área del mantenimiento, ya que el el objetivo final de la aplicación del campo de Machine/Deep Learning es aproximar la vida útil remanente de un activo físico para así obtener una programación muy precisa de las tareas y actividades de mantenimiento, ahorrando costes y minimizando los tiempos de inactividad.

El ejemplo más básico en tareas de regresión corresponde a la **Regresión Lineal**, donde el objetivo es aprender el valor de los coeficientes α y β que se aproximan y ajustan mejor a los datos disponibles para resolver un sistema del tipo $\alpha \cdot X + \beta \cdot Y = Z$, con el objetivo de

aproximar un buen valor para Z sobre un espacio de valores continuos para X e Y. Además de la regresión lineal, en la literatura se suelen encontrar los siguientes métodos para resolver tareas de regresión:

- Kernel regression (*Mayor precisión*).
- Regresión de proceso gaussiano (*Gaussian process regression*).
- Árboles de regresión (*Regression trees*).
- Regresión de vectores de apoyo (*Support Vector Regression (SVR)*).
- LASSO / Ridge.
- Deep Learning.
- Bosques aleatorios (Random forests).

Clasificación

Esta es una de las tareas principales del machine/deep learning. Predominantemente, esta tarea es de naturaleza supervisada, ya que se basa en las etiquetas para enseñar al modelo a identificar patrones que aparten las diferentes clases. Las tareas de clasificación están relacionadas con la predicción de una categoría de datos (variables discretas) Y a partir de las características de una muestra respectiva de características X. Para resolver estas tareas se le pregunta al algoritmo de aprendizaje automático programa específicamente a cual categoría K el dato de entrada pertenece, para ello el algoritmo produce una función $f : \mathbb{R}^n \rightarrow \{1, ..., k\}$, entonces el modelo $y = f(x)$ asigna a un vector X una categoría Y según la clasificación numérica [28]. Por lo tanto, técnicamente, lo que el modelo está aprendiendo es el hiperplano de separación de límites entre diferentes clases que maximizan la segregación en los datos.

Algunos de los casos de uso más comunes pueden encontrarse en el ámbito de la sanidad, como por ejemplo si una persona padece o no una determinada enfermedad. También tiene su aplicación en casos de uso financiero, como determinar si una transacción es un fraude o no. Los siguientes métodos suelen aplicarse para resolver tareas de clasificación:

- Kernel discriminant analysis (*Mayor precisión*).
- K-Nearest Neighbors.
- Redes neuronales artificiales (*Artificial Neural Networks (ANNs)*).
- Máquina de vectores de apoyo (*Support Vector Machine (SVM)*).
- Bosques aleatorios (Random forests).
- Árboles de decisión (*Decision trees*).

- Regresión logística.
- Naive bayes.
- Deep Learning.

Clustering

El clustering es una tarea dedicada a encontrar patrones en los datos que permitan agrupar muestras similares (etiquetadas, no etiquetadas o ambas), generando una segmentación de los datos con fines de visualización, reducción de la dimensionalidad o "etiquetado generalizado"[1]. Algunos de los ejemplos más comunes son la segmentación de clientes y la identificación de características de productos para la hoja de ruta de productos. En la literatura se suelen encontrar los siguientes algoritmos de aprendizaje automático para resolver tareas de clustering:

- Desplazamiento medio (*Mayor precisión*).
- Agrupación jerárquica (*Hierarchical Cluster Analysis (HCA)*).
- K - Means.
- Topic Models.
- DBSCAN.

En particular, la agrupación jerárquica (*Hierarchical Cluster Analysis (HCA)*) se puede utilizar para medir el nivel de incidencia de las características de un conjunto de datos en el proceso de toma de decisiones del modelo de aprendizaje automático. El resultado de la visualización gráfica de una tarea de agrupación jerárquica suele representarse a través de un dendrograma, el cual corresponde a un tipo de representación gráfica o diagrama de datos en forma de árbol que organiza los datos en subcategorías que se van dividiendo en otros hasta llegar al nivel de detalle deseado.

Reducción de dimensionalidad

La reducción de la dimensionalidad corresponde a la tarea de transformar los datos de entrada desde un espacio de alta dimensión a un espacio de baja dimensión, de manera que la representación de los datos en el espacio baja dimensión sea capaz de conservar algunas de las propiedades significativas de los datos de entrada originales. Por lo tanto, la reducción de la dimensionalidad busca reducir el número de variables aleatorias consideradas, ya que

[1] Si se proporcionan unas pocas muestras etiquetadas, se etiqueta todo el cluster respectivo con esa etiqueta para convertir una tarea no supervisada a una tarea supervisada

trabajar en espacios de alta dimensión puede ser indeseable debido a que los datos brutos suelen ser escasos y su análisis suele ser computacionalmente intratable (difícil de controlar o tratar). La reducción de la dimensionalidad suele ser habitual en los campos que tratan con un gran número de observaciones y/o un gran número de variables, como el procesamiento de señales, el reconocimiento del habla, la neuroinformática y la bioinformática [98]. A continuación se presentan algunos métodos de aprendizaje automático que suelen utilizarse para la reducción de la dimensionalidad:

- Análisis de componentes principales (*Principal Component Analysis (PCA)*).
- Aprendizaje múltiple / KPCA (*Kernel PCA*).
- Locally Linear Embedding (LLE).
- *t*-Distribuited Stochastic-
- Neighboor Embedding (*t*-SNE).

Por otro lado, las tareas de reducción de dimensionalidad pueden dividirse entre la **selección** de características y la **extracción** de características. En particular, la tarea selección de características se utiliza sobre todo al momento de construir modelos, seleccionando el subconjunto adecuado de características que contengan la mayor cantidad de información y minimizando las redundancias mediante técnicas estadísticas, de envoltura o de regularización que caracterizan la interacción de las características entre sí. Por ejemplo, la prueba estadística de selección de características Chi-Cuadrado se utiliza para probar la independencia de dos eventos, donde dadas dos variables, O_i es la frecuencia de las muestras observadas y E_i es la frecuencia esperada. Esta prueba mide cómo se desvían E_i y O_i entre sí, como se muestra en la Ecuación 2.2.

$$\chi_c^2 = \sum \frac{(O_i - E_i)^2}{E_i} \tag{2.2}$$

De este modo, el método pretende captar las características que son altamente dependientes de la respuesta observada, donde cuanto más alta sea la puntuación χ_c^2, la característica será más dependiente de la respuesta y es más probable que ésta contenga información significativa y útil para la extracción de patrones por un modelo de aprendizaje automático.

Detección de anomalías

La detección de anomalías corresponde al proceso de identificar patrones inusuales en los datos y que no se ajustan a su comportamiento esperado. La detección de anomalías se suele

utilizar en una amplia gama de aplicaciones, como la detección de actividades fraudulentas en los datos financieros, la detección de comportamientos maliciosos en los datos de tráfico de la red y la identificación de fallas en los datos provenientes de los sensores. La detección de anomalías puede llevarse a cabo utilizando una variedad de modelos de aprendizaje automático, como los métodos basados en la densidad, los métodos basados en clusters y los métodos basados en reglas. En la literatura se suelen encontrar los siguientes métodos para resolver tareas de detección de anomalías:

- One-class SVM.
- Isolation Forest.
- Variational Autoencoders.

En la Figura 2.7 se muestra una representación esquemática que agrupa los tipos de aprendizaje automático según su naturales con las tareas típicas asociadas a cada uno de dichos tipos.

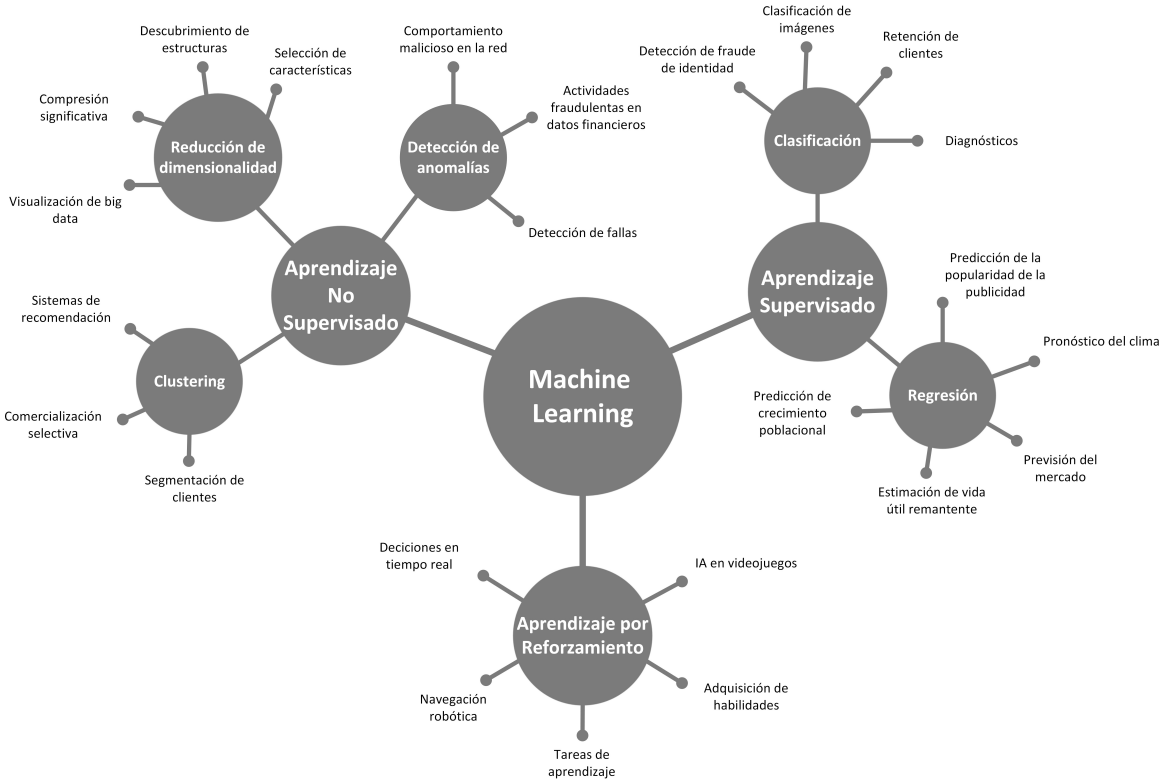


Figura 2.7: Diagrama resumen tipos de machine learning.
(Fuente: Elaboración propia a partir de "Proyecto de Machine Learning en Mantenimiento: Cómo enfocarlo para el éxito" - Antonio Marín Écija. [117])

2.2.4. Etapas en el desarrollo de modelos de Machine Learning

A la hora de desarrollar o diseñar un modelo de aprendizaje automático para su aplicación, por ejemplo en el mantenimiento predictivo, se deben tomar en consideración algunos pasos, los cuales son: etapa de *selección de datos históricos*; etapa de *preprocesamiento de datos*; etapa de *selección, entrenamiento y validación del modelo*; y etapa de *mantenimiento del modelo* [92]. La Figura 2.8 ilustra estos pasos.

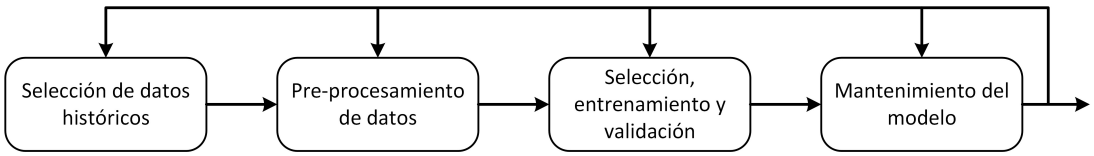


Figura 2.8: Pasos principales para el diseño de un modelo de machine learning.
(Fuente: "Ensemble learning methodologies for soft sensor development in industrial processes" - Soares, S. G. [92])

La etapa de la selección de los datos históricos identifica cómo es que los datos son recolectados y almacenados de modo que los datos más valiosos son seleccionados para alimentar los modelos de machine learning. En el caso específico del mantenimiento predictivo esta etapa también se conoce como "etapa de adquisición de datos", y apunta a la obtención de datos relevantes y relacionados a la salud de los sistemas [40].

La etapa de pre-procesamiento de los datos procesa y transforma los datos de modo que estos puedan ser procesados de manera eficiente por los modelos de machine learning. Esta etapa incluye la transformación de los datos (p.e. la normalización de los datos), la limpieza de los datos (p.e. tratamiento de los datos faltantes y eliminación de los valores atípicos) y reducción de datos (p.e. reducción de la dimensionalidad). En el mantenimiento predictivo, la etapa del preprocesamiento de los datos se encarga del análisis de los datos recolectados para generar un mejor entendimiento e interpretación de la información.

La etapa de selección, entrenamiento y validación del modelo involucra la selección del modelo de machine learning adecuado, el entrenamiento del mismo (p.e. el desarrollo del modelo) y la validación del modelo (p.e procedimiento que evalúa si el modelo puede representar el sistema subyacente). En el mundo del mantenimiento predictivo, esta etapa es denominada como "etapa de toma de decisiones de mantenimiento", y apunta a decidir el mejor algoritmo para la aplicación en el mantenimiento predictivo.

La etapa de mantenimiento del modelo tiene por objetivo mantener o mejorar el desempeño del modelo generado a lo largo del tiempo. Lo anterior se debe a que a menudo se presentan ambientes cambiantes donde las aplicaciones de los modelos pueden cambiar con el tiempo, lo que lleva a que el modelo se degrade debido a que en un principio éste fue generado a partir de condiciones que ya no son representativas del problema. Esta etapa considera el monitoreo del modelo generado, y en caso de que el rendimiento del modelo se degrade, los modelos deben volver a pasar por la etapa de entrenamiento y validación.

2.3. Machine Learning en el Mantenimiento Predictivo

El mantenimiento predictivo requiere de un buen conocimiento y caracterización de los patrones presentes en la información recabada para generar un modelo robusto, pero en muchas ocasiones resulta difícil determinar los grupos de flujos de información relevante para la detección de una falla, así como también la ventana de tiempo en la cual dichos flujos de información producen los patrones característicos que son indicadores de la falla [44]. Por lo tanto, en general se pude decir que el éxito de los modelos de mantenimiento predictivo depende de tres componentes principales: disponer de los datos adecuados, enmarcar el problema de forma apropiada y evaluar las predicciones de manera pertinente [78].

Al igual que en el desarrollo de modelos de machine learning, para construir un modelo de fallas para el mantenimiento predictivo, es necesaria una suficiente cantidad de datos históricos que permitan captar la información sobre los acontecimientos que conducen al fallo. Además, las características generales del sistema, consideradas comúnmente como "estáticas", como las propiedades mecánicas, el uso promedio y las condiciones de funcionamiento de los activos también pueden aportar información valiosa, aunque hay que tener la precaución de que contar más datos no siempre es mejor.

Por otro lado, a la hora de enmarcar un modelo de mantenimiento predictivo, es necesario tener en consideración qué tipo de output o salida se busca que entregue el modelo (p.e. predicción del *RUL*, predicción de fallas, detección de anomalías, etc.). También es pertinente considerar si es que los datos históricos disponibles son suficientes, o si cada evento esta propiamente "etiquetado" (p.e. cuales mediciones corresponde a un buen funcionamiento y estado del activo, y cuales no), la proporción entre datos etiquetados con buen funcionamien-

to y mal funcionamiento, y la antelación que se busca para que el modelo indique que se producirá un fallo.

Luego de haber enmarcado apropiadamente al modelo de mantenimiento predictivo, es necesario decidir qué estrategia de modelización se ajusta mejor a los datos disponibles y al resultado deseado. En este contexto, el machine learning toma relevancia como una alternativa atractiva al momento de analizar la información proveniente de los sistemas productivos y de manufactura, ya que posee la habilidad de manejar datos multivariados[2] de gran dimensionalidad, proporcionando un enfoque complementario a la planificación del mantenimiento mediante el análisis de conjuntos de datos significativos de variables individuales de rendimiento de la máquina y del entorno, identificando firmas y perfiles de fallos, y proporcionando una predicción de fallos para piezas individuales.

Existen múltiples estrategias basadas en el aprendizaje automático para el desarrollo de modelos de mantenimiento predictivo, donde las más utilizadas son las siguientes:

- Modelos de ***regresión*** para predecir la Vida Útil Remanente (RUL) de los activos.
- Modelos de ***clasificación*** para la predicción de fallas dentro de una ventana de tiempo.
- Señalización de los comportamientos anómalos en los activos mediante modelos de ***detección de anomalías***.
- Modelos de supervivencia para la predicción de la probabilidad de fallo en el tiempo, enfocados en el propio proceso de degradación y la probabilidad de fallo resultante.

2.4. Principales desafíos en Machine Learning

2.4.1. Calidad de los datos

Uno de los desafíos principales que los profesionales se pueden encontrar a la hora de diseñar un modelo de aprendizaje automático corresponde a la ausencia de datos de buena calidad, ya que un modelo presenta mayor o menor desempeño en función de los datos datos de entrenamiento del mismo. Los datos poco limpios y con presencia de "ruido" pueden causar que toda la etapa del pre-procesamiento de los datos sea larga y cuantiosa, sin embargo, como se vio en secciones anteriores, la etapa del pre-procesamiento es esencial para

[2]Datos que se obtienen a partir de más de una variable

los modelos, ya que en ausencia de esta, muy probablemente los algoritmos incurrirán en predicciones inexactas y defectuosas. Por lo tanto, el proceso de limpieza de los datos para la etapa de entrenamiento, que incluye la eliminación de los valores atípicos, el filtrado de los valores perdidos y tener muestras de datos equiespaciadas en el tiempo, debe ser realizada a la perfección.

2.4.2. Desajustes en el entrenamiento

El ***Overfitting*** o sobreajuste de los datos se refiere a un modelo que modela los datos de entrenamiento demasiado bien. Esto se produce cuando un modelo aprende las particularidades y el ruido presente en los datos de entrenamiento, hasta el punto en que esto repercute negativamente en el rendimiento del modelo, presentando un ***test error*** alto y siendo incapaz de generalizar para datos nuevos. Lo anterior significa que el ruido o las fluctuaciones aleatorias de los datos de entrenamiento son recogidos y aprendidos como conceptos por el modelo, donde el problema radica en que estos conceptos no se aplican a los nuevos datos y afectan negativamente a la capacidad de los modelos para generalizar.

Por otro lado, el ***Underfitting*** o subajuste se refiere a un modelo que no es capaz de modelar ni ajustarse correctamente a los datos de entrenamiento, presentando un ***train error*** alto. El ajuste insuficiente destruye la precisión del modelo de aprendizaje automático y su aparición simplemente significa que el modelo o el algoritmo no se ajusta a los datos lo suficientemente bien. Suele suceder cuando se cuenta con menos datos para construir un modelo preciso y también cuando se intenta construir un modelo lineal con datos no lineales. La Figura 2.9 muestra de manera sencilla y representativa los conceptos anteriores.

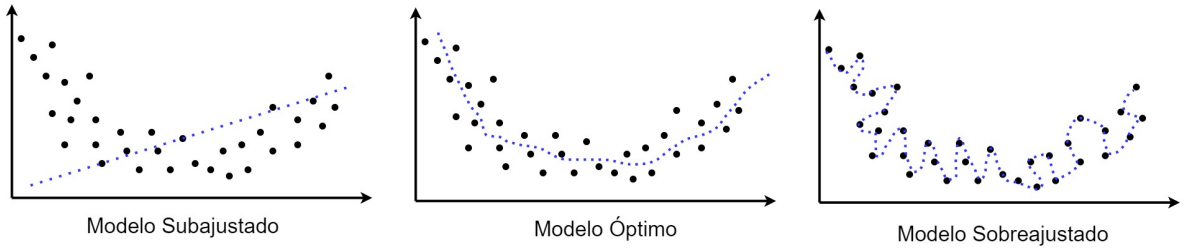


Figura 2.9: Modelo Subajustado, Óptimo y Sobreajustado en aprendizaje automático.
(Fuente: Elaboración propia.)

2.4.3. Falta de datos en el entrenamiento

La tarea más importante a realizar en el proceso de aprendizaje automático corresponde al entrenamiento del modelo con suficientes datos para conseguir un resultado preciso. Una cantidad menor de datos de entrenamiento producirá predicciones inexactas o muy sesgadas al momento de tomar decisiones por ejemplo, en el caso que se requiera realizar un algoritmo de reconocimiento de perros, y al modelo sólo se le entrene con imágenes de perros negros y blancos, al momento de mostrarle al modelo un perro de dos colores, éste no va a saber como clasificarlo y realizará una predicción errónea.

2.5. Redes Neuronales

¿Cómo detecta un sistema biológico la información sobre el mundo físico?

¿En qué forma se almacena o se recuerda la información?

¿Cómo influye la información contenida en el almacenamiento, o en la memoria, el comportamiento y el reconocimiento?

F. Rosenblatt intentó responder estas tres preguntas en 1958 [82] con su modelo probabilístico para el almacenamiento de la información en el cerebro. Sin embargo, a pesar de su antigüedad, estas preguntas siguen definiendo la mayor parte de lo que se conoce hoy en día como ***Red Neuronal*** y ***Aprendizaje Profundo***. La primera de estas cuestiones pertenece al ámbito de la fisiología sensorial, mientras que la segunda plantea una posible codificación de la información, y la tercera plantea que este tipo de codificación afecta a las propiedades de una Red Neuronal. Esto puede verse, por ejemplo, en los diferentes tipos de Redes Neuronales (codificadores) y su aplicación habitual (su rendimiento). Una ***Red Neuronal Convolucional*** puede procesar un mapa de características con gran eficacia, mientras que una ***Red Neuronal Recurrente*** puede modelar con éxito series temporales. Ambos tipos de redes tienen una dinámica única dada por las ecuaciones que las determinan. De este modo, se puede ver que este campo de estudios continúa con el mismo espíritu que sus pioneros, que en realidad buscaban pistas sobre lo que define el comportamiento de la mente de las personas.

Una ***Red Neuronal*** corresponde a un conjunto de múltiples capas formadas por un con-

junto de unidades elementales conocidas comúnmente como ***perceptrón*** o ***neurona***. Estas unidades elementales se conectan entre capas unas a otras a través de un enlace o peso (fully-connected), en donde cada perceptrón recibe información de la capa anterior que luego de una transformación, producto de una operación lineal entre la salida de la capa anterior y una matriz de peso asociado al enlace de cada neurona, más la acción de un vector bias (vector de sesgo)[3] y una función de activación, se produce la salida del perceptrón en cuestión. La salida de estas neuronas son traspasadas a la siguiente capa de neuronas y así la red va aprendiendo representaciones más complejas en cada una de las capas. La capa de salida entrega valores distintos dependiendo de si el objetivo es usar la red para clasificación binaria, o valores continuos en un modelo de regresión, o valores discretos si el objetivo es construir un modelo de clasificación.

2.5.1. Perceptrón

El perceptrón o neurona es la unidad más básica de las redes neuronales, en donde el output o salida de cada neurona se obtiene según la siguiente expresión:

$$y = f(z) \tag{2.3}$$

donde z es igual a:

$$z = w_1 \cdot x_1 + w_2 \cdot x_2 + ... + w_n \cdot x_n \tag{2.4}$$

por lo que la Ecuación 2.3 se puede reescribir como:

$$y = f(z) = f\left(\sum_{i=1}^{n} w_i \cdot x_i + b\right) \tag{2.5}$$

donde f corresponde a una función de activación, los valores de w conforman una matriz de pesos transpuesta, los valores de x conforman un vector que contiene los datos de entrada y b es un sesgo escalar. La expresión matricial de la Ecuación 2.5 se denota de la siguiente manera:

[3]El sesgo representa una neurona adicional incluida con cada capa de pre-salida y almacena un valor unitario para cada acción

$$\begin{bmatrix} w_1 & w_2 & w_3 & \cdots & w_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} + b = z \tag{2.6}$$

$$WX^T + b = z \Longrightarrow y = f(z) \tag{2.7}$$

y su interpretación gráfica se puede entender de la siguiente manera:

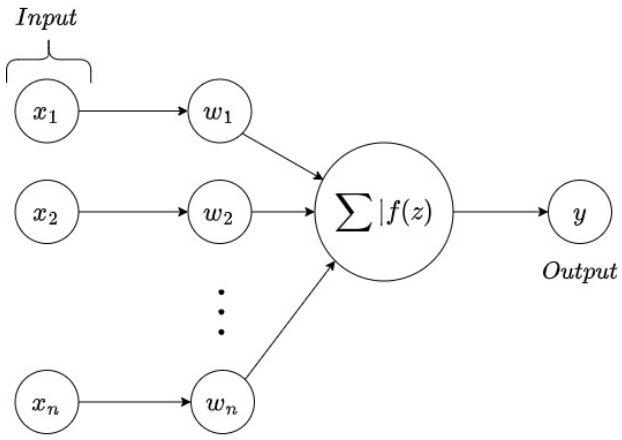


Figura 2.10: Representación gráfica de un perceptrón o neurona con n datos de entrada n pesos y una función de activación f. (Fuente: Elaboración propia a partir de "Neural Networks and Learning Machines" - Simon S. Haykin [32])

2.5.2. Modelo de Red Neuronal

Un modelo de red neuronal o ***Perceptrón multicapa*** no es más que un arreglo formado por una serie de capas que agrupan una serie de neuronas (ver Figura 2.11). Cada una de las capas entre las de entrada y salida (a excepción de éstas) se considera una capa oculta. Se define la salida y_j^1 para la primera capa oculta[4], siendo $j = 1, 2, ..., m$ el índice de cada perceptrón dentro de la capa, de la siguiente manera:

$$y_j^{[1]} = f\left(\sum_{i=1}^{n} w_{j,i} \cdot x_i + b_j^{[1]}\right) \tag{2.8}$$

[4]El superíndice $[i]$ indica el número de capa oculta i

En su representación matricial, la anterior ecuación se escribe de la forma:

$$\begin{bmatrix} w_{j,1} & w_{j,2} & w_{j,3} & \cdots & w_{j,n} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} + b_j^{[1]} = z_j^{[1]} \Longrightarrow f\left(z_j^{[1]}\right) = y_j^{[1]} \tag{2.9}$$

Con la primera capa oculta conformada por m neuronas, se pueden expresar las salidas de forma matricial de la siguiente forma:

$$\begin{bmatrix} W_1^{[1]^T} \\ W_2^{[1]^T} \\ W_3^{[1]^T} \\ \vdots \\ W_m^{[1]^T} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} b_1^{[1]} \\ b_2^{[1]} \\ b_3^{[1]} \\ \vdots \\ b_m^{[1]} \end{bmatrix} = \begin{bmatrix} z_1^{[1]} \\ z_2^{[1]} \\ z_3^{[1]} \\ \vdots \\ z_m^{[1]} \end{bmatrix} \Longrightarrow \begin{bmatrix} f\left(z_1^{[1]}\right) = y_1^{[1]} \\ f\left(z_2^{[1]}\right) = y_2^{[1]} \\ f\left(z_3^{[1]}\right) = y_3^{[1]} \\ \vdots \\ f\left(z_m^{[1]}\right) = y_m^{[1]} \end{bmatrix} \tag{2.10}$$

Para el resto de las capas ocultas se obtiene la salida y_j^k, con $k = 2,3,...,L$, según la siguiente expresión:

$$y_j^{[k]} = f\left(\sum_{i=1}^{m} w_{j,i} \cdot y_i^{[k-1]} + b_j^{[k]}\right) \tag{2.11}$$

Siendo la representación matricial de las salidas de las k capas ocultas como se muestra a continuación:

$$W^{[k]} y^{[k-1]} + b^{[k]} = z^{[k]} \Longrightarrow f\left(z^{[k]}\right) = y^{[k]} \tag{2.12}$$

Siendo:

- $W^{[k]}$: Matriz de pesos de la k-ésima capa oculta.
- $y^{[k-1]}$: Matriz de salida de las neuronas pertenecientes a la capa oculta $k-1$.
- $b^{[k]}$: Matriz de sesgos (bias) de la k-ésima capa oculta.
- $y^{[k]}$: Matriz de salida de las neuronas pertenecientes a la k-ésima capa oculta.

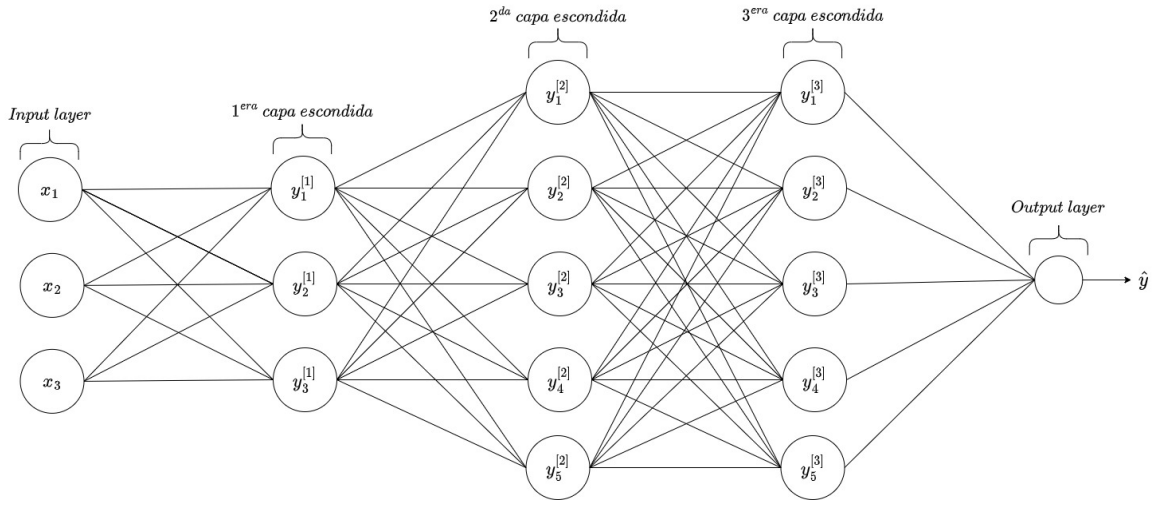


Figura 2.11: Ejemplo de representación de una red neuronal multicapa (fully-connected) con 3 entradas, 3 capas ocultas y una capa de salida. (Fuente: Elaboración propia a partir de "Neural Networks and Learning Machines" - Simon S. Haykin [32])

2.5.3. Función de Activación

En redes neuronales, la salida de una neurona generalmente pasa a través de un filtro, conocido como ***función de activación***, que limita o modifica el valor generado. Este filtro introduce no linealidad en el modelo, permitiendo que la red neuronal aprenda relaciones complejas. La función de activación también ayuda a acotar los valores de salida, reduciendo la complejidad computacional del modelo [32].

Definición 2.1. *Una función de activación f está definida por:*

$$f : \Omega \subset \mathbb{R} \rightarrow \Lambda \in \mathbb{R} \tag{2.13}$$

Esta función opera elemento a elemento sobre una salida de una capa de la red, controlando la forma de dicha salida y ajustando un umbral sobre la magnitud de la misma [28]. Un parámetro adicional, conocido como temperatura T, puede incluirse para ajustar la función: $f(a/T)$, donde a es la entrada a la función [69].

Definición 2.2. *Una función de activación f se considera saturada si:*

$$\forall z \in \mathbb{R} || \forall f(z) \in \mathbb{R} || \exists s \in \mathbb{R} || tq | lim_{n \rightarrow \infty} f(z)| \leq s \tag{2.14}$$

Ejemplos populares incluyen la función sigmoide y la tangente hiperbólica, que limitan los valores de salida en rangos específicos [32].

Función Sigmoide σ:

La función sigmoide es una de las funciones de activación más utilizadas en redes neuronales. Se caracteriza por ser estrictamente creciente, equilibrando el comportamiento lineal y no lineal [32]. Se define como:

$$f(z) = \sigma(z) = \frac{1}{1 + e^{-z}} \tag{2.15}$$

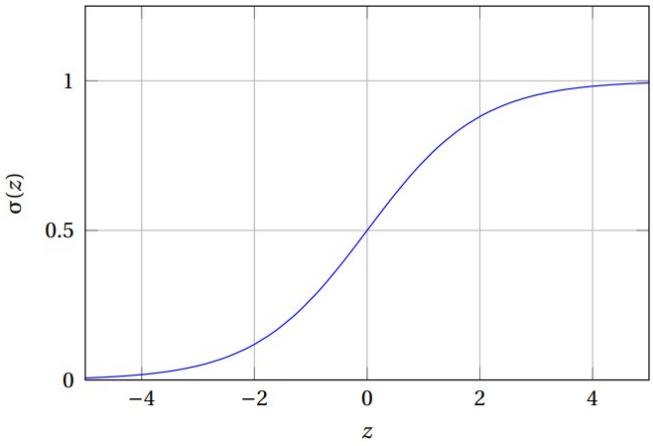


Figura 2.12: Función de activación Sigmoide σ.
(Fuente: Convolutional Recurrent Neural Networks for Remaining Useful Life prediction in mechanical systems - N. Oyharcabal A. [69])

La función sigmoide convierte las entradas a un rango entre 0 y 1. Al ajustar un parámetro de pendiente a, se pueden obtener funciones con diferentes grados de pendiente. En el límite, cuando el parámetro a tiende a infinito, la función se aproxima a una función de umbral [32]. Además, su naturaleza derivable la hace útil en problemas de clasificación binaria [28].

Esta función es especialmente útil en la capa de salida para clasificaciones binarias, donde los resultados son 0 o 1, según un umbral comúnmente fijado en 0.5 [28].

Función Tangente hiperbólica $Tanh$:

La función tangente hiperbólica es otra función comúnmente utilizada en capas ocultas de las redes neuronales. Se define como:

$$f(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} \tag{2.16}$$

A diferencia de la sigmoide, la tangente hiperbólica transforma las entradas al rango $[-1, 1]$, lo que ayuda a centrar los datos alrededor de cero, mejorando la eficiencia del aprendizaje [28].

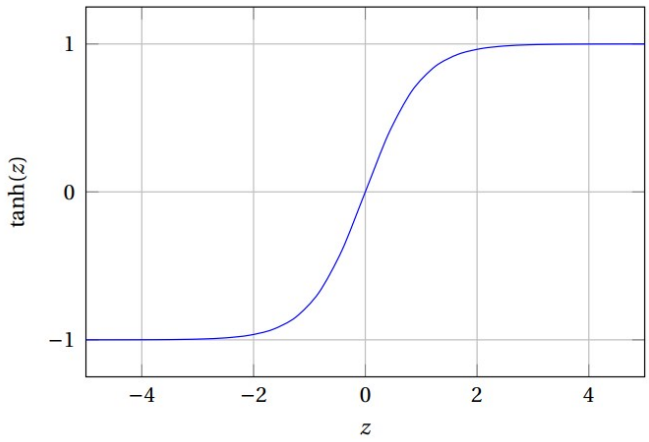


Figura 2.13: Función de activación Tanh.
(Fuente: Convolutional Recurrent Neural Networks for Remaining Useful Life prediction in mechanical systems - N. Oyharcabal A. [69])

Definición 2.3. *Una función de activación f se considera no saturada si:*

$$\forall z \in \mathbb{R} | lim_{n \to \infty} f(z)| \to +\infty \tag{2.17}$$

Ejemplos populares incluyen ReLU y ELU, que permiten que los valores de salida no estén acotados [48].

Función ReLU:

La función ReLU, definida como:

$$f(z) = \text{máx}(0, z) \tag{2.18}$$

Es ampliamente utilizada en redes neuronales profundas, dado que mejora la eficiencia del aprendizaje y evita el problema de los gradientes desaparecidos, común en funciones como la sigmoide [48].

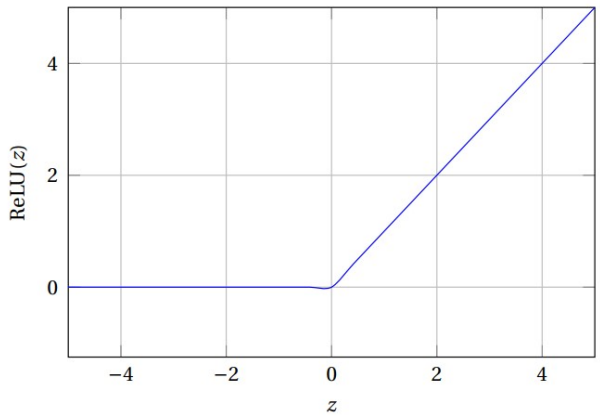


Figura 2.14: Función de activación ReLU.
(Fuente: Convolutional Recurrent Neural Networks for Remaining Useful Life prediction in mechanical systems - N. Oyharcabal A. [69])

Función ELU:

Finalmente, la función ELU es una variante de ReLU que introduce una saturación en los valores negativos. Se define como:

$$f(z) = \begin{cases} \alpha(e^z - 1) & \text{si } z < 0 \\ z & \text{si } z \geq 0 \end{cases} \tag{2.19}$$

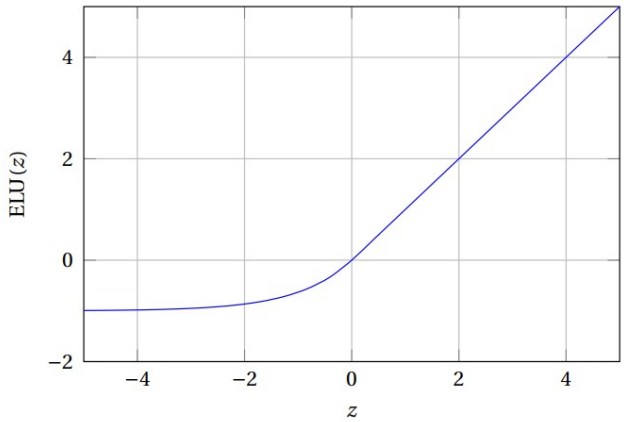


Figura 2.15: Función de activación ELU.
(Fuente: Convolutional Recurrent Neural Networks for Remaining Useful Life prediction in mechanical systems - N. Oyharcabal A. [69])

Esta función mejora la robustez de la red, especialmente en modelos profundos, al mantener activaciones negativas controladas [48].

Función Softmax:

La función ***Softmax*** es comúnmente utilizada en problemas de clasificación multiclase. Esta función comprime un vector de entradas reales en un vector de probabilidades:

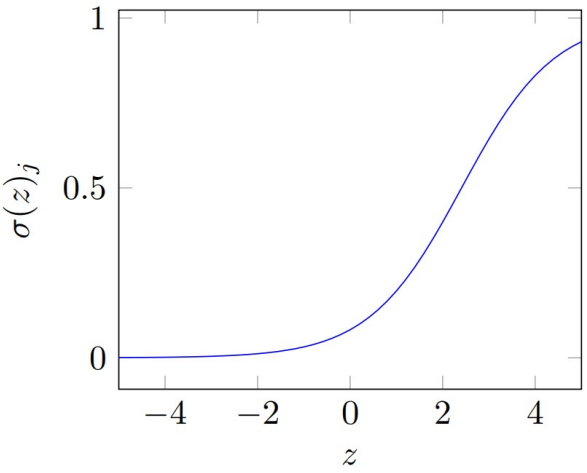


Figura 2.16: Función de activación Softmax.
(Fuente: Detección de anomalías aplicado al mantenimiento predictivo, mediante técnicas de Deep Learning no supervisado, utilizando autoencoders, Long Short-Term Memory y Self-Attention- L. F. Jeréz [41])

La función Softmax se utiliza principalmente en la capa de salida para asignar probabilidades a las diferentes clases [28].

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \quad \text{para } j = 1, \ldots, K \tag{2.20}$$

2.5.4. Función de Coste o Pérdida

La función de coste o pérdida $\mathcal{L}(\theta)$ es una técnica de evaluación del rendimiento de un modelo de redes neuronales. Esta función toma los resultados predichos por el modelo $\hat{y}$ y los resultados reales y, y calcula la magnitud del error cometido en la predicción. Es fundamental para guiar el proceso de ajuste de los parámetros del modelo, ya que los métodos de optimización buscan minimizar esta función para mejorar las predicciones [28] [11].

$$\mathcal{L} = \frac{1}{N} \sum error\left(\hat{y}^{(i)}(\theta), y^{(i)}(\theta)\right) \tag{2.21}$$

Si las predicciones difieren significativamente de los valores reales, el valor de la función de coste será mayor. A medida que se ajusta el modelo adaptando los parámetros para mejorar las predicciones, la función de coste actúa como un indicador de cuánto ha mejorado el modelo [11].

Existen diferentes tipos de función de coste para las diversas tareas de un modelo de aprendizaje automático. A continuación, se definen algunas de las más comunes:

Función de coste para tareas de regresión:

- **Error cuadrático medio:** Se define como la media de la suma de las diferencias al cuadrado entre las predicciones y las observaciones reales.

$$ECM\left(l_2 - loss\right) = \frac{\sum_{i=0}^{n}\left(y - \hat{y}\right)^2}{n} \tag{2.22}$$

- **Error medio absoluto:** Se define como la media de la suma de las diferencias absolutas entre las predicciones y las observaciones reales [31].

$$MAE\left(l_1 - loss\right) = \frac{\sum_{i=0}^{n} \mid \left(y - \hat{y}\right) \mid}{n} \tag{2.23}$$

Función de coste para tareas de clasificación:

- **Cross-Entropy Loss:** También denominada pérdida logarítmica o pérdida logística. El propósito de la entropía cruzada es tomar las probabilidades de salida $q(x)$ y medir su distancia a los valores reales. Esta función penaliza fuertemente los errores de clasificación, siendo utilizada en tareas de clasificación [28].

$$\mathcal{L}_{CE}\left(p,q\right) = \sum p(x) log\left(\frac{1}{q(x)}\right) \tag{2.24}$$

2.5.5. Teorema Universal de Aproximación

El *Teorema Universal de Aproximación* establece que una red neuronal de una sola capa oculta puede aproximar cualquier función continua en un espacio compacto, siempre que se utilicen funciones de activación adecuadas, como la sigmoide [21] [37].

Definición 2.4. *Sea F una función continua $[0,1]^k \to [0,1]$, en donde para todo $\epsilon > 0$ existe una matriz W, un vector b y una matriz U, tal que:*

$$F(x) = \sigma(xW + b)U \tag{2.25}$$

$$|f(x) - F(x)| < \epsilon \quad ,\forall x \in [0,1]^k \tag{2.26}$$

En otras palabras, sin importar cuán cerca quieras aproximarte a una función, siempre existe una matriz W, un vector b y una matriz U tal que la diferencia entre la función objetivo y la función de aproximación sea menor que el error ϵ definido. Esto muestra que las redes neuronales tienen la capacidad de aproximar funciones complejas con una estructura relativamente simple [28] [32].

2.5.6. Backpropagation

El algoritmo de Backpropagation es ampliamente utilizado para ajustar los pesos en redes neuronales, con el objetivo de minimizar el error en las predicciones del modelo y mejorar su precisión. Este método se basa en el descenso de gradiente, donde se calcula de manera eficiente el gradiente de la función de pérdida utilizando la regla de la cadena y principios de la programación dinámica, permitiendo la actualización de los pesos a lo largo de las capas de la red neuronal. Ver Ecuación 2.31. El Backpropagation fue propuesto por primera vez por Rumelhart, Hinton y Williams [83] y ha sido un pilar fundamental en el entrenamiento de redes neuronales modernas debido a su capacidad para manejar redes multicapa y grandes conjuntos de datos [28].

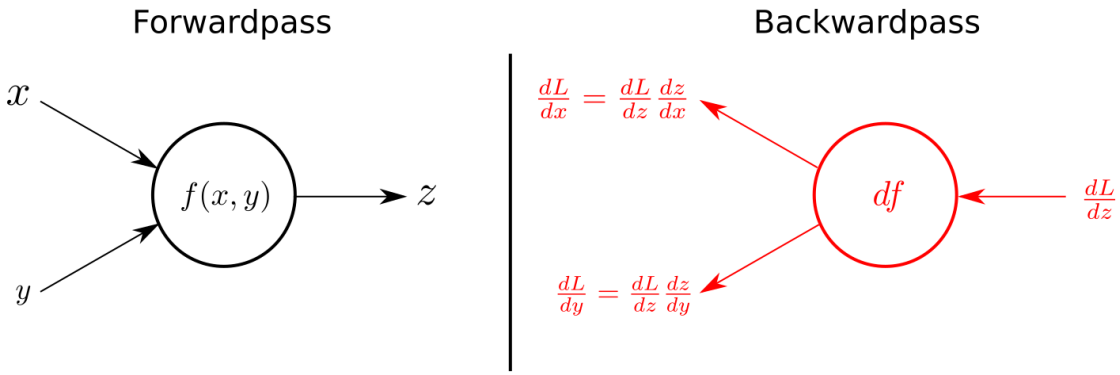


Figura 2.17: Interpretación de grafos computacionales del Backpropagation.
(Fuente: Mayank Agarwal [4])

2.5.7. Algoritmos de optimización

Descenso de gradiente

El descenso de gradiente es un algoritmo optimizador iterativo de primer orden utilizado para encontrar un mínimo local de una función diferenciable. El algoritmo toma pasos en la dirección opuesta al gradiente ∇f de la función en un punto dado [66] [68]. Si f es diferenciable a lo largo de cada eje, el vector gradiente se denota según la siguiente expresión:

$$\nabla f(x) = \left(\frac{\partial f(x)}{\partial x_1}, \cdots, \frac{\partial f(x)}{\partial x_p} \right)^T \in \mathbb{R}^p \tag{2.27}$$

Donde $\nabla f(x) : \mathbb{R}^p \to \mathbb{R}^p$ es un campo vectorial que puede interpretarse como la dirección en la que la función f crece con mayor magnitud [13]. Por otra parte, la expansión de Taylor entrega una aproximación de la función f cerca de x según la siguiente expresión:

$$f(z) = T_x(z) + o(||x - z||) \tag{2.28}$$

en donde

$$T_x(z) = f(x) + \langle \nabla f(x), z - x \rangle \tag{2.29}$$

Ahora bien, dado que el gradiente de una función apunta en la dirección de mayor crecimiento, para llegar al mínimo es necesario moverse en la dirección contraria, es decir, en la dirección de $-\nabla f(x)$ [66]. Por lo tanto, el algoritmo de descenso de gradiente es el siguiente:

$$x_{k+1} = x_k - \lambda \nabla f(x_k) \tag{2.30}$$

donde el algoritmo comienza con un valor inicial arbitrario $x_0 \in \mathbb{R}^p$, y $\lambda > 0$ corresponde a la magnitud del paso del algoritmo (también conocida como "tasa de aprendizaje"). Por lo tanto, si f es diferenciable en x, según la expansión de Taylor se obtiene la siguiente relación:

$$f(x - \lambda \nabla f(x)) = f(x) - \lambda \langle \nabla f(x), \nabla f(x) \rangle + o(\lambda) = f(x) - \lambda ||\nabla f(x)||^2 + o(\lambda) \tag{2.31}$$

Entonces, existen dos posibilidades: en el caso en que $\nabla f(x) = 0$, entonces se ha alcanzado un mínimo local, pero si λ es lo suficientemente pequeño, $f(x - \lambda \nabla f(x)) < f(x)$, lo que indica que se está acercando al mínimo local más cercano de la función a optimizar [68].

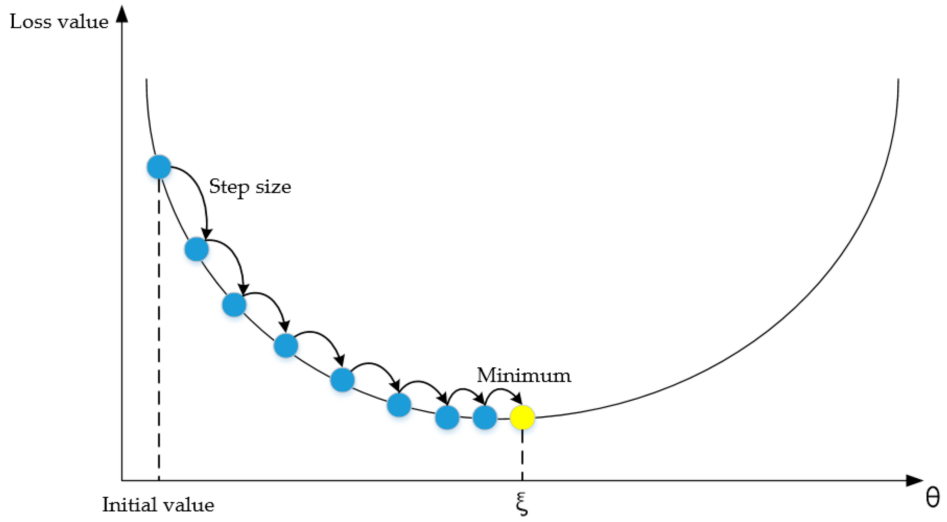


Figura 2.18: Representación gráfica del descenso de gradiente.
(Fuente: Chen et al. 2019 [16])

Por otro lado, en una red neuronal con $\hat{\theta}$ parámetros, el descenso de gradiente tiene la siguiente forma:

$$\hat{\theta}_{k+1} = \hat{\theta}_k - \lambda \frac{\partial \mathcal{L}}{\partial \theta}\bigg|_{\hat{\theta}} \tag{2.32}$$

Luego, en una red neuronal con L capas ocultas, $\hat{w}^{(i)}$ y $\hat{b}^{(i)}$ parámetros y una capa de salida con $\hat{U}$ y $\hat{C}$, el algoritmo de descenso de gradiente se actualiza de la siguiente manera [28]:

$$\begin{bmatrix} \hat{w}_{k+1}^{(i)} = \hat{w}_k^{(i)} - \lambda \frac{\partial \mathcal{L}}{\partial w^{(i)}}|_\theta \\ \hat{b}_{k+1}^{(i)} = \hat{b}_k^{(i)} - \lambda \frac{\partial \mathcal{L}}{\partial b^{(i)}}|_\theta \\ \hat{U}_{k+1} = \hat{U}_k - \lambda \frac{\partial \mathcal{L}}{\partial U}|_\theta \\ \hat{C}_{k+1} = \hat{C}_k - \lambda \frac{\partial \mathcal{L}}{\partial C}|_\theta \end{bmatrix} \Longrightarrow \nabla \mathcal{L} = \left[\frac{\partial \mathcal{L}}{\partial w^{(i)}}, \frac{\partial \mathcal{L}}{\partial b^{(i)}}, \cdots, \frac{\partial \mathcal{L}}{\partial C} \right] \tag{2.33}$$

Descenso estocástico de gradiente

El descenso estocástico de gradiente (SGD) introduce el concepto de promedio móvil exponencial a los parámetros de la red, lo que otorga mayor relevancia a los nuevos valores respecto a los antiguos [81]. Anteriormente se tenia la siguiente expresión en el descenso de gradiente $w := w - \lambda \frac{\partial \mathcal{L}}{\partial w}$[5]. Pero ahora, al integrar el promedio móvil exponencial la expresión queda de la siguiente manera:

[5]La expresión $:=$ considera una actualización de valores, mas no una igualdad.

$$w := w - \lambda \bar{\partial} w \tag{2.34}$$

$$\bar{\partial} w := \beta \bar{\partial} w + (1 - \beta) \frac{\partial \mathcal{L}}{\partial w} \tag{2.35}$$

donde, luego de un arreglo algebraico y un cambio de variable se obtiene la siguiente expresión:

$$\begin{bmatrix} V_{\partial w} := \beta V_{\partial w} - \lambda^{'} \frac{\partial \mathcal{L}}{\partial w} \\ w := w + V_{\partial w} \end{bmatrix} \tag{2.36}$$

donde β sería el parámetro de "momentum" (o parámetro de la media móvil)[73] y $\lambda^{'}$ la taza de aprendizaje. Siendo el momentum una capacidad del modelo para seguir con inercia en cada paso, es decir, actualizar el parámetro según el paso anterior.

AdaGrad

AdaGrad se considera como un algoritmo adaptativo, ya que cuanta con la capacidad de adaptar individualmente los parámetros del modelo, escalándolos de forma inversamente proporcional a la raíz cuadrada de la suma histórica de los gradientes [23]. Es decir, los parámetros con elevados gradientes de la función de perdida, representan un decrecimiento rápido en la taza de aprendizaje, y por el contrario, a valores pequeños de gradientes de la función de perdida, la taza de aprendizaje decrece de forma más lenta.

$$\begin{bmatrix} V_{\partial w} := V_{\partial w} + \frac{\partial \mathcal{L}}{\partial w} \cdot \frac{\partial \mathcal{L}}{\partial w} \\ w := w - \frac{\lambda}{\sqrt{V_{\partial w}}} \cdot \frac{\partial \mathcal{L}}{\partial w} \end{bmatrix} \tag{2.37}$$

Este algoritmo tiene propiedades interesantes para funciones convexas, sin embargo, en la mayoría de los casos de deep learning el modelo hace que al comienzo del entrenamiento el gradiente cuadrado acumulado crezca rápidamente y la taza de aprendizaje disminuya de forma excesiva, corriendo el riesgo de quedarse en mínimos locales o no converger.

RMS prop

RMSprop busca solucionar el problema de la rápida disminución de la tasa de aprendizaje en AdaGrad, utilizando un promedio móvil exponencial de los gradientes al cuadrado [34]:

$$\begin{bmatrix} S_{\partial w} := \beta S_{\partial w} + (1 - \beta) \frac{\partial \mathcal{L}}{\partial w} \cdot \frac{\partial \mathcal{L}}{\partial w} \\ w := w - \frac{\lambda}{\sqrt{S_{\partial w}}} \cdot \frac{\partial \mathcal{L}}{\partial w} \end{bmatrix} \tag{2.38}$$

Adam

Este algoritmo sigue siendo un algoritmo adaptativo de la taza de aprendizaje. Adam se considera una variante de una combinación entre RMS prop y Momentum con importantes distinciones. Una de las claves de Adam es que utiliza el promedio móvil exponencial para obtener una estimación tanto del primero como del segundo momento del gradiente [43], es decir, utiliza las variables de estado:

$$P_{\partial w} := \beta_1 P_{\partial w} + (1 - \beta_1)\frac{\partial \mathcal{L}}{\partial w} \tag{2.39}$$

$$S_{\partial w} := \beta_2 S_{\partial w} + (1 - \beta_2)\frac{\partial \mathcal{L}}{\partial w} \cdot \frac{\partial \mathcal{L}}{\partial w} \tag{2.40}$$

Donde β_1, $\beta_2 \in [0, 1]$, comúnmente se utilizan $\beta_1 = 0{,}9$ y $\beta_2 = 0{,}999$. Además, hay que observar que al inicio del entrenamiento se tiene que $P_0 = S_0 = 0$, es decir, una cantidad significativa de sesgo inicialmente hacia los valores más pequeños, por esta razón se normalizan estos estados según:

$$\bar{P_{\partial w}} = \frac{P_{\partial w}}{1 - \beta_1^t} \qquad \text{y} \qquad \bar{S_{\partial w}} = \frac{S_{\partial w}}{1 - \beta_2^t} \tag{2.41}$$

Dado que $\beta \in [0, 1]$, la variable β^t solo importa al inicio, ya que luego al estar elevado a t se irá rápidamente a cero. Finalmente el algoritmo de optimización de Adam converge a la siguiente expresión:

$$w := w - \frac{\lambda}{\sqrt{\bar{S_{\partial w}}}}\bar{P_{\partial w}} \tag{2.42}$$

2.6. Redes Neuronales Convolucionales (CNNs)

Las ***redes neuronales convolucionales*** **(CNNs)** son análogas a las redes neuronales artificiales tradicionales, ya que están compuestas por neuronas que se auto-optimizan mediante el aprendizaje. Sin embargo, las CNNs están diseñadas específicamente para trabajar con datos que tienen una estructura en forma de grilla, como las imágenes, lo que las hace especialmente útiles en tareas de reconocimiento de patrones visuales [49] [45], lo que las hace particularmente útiles para tareas de visión por computadora debido a su capacidad para aprender características jerárquicas a partir de datos de imagen, capturando desde bordes simples en capas iniciales hasta estructuras más complejas en capas más profundas.

Una CNN suele tomar como entrada un tensor tridimensional, como una imagen con H filas, W columnas y 3 canales de color (RGB). Sin embargo, las CNNs también pueden procesar tensores de orden superior. La transformación de los datos en una CNN ocurre mediante la convolución entre un filtro o kernel y un subconjunto del tensor de entrada. Cada paso de procesamiento en una CNN se denomina "capa". La operación de convolución entre una imagen I con dimensiones H (altura) y W (ancho), y un kernel K, se define como:

$$C = (I \cdot K)(i,j) = \sum_{m=1}^{H} \sum_{n=1}^{W} I(i+n, j+m) K(n,m) \tag{2.43}$$

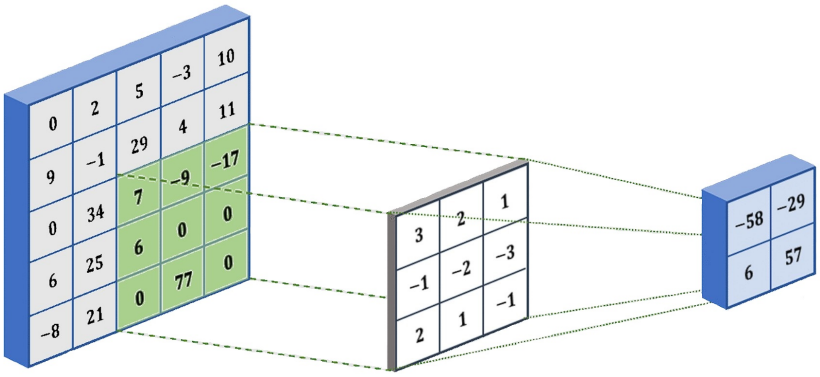


Figura 2.19: Representación gráfica del proceso de convolución entre una matriz de entrada (imágen) y un kernel.
(Fuente: Playing Super Mario Bros with Proximal Policy Optimization - Brandon Da Silva [90])

Esta operación genera una salida de menor dimensión en comparación con la entrada (ver Figura 2.19), lo cual se denomina ***valid padding***. Para preservar la dimensión de la entrada, se puede usar la técnica de ***zero padding***, que añade ceros en los bordes de la imagen [24].

La técnica de ***zero padding*** consiste en rellenar el borde de la entrada con ceros, lo que permite controlar mejor la dimensionalidad de la salida. Esto es particularmente útil cuando se desean conservar las dimensiones espaciales después de las operaciones de convolución. Para calcular la dimensión de salida en función del tamaño de la entrada, el tamaño del kernel, el padding y el paso de la convolución, se puede utilizar la siguiente fórmula [28]:

$$\frac{(V-R)+2Z}{S}+1 \tag{2.44}$$

Donde V representa el tamaño del volumen de entrada (altura × ancho × profundidad), R representa el tamaño del campo receptivo, Z es la cantidad de *zero padding*, y S es el

valor del stride (o paso). Si el resultado no es un número entero, entonces el stride ha sido incorrectamente configurado, lo que impide que las neuronas encajen correctamente sobre la entrada dada.

2.6.1. Pooling Layers

Además de las capas convolucionales, las CNNs suelen incluir capas de agrupación, conocidas como ***pooling layers***, cuyo objetivo principal es reducir progresivamente la dimensionalidad espacial de la representación aprendida, disminuyendo el número de parámetros y la complejidad computacional del modelo. Las capas de agrupación también ayudan a controlar el sobreajuste y hacen las representaciones más invariantes a pequeñas transformaciones en los datos de entrada [28]. Los dos tipos más comunes de ***pooling layers*** son:

- **Max-pooling:** Este tipo de capa de agrupación calcula el valor máximo en un parche (sub-muestra) de un mapa de características y lo usa para reducir la dimensionalidad de salida. Max-pooling es eficaz para extraer características dominantes dentro de una sub-región espacial [86].
- **Average-pooling:** A diferencia de Max-pooling, Average-pooling calcula el promedio de los valores dentro de un parche, generando una salida suavizada. Este tipo de agrupación proporciona una pequeña cantidad de invariancia a la traducción, ya que los pequeños cambios en la ubicación de los píxeles no afectan significativamente el resultado agrupado [12].

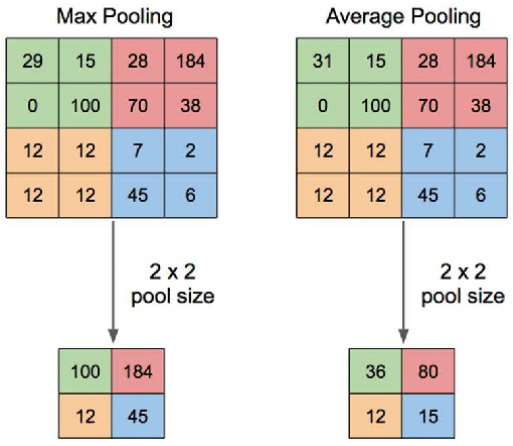


Figura 2.20: Representación gráfica de los procesos de Max-pooling y Average-pooling
(Fuente: Application of Transfer Learning Using Convolutional Neural Network Method for Early Detection of Terry's Nail - Y. Muhamad [107])

2.7. Redes Neuronales Recurrentes (RNNs)

Las redes neuronales recurrentes **(RNNs)** son una arquitectura especializada en el modelado de datos secuenciales, introduciendo la capacidad de mantener una "memoria"de estados anteriores [70] [25]. Estas redes son particularmente efectivas para tareas que implican secuencias temporales, como series de tiempo, lenguaje natural y procesamiento de audio [87].

Las RNNs procesan una secuencia de vectores de entrada $x = (x_1, x_2, x_3, \cdots, x_n)$, y en cada paso de tiempo t, producen una capa oculta $h^{[t]}$, que es utilizada para hacer una predicción $\hat{y} = (\hat{y}_1, \hat{y}_2, \hat{y}_3, \cdots, \hat{y}_n)$. La principal característica de las RNNs es la presencia de una conexión de retroalimentación, que permite que el estado oculto $h^{[t]}$ dependa tanto de la entrada actual $x^{[t]}$ como del estado oculto anterior $h^{[t-1]}$, estableciendo una recurrencia [28]:

$$h^{[t]} = f\left(x^{[t]}U + h^{[t-1]}W\right) \tag{2.45}$$

De esta forma, el estado oculto $h^{[t]}$ puede interpretarse como un resumen de la secuencia de entradas hasta el paso de tiempo t [56].

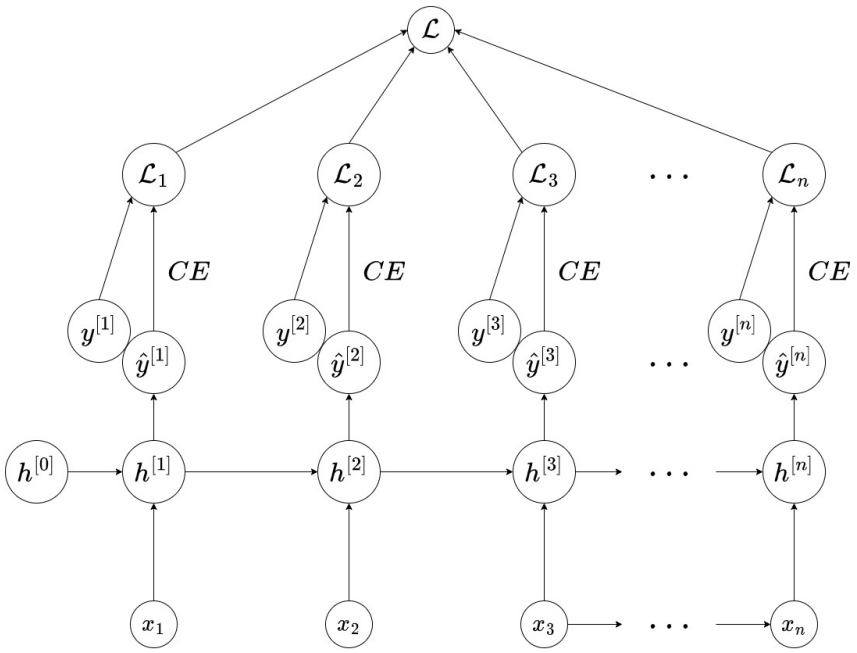


Figura 2.21: Representación gráfica de una Fully Recurrent Neural Network.
(Fuente: Detección de anomalías aplicado al mantenimiento predictivo, mediante técnicas de Deep Learning no supervisado, utilizando autoencoders, Long Short-Term Memory y Self-Attention - Luis Felipe Jeréz [41])

Las ecuaciones que gobiernan una RNN (ver Figura 2.21) son las siguientes:

$$\begin{bmatrix} h^{[t]} = \tanh\left(x^{[t]}U + h^{[t-1]}W + b\right) \\ \hat{y}^{[t]} = Softmax\left(h^{[t]}O + c\right) \\ \mathcal{L}^{[t]} = \mathcal{L}\left(\hat{y}^{[t]}, y^{[t]}\right) \\ \mathcal{L}\left(\hat{y}^{[t]}, y^{[t]}\right) = \frac{1}{N}\sum_{t=1}^{n}\mathcal{L}^{[t]} \end{bmatrix} \tag{2.46}$$

El modelo se entrena mediante el algoritmo de "Backpropagation Through Time" (BPTT), que extiende el algoritmo de backpropagation para secuencias. El gradiente de la función de pérdida $\mathcal{L}$ con respecto a los parámetros θ se calcula acumulando los gradientes a lo largo del tiempo:

$$\frac{\partial \mathcal{L}^T}{\partial \theta} = \sum_{t=1}^{T} \frac{\partial \mathcal{L}^T}{\partial \theta}\bigg|_{(t)} \tag{2.47}$$

A pesar de su eficacia, las RNNs presentan el problema del desvanecimiento del gradiente, que ocurre cuando los gradientes retropropagados disminuyen exponencialmente a medida que se retrocede a través de la secuencia. Este problema se debe al uso de funciones de activación como $\tanh$ o $sigmoid$, lo que dificulta que el modelo capture dependencias a largo plazo [10]. En consecuencia, cuando los gradientes tienden a cero, la red deja de aprender, afectando su capacidad para modelar secuencias largas. Para mitigar este problema, se han propuesto arquitecturas avanzadas como ***Long Short-Term Memory (LSTM)*** y ***Gated Recurrent Unit (GRU)*** [35] [18].

2.7.1. Long Short-Term Memory (LSTM)

introduce una celda de memoria que permite almacenar información a lo largo de secuencias extendidas, controlada por tres compuertas: la compuerta de entrada $i^{[t]}$, la compuerta de olvido $f^{[t]}$, y la compuerta de salida $o^{[t]}$. Estas compuertas determinan cuándo actualizar, olvidar o utilizar la información almacenada en la celda de memoria, lo que ayuda a mitigar el problema del desvanecimiento del gradiente(ver Figura 2.22), ya que la introducción de las compuertas en las LSTM permite que la red aprenda cuándo retener información de entradas anteriores y cuándo actualizar la celda de memoria, lo que mejora considerablemente el rendimiento en tareas con secuencias largas [26]. Las ecuaciones[6] que gobiernan la arquitectura

[6]La operación $\cdot$ corresponde al producto tensorial punto a punto.

LSTM de la Figura 2.22 son las siguientes:

$$\begin{bmatrix} o^{[t]} = \sigma\left(x^{[t]}U^o + h^{[t-1]}W^o + b_o\right) \\ i^{[t]} = \sigma\left(x^{[t]}U^i + h^{[t-1]}W^i + b_i\right) \\ f^{[t]} = \sigma\left(x^{[t]}U^f + h^{[t-1]}W^f + b_f\right) \\ \hat{c}^{[t]} = \tanh\left(x^{[t]}U + h^{[t-1]}W + b_c\right) \\ c^{[t]} = i^{[t]} \cdot \hat{c}^{[t]} + f^{[t]} \cdot c^{[t-1]} \\ h^{[t]} = o^{[t]} \cdot \tanh(c^{[t]}) \end{bmatrix} \tag{2.48}$$

Donde:

- $x^{[t]} \in \mathbb{R}^d$: Vector de entrada en el tiempo t.
- $f^{[t]} \in (0,1)^h$: Activación de la compuerta *forget*.
- $i^{[t]} \in (0,1)^h$: Activación de la compuerta *input*.
- $o^{[t]} \in (0,1)^h$: Activación de la compuerta *output*.
- $h^{[t]} \in (-1,1)^h$: Vector de salida de la capa oculta en el tiempo t.
- $\hat{c}^{[t]} \in (-1,1)^h$: Estado de la celda de memoria.
- $c^{[t]} \in \mathbb{R}^h$: Estado de la celda de memoria en el tiempo t.
- $W \in \mathbb{R}^{h \times d}, U \in \mathbb{R}^{h \times h}$ y $b \in \mathbb{R}^h$: Matrices de pesos y vectores de sesgo.

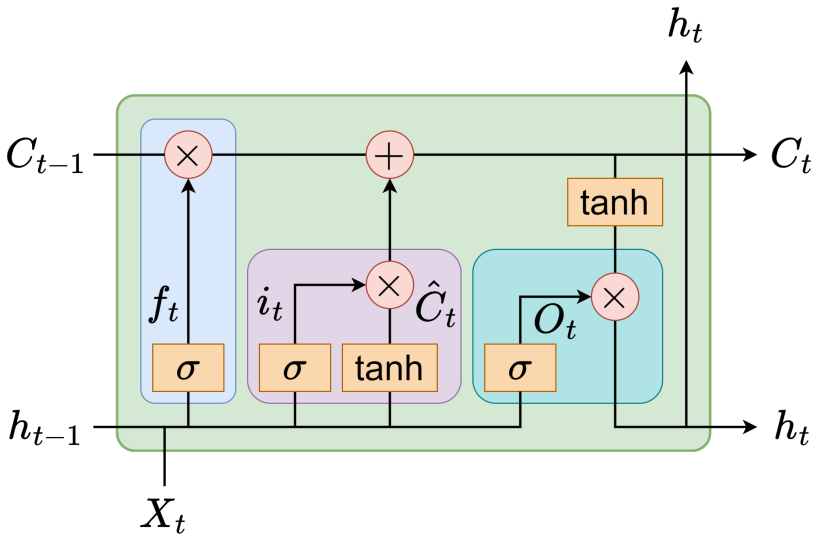


Figura 2.22: Representación gráfica de una actualización de la capa oculta $h^{[t]}$ en una arquitectura LSTM.
(Fuente: Elaboración propia a partir de: A Directed Acyclic Graph Network Combined with CNN and LSTM for Remaining Useful Life Prediction - Li, Jialin Xueyi, Li [53])

Capítulo 3

Estado del Arte

En las últimas décadas, el panorama industrial ha experimentado transformaciones significativas debido a la convergencia de diversas tecnologías emergentes como la nanotecnología, la biotecnología, la producción digital avanzada, los avances en monitoreo y sensorización, entre otros. El fenómeno anterior es conocido como la Cuarta Revolución Industrial, término que fue propuesto por primera vez en 2016 por Klaus Schwab [88]. Desde entonces, la Cuarta Revolución Industrial ha ido impulsado una integración paulatina, pero cada vez más profunda, de los dominios biológico, físico y digital, creando un entorno industrial más sofisticado e interconectado [47], lo que a su vez ha ido generando nuevos paradigmas en los entornos de producción. En este contexto, el mantenimiento predictivo ha surgido como una aplicación crítica en la gestión de activos, ya que uno de los objetivos de la Cuarta Revolución Industrial es permitir que las máquinas actúen de manera autónoma en base a los datos y la información recolectada de sus condiciones operacionales, manteniendo así su rendimiento óptimo a través de la auto-corrección de las condiciones que puedan resultar en fallos. En este contexto, la explicabilidad y el análisis de causa raíz de las fallas son componentes clave. Si bien este objetivo aún no ha sido alcanzado, existen avances significativos en el campo de la gestión de la salud de los activos, principalmente en la estimación de la Vida Útil Remanente (RUL) y en la detección de anomalías utilizando métodos y modelos de basados en la inteligencia artificial.

3.1. Enfoques en la estimación del RUL

Los métodos de regresión basados en inteligencia artificial han emergido como herramientas poderosa para la estimación del RUL en el contexto del mantenimiento predictivo. Estos métodos operan bajo dos enfoques principales: la estimación directa del RUL a partir de datos crudos de sensores, y la predicción indirecta basada en el índice de salud del sistema (HI). Por su parte, el RUL se define como el tiempo restante hasta que ocurra un fallo en el sistema, mientras que el HI es una medida intermedia que cuantifica el estado general de salud del sistema. Ahora bien, aunque diversos algoritmos se emplean para estimar el RUL a partir de señales de entrada, la mayoría de estos modelos se basan en la premisa de que existen múltiples estados de degradación previos al fallo. Esta suposición implica que la degradación del sistema es un proceso gradual que puede atravesar varias fases antes de que se comprometa significativamente la funcionalidad general del sistema [75]. En sistemas complejos, la relación entre el RUL y el estado de salud del sistema rara vez es lineal, por lo que asumir una correlación lineal podría subestimar la complejidad inherente del sistema y llevar a predicciones inexactas. Por lo tanto, además de las condiciones operativas capturadas por los sensores, diversos factores externos como las condiciones ambientales, la carga de trabajo y la edad del equipo pueden influir significativamente en la degradación y el eventual fallo del sistema.

Es importante destacar que, si bien muchos estudios en la estimación del RUL se centran en activos específicos y modos de falla particulares, el objetivo general de estos trabajos es establecer marcos de trabajo y validar configuraciones que puedan aplicarse de manera más amplia en la estimación del RUL, por lo que están diseñados con la intención de ser adaptables y generalizables a diversos escenarios en el contexto mantenimiento predictivo.

3.1.1. Redes Neuronales Convolucionales (CNN)

Uno de los tipos de algoritmos de Machine Learning (ML) más frecuentes en el campo del pronóstico y la gestión de salud de los activos son las Redes Neuronales Convolucionales (CNN). Las CNN se utilizan principalmente en el procesamiento de imágenes por su capacidad de reconocimiento de patrones, pero estas pueden adaptarse fácilmente para tratar diferentes tipos de datos y estructuras, expandiendo su capacidades para el reconocimiento de patrones a muchos problemas diferentes, como por ejemplo, en la predicción del RUL [54].

Por su parte, las Redes Neuronales Convolucionales Profundas (Deep CNN) han emergido como una herramienta prometedora para la predicción del RUL, ya que a diferencia de las CNN normales, las Deep CNN buscan predecir la densidad de probabilidad del RUL, en lugar de una estimación puntual única [115]. Este enfoque permite generar intervalos de confianza que se ajustan con el tiempo, ofreciendo rangos de posibles fallos en cada punto temporal.

Sin embargo, las CNN presentan ciertas limitaciones, principalmente debido a su naturaleza de "caja negra", la cual dificulta la interpretación de los procesos internos durante la ejecución [36]. Además, requieren un ajuste meticuloso de hiperparámetros para cada conjunto de entrenamiento, lo que puede resultar en problemas de sobreajuste o subajuste más pronunciados que en otros tipos de redes o arquitecturas [76]. Para abordar algunos de estos inconvenientes, Wang et al. propuso una Red Convolucional Temporal (TCN) que utiliza dos métodos de convolución: causales y dilatadas [101]. Esta combinación mejora la captura de dependencias locales y la precisión general. Por su parte, Zhou et al. destaca que alternar entre estos tipos de convoluciones previene filtraciones de información futura a pasos anteriores en la secuencia [116], permitiendo acceder a gran parte de las secuencias de la TCN sin comprometer la integridad temporal de los datos.

Aunque los estudios mencionados se centran en aplicaciones específicas, la flexibilidad de las CNN permite su aplicación a diferentes tipos de datos y modos de falla, siempre que se realicen los ajustes necesarios en la estructura de las redes y el preprocesamiento de datos.

3.1.2. Redes de Memoria a Corto y Largo Plazo (LSTM)

Las redes neuronales recurrentes de Memoria a Corto y Largo Plazo (LSTM) han emergido como una arquitectura predominante y versátil. Numerosos estudios han explorado y refinado el uso de LSTM para este propósito, demostrando su eficacia en diversos conjuntos de datos [100] [102]. La investigación en este ámbito ha abarcado múltiples aspectos de optimización y mejora, desde el análisis del impacto de diferentes parámetros en el rendimiento de las redes LSTM [46], hasta la propuesta de la incorporación de simulaciones de Monte Carlo para proporcionar predicciones probabilísticas de RUL [113].

Por otro lado, la fusión de diferentes capas LSTM se ha convertido en una práctica común [55], con propuestas que van desde la extensión de datos de entrada a dimensiones superiores [52], hasta la creación de capas de fusión LSTM que aprenden de redes entrenadas con

diferentes tamaños de ventana temporal [112]. Enfoques más sofisticados han introducido unidades de división de nivel para evaluar la importancia de los datos de entrada [104] y han explorado la amplificación de pesos en redes LSTM [105].

Si bien estos trabajos abordan casos particulares, es importante resaltar que las arquitecturas LSTM presentadas son inherentemente adaptables. Los principios y estructuras descritos pueden aplicarse a una amplia gama de problemas de estimación del RUL, ajustándose según las características específicas de los datos y los modos de falla en cuestión.

3.1.3. Redes Híbridas

Últimamente, las arquitecturas híbridas que combinan diferentes tipos de redes neuronales han emergido como una solución prometedora para capturar tanto las características espaciales como temporales de los datos de degradación. Estas redes híbridas pueden integrar diversos tipos de capas en su estructura. Por ejemplo, se puede utilizar una red para preprocesar datos antes de pasarlos a otros procesos o redes [17]. Un enfoque particularmente popular consiste en combinar redes CNN con redes recurrentes como GRU o LSTM, lo que permite aprovechar la capacidad de extracción de características de las CNN y el modelado temporal de las redes recurrentes [33] [108]. Estas arquitecturas híbridas colaborativas pueden mejorar significativamente el rendimiento en tareas como la predicción del RUL [33] [5], ya que este tipo de problemas abordan esencialmente series de tiempo predominantemente no lineales.

En el trabajo de Niu et al. [67], se propone una arquitectura híbrida para predecir el RUL de herramientas de mecanizado. La red empleada es una combinación de capas CNN y LSTM, diseñada para aprovechar las capacidades de extracción de características locales de las CNN y la capacidad de las LSTM para manejar dependencias a largo plazo en datos de series temporales. Según los autores, los resultados del estudio indican que su red CNN-LSTM no solo mejora la precisión de las predicciones, sino que también muestra una estabilidad considerable en las evaluaciones realizadas. En el trabajo, además, se realizan comparaciones con otras metodologías, incluyendo CNN y LSTM individuales, donde sugieren que, bajo ciertas condiciones, los enfoques en arquitecturas híbridas son superiores en términos de precisión y robustez, destacando su viabilidad para aplicaciones industriales complejas.

De manera similar, en el trabajo de Zhang et al. [111], los autores presentan una metodología para mejorar la precisión y eficiencia en la predicción del RUL de herramientas de corte al combinar el exponente de Hurst para la partición de señales con una arquitectura híbrida CNN-LSTM para el análisis de datos multivariados recopilados durante el ciclo de vida de una herramienta de corte. La arquitectura híbrida CNN-LSTM propuesta es secuencial, comenzando con sub-modelos CNN dispuestos en paralelo para la extracción de características de cada tipo de señal recogida de manera individual. Estas características se fusionan en una capa de concatenación y luego se envían a la capa LSTM para la predicción final. Esta estructura secuencial aprovecha la capacidad de las CNN para extraer características espaciales y la habilidad de las LSTM para modelar dependencias temporales a largo plazo en las señales.

Por su parte, Jafari et al. [38] introducen una arquitectura híbrida que integra de manera fluida las capacidades de las redes CNN y las unidades recurrentes GRU para mejorar la precisión en la predicción del RUL de baterías de ion litio (LIBs). El modelo propuesto utiliza ramas paralelas de CNN para procesar individualmente cada característica, incluyendo voltaje, corriente, temperatura y capacidad, permitiendo una extracción más eficiente de información relevante de los perfiles de carga multicanal. Posteriormente, estas características se integran de manera efectiva en una capa GRU, que se especializa en modelar dependencias temporales, lo que, según los autores, juega un papel crucial en la predicción precisa del RUL, adaptándose dinámicamente a las variaciones en la capacidad de las baterías a lo largo del tiempo.

Por otro lado, Li et al. [53] propusieron una arquitectura de Red de Grafo Acíclico Dirigido (DAG) que combina las capacidades de las redes CNN y las redes LSTM en una estructura de Grafo Acíclico Dirigido (DAG). Este enfoque representa un avance significativo en comparación con los modelos híbridos anteriores, que típicamente combinaban CNN y LSTM de manera secuencial, utilizando la CNN principalmente como extractor de características para la LSTM. La novedad clave de esta propuesta radica en su estructura paralela, donde las ramas CNN y LSTM operan de forma independiente pero complementaria dentro de la red DAG. Esta configuración supera una limitación crítica de los modelos secuenciales previos, permitiendo que ambas ramas se ajusten mutuamente durante el entrenamiento en respuesta al error de predicción global. Tal diseño facilita una integración más orgánica de las capacidades de extracción de características espaciales de la CNN y el modelado de dependencias temporales de la LSTM, aprovechando así las ventajas de cada tipo de red por separado.

Además de su arquitectura innovadora, el método propuesto introduce mejoras en el preprocesamiento de datos, empleando una técnica de ventana deslizante para generar secuencias a corto plazo. Este enfoque aborda hábilmente el desafío de la variabilidad en la longitud de las secuencias de entrada, un problema común en los conjuntos de datos de degradación. La implementación de una función RUL piece-wise, en lugar de la función lineal tradicional, demuestra una consideración cuidadosa del mecanismo de degradación subyacente, potencialmente mejorando la precisión del modelo en diferentes etapas del ciclo de vida del equipo.

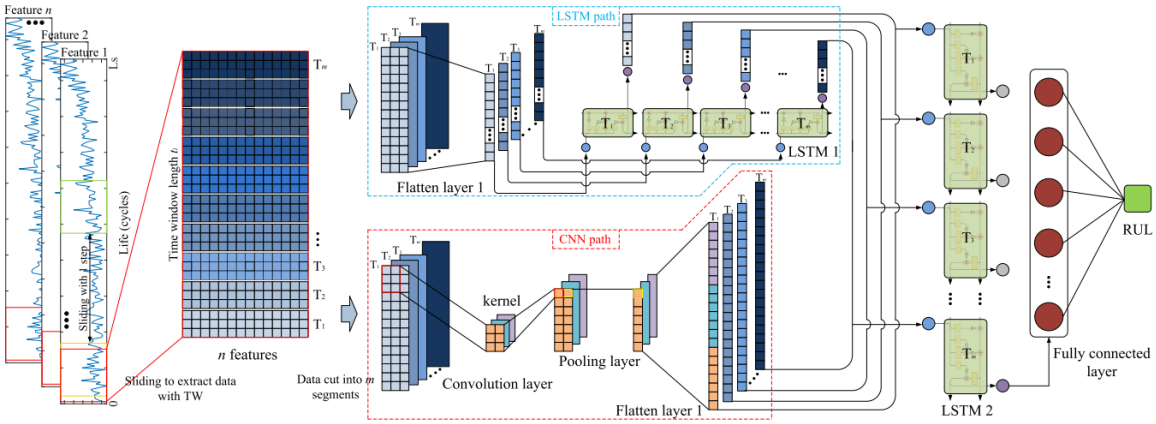


Figura 3.1: DAG Framework.
(Fuente: *A Directed Acyclic Graph Network Combined With CNN and LSTM for Remaining Useful Life Prediction* - Jianlin Li, Xueyi Li, and David He)

Es importante destacar que, aunque los ejemplos citados en esta revisión corresponden a casos específicos con datos y modos de falla particulares, los enfoques y arquitecturas presentados para estimar el RUL son, en esencia, generales y adaptables. La versatilidad de estas técnicas de aprendizaje automático permite que las arquitecturas diseñadas por los autores citados puedan aplicarse a diversos conjuntos de datos y escenarios de falla. Por ejemplo, la arquitectura CNN-LSTM propuesta por Niu et al. [67] para herramientas de mecanizado, o el modelo híbrido CNN-GRU de Jafari et al. [38] para baterías de ion litio, ilustran cómo estas estructuras pueden aplicarse a diferentes contextos. Sin embargo, la fortaleza de estos enfoques radica en su capacidad de generalización: un investigador o ingeniero podría tomar estas arquitecturas y aplicarlas a sus propios datos, independientemente del tipo específico de equipo o modo de falla que esté estudiando.

Capítulo 4

Metodología

Para cumplir con los objetivos propuestos para este trabajo de titulación, se establece la siguiente estructura lógica para la metodología: 1) Modelado de la Bomba tríplex; 2) Generación de Datos Sintéticos Simulados; 3) Pre-procesamiento de los Datos Sintéticos Simulados; 4) División de Datos y Definición de Escenarios Experimentales; 5) Desarrollo de los Modelos Predictivos; 6) Entrenamiento de los Modelos Predictivos; y 7) Evaluación del Rendimiento de los Modelos Predictivos.

4.1. Modelado de la Bomba tríplex

En esta sección se define el modelo simulado de una bomba tríplex, detallando las ecuaciones que describen su dinámica. El modelo se estructura considerando la interacción de tres bloques principales, como se ilustra en la Figura 4.1: 1) Bloque Motor; 2) Bloque Bomba; y 3) Bloque Carga/Descarga.

Este enfoque modular permite una representación detallada de cada componente y facilita el análisis de las interacciones entre los distintos elementos del sistema. A continuación, se describirán las ecuaciones y parámetros que gobiernan el comportamiento de cada bloque, proporcionando una base sólida para la simulación y el posterior análisis de degradación y falla del equipo.

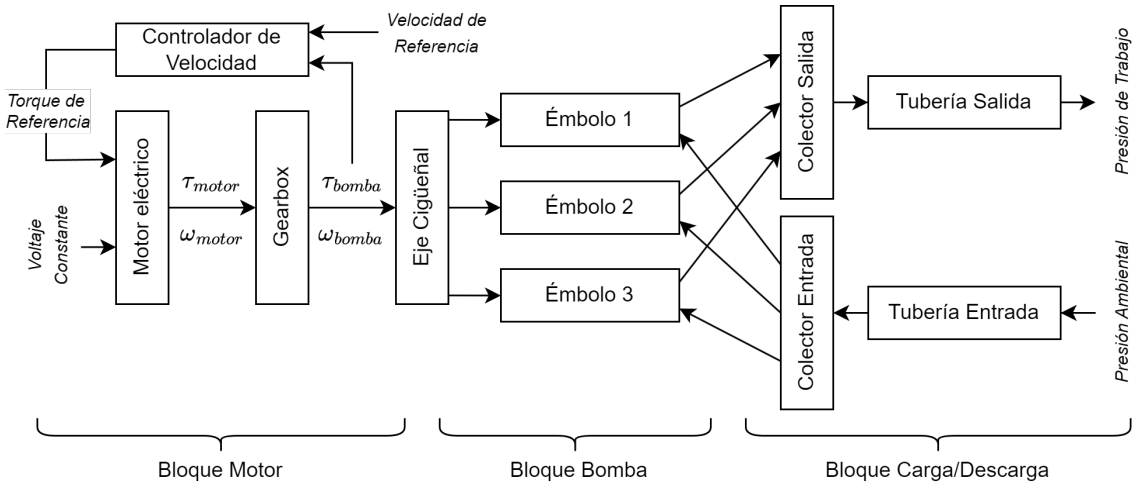


Figura 4.1: Esquema que representa los distintos módulos y conexiones presentes en el modelo simulado de la bomba tríplex.

(Fuente: Elaboración Propia.)

4.1.1. Definición del Sistema

El modelo simulado de este trabajo toma como referencia el modelo simulado del sistema de una bomba de desplazamiento positivo reciprocante de tres émbolos CAT 1051, desarrollado por Miller et al. [62] en el entorno Simulink de MATLAB. Este modelo incluye los efectos de componentes defectuosos al simular el comportamiento degradado de la bomba debido al desgaste de los rodamientos del eje cigüeñal, a la fuga de fluido en los cilindros y al bloqueo en las válvulas de entrada. Las fallas simuladas en dicho modelo se definen únicamente como parámetros de estado que no dependen del tiempo, por lo que no considera la propagación de la degradación en los componentes del sistema.

La ausencia de un modelo de propagación de la degradación implica que los datos generados a partir del modelo simulado del activo no contienen trayectorias de degradación hasta el fallo, lo que representa un impedimento, ya que el desarrollo de modelos predictivos requiere la disponibilidad de conjuntos de datos con trayectorias de operación hasta el fallo, las cuales deben estar compuestas por conjuntos de señales correspondientes al monitoreo de la condición de los sistemas, ordenados en series de tiempo, junto con las etiquetas de tiempo hasta el fallo correspondientes a cada muestra [7]. Por lo tanto, para generar trayectorias de degradación hasta el fallo, es necesario modificar el modelo referencial, de modo que este disponga de un modelo de degradación adecuado.

Los parámetros hidráulicos, mecánicos y eléctricos de la bomba tríplex se definen al simular las interacciones dinámicas entre los distintos bloques del sistema. La interacción entre el subsistema eléctrico, compuesto por el motor eléctrico y su correspondiente controlador de velocidad, y el subsistema hidráulico, compuesto por el eje cigüeñal, la interacción mecánico-traslacional entre cilindros y émbolos, y el funcionamiento de las válvulas de retención que controlan el flujo hacia los colectores de entrada y salida, se puede apreciar en el diagrama de la Figura 4.1.

Bloque Motor

El bloque del motor representa a un motor eléctrico genérico sin escobillas de Corriente Alterna (DC) con control de torque en bucle cerrado. El bloque abstrae el comportamiento combinado del motor, la electrónica de accionamiento y el control tanto del torque como de la velocidad angular del rotor al considerar un controlador proporcional-integral (PI) encargado de ajustar el nivel del torque generado por el motor eléctrico para alcanzar las revoluciones requeridas. Las pérdidas resistivas producidas debido a la tendencia de los devanados del inducido a resistirse al flujo de electrones se modelan en función de la demanda de torque según la siguiente expresión:

$$P_{losses} = k \cdot \tau_{elec}^2 \tag{4.1}$$

donde k es la constante de proporcionalidad para las pérdidas de resistencia.

$$k = \frac{\omega_\eta \cdot (1 - \eta/100)}{\tau_\eta \cdot \eta/100} \tag{4.2}$$

Luego, la potencia eléctrica se puede expresar al incluir las pérdidas resistivas como:

$$P_{elec} = P_{losses} + \omega \cdot \tau_{elec} \tag{4.3}$$

La ley de Joule define el índice de conversión de la energía eléctrica en energía térmica, de modo que es posible obtener el valor de la corriente eléctrica según la siguiente expresión:

$$I = \frac{P_{elec}}{V} \tag{4.4}$$

donde:

- ω es la velocidad angular del rotor
- τ_{elec} es la demanda de torque

- η es la eficiencia global del motor
- ω_η es la velocidad angular a la que se mide la eficiencia global del motor
- τ_η es el torque al que se mide la eficiencia global del motor
- V es la tensión en los terminales del motor
- I es la corriente en los terminales del motor

Al considerar el caso ideal en el que el motor eléctrico posee un valor de eficiencia global del 100 %, la ecuación 4.4 se puede expresar de manera simplificada de la siguiente manera:

$$I = \frac{\omega \cdot \tau_{elec}}{V} \tag{4.5}$$

El motor acciona el eje cigüeñal de la bomba mediante una caja de cambios simple de relación 2:1, por lo que la velocidad angular del motor eléctrico es el doble de la velocidad angular del eje cigüeñal de la bomba y el torque que genera el motor es la mitad que el torque transmitido al eje cigüeñal de la bomba. Se asume que la transmisión mecánica de energía entre el motor eléctrico y la bomba es ideal y sin pérdidas.

$$\omega = \omega_{motor} = 2 \cdot \omega_{bomba} \qquad \wedge \qquad \tau_{elec} = \tau_{motor} = \frac{\tau_{bomba}}{2} \tag{4.6}$$

El controlador de velocidad se representa como un controlador PI que regula el torque generado por el motor eléctrico mediante un bucle de retroalimentación. El controlador PI compara el valor de la variable real con el valor deseado, para luego intentar minimizar el error entre ambos valores a través de un algoritmo de control que consta de dos componentes: el componente proporcional y el componente integral. El componente proporcional se calcula en función del error actual, mientras que el componente integral se basa en la suma acumulada de los errores pasados. El esquema de la Figura 4.2 muestra una representación del controlador PI utilizado para ajustar el torque del motor en función del error entre la velocidad angular del eje cigüeñal de la bomba y la velocidad angular nominal de referencia.

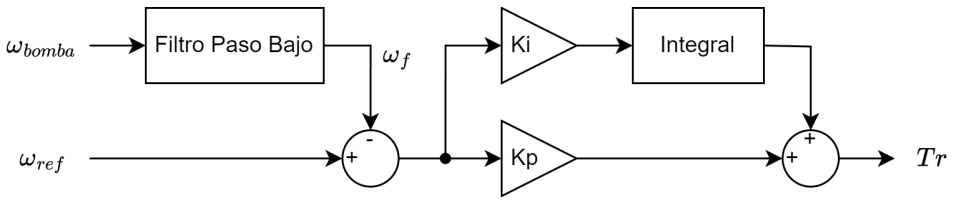


Figura 4.2: Esquema del controlador PI.
(Fuente: Elaboración Propia.)

La representación esquemática del controlador PI se traduce en la siguiente expresión:

$$Tr(t) = K_p \cdot e(t) + K_i \cdot \int_0^t e(t)\, dt \tag{4.7}$$

donde Tr es el torque de referencia que utiliza el motor para actualizar el valor del torque a generar, y $e(t) = \omega_{ref} - \omega_f$ corresponde al error entre la velocidad angular nominal de referencia ω_{ref} y la velocidad angular filtrada del eje cigüeñal de la bomba ω_f.

A la señal de la velocidad angular del eje cigüeñal de la bomba ω_{bomba} se le aplica un filtro paso bajo debido a que el controlador PI responde de manera directa a los cambios en la señal de entrada, por lo que las altas frecuencias en dicha señal pueden causar oscilaciones no deseadas en la respuesta del controlador. Un filtro paso bajo suaviza dichas oscilaciones, permitiendo que el controlador PI responda de manera mucho más estable.

Una forma común de describir un filtro paso bajo consiste en la caracterización de la respuesta de frecuencia mediante la función de transferencia[1].

$$H(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{\omega_0}{(s + \omega_0)} \tag{4.8}$$

Luego, al pasar al dominio del tiempo se tiene que:

$$v_{out}(t) = v_{in}(t) - \frac{1}{\omega_0} \cdot \frac{dv_{out}}{dt} \tag{4.9}$$

donde la señal de entrada v_{in} representa la velocidad angular del eje cigüeñal de la bomba, la señal de salida v_{out} representa el valor de la velocidad angular filtrada y ω_0 es igual a $2 \cdot \pi \cdot fc$, donde fc corresponde a la frecuencia de corte del filtro. Luego, la expresión para la velocidad angular filtrada ω_f en función de la señal de entrada de la velocidad angular del eje cigüeñal de la bomba ω_{bomba} es la siguiente:

$$\omega_f(t) = \omega_{bomba}(t) - \frac{1}{2 \cdot \pi \cdot fc} \cdot \frac{d\omega_f}{dt} \tag{4.10}$$

El valor del torque generado por el motor eléctrico τ_{motor} se obtiene a partir del valor del torque de referencia Tr y de una constante de tiempo Tc según la siguiente expresión:

$$\tau_{motor}(t) = Tr - Tc \cdot \frac{d\tau_{motor}}{dt} \tag{4.11}$$

La constante de tiempo Tc representa el seguimiento temporal de la señal de control, donde su recíproco $1/Tc$ representa la frecuencia de actualización del valor del torque del motor respecto al valor del torque de referencia, es decir, el valor de τ_{motor} debe alcanzar el valor de Tr en una ventana de tiempo Tc. Adicionalmente, se impone un límite de 5 Nm al valor máximo de torque generado por el motor eléctrico durante la operación continua.

La dinámica del sistema motor eléctrico - eje cigüeñal se obtiene al considerar la segunda ley de Newton aplicada al movimiento rotacional:

$$\sum_{i=1}^{N} \tau_i = I_{bomba} \cdot \alpha_{bomba} \tag{4.12}$$

donde I_{bomba} es el momento de inercia del eje cigüeñal de la bomba y α_{bomba} es su aceleración angular. El término de la izquierda representa todos los torques que actúan sobre el eje cigüeñal de la bomba, los cuales corresponden al torque positivo transmitido desde el motor eléctrico τ_{bomba} y los torques que se oponen al movimiento rotacional τ_{roce} y τ_{carga}. Al expresar τ_{roce} como el producto entre el coeficiente de amortiguación B_{bomba}, el cual indica los efectos viscosos o de roce en los rodamientos, y la velocidad angular del eje cigüeñal ω_{bomba}, la ecuación 4.12 puede expresarse de la siguiente manera:

$$\tau_{bomba} = I_{bomba} \cdot \alpha_{bomba} + B_{bomba} \cdot \omega_{bomba} + \tau_{carga} \tag{4.13}$$

donde τ_{carga} representa los esfuerzos rotacionales que debe superar el eje para el desplazamiento de los émbolos y generar la presión de trabajo. Se asume que las fluctuaciones en la velocidad del motor, causadas por la carga mecánica, no afectan la medición y ajuste del valor del torque del motor.

Dado que la aceleración angular del eje cigüeñal α_{bomba} también corresponde a la derivada de la velocidad angular ω_{bomba} en el tiempo, la ecuación 4.13 se puede reescribir de la siguiente manera:

$$\frac{d\omega_{bomba}}{dt} = (\tau_{bomba} - B_{bomba} \cdot \omega_{bomba} - \tau_{carga}) \cdot \frac{1}{I_{bomba}} \tag{4.14}$$

Luego, al reordenar las ecuaciones 4.10 y 4.11, al derivar en función del tiempo la expresión para el torque de referencia de la ecuación 4.7, y al agregar la expresión para la derivada del desplazamiento angular en el tiempo, se obtiene el siguiente sistema de ecuaciones dife-

renciales ordinarias, el cual define la evolución temporal del bloque del motor:

$$\begin{aligned}
\frac{d\tau_{motor}}{dt} &= (Tr - \tau_{motor}) \cdot \frac{1}{Tc} \\
\frac{d\omega_{bomba}}{dt} &= (2 \cdot \tau_{motor} - B_{bomba} \cdot \omega_{bomba} - \tau_{carga}) \cdot \frac{1}{I_{bomba}} \\
\frac{d\theta_{bomba}}{dt} &= \omega_{bomba} \\
\frac{d\omega_f}{dt} &= (\omega_{bomba} - \omega_f) \cdot 2 \cdot \pi \cdot fc \\
\frac{dTr}{dt} &= -K_p \cdot \frac{d\omega_f}{dt} + K_i \cdot (\omega_{ref} - \omega_f)
\end{aligned} \tag{4.15}$$

La información acerca de las interacciones con el bloque de la bomba y el bloque de carga/descarga están presentes en el sistema mediante la variable τ_{carga}, la cual representa el esfuerzo adicional que debe ser generado por el motor eléctrico para accionar el eje cigüeñal de la bomba debido al fluido que debe ser desplazado por los tres émbolos y a las pérdidas de carga en las tuberías de entrada y salida.

Bloque Bomba

El bloque de la bomba modela las interacciones entre el eje cigüeñal de la bomba, los tres émbolos que este acciona y las válvulas de retención que controlan la entrada y salida del fluido de trabajo en las cámaras de los cilindros. El esquema de la Figura 4.3 muestra una representación de un émbolo y las válvulas de retención de entrada y salida.

El modelo considera un fluido de trabajo con una pequeña cantidad de aire encerrado. En general esta cantidad de aire encerrado no permanece constante debido a que se generan efectos de disolución del aire en el fluido [2]. Sin embargo, en este caso se asume que la cantidad de aire encerrado en el fluido de trabajo se mantiene constante.

Al contener una cantidad de aire encerrado, el fluido de trabajo se modela como un fluido politrópico de flujo compresible, en el cual la presión del aire y la presión del líquido son idénticas. Para el modelo se asume un proceso politrópico de índice igual 1, es decir, el fluido de trabajo se considera isotérmico[1].

[1]En el flujo isotérmico, la densidad del fluido varía con la presión mientras que la temperatura se mantiene constante.

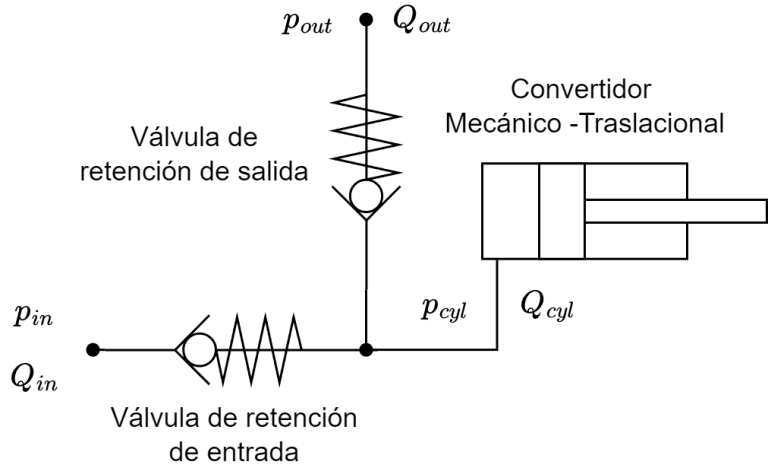


Figura 4.3: Esquema del sistema del émbolo.
(Fuente: Elaboración Propia.)

La densidad de una mezcla isotérmica de líquido-aire a una presión de referencia se define como la masa total del líquido y del aire encerrado sobre el volumen total del líquido y del aire encerrado a dicha presión. Mientras que la masa total de la mezcla se conserva al variar la presión, el volumen de la mezcla no permanece constante. El volumen que ocupa el aire encerrado en la mezcla viene dado por una fracción volumétrica:

$$\alpha_0 = \frac{V_{g0}}{V_{g0} + V_{L0}} \tag{4.16}$$

donde:

- α_0 es la relación volumétrica aire/líquido a la presión de referencia (atmosférica)
- V_{g0} es el volumen de aire a la presión de referencia
- V_{L0} es el volumen de líquido puro a la presión de referencia

Luego, el volumen que ocupa el aire encerrado en la mezcla V_g a una presión dada p se obtiene según la ecuación de los gases politrópicos:

$$V_g = V_{g0} \cdot \left(\frac{p_0}{p}\right)^{1/n} \tag{4.17}$$

donde:

- p_0 es la presión de referencia (atmosférica)
- n es el coeficiente politrópico del aire (1 cuando se considera proceso isotérmico)

Al considerar al líquido puro como un fluido incompresible, se asume que su módulo de volumen [2] es constante. Por lo tanto, bajo este supuesto la densidad de la mezcla líquido-aire se expresa como:

$$\rho_{mix} = \frac{\rho_{L0} + \left(\frac{\alpha_0}{1-\alpha_0}\right) \cdot \rho_{g0}}{e^{-(p-p_0)/\beta_L} + \left(\frac{\alpha_0}{1-\alpha_0}\right) \cdot \left(\frac{p_0}{p}\right)^{1/n}} \tag{4.18}$$

donde:

- ρ_{L0} es la densidad del líquido puro a la presión de referencia p_0
- ρ_{g0} es la densidad del aire a la presión de referencia p_0
- β_L es el módulo de volumen del líquido puro

Luego el módulo de volumen de la mezcla se define como:

$$\beta_{mix} = \beta_L \cdot \frac{e^{-(p-p_0)/\beta_L} + \left(\frac{\alpha_0}{1-\alpha_0}\right) \cdot \left(\frac{p_0}{p}\right)^{1/n}}{e^{-(p-p_0)/\beta_L} + \beta_L \cdot \frac{1}{n} \cdot \left(\frac{\alpha_0}{1-\alpha_0}\right) \cdot \left(\frac{p_0^{1/n}}{p^{1/n+1}}\right)} \tag{4.19}$$

Finalmente la derivada parcial de la densidad de la mezcla en función de la presión p se define como[27]:

$$\frac{\partial \rho_{mix}}{\partial p} = \frac{\left(\rho_{L0} + \left(\frac{\alpha_0}{1-\alpha_0}\right) \cdot \rho_{g0}\right) \cdot \left(\frac{1}{\beta_L} \cdot e^{-(p-p_0)/\beta_L} + \frac{1}{n} \cdot \left(\frac{\alpha_0}{1-\alpha_0}\right) \cdot \left(\frac{p_0^{1/n}}{p^{1/n+1}}\right)\right)}{\left(e^{-(p-p_0)/\beta_L} + \left(\frac{\alpha_0}{1-\alpha_0}\right) \cdot \left(\frac{p_0}{p}\right)^{1/n}\right)^2} \tag{4.20}$$

Por otro lado, en la Figura 4.3, se muestra la interacción cilindro-émbolo como un convertidor mecánico-traslacional. Un convertidor mecánico-traslacional modela la interfaz entre un fluido isotérmico y la mecánica traslacional al convertir la presión que ejerce el fluido al interior de la cámara del cilindro en fuerza mecánica y viceversa. Las ecuaciones de conservación de la masa en el volumen del convertidor mecánico son:

$$\dot{m} = \varepsilon \cdot \rho_{cyl} \cdot S \cdot v + \frac{1}{\beta_{cyl}} \cdot \frac{dp_{cyl}}{dt} \cdot \rho_{cyl} \cdot V \tag{4.21}$$

[2] El módulo de volumen es una medida de la resistencia de una sustancia a la compresión volumétrica y se define como la relación entre el aumento infinitesimal de la presión y la disminución relativa del volumen resultante.

$$v = \frac{dx}{dt} \tag{4.22}$$

$$V = V_{dead} + \varepsilon \cdot S \cdot x \tag{4.23}$$

donde:

- $\dot{m}$ es el flujo de masa que entra o sale del cilindro
- ε es la orientación mecánica del convertidor (1 si el aumento de la presión del fluido provoca un desplazamiento positivo del émbolo, -1 si el aumento de la presión del fluido provoca un desplazamiento negativo del émbolo)
- ρ_{cyl} es la densidad del fluido al interior del cilindro
- β_{cyl} es el módulo de volumen del fluido interior del cilindro
- S es el área de la sección transversal del émbolo
- v es la velocidad traslacional del émbolo
- x es el desplazamiento relativo del émbolo
- V es el volumen del fluido al interior del cilindro
- V_{dead} es el volumen muerto, es decir, el volumen del fluido cuando el desplazamiento es igual al cero
- p_{cyl} es la presión del fluido al interior del cilindro

Tanto la velocidad como el desplazamiento del émbolo dependen directamente del movimiento rotacional del eje cigüeñal de la bomba, por lo que es posible escribir las ecuaciones de desplazamiento $x(t)$ y de velocidad relativa $v(t)$ del émbolo de la siguiente forma:

$$x(t) = \frac{c}{2} - \frac{c}{2} \cdot \cos\left(\theta_{bomba}\right) \tag{4.24}$$

$$v(t) = \frac{c}{2} \cdot \omega_{bomba} \cdot \sin\left(\theta_{bomba}\right) \tag{4.25}$$

donde c es la carrera del émbolo [3].

Al reordenar de la ecuación 4.21 se obtiene la ecuación diferencial que describe el comportamiento de la presión al interior del cilindro para un fluido compresible.

[3]La carrera de un émbolo se refiere a la distancia recorrida por el émbolo entre el punto muerto superior y el punto muerto inferior, es decir, es la longitud del camino que el émbolo recorre dentro del cilindro.

$$\frac{dp_{cyl}}{dt} = (\dot{m} - \varepsilon \cdot \rho_{cyl} \cdot S \cdot v) \cdot \frac{\beta_{cyl}}{\rho_{cyl} \cdot V} \tag{4.26}$$

Luego la ecuación de la conservación de momento al interior del volumen del convertidor mecánico es:

$$F_{cyl} = \varepsilon \cdot (p_{atm} - p_{cyl}) \cdot S \tag{4.27}$$

donde F_{cyl} es la fuerza que ejerce el líquido sobre el émbolo y p_{atm} es la presión ambiental fuera del cilindro. En la expresión anterior se considera que las paredes del cilindro son perfectamente rígidas y que tanto la resistencia al flujo entre la entrada y el interior del cilindro como la energía cinética del fluido al interior del cilindro son despreciables [27].

La fuerza de fricción como consecuencia del contacto entre cuerpos en movimiento debido al desplazamiento del émbolo al interior del cilindro, se simula como función de la velocidad relativa del émbolo, y se asume como la suma de las fuerzas de fricción de Stribeck, de Coulomb y los componentes viscosos, como se muestra en la siguiente figura:

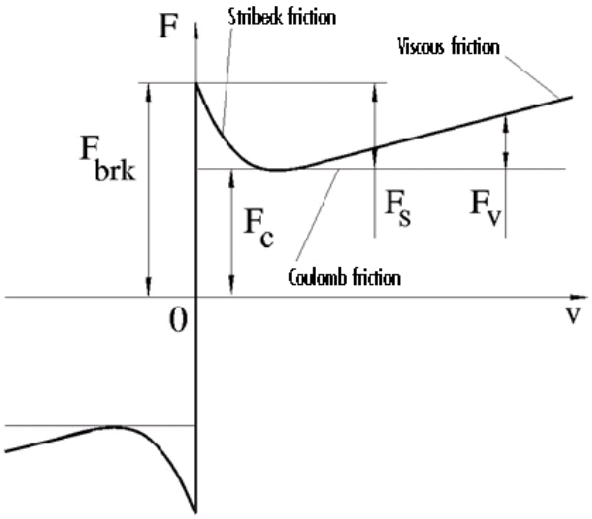


Figura 4.4: Representación de la fuerza de fricción como suma de las fuerzas de fricción de Stribeck, de Coulomb y los componentes viscosos.
(Fuente: Armstrong, B. and C.C. de Wit (1995). Friction Modelling and Compensation, The Control Handbook. [8])

La fricción de Stribeck F_S es la característica de inclinación negativa que se produce a velocidades bajas [8]. Por su parte, la fricción de Coulomb F_C tiene como resultado una

fuerza constante a cualquier velocidad y la fricción viscosa F_V es aquella que se opone al movimiento con una fuerza directamente proporcional a la velocidad relativa de los cuerpos en contacto. Se suele referir a la suma de las fricciones de Coulomb y Stribeck cerca de la velocidad cero como fricción de arranque F_{brk}. La fuerza de fricción F_{fric} se aproxima mediante las siguientes ecuaciones:

$$F_{fric} = \sqrt{2 \cdot e} \cdot (F_{brk} - F_C) \cdot e^{-\left(\frac{v}{v_{St}}\right)^2} \cdot \frac{v}{v_{St}} + F_C \cdot \tanh\left(\frac{v}{v_{Coul}}\right) + f \cdot v \tag{4.28}$$

$$v_{St} = v_{brk} \cdot \sqrt{2} \tag{4.29}$$

$$v_{Coul} = v_{brk}/10 \tag{4.30}$$

donde:

- F_{fric} es la fuerza de fricción
- F_C es la fricción de Coulomb
- F_{brk} es la fricción de arranque
- v_{brk} es la velocidad de fricción de arranque
- v_{St} es el umbral de velocidad de Stribeck
- v_{Coul} es el umbral de velocidad de Coulomb
- v es la velocidad relativa
- f es el coeficiente de fricción viscosa

La función exponencial utilizada en la parte de Stribeck de la ecuación de fuerza es continua y cae a magnitudes de velocidad superiores a la velocidad de fricción de arranque. La función tangente hiperbólica utilizada en la parte de Coulomb de la ecuación de fuerza garantiza que la ecuación sea fluida y continua en $v = 0$, pero alcanza rápidamente su valor total a velocidades distintas de cero [8].

Luego el torque que debe superar el eje de la bomba para desplazar el fluido dentro y fuera de los cilindros se expresa en función de las fuerzas que el líquido ejerce sobre el émbolo F_{cyl} y la fuerza de fricción F_{fric} de la siguiente manera:

$$\tau_{cyl} = (F_{cyl} + F_{fric}) \cdot r \cdot \sin(\theta_{bomba}) \tag{4.31}$$

donde r es la longitud del brazo donde se aplica la fuerza en el eje cigüeñal, el cual corresponde a la distancia de excentricidad entre el eje de rotación del cigüeñal y el eje del muñón de biela (Figura 4.5). La función de seno que aparece en la expresión se debe a que la longitud del brazo r varia según la rotación del eje cigüeñal.

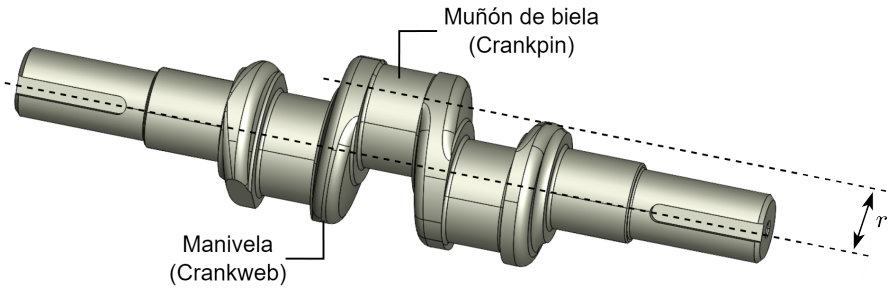


Figura 4.5: Eje cigüeñal de la bomba CAT 1051.
(Fuente: Steve Miller (2024). Predictive Maintenance in a Hydraulic Pump [62])

La variable τ_{carga} presente en el sistema de ecuaciones 4.15 se calcula como la suma de los torques asociados a los tres convertidores mecánicos obtenidos según la ecuación 4.31, pero agregando el desfase correspondiente a cada ecuación de modo que:

$$\tau_{carga} = \tau_{cyl_1} + \tau_{cyl_2} + \tau_{cyl_3} \tag{4.32}$$

donde:

$$\begin{aligned} \tau_{cyl_1} &= (F_{cyl_1} + F_{fric_1}) \cdot r \cdot \sin(\theta_{bomba}) \\ \tau_{cyl_2} &= (F_{cyl_2} + F_{fric_2}) \cdot r \cdot \sin(\theta_{bomba} - \frac{2\pi}{3}) \\ \tau_{cyl_3} &= (F_{cyl_3} + F_{fric_3}) \cdot r \cdot \sin(\theta_{bomba} - \frac{4\pi}{3}) \end{aligned} \tag{4.33}$$

Según la Figura 4.3 la entrada y salida del fluido de trabajo en los cilindros se controla mediante las válvulas de retención. Estas válvulas permiten el flujo de fluido en una dirección y lo bloquean en la dirección opuesta. El área de apertura de la válvula A_{valve} se modela como una parametrización lineal en función de la diferencia de presión Δp en los terminales de la válvula. La apertura de la válvula comienza cuando Δp alcanza o supera el valor de la presión de apertura p_{crak}, momento en el cual el área de apertura de la válvula crece linealmente hasta alcanzar su máximo valor A_{max} cuando la diferencia de presión Δp alcanza el valor de la presión de apertura máxima p_{max}.

$$A_{valve} = \hat{p} \cdot (A_{max} - A_{leak}) + A_{leak} \tag{4.34}$$

donde la presión normalizada $\hat{p}$ es:

$$\hat{p} = \frac{\Delta p - p_{crak}}{p_{max} - p_{crak}} \tag{4.35}$$

Como se puede apreciar en la ecuación 4.34, el área de apertura de la válvula nunca es cero, puesto que su límite inferior es igual a A_{leak}. Esto implica que, aún cuando la válvula de retención se considere "cerrada", siempre existirá un flujo de fluido que atraviesa la válvula de un lado al otro, cuya dirección dependerá de la diferencia de presión entre los terminales de la válvula. El flujo másico a través de la válvula en función del área de apertura se expresa de la siguiente manera:

$$\dot{m}_{valve} = \frac{C_d \cdot A_{valve} \cdot \sqrt{2 \cdot \bar{\rho}} \cdot \Delta p}{\left(\Delta p^2 + p_{crit}^2\right)^{1/4}} \tag{4.36}$$

donde:

- C_d es el coeficiente de descarga
- A_{valve} es el área de apertura instantánea de la válvula
- $\bar{\rho}$ es la densidad promedio del fluido
- Δp es la diferencia de presión entre los terminales de la válvula

La diferencia de presión crítica, p_{crit}, es la diferencia de presión asociada al número de Reynolds crítico, Re_{crit}, y corresponde al punto de transición del régimen de flujo entre flujo laminar y turbulento:

$$p_{crit} = \frac{\pi \cdot \bar{\rho}}{8 \cdot A_{valve}} \cdot \left(\frac{\nu \cdot Re_{crit}}{C_d}\right)^2 \tag{4.37}$$

Las válvulas de retención de salida y entrada para todos los cilindros se consideran idénticas y se modelan con las mismas ecuaciones. La única diferencia que existe entre las válvulas de retención de entrada y salida es valor de la diferencia de presión Δp considerado para cada caso, ya que la válvula de retención de entrada compara la presión del fluido a la entrada de la bomba con la presión al interior del cilindro ($\Delta p_{in} = p_{in} - p_{cyl}$), mientras que la válvula de retención de salida compara la presión al interior del cilindro con la presión a la salida de la bomba o presión de trabajo ($\Delta p_{out} = p_{cyl} - p_{out}$).

El flujo másico a través de la válvula de la ecuación 4.36 corresponde al mismo flujo másico que entra o sale del cilindro en la ecuación 4.26. Si se calcula el flujo másico que atraviesa una válvula de retención de entrada, entonces, en la práctica se obtiene el flujo másico que entra al cilindro. Lo mismo aplica si se calcula el flujo másico que atraviesa alguna válvula de retención de salida. Por lo tanto, dado que la razón entre el flujo másico a través de alguna válvula de retención y la densidad promedio del fluido que atraviesa dicha válvula corresponde al flujo volumétrico, entonces caudal de entrada y salida de cada cilindro se puede obtener a partir de las siguientes expresiones:

$$\begin{aligned} Q_{in,cyl_1} &= \frac{\dot{m}_{in,cyl_1}}{\bar{\rho}_{in,cyl_1}} \quad \wedge \quad Q_{out,cyl_1} = \frac{\dot{m}_{out,cyl_1}}{\bar{\rho}_{out,cyl_1}} \\ Q_{in,cyl_2} &= \frac{\dot{m}_{in,cyl_2}}{\bar{\rho}_{in,cyl_2}} \quad \wedge \quad Q_{out,cyl_2} = \frac{\dot{m}_{out,cyl_2}}{\bar{\rho}_{out,cyl_2}} \\ Q_{in,cyl_3} &= \frac{\dot{m}_{in,cyl_3}}{\bar{\rho}_{in,cyl_3}} \quad \wedge \quad Q_{out,cyl_3} = \frac{\dot{m}_{out,cyl_3}}{\bar{\rho}_{out,cyl_3}} \end{aligned} \tag{4.38}$$

Bloque Carga/Descarga

El diagrama de la Figura 4.1 establece que el bloque de carga y descarga presenta tuberías tanto a la entrada como a la salida de la bomba. Dichas tuberías se consideran idénticas y se asume que su presencia genera pérdidas o caídas de presión en tuberías según la ecuación de Darcy-Weisbach (4.39).

$$p_{loss} = f \cdot \frac{8 \cdot L_e \cdot \rho \cdot Q^2}{\pi^2 \cdot D^5} \tag{4.39}$$

donde:

- f es factor de fricción de Darcy-Weisbach
- L_e es la longitud equivalente de la tubería
- ρ es la densidad del fluido que atraviesa la tubería
- Q es el caudal que atraviesa la tubería
- D es el diámetro interior de la tubería

El caudal que aparece en la expresión anterior corresponde al caudal que entra o sale de la bomba según se calculan las pérdidas en la tubería de entrada o de salida. En la tubería de entrada, el caudal corresponde a la suma de los caudales individuales que entran a los cilindros de la ecuación 4.38. Lo mismo aplica para el caudal en la tubería de salida.

$$
\begin{aligned}
Q_{in} &= Q_{in,cyl_1} + Q_{in,cyl_2} + Q_{in,cyl_3} \\
Q_{out} &= Q_{out,cyl_1} + Q_{out,cyl_2} + Q_{out,cyl_3}
\end{aligned}
\tag{4.40}
$$

El coeficiente de fricción de Darcy-Weisbach f puede variar dependiendo de si el flujo dentro de la tubería sigue un régimen laminar o turbulento. Según el diagrama de Moody [9], es posible definir cuatro regiones diferenciadas: región laminar, donde la rugosidad relativa de las paredes de la tubería no tienen efectos apreciables; la zona de transición que representa una región intermedia donde el factor de fricción depende de la rugosidad relativa y el número de Reynolds; el régimen de flujo hidráulico liso conocido por el grado moderado de rugosidad relativa, donde la tubería actúa como una tubería lisa; y la región de turbulencia completamente desarrollada, donde el factor f es independiente del número de Reynolds. Para la región laminar, es decir, para $0 \leq Re < 2300$, el coeficiente de fricción f depende únicamente del número de Reynolds.

$$
f = \frac{64}{Re} \tag{4.41}
$$

donde el Número de Reynolds representa la razón entre los esfuerzos inerciales y viscosos presentes en un fluido. Un número de Reynolds pequeño indica que las fuerzas viscosas son dominantes y un número de Reynolds grande indica que las fuerzas de inercia son dominantes [80].

$$
Re = \frac{4 \cdot Q}{\pi \cdot \nu \cdot D} \tag{4.42}
$$

Según las investigaciones de Nikuradse sobre flujo turbulento en tuberías, el coeficiente de fricción n la región de turbulencia completamente desarrollada depende únicamente de la rugosidad relativa [109].

$$
f = \left(-2 \cdot \log \left(\frac{\varepsilon / D}{3{,}7} \right) \right)^{-2} \tag{4.43}
$$

donde:

- ε es la altura media de la rugosidad
- D es el diámetro interno de la tubería

Antes de 1939, cuando se publicó la ecuación implícita de Colebrook-White para el régimen turbulento en tuberías lisas, se utilizaba ampliamente la ecuación implícita de Prandtl, en la cual derivó una fórmula a partir del perfil logarítmico de velocidad y datos experimentales disponibles sobre tuberías lisas [22].

$$f = \left(-2 \cdot \log \left(\frac{2{,}51}{Re \cdot \sqrt{f}} \right) \right)^{-2} \tag{4.44}$$

La fórmula más conocida a día de hoy para describir el factor de fricción en la zona de transición es la ecuación implícita de Colebrook-White [20], la cual considera los términos presentados por Nikuradse y Prandtl.

$$f = \left(-2 \cdot \log \left(\frac{\varepsilon/D}{3{,}7} + \frac{2{,}51}{Re \cdot \sqrt{f}} \right) \right)^{-2} \tag{4.45}$$

El problema con la ecuación de Colebrook-White radica en que calcular el factor de fricción f no es una tarea sencilla debido a que dicho término aparece a ambos lados de la ecuación, es decir, la ecuación de Colebrook-White es implícita. La resolución de este tipo de ecuaciones requiere un método de resolución iterativo que puede ser costoso en términos computacionales y que, de no ser manejado apropiadamente, puede acumular grandes errores. Por esta razón, se han propuesto varias soluciones explícitas aproximadas a la ecuación de Colebrook-White, ya que una solución explícita no requiere de un cálculo iterativo y por ende su impacto en el coste computacional es mínimo. En [110], los autores describen más de 30 ecuaciones que plantean una solución explícita a la ecuación de Colebrook-White, donde además realizan un estudio y un análisis comparativo de todas las aproximaciones en comparación con la solución de la ecuación de Colebrook-White para todo el rango del diagrama de Moody ($2300 \leq Re \leq 10^8$ y $0 \leq \varepsilon/D \leq 5{\cdot}10^{-2}$). En el estudio, la mejor aproximación corresponde a la ecuación propuesta por Vatankhah et al [99]:

$$f = \left(\frac{2{,}51/Re + 1{,}1513 \cdot \delta}{\delta - (\varepsilon/D)/3{,}71 - 2{,}3026 \cdot \delta \cdot \log(\delta)} \right)^2 \tag{4.46}$$

donde:

$$\delta = \frac{6{,}0173}{Re \cdot (0{,}07 \cdot (\varepsilon/D) + Re^{-0{,}885})^{0{,}109}} + \frac{\varepsilon/D}{3{,}71} \tag{4.47}$$

En su trabajo, Vatankhah et al [99] obtiene un error relativo de su aproximación menor al 0.008 %, por lo que su solución explícita de dos pasos a la ecuación de Colebrook-White se puede considerar casi exacta.

Con lo anterior, las presiones a la salida y entrada de la bomba se pueden expresar de la siguiente manera:

$$p_{in} = p_{atm} - p_{loss_{in}} \tag{4.48}$$

$$p_{out} = (p_{atm} + \Delta p) + p_{loss_{out}} \tag{4.49}$$

donde las pérdidas en tuberías $p_{loss_{in}}$ y $p_{loss_{out}}$ se obtiene según la ecuación 4.39 y el factor de fricción se calcula según las ecuaciones de Vatankhah et al 4.46 o con 4.41 dependiendo si el régimen es laminar o turbulento. ($0 \leq Re < 2300$ para laminar y $2300 \leq Re \leq 10^8$ para régimen turbulento).

4.1.2. Modos de Falla

Como se mencionó anteriormente, el modelo simulado de la bomba desarrollado en el entorno Simulink de MATLAB, usado como referencia [62], incluye los efectos de componentes defectuosos al simular el comportamiento degradado de la bomba debido al desgaste de los rodamientos del eje del cigüeñal, la fuga de fluido en los cilindros debido al deterioro de los sellos de la junta, y el bloqueo en las válvulas de retención en la entrada de los cilindros. Estos defectos representan los modos de falla considerados para el modelo. La condición de desgaste en los rodamientos en el eje cigüeñal de la bomba se puede inducir al variar el valor del parámetro B_{bomba} en el sistema de ecuaciones 4.15. Dicho parámetro representa el coeficiente de amortiguación capaz de explicar la disipación de energía debida a las fricciones viscosas que frenan el movimiento rotacional del eje. Mientras mayor sea el valor de este parámetro, mayores serán los efectos del roce y, por ende, mayor será el efecto de desgaste o deterioro en los rodamientos.

Para simular la fuga de fluido en los cilindros, se considera un pequeño orificio como punto de fuga en lugar de modelar la degradación detallada de los sellos de la junta. Esta simplificación es práctica, ya que el efecto principal de este tipo de fallo es una disminución en el caudal de salida de la bomba. Al colocar este pequeño orificio entre la salida de cada cilindro y su respectiva válvula de retención, se produce como resultado que una fracción del flujo másico que sale del cilindro no llegue a la válvula. Por lo tanto, cuando se alcanza la presión de apertura de la válvula, el flujo másico que la atraviesa es menor que el flujo que originalmente salió del cilindro. Dicho orificio se puede modelar según la siguiente expresión:

$$\dot{m}_{orifice} = \frac{C_d \cdot A_{orifice} \cdot \sqrt{2 \cdot \bar{\rho}} \cdot (p_A - p_B)}{\left((p_A - p_B)^2 + p_{crit}^2\right)^{1/4}} \tag{4.50}$$

donde:

- C_d es el coeficiente de descarga
- $A_{orifice}$ es el área de apertura instantánea del orificio
- $\bar{\rho}$ es la densidad promedio del fluido

Si el área transversal del orificio $A_{orifice}$ aumenta, mayor será el efecto de fuga de fluido en cada cilindro, por lo que, de forma análoga, mayor será el efecto de desgaste o deterioro en los sellos de la junta.

Los términos p_A y p_B en la ecuación 4.50 representan las presiones en los terminales del orificio, las cuales corresponden a la presión interior del cilindro y a la presión atmosférica respectivamente. Es posible notar que, cuando la presión interior del cilindro (p_A) es menor que la presión de referencia o atmosférica (p_B), el valor del flujo másico del orificio $\dot{m}_{orifice}$ se vuelve negativo. Esto sugiere que, debido al gradiente de presión, debería entrar fluido al cilindro en lugar de salir. Sin embargo, ya que el orificio representa una fuga de fluido hacia el ambiente y no hacia algún reservorio que permita la re circulación del fluido, es necesario limitar el valor del flujo másico $\dot{m}_{orifice}$ a cero cuando $p_B > p_A$.

El término de diferencia de presión crítica, p_{crit}, corresponde la diferencia de presión asociada al número de Reynolds crítico, Re_{crit}, y se obtiene de igual manera que en caso del cálculo para las válvulas de retención (ecuación 4.37), pero reemplazando el término del área de apertura de la válvula A_{valve} por el área del orificio $A_{orifice}$.

Finalmente, el efecto del bloqueo de las válvulas de retención de entrada se modela al variar el valor de A_{max} en la ecuación 4.34. Dicho valor representa la máxima apertura que puede alcanzar la válvula cuando el diferencial de presión al que está sometida es igual a la presión de apertura máxima, p_{max}. Al disminuir el valor de A_{max}, se induce el efecto de bloqueo en las válvula de retención de entrada.

4.1.3. Modelo de Degradación

Para desarrollar y evaluar modelos basados en datos para la estimación del RUL, los conjuntos de datos generados por el modelo simulado descrito en las secciones anteriores deben incluir trayectorias de degradación que representen el comportamiento del sistema a lo largo

del tiempo. En [7], se define que la degradación puede ser modelada como una combinación de tres términos contribuidores: una degradación inicial, una degradación normal y una degradación anormal.

Degradación inicial. Este término asume que, debido a las tolerancias en los procesos de manufactura y montaje, cada unidad presentará desgastes iniciales ligeramente diferentes en sus componentes. La degradación debida a estos desgastes iniciales no se considera anormal, pero puede marcar la diferencia en la vida útil de un componente. Este desgaste inicial se modela al generar variaciones en los valores iniciales de los parámetros B_{bomba}, A_{max} (para la válvula de retención de entrada) y $A_{orifice}$. Para cada uno de estos parámetros se asume una distribución uniforme $U(0, 0.05)$, es decir, se impone una variación aleatoria del 5 % respecto a los valores nominales. La degradación inicial se denota como δ_0.

Degradación Normal. Además del desgaste inicial, se asume que los componentes del sistema también están sujetos a experimentar un proceso de degradación por desgaste debido al uso continuado. Este tipo de degradación se considera normal y se puede modelar como una tendencia lineal decreciente, dada por la siguiente expresión:

$$\delta_n(t) = a_n \cdot t + \delta_0 \quad \forall \quad t \leq t_s \tag{4.51}$$

donde a_n es la pendiente de la curva de degradación. Al imponer una degradación máxima del 10 % ($m = 0.1$) respecto del valor de la degradación inicial para los parámetros B_{bomba}, A_{max} y $A_{orifice}$, cuando $t = t_s$, la pendiente a_n se puede definir de la siguiente manera:

$$a_n = \varepsilon \cdot \frac{m \cdot \delta_0}{t_s} \tag{4.52}$$

donde ε es igual a 1 si la pendiente es positiva y -1 si la pendiente es negativa.

Por otra parte, durante el ciclo de vida operativo de los equipos, en algún momento, el estado de salud puede cambiar hacia un comportamiento *anormal* debido a la presencia de algún modo de falla específico. Este momento se conoce como el tiempo de transición t_s. Posteriormente, la falla correspondiente genera una degradación paulatina hasta que finalmente ocurre la falla en el tiempo t_{EOL} (end-of-life).

La aparición de una condición *anormal* como resultado de un modo de falla específico puede ser modelada al asumir un proceso estocástico con una distribución de probabilidad Weibull[4]. Si X es una variable aleatoria continua, entonces se dice que dicha variable tiene una distribución Weibull con parámetros $\beta, \eta > 0$ si su función de densidad de probabilidad es:

$$f(x) = \left(\frac{\beta x^{\beta-1}}{\eta^{\beta}}\right) \cdot e^{\left(-\frac{x}{\eta}\right)^{\beta}} \tag{4.53}$$

donde β es el parámetro de forma y η es el parámetro de escala de la distribución. Si se asume que la tasa de falla es proporcional a una potencia de tiempo, entonces es posible caracterizar los modos de falla de la siguiente manera:

- Un valor de $\beta < 1$ indica que la tasa de falla decrece con el tiempo
- Cuando $\beta = 0$, la tasa de falla es constante en el tiempo
- Un valor de $\beta > 1$ indica que la tasa de falla crece con el tiempo

El parámetro de escala η estira o comprime la curva de la distribución. Este parámetro proporciona una estimación de la vida característica del activo, es decir, el tiempo en el que aproximadamente el 63.2 % de los activos habrán presentado la falla [3].

Luego, la función de distribución acumulada de una variable aleatoria $X \sim$ Weibull(β, η) es:

$$F(x) = 1 - e^{\left(-\frac{x}{\eta}\right)^{\beta}} \tag{4.54}$$

En ausencia de datos históricos sobre el historial de funcionamiento del sistema, se establece de manera arbitraria que el tiempo hasta la falla t_{EOL} corresponde a 55 días (1320 horas) de operación continua, y que el factor de forma β de la función de distribución de probabilidad Weibull es igual a 2. Considerando que la función de distribución acumulada de fallas $F(t)$ de Weibull es 0 para $t = 0$ horas (es decir, la confiabilidad R es igual a 1 al inicio de la operación), y que $F(t)$ se aproxima a 1 para $t = 1320$ horas (R se aproxima a 0), se puede ajustar la curva de la ecuación 4.54, obteniendo un parámetro de escala η de aproximadamente 521 horas. Por lo tanto, bajo las suposiciones anteriores, después de 521 horas de operación continua, la probabilidad de que el sistema presente una condición *anormal* debido a algún modo de falla es aproximadamente del 63.21 %.

[4]Se refiere a un proceso completamente aleatorio, en el que la variable aleatoria sigue una distribución Weibull [103], principalmente usada en análisis de supervivencia ya que puede describir cómo cambia la probabilidad de fallo de un sistema.

Luego, se propone utilizar el valor del parámetro de escala η igual a 521 horas como el tiempo de transición t_s. Para tener en cuenta la variabilidad en la aparición del comportamiento *anormal* entre diferentes bombas y para diferentes modos de falla, se sugiere usar el valor del tiempo de transición t_s como la mediana de una distribución normal con una desviación estándar de 72 horas. Del mismo modo, se sugiere utilizar el valor del tiempo hasta la falla t_{EOL} como la mediana de una distribución normal con una desviación estándar de 72 horas para representar la variabilidad en el tiempo de vida para diferentes bombas.

Degradación Anormal. En [85], se menciona que los modelos de degradación más comúnmente utilizados en diferentes dominios de aplicación incluyen al modelo de Arrhenius, el modelo de Coffin-Manson para el crecimiento de grietas y el modelo de Eyring.

El modelo de Arrhenius se ha utilizado ampliamente para modelar una gran variedad de fallas, especialmente en aquellos que dependen de reacciones químicas, procesos de difusión o procesos de migración. Lo anterior cubre la mayoría de los modos de falla no mecánicos (o que no se relacionan con la fatiga del material) en equipos electrónicos. La ecuación de Arhhenius es:

$$t_f = A \cdot e^{\frac{\Delta H}{kT}} \tag{4.55}$$

donde:

- t_f es el tiempo hasta la falla
- T es la temperatura en el punto cuando el proceso de falla está ocurriendo
- k es la constante de Boltzmann
- A es un factor de escala
- ΔH es la energía de activación

El modelo modificado de Coffin-Mason ha sido más típicamente aplicado para el modelamiento de las fallas mecánicas, la fatiga de materiales y la deformación de lo materiales. Este modelo se ha utilizado para modelar el crecimiento de grietas en soldaduras y otros metales debido a los ciclos de temperatura repetitivos a los que se someten los equipo. La ecuación operativa del modelo es:

$$N_f = A \cdot f^{-\alpha} \cdot \Delta T^{-\beta} \cdot G(T_{max}) \tag{4.56}$$

donde:

- N_f es el número de ciclos hasta la falla
- A es un factor de escala

- f es la frecuencia de ciclado
- ΔT es el rango de temperatura durante un ciclo
- $G(T_{max})$ es un término de Arrhenius evaluado en la temperatura máxima alcanzada en cada ciclo
- α es el exponente de frecuencia de ciclado
- β es el exponente de rango de temperatura

Por otro lado, el modelo de Eyring tiene su origen en la teoría de la velocidad de reacción química[5] y su base teórica en la química y la mecánica cuántica. El modelo describe cómo varía el tiempo hasta el fallo en función de la tensión. El modelo básico incluye los efectos de la temperatura y puede ampliarse para incluir otras tensiones relevantes. El término de temperatura por sí mismo es muy similar al del modelo empírico de Arrhenius, lo cual explica el éxito de ese modelo a la hora de establecer la conexión entre el parámetro ΔH y el concepto de la teoría cuántica de "energía de activación necesaria para cruzar una barrera de energía e iniciar una reacción". El modelo de Eyring para los efectos de temperatura y tensiones relevantes adicionales tiene la siguiente expresión general:

$$t_f = A \cdot T^{\alpha} \cdot e^{\left(\frac{\Delta H}{kT} + \left(B + \frac{C}{T}\right) \cdot S_1 + \left(D + \frac{E}{T}\right) \cdot S_2\right)} \tag{4.57}$$

donde,

- t_f es el tiempo hasta la falla
- α, β, A, B, C, D y E son constantes que determinan la aceleración entre combinaciones de tensión
- S_1 y S_2 son los esfuerzos relevantes (por ejemplo, alguna función del voltaje o la corriente)
- T es la temperatura en grados Kelvin
- k es la constante de Boltzmann

Un factor en común entre todos los modelos mencionados anteriormente, es que comparten un comportamiento exponencial para describir la evolución de las fallas. Según lo anterior, para mantener el modelo de degradación relativamente simple, se propone abstraer todo el proceso de desgaste al nivel macro sin poner énfasis el nivel micro al adoptar el enfoque

[5]La teoría de la velocidad de reacción química se ocupa de cómo la velocidad de una reacción química cambia en función de las condiciones del sistema, como la concentración de los reactivos, la temperatura, y la presencia de catalizadores.

usado en [85] y [7] basado en el uso de término exponencial con una ecuación generalizada para modelar la degradación del sistema.

$$w = Ae^{B(t)} \tag{4.58}$$

Si además se asume un umbral superior de desgaste th_w, el cual denota un límite operativo más allá del cual el componente o subsistema no puede seguir siendo utilizado, la ecuación de desgaste generalizada (4.58) puede reescribirse como un índice de degradación anormal variable en el tiempo $\delta_a(t)$ al restar el término del umbral superior de desgaste th_w y normalizar la expresión con respecto a dicho umbral, obteniendo que:

$$\delta_a(t) = 1 - \frac{Ae^{B(t)}}{th_w} \tag{4.59}$$

Si se asume que $A/th_w = e^a$ y al expresar $B(t) = t^b$, entonces la expresión para la degradación anormal puede ser reescrita como:

$$\delta_a(t) = 1 - e^{at^b} \tag{4.60}$$

Luego, al considerar que al momento de la aparición de un comportamiento *anormal* en la degradación del sistema, ya existe cierto nivel previo de degradación dado por el estado de degradación inicial y posteriormente la evolución de la degradación normal del sistema, el modelo de degradación anormal que se asume para modelar el comportamiento de los parámetros B_{bomba}, A_{max} y $A_{orifice}$ es el siguiente:

$$\delta_a(t) = 1 - e^{at^b} + \delta_n(t_s) \tag{4.61}$$

La degradación inicial en la ecuación de degradación anormal (4.61) corresponde al nivel de degradación normal evaluado en t_s. Para poder definir los parámetros a y b del término exponencial en la ecuación de degradación anormal, es necesario considerar el tiempo t_{EOL} y un valor de degradación máxima o umbral superior de desgaste th_w. Al asumir un valor arbitrario para el parámetro b igual a una distribución uniforme aleatoria $U(2.0, 3.0)$, el valor del parámetro a se puede despeja según la siguiente expresión:

$$a = \frac{\ln(1 + \delta_n(t_s) - th_w)}{(t_{EOL} - t_s)^b} \tag{4.62}$$

En el caso particular del modelo de la bomba tríplex, se considera que la falla del sistema

se logra cuando se produce un bloqueo total de alguna de las válvulas de retención de entrada, cuando el área de fuga en alguno de los cilindros es la máxima posible, o bien cuando el desgaste en los rodamientos del cigüeñal de la bomba es tal que el caudal de salida cae por debajo del 66 % del caudal nominal. Bajo estas condiciones, la bomba no se alcanza los requerimientos nominales de caudal, y por ende se considera que la bomba ha fallado. Por lo tanto, los umbrales de máxima degradación th_w definidos para los parámetros B_{bomba}, A_{max} y $A_{orifice}$, son los siguientes:

- $th_w(B_{bomba}) = 0{,}067780 \quad \left[\frac{N \cdot m}{rad/s}\right]$
- $th_w(A_{max}) = 1 \times 10^{-12} \quad [m^2]$
- $th_w(A_{orifice}) = 1 \times 10^{-4} \quad [m^2]$

Estos valores se determinaron considerando las condiciones específicas de falla de la bomba. El umbral $th_w(A_{max})$ refleja una válvula de retención completamente cerrada, cuyo valor es igual a área de fuga mínima inevitable A_{leak} de la ecuación (4.34). El umbral $th_w(A_{orifice})$ se establece igual al área de máxima apertura de las válvulas de retención de la ecuación (4.34), ya que, el flujo desplazado por un émbolo no sea efectivo, el flujo a través del orificio de fuga, y por ende el área del orificio de fuga, debe ser igual al de la válvula de retención de salida. Finalmente, el valor del umbral $th_w(B_{bomba})$ se obtuvo simulando una bomba con una eficiencia de desplazamiento volumétrica del 66 %, a causa del desgaste crítico de los rodamientos del eje cigüeñal de la bomba.

4.2. Generación de Datos Sintéticos Simulados

4.2.1. Ruido de Sensores

El ruido en las mediciones es una fuente importante de variabilidad presente en los sistemas reales. Un enfoque típico para modelar el ruido de sensores consiste en añadir ruido blanco a la respuesta simulada del sistema. Dado que para este trabajo no se dispone de datos reales para caracterizar los niveles reales de ruido, se propone agregar ruido Gaussiano a las señales de salida x_s del sistema simulado.

La potencia de la señal de salida x_s se calcula de la siguiente manera:

$$P_{x_S} = \frac{1}{N} \cdot \sum_{i=1}^{N} |x_S(t)|^2 \tag{4.63}$$

Por otro lado, la potencia de la salida P_{x_S} se puede expresar en decibeles por medio de la siguiente ecuación:

$$P_{x_S,\text{db}} = 10 \cdot \log_{10}(P_{x_S}) \tag{4.64}$$

Si se conoce el valor de la razón señal-ruido (SNR), el ruido en la señal se puede modelar según:

$$P_{noise,\text{db}} = P_{x_S,\text{db}} - \text{SNR}_{\text{db}} \tag{4.65}$$

Luego, la magnitud del ruido se obtiene a partir de la operación inversa al valor de la potencia del ruido en decibeles $P_{noise,\text{db}}$, y al aplicar un distribución uniforme aleatoria:

$$noise = \text{U}(0, \sqrt{10^{(P_{noise,\text{db}}/10)}}) \tag{4.66}$$

4.2.2. Discretización de Señales

Además de la presencia de ruido en las mediciones reales de sensores, un factor importante es la consideración de que las señales registradas son de carácter discreto, es decir, los sensores capturan valores puntuales a intervalos fijos de tiempo, los cuales se usan para conocer posteriormente la señal de entrada. Por lo tanto, para considerar la discretización de la señal continua de la simulación se usa el zero-order-hold (ZOH). El ZOH es un modelo matemático de la reconstrucción práctica de la señal realizada por un convertidor digital-analógico (DAC) convencional. Es decir, describe el efecto de convertir una señal de tiempo discreto en una señal de tiempo continuo manteniendo cada valor de muestra durante un intervalo de muestra.

El ZOH reconstruye la siguiente forma de onda en tiempo continuo a partir de una secuencia de muestras x[n], suponiendo una muestra por intervalo de tiempo T:

$$x_{ZOH}(t) = \sum_{n=-\infty}^{\infty} x[n] \cdot \text{rect}\left(\frac{t - T/2 - nT}{T}\right) \tag{4.67}$$

donde la función rect $(\cdot)$ es la función rectangular, la cual se define como:

$$\text{rect}\left(\frac{t}{a}\right) = \begin{cases} 0, & \text{si } |t| > \frac{a}{2} \\ \frac{1}{2}, & \text{si } |t| = \frac{a}{2} \\ 1, & \text{si } |t| < \frac{a}{2} \end{cases} \tag{4.68}$$

4.2.3. Construcción del Conjunto de Datos

A partir de todas las ecuaciones y consideraciones descritas en la sección 4.1.1, se generaron 14 escenarios distintos que representan la simulación de la operación hasta el fallo de 14 bombas tríplex con presencia de distintos modos de falla. Dado que el modelo describe 3 tipos de degradación: 1) desgaste de los rodamientos del eje del cigüeñal de la bomba, 2) fuga de fluido en los cilindros debido al deterioro de los sellos de la junta, y 3) bloqueo en las válvulas de retención en la entrada de los cilindros, se consideran las 7 posibles combinaciones de estos 3 tipos de degradación distintos para formar los 7 modos de falla considerados. De esta manera, para cada modo de falla definido, se generaron 2 simulaciones de operación hasta el fallo de las bombas tríplex.

Para cada simulación se usaron condiciones iniciales en régimen estacionario pero distintas entre sí, y los valores iniciales de los parámetros de desgaste también se establecieron de manera aleatoria según lo descrito en el modelo de degradación de la sección 4.1.3.

La resolución temporal utilizada para el ejercicio de simulación de las 14 bombas fue igual a 0.0001 segundos, el cual demostró ser un paso de tiempo suficiente para que el sistema de ecuaciones desarrollado convergiera (Ver Apéndice 1). Luego, las variables consideradas para la construcción del conjunto de datos que representa la operación hasta el fallo de cada bomba tríplex son las que se muestran en la Tabla 4.1. Para cada variable se eligió de manera arbitraria los siguientes valores de SNR (signal-to-noise-ratio): 80, 86, 42, 56 y 76 decibeles para los sensores de corriente, velocidad angular, caudal, presión de entrada y presión de salida respectivamente. Finalmente, se aplicó el ZOH para generar las señales simuladas de sensores, con una tasa de muestreo de 1000 Hertz (0.001s).

4.3. Preprocesamiento de los Datos

El preprocesamiento de los datos de entrada constituye una etapa fundamental en el desarrollo de modelos predictivos, ya sean estos de naturaleza estadística, matemática o basados

Tabla 4.1: Variables medibles del sistema simulado de la bomba tríplex.

#	Simbolo	Descripción	Unidades
1	iMotor_meas	Corriente demandada por el motor eléctrico	A
2	wMotor_meas	Velocidad angular del rotor del motor eléctrico	rpm
3	Q_in_meas	Caudal de entrada de fluido a la bomba	lpm
4	Q_out_meas	Caudal de salida de fluido a la bomba	lpm
5	P_in_meas	Presión a la entrada de la bomba	bar
6	P_out_meas	Presión a la salida de la bomba	bar

en inteligencia artificial. Esta fase inicial es crucial para garantizar la calidad y la relevancia de los datos que alimentarán cada modelo, optimizando así su rendimiento y fiabilidad. En este trabajo, dado que se utilizan datos simulados, se ha implementado una serie de técnicas de preprocesamiento específicamente diseñadas para abordar las características particulares de estos datos. El proceso comienza con el suavizado de las señales para eliminar el ruido inherente, seguido de una generación de características tanto en el dominio del tiempo como en el de la frecuencia. Posteriormente, se lleva a cabo una cuidadosa selección de las características más significativas, se aplican técnicas de normalización y remuestreo para asegurar la consistencia y representatividad de los datos, y finalmente se redefine la variable objetivo (RUL) para que represente de mejor manera la naturaleza de los datos.

4.3.1. Suavizado de las Señales de Entrada

Al simular las condiciones de sensores reales, se introdujo deliberadamente un componente de ruido en los datos para emular las variaciones e incertidumbres inherentes a las mediciones en entornos reales, lo que permite trabajar con un conjunto de datos que se aproxima más fielmente a las situaciones que se encuentran en aplicaciones prácticas. Sin embargo, este mismo ruido puede perjudicar tremendamente el proceso de entrenamiento de modelos predictivos si no se trata correctamente.

En este trabajo se eligió la función de suavizado exponencial (4.69), el cual asigna pesos decrecientes exponencialmente a los datos pasados. Los datos más recientes son los que tienen un peso mayor en el promedio suavizado. La fórmula típica para el suavizado exponencial simple es:

$$S_t = \alpha \cdot Y_t + (1 - \alpha) \cdot S_{t-1} \tag{4.69}$$

Donde S_t es el valor suavizado en el tiempo t, X_t es el valor original en el tiempo t, y α

es el factor de suavizado $(0 < \alpha < 1)$. Este método introduce menos retraso en la serie temporal que la media móvil y es lo suficientemente flexible como para ajustarse para capturar tendencias y patrones en los datos, pero es más adecuado para datos sin una tendencia clara o un patrón estacional fuerte. Por otro lado, el método es bastante sensible al valor del factor de suavizado α, lo cual puede afectar en gran medida la calidad del suavizado. .

Para la función de suavizado exponencial se utiliza un valor de α igual a 0.3, ya que la inspección visual indicó un buen nivel de suavizado con eliminación del ruido, pero sin pérdida de la información más relevante.

4.3.2. Generación de Características

Los datos provenientes de los 6 sensores de la Tabla 4.1 para cada bomba tríplex simulada, representan muestras de 1 segundo de duración con una tasa de muestreo de 1000 Hz, por lo que cada muestra contiene 1000 valores para cada sensor. Adicionalmente, las muestras fueron tomadas a intervalos de una hora durante la operación simulada de los equipos, por lo que los datos de entrada poseen una alta dimensionalidad (6000 valores por hora de operación), lo que representa desafíos significativos para el procesamiento directo mediante algoritmos de deep learning. En este contexto, el proceso de la generación o extracción de características es fundamental, ya que este enfoque transforma los datos brutos de entrada en representaciones más informativas y manejables para los algoritmos de aprendizaje [65] en forma de características estadísticas en el dominios del tiempo como las mostradas en la Tabla 4.2, o en el dominio de la frecuencia como las mostradas en la Tabla 4.3. La importancia de este proceso radica en su capacidad para reducir la dimensionalidad, eliminar redundancias y resaltar patrones significativos, por lo que, en el contexto del deep learning, estas técnicas mejoran la eficiencia y precisión de los modelos, facilitando la generalización y mitigando el sobreajuste [30].

La aplicación de técnicas de extracción de características en los dominios del tiempo, y de la frecuencia, permite condensar la información en un conjunto más compacto de características representativas. Este proceso no solo reduce la dimensionalidad, sino que también puede revelar patrones y anomalías ocultos en los datos, por lo que, en última instancia, este enfoque facilita la detección de patrones, la identificación de anomalías y la predicción de fallos en el equipo monitoreado, transformando datos de alta dimensionalidad en una representación más informativa y eficiente.

Tabla 4.2: Características del dominio del tiempo.

No.	Característica	Definición
1	Media	$T_m = \frac{1}{n} \cdot \sum_{i=1}^{n} x_i$
2	Mediana	$T_{med} = \begin{cases} x_{\left(\frac{n+1}{2}\right)} & \text{si } n \text{ es impar} \\ \frac{x_{\left(\frac{n}{2}\right)} + x_{\left(\frac{n}{2}+1\right)}}{2} & \text{si } n \text{ es par} \end{cases}$
3	Desviación Estándar	$T_{std} = \sqrt{\frac{1}{n-1} \cdot \sum_{i=1}^{n} (x_i - \overline{x})^2}$
4	Varianza	$T_{var} = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \overline{x})^2$
5	Máximo	$T_{max} = \max(x)$
6	Mínimo	$T_{min} = \min(x)$
7	Valor Peak	$T_{peak} = \max(\lvert x \rvert)$
8	Valor RMS	$T_{rms} = \sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} x_i^2}$
9	Curtosis	$T_{kur} = \frac{\sum_{i=1}^{n} (x_i - \overline{x})^4}{(n-1) \cdot T_{std}^4}$
10	Asimetría	$T_{skew} = \frac{\sum_{i=1}^{n} (x_i - \overline{x})^3}{(n-1) \cdot T_{std}^3}$
11	Crest factor	$T_{cf} = \frac{T_{peak}}{T_{rms}}$
12	Shape factor	$T_{sf} = \frac{T_{rms}}{\lvert \overline{x} \rvert}$
13	Impulse factor	$T_{if} = \frac{T_{peak}}{\lvert \overline{x} \rvert}$
14	Clearence factor	$T_{clf} = \frac{T_{peak}}{\left(\frac{1}{n} \cdot \sum_{i=1}^{n} \lvert x_i \rvert^{1/2}\right)^2}$

En la Tabla 4.2, x_i es una serie temporal de señales para $i = 1, 2, \ldots, n$, donde n es el número de puntos de datos, y $\overline{x} = \frac{1}{n} \cdot \sum_{i=1}^{n} x_i$ es el promedio de la distribución de datos.

Adicionalmente se considera la pendiente (Slope), y las primera y segunda derivadas de las distribución de datos en el dominio del tiempo, ya que estas características adicionales aportan información sobre tendencias, tasas de cambio y aceleraciones en las señales de los sensores, enriqueciendo el conjunto de características y potencialmente mejorando la capacidad de los modelos para capturar patrones complejos y eventos significativos en la operación del equipo.

Tabla 4.3: Características del dominio de la frecuencia.

No.	Característica	Definición
1	Centro de frecuencia	$F_{fc} = \frac{\sum_{i=1}^{N} f_i \cdot p_i}{\sum_{i=1}^{N} p_i}$
2	RMS de frecuencia	$F_{rms} = \sqrt{\frac{\sum_{i=1}^{N} p_i^2 \cdot f_i^2}{\sum_{i=1}^{N} p_i^2}}$
3	Desv. estándar de frecuencia	$F_{std} = \sqrt{\frac{\sum_{i=1}^{N} p_i \cdot (f_i - F_{fc})^2}{\sum_{i=1}^{N} p_i}}$
4	Relación RSPPB	$\text{RSPPB} = \frac{\max(\lvert p_i \rvert)}{\overline{p}}$
5	Curtosis espectral	$P_{kur} = \frac{\sum_{i=1}^{N} (p_i - \overline{p})^4}{(N-1) \cdot \left(\sqrt{\frac{1}{N-1} \cdot \sum_{i=1}^{N} (p_i - \overline{p})^2} \right)^4}$
6	Asimetría espectral	$P_{skew} = \frac{\sum_{i=1}^{N} (p_i - \overline{p})^3}{(N-1) \cdot \left(\sqrt{\frac{1}{N-1} \cdot \sum_{i=1}^{N} (p_i - \overline{p})^2} \right)^3}$

En la Tabla 4.3, p_i es el espectro de potencia de $x_i(t)$, $i = 1, 2, \ldots, N$, N es el número de líneas del espectro, y f_i es el valor de frecuencia de la i-ésima línea del espectro. $\overline{p} = \frac{1}{N} \cdot \sum_{i=1}^{N} p_i$ puede indicar la energía de vibración en el dominio de la frecuencia, F_{fc} y F_{rms} describen el cambio de posición de la frecuencia principal, y F_{std} puede mostrar el grado de centralización o descentralización de la energía del espectro de potencia.

Por lo tanto, a partir de cada muestra obtenido de los 6 sensores iniciales, se generan 23 características por sensor, transformando una muestra de 1000×6, en una de $1 \times (23 \times 6)$.

4.3.3. Selección de Características

No todas las características generadas en la sección 4.3.2 aportan igual cantidad de información o guardan relación con la variable objetivo RUL, por lo que es necesario hacer una selección de las características generadas más importantes y relevantes.

Entre los métodos más efectivos para la selección de características se encuentran los basados en árboles, como Random Forest, ExtraTrees y Gradient Boosting. Estos métodos se destacan por varias razones. Son robustos y manejan bien datos no lineales y con interacciones complejas. Al ser no paramétricos, no asumen una distribución específica de los datos, lo que los hace versátiles en diversos escenarios. Tienen la capacidad de capturar interacciones importantes entre características, lo que es crucial en muchos problemas del mundo real. Además, son menos sensibles a outliers y ruido en los datos, proporcionando así una mayor estabilidad en sus resultados.

Para este trabajo la selección de características se realizó en base a los métodos Random Forest, ExtraTreesRegressor y Gradient Boosting. Para cada método se utilizaron 100 estimadores, y luego se seleccionaron sólo las características que explican el 95 % de la importancia para cada método. Luego, se eligen sólo las características que aparecen en al menos dos de las tres listas de características generadas (una por cada método), para obtener una lista de 16 características. Finalmente se seleccionan las primeras 10 características de la lista obtenida a partir de los métodos Random Forest, ExtraTreesRegressor y Gradient Boosting.

4.3.4. Normalización

La normalización de datos es un paso crítico en el preprocesamiento, especialmente cuando se trabaja con modelos basados en redes neuronales. Su importancia radica en dos aspectos fundamentales que afectan el rendimiento y la eficacia del modelo. En primer lugar, la normalización aborda el problema de la disparidad de escalas entre las diferentes características de entrada. Esta disparidad puede tener un impacto significativo en el comportamiento de las funciones de activación dentro de las redes neuronales. En segundo lugar, la normalización mejora el proceso de optimización del modelo, ya que cuando las características tienen es-

calas diversas, la función de costo a minimizar también reflejará estas diferencias de escala. Esto puede afectar negativamente el algoritmo de gradiente descendiente, haciendo que converja más lentamente o incluso que tenga dificultades para encontrar el mínimo global. Por lo tanto, al normalizar los datos, se reduce la variabilidad en las distancias dentro del espacio de características, lo que generalmente resulta en una optimización más eficiente y efectiva.

Para abordar estos desafíos, en este trabajo se utilizó la normalización MinMax. Esta técnica realiza una transformación lineal de cada característica, mapeando sus valores a un rango específico, típicamente [0, 1]. La fórmula utilizada para esta normalización es:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \cdot (\max(x') - \min(x')) + \min(x') \tag{4.70}$$

Donde x' es el valor normalizado, x es el valor original, y $min(x)$ y $max(x)$ son los valores mínimo y máximo de la característica respectivamente.

Luego, al considerar el rango de [min(x'), max(x')] = [0, 1], la ecuación anterior queda expresada como:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{4.71}$$

4.3.5. Remuestreo con Ventana Deslizante

Los conjuntos de datos generados según lo descrito en la sección 4.2.3 tienen la forma (N° Muestras, N° Características), dónde N° Muestras corresponde a las filas y N° Características corresponde a las columnas del conjunto de datos generado. Luego en las secciones 4.3.2 y 4.3.3 se generan y seleccionan nuevas características a partir de los datos de sensores inicialmente considerados respectivamente. Pero luego de estos pasos los conjuntos de datos mantienen esencialmente la forma (N° Muestras, N° Características). Se requiere modificar la forma de los datos debido a que a menudo los modelos de deep utilizan capas cuya entrada debe estar en un formato de mayor dimensión, como por ejemplo las capas LSTM y Convolucionales 1D, donde su entrada debe estar en la forma (batch, timesteps, features) [35].

Para abordar esta discrepancia de formato, se puede implementar la técnica de ventanas deslizantes (sliding windows) para el remuestreo de los datos en la forma requerida. El enfoque de ventanas deslizantes consiste en segmentar la serie temporal en intervalos super-

puestos, donde cada segmento representa una muestra de entrada para el modelo. La técnica de ventanas deslizantes es ampliamente utilizada en el análisis de series temporales y deep learning debido a su capacidad para capturar patrones temporales y reducir la dimensionalidad de los datos [59]. La ecuación que describe la creación de una ventana temporal se puede expresar como:

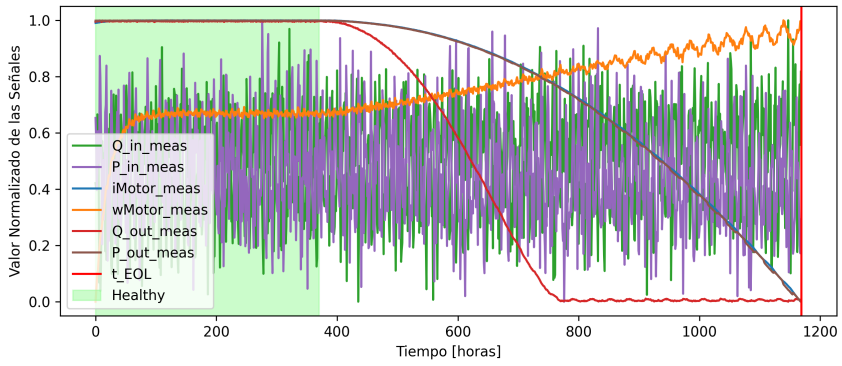
$$W_i = [X_{t-w+1}, X_{t-w+2}, ..., X_t] \tag{4.72}$$

donde W_i representa la i-ésima ventana, X_t es el vector de características en el tiempo t, y w es el tamaño de la ventana. Este proceso se repite para cada punto temporal, generando un conjunto de datos remuestreado que preserva la estructura temporal de la serie original mientras se adapta a los requerimientos de entrada de los modelos de deep learning mencionados anteriormente.

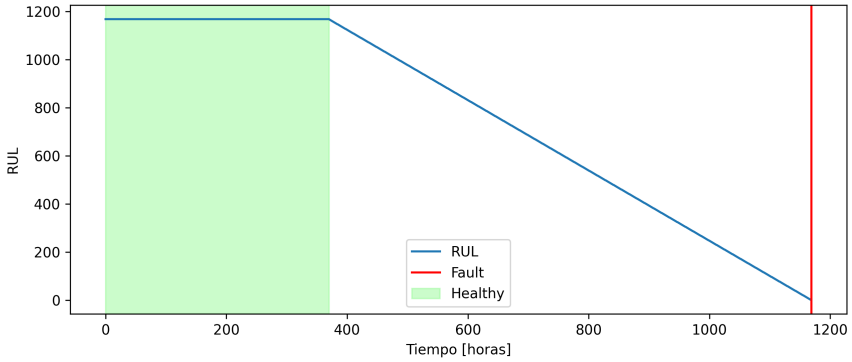
Ahora bien, luego del proceso de generación de características (sec. 4.3.2), cada muestra representa las condiciones de los equipos para cada hora de su ciclo de vida, de modo que, al seleccionar una ventana temporal de tamaño 5, se agrupan 5 muestras que representan el estado de salud de cada bomba durante 5 horas de operación. En este trabajo se utiliza un tamaño de ventana deslizante igual a 24, de modo que cada ventana recoge y extrae patrones en un marco temporal de 24 horas para cada bomba.

4.3.6. Definición de Función Objetivo

Los conjuntos de datos generados según lo descrito en la sección 4.2.3 consideran que la variable objetivo RUL decrece de manera lineal con el tiempo, de modo que todas las filas que corresponden a una misma muestra tienen el mismo valor de RUL = RUL_i, pero los datos de la siguiente muestra tienen un valor de RUL = $\text{RUL}_i - 1$, ya que la configuración de la simulación considera la toma de una muestra cada 1 hora de operación. Sin embargo, según Li et al. [53], en aplicaciones prácticas, la degradación de los equipos durante las etapas iniciales de operación suele ser poco evidente. En la Figura 4.6(a) se ve como las señales que no muestran una tendencia significativa dentro de un rango considerado saludable para el equipo. Para abordar esta discrepancia, en [53], los autores propusieron un enfoque más refinado utilizando una función lineal por partes (Fig. 4.6(b)).



(a)



(b)

Figura 4.6: Unidad B002 del conjunto de datos simulado: (a) Datos de los 6 sensores normalizados, y (b)Función lineal por partes del RUL.

Esta representación más precisa del RUL permite una mejor planificación del mantenimiento y una gestión más eficiente de la vida útil del equipo, superando las limitaciones de la simple aproximación lineal comúnmente utilizada.

4.4. División de Datos y Definición de Escenarios Experimentales

Para entrenar modelos predictivos que sean capaces de estimar el RUL de los activos de manera precisa y confiable, es necesario separar los datos disponibles en conjuntos de entrenamiento, validación y prueba. En la sección 4.3 se le aplicó un preprocesamiento a los datos sintéticos generados en la sección 4.2, los cuales representan las trayectorias de

degradación hasta el fallo de un grupo simulado de 14 bombas tríplex. Como se menciona en la sección 4.2.3, entre las 14 bombas simuladas existen 7 modos de falla distintos, donde cada modo de falla está representado por dos bombas diferentes en el conjunto de datos simulados. Considerando lo anterior, se propone la generación de dos escenarios distintos, para entrenar y evaluar el desempeño de los distintos modelos de deep learning. La separación de los datos en dos escenarios distintos en lugar de usar las 14 bombas en un único escenario se fundamenta en la intención de evaluar de manera más rigurosa y exhaustiva la capacidad de generalización y la robustez de los distintos modelos.

4.4.1. Justificación de la Estrategia de División de los Datos

- Al separar los datos en dos escenarios distintos, los modelos predictivos pueden ser evaluados en diferentes condiciones, asegurando que se pruebe tanto su capacidad para generalizar dentro de modos de falla conocidos, como su capacidad para adaptarse a modos de falla completamente nuevos y no vistos durante el proceso de entrenamiento.

- Entrenar y evaluar en un único escenario podría conducir a una evaluación sesgada, donde los modelos predictivos pueden estar sobre ajustado a los datos específicos de dicho escenario. La separación en dos escenarios distintos mitiga este riesgo al incluir múltiples condiciones y configuraciones de falla.

4.4.2. Definición de Escenarios Experimentales

Escenario 1: Generalización a Modos de Falla No Vistos

En el caso del escenario 1, se propone una generalización a nuevos modos de falla no vistos durante el entrenamiento, por lo que para el conjunto de datos de entrenamiento se seleccionan 8 bombas, representando 4 modos de falla diferentes (2 bombas por modo de falla), mientras que para el conjunto de datos de validación se seleccionan 2 bombas, representando un único modo de falla, y para el conjunto de datos de prueba se seleccionan 4 bombas, representando 2 modos de falla diferentes (2 bombas por modo de falla). Por lo tanto, en el escenario 1, las bombas B001, B003, B004, B006, B007, B010, B012 y B013 se usaron como el conjunto de datos de entrenamiento, mientras que las bombas B002 y B011 se usaron como el conjunto de datos de validación, y las bombas B005, B008, B009 y B014 se usaron como el conjunto de datos de prueba. En este escenario se evalúa la capacidad de los modelos para

generalizar a modos de falla completamente nuevos, donde el conjunto de datos validación ayuda para que los modelos no se sobre ajusten a los modos de falla específicos del conjunto de datos entrenamiento. El conjunto de prueba evalúa cómo los modelos se comportan con modos de falla que no estuvieron presentes en el proceso de entrenamiento.

Escenario 2: Generalización dentro de Modos de Falla Conocidos

Para el escenario 2 se propone una generalización dentro de todos los modos de falla conocidos, por lo que para el conjunto de datos de entrenamiento se seleccionan 7 bombas, cada una representando uno de los modos de falla descritos, mientras que para el conjunto de datos de validación se seleccionan 3 bombas que representen 3 de los 7 modos de falla diferentes. Por último, el conjunto de datos de prueba se configura con 4 bombas representando los 4 modos de falla restantes. Por lo tanto, en el escenario 2, las bombas B001, B002, B003, B004, B005, B006 y B009 se usaron como el conjunto de datos de entrenamiento, mientras que las bombas B010, B011 y B012 se usaron como el conjunto de datos de validación, y las bombas B007, B008, B013 y B014 se usaron como el conjunto de datos de prueba. Esta separación permite que los modelos vean ejemplos de todos los modos de falla durante el entrenamiento, donde el conjunto de datos de validación ayuda a ajustar los modelos para generalizar entre diferentes bombas con el mismo modo de falla y el conjunto de datos de prueba evalúa cómo los modelos generalizan a datos de bombas no vistas durante el entrenamiento, pero que presentan modos de falla conocidos.

La Tabla 4.4 muestra la separación de las distintas bombas en los conjuntos de datos de entrenamiento, validación y prueba para cada uno de los escenarios descritos en los párrafos anteriores.

Tabla 4.4: Configuración de unidades en los conjuntos de entrenamiento, validación y prueba en los escenarios 1 y 2.

Escenario	Entrenamiento	Validación	Prueba	Escenario	Entrenamiento	Validación	Prueba
1	B001, B003	B002	B005	2	B001, B002	B010	B007
	B004, B006	B011	B008		B003, B004	B011	B008
	B007, B010		B009		B005, B006	B012	B013
	B012, B013		B014		B009		B014

4.5. Desarrollo de los Modelos Predictivos

Como se mencionó en el Estado del Arte y en el Marco Teórico, las redes neuronales convolucionales (CNN) y las redes de memoria a corto y largo plazo (LSTM) presentan ventajas específicas en el manejo de diferentes tipos de datos. Las CNN son especialmente efectivas para el reconocimiento de patrones espaciales en datos estructurados debido a su capacidad para detectar características locales y su invariancia a traslaciones y escalas [48] [77]. Por otro lado, las LSTM sobresalen en la modelación de datos secuenciales y series temporales, ya que su arquitectura les permite capturar dependencias a largo plazo y mitigar problemas de gradientes desvanecientes y explosivos [35] [29]. Cada una de estas arquitecturas tiene sus propios puntos fuertes y débiles: mientras que las CNN pueden tener dificultades para manejar datos secuenciales debido a su naturaleza estática, las LSTM pueden ser menos eficientes en la extracción de características locales detalladas. Por lo tanto, dada la complementariedad en el enfoque individual de estas arquitecturas, en este trabajo se propone un análisis comparativo de cuatro configuraciones diferentes de redes neuronales.

La primera configuración es una red CNN simple, diseñada principalmente para la extracción de características locales a partir de datos secuenciales como series temporales. La segunda configuración presenta una red LSTM diseñada para para procesar y capturar dependencias a largo plazo en series temporales. La tercera configuración es una combinación de las redes CNN y LSTM en serie, tomando como inspiración los trabajos de Niu et al. [67] y Zhang et al. [111], pero simplificando las configuraciones al usar redes CNN y LSTM de manera secuencial. La última es una configuración más compleja basada principalmente en la arquitectura propuesta por Li et al. [53], donde los datos se analizan en caminos paralelos por las redes CNN y LSTM. Todos estos modelos se desarrollaron utilizando la API de Keras debido a su gran flexibilidad y facilidad de uso [19].

4.5.1. Modelo CNN

El modelo CNN, cuya representación esquemática se observa en la Figura 4.7, corresponde a una red convolucional simple, cuya configuración comienza con una capa de entrada que acepta datos con una forma específica definida por “input_shape”. A continuación, se aplican tres capas convolucionales consecutivas: la primera capa convolucional tiene 32 filtros con un tamaño de kernel de 3, función de activación ReLU y un padding “causal”, lo que resulta en convoluciones causales (dilatadas) donde la salida[t] no depende de la entrada[t+1:]. Esto

es útil para modelar datos temporales donde no se debe violar el orden temporal [97]. La segunda capa convolucional tiene 64 filtros con un tamaño de kernel de 3, función de activación ReLU y un padding causal; y la tercera capa convolucional tiene 128 filtros con un tamaño de kernel de 5, función de activación ReLU y un padding causal. Estas capas convolucionales son responsables de la extracción de características locales de la secuencia de entrada, permitiendo la detección de patrones relevantes a diferentes escalas. La salida de estas capas se aplanan mediante una capa flatten, que convierte las salidas bidimensionales en una única dimensión. Luego, se incluye una capa de regularización Dropout con una tasa del 50 %, que se utiliza para prevenir el sobreajuste al desactivar aleatoriamente un porcentaje de neuronas durante el entrenamiento. Finalmente, el modelo culmina con una capa densa que tiene una única neurona con una función de activación lineal, proporcionando una única predicción continua adecuada para problemas de regresión como la estimación del RUL.

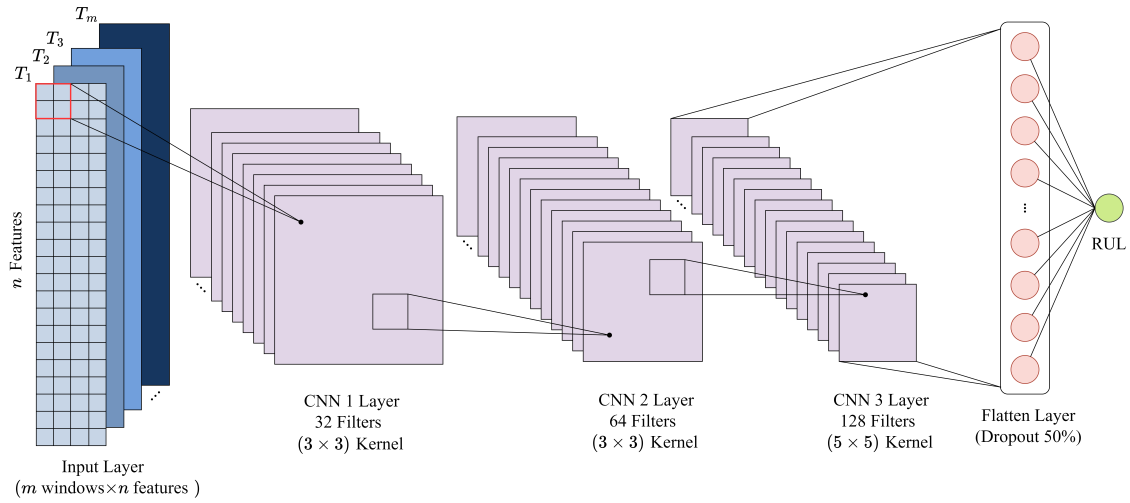


Figura 4.7: Diagrama esquemático de la arquitectura del modelo CNN.

4.5.2. Modelo LSTM

El modelo LSTM cuya representación esquemática se observa en la Figura 4.5.2, presenta una red neuronal de memoria a corto y largo plazo, cuya configuración comienza con una capa de entrada que acepta datos con una forma específica definida por "input_shape". A continuación, se aplican dos capas LSTM consecutivas: la primera capa LSTM tiene 64 unidades, mientras que la segunda tiene 128 unidades. Ambas capas LSTM utilizan la función de

activación tangente hiperbólica (tanh) para la salida y la función de activación sigmoide para las puertas recurrentes. Además, ambas capas LSTM están configuradas para que devuelvan la secuencia completa de salidas en cada paso de tiempo, permitiendo que la siguiente capa LSTM procese toda la secuencia de datos. Después de las capas LSTM, se aplica una capa de aplanamiento flatten para convertir la salida bidimensional de la última capa LSTM en un vector unidimensional. Luego, se incluye una capa de regularización Dropout con una tasa del 50 %, que se utiliza para prevenir el sobreajuste al desactivar aleatoriamente la mitad de las neuronas durante el entrenamiento, mejorando así la capacidad de generalización del modelo. Finalmente, el modelo culmina con una capa densa que tiene una única neurona con una función de activación lineal, proporcionando una única predicción continua.

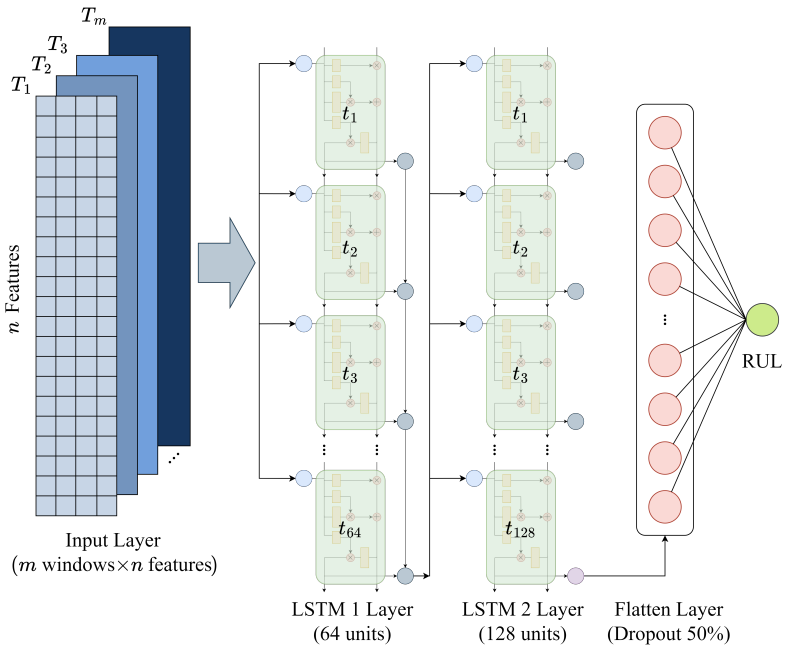


Figura 4.8: Diagrama esquemático de la arquitectura del modelo LSTM.

4.5.3. Modelo CNN-LSTM

La tercera configuración, cuya representación esquemática se observa en la Figura 4.9, presenta una red neuronal híbrida que combina capas CNN y capas LSTM, diseñada para aprovechar las ventajas de ambas arquitecturas en el procesamiento de datos secuenciales y temporales. Esta configuración en principio, es capaz de extraer características locales y capturar dependencias a largo plazo en series temporales, lo que la hace adecuada para tareas

como la predicción del RUL.

La arquitectura comienza con una capa de entrada que acepta datos con una forma específica definida por "input_shape". A continuación, se aplican tres capas convolucionales consecutivas: la primera capa convolucional tiene 32 filtros con un tamaño de kernel de 3, la segunda tiene 64 filtros con un tamaño de kernel de 3, y la tercera tiene 128 filtros con un tamaño de kernel de 5. Todas estas capas convolucionales utilizan la función de activación ReLU y un padding causal. Estas capas convolucionales son responsables de la extracción de características locales de la secuencia de entrada. Después de las capas convolucionales, se aplican dos capas LSTM consecutivas: la primera capa LSTM tiene 64 unidades, mientras que la segunda tiene 128 unidades. Ambas capas LSTM utilizan la función de activación tanh para la salida y la función de activación sigmoide para las puertas recurrentes. Además, ambas capas LSTM están configuradas para que devuelvan la secuencia completa de salidas en cada paso de tiempo. Estas capas LSTM permiten al modelo capturar dependencias a largo plazo en los datos procesados por las capas convolucionales. Tras las capas LSTM, se aplica una capa de aplanamiento (flatten) para convertir la salida multidimensional en un vector unidimensional. Luego, se incluye una capa de regularización Dropout con una tasa del 50 %, que se utiliza para prevenir el sobreajuste al desactivar aleatoriamente la mitad de las neuronas durante el entrenamiento, mejorando así la capacidad de generalización del modelo. Finalmente, el modelo culmina con una capa densa que tiene una única neurona con una función de activación lineal, proporcionando una única predicción continua.

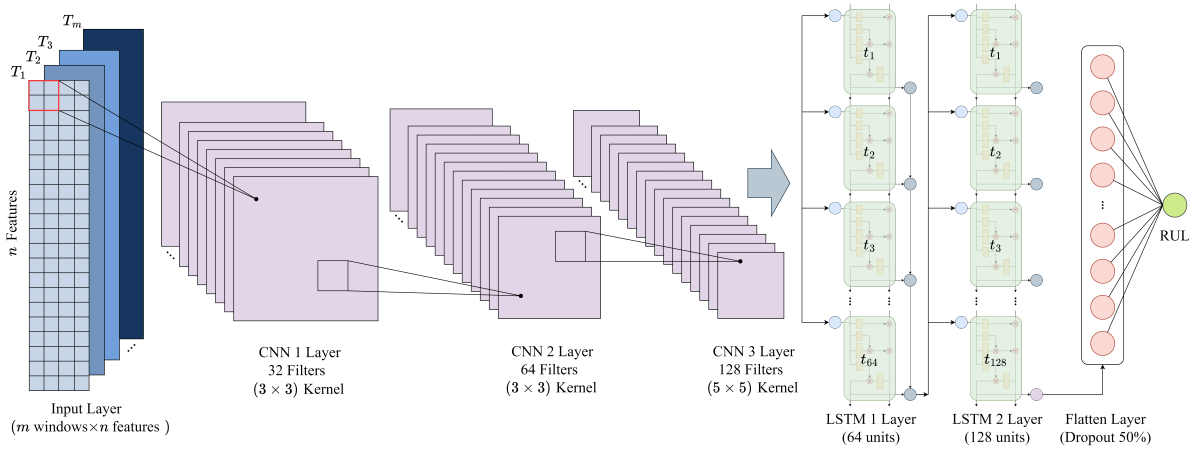


Figura 4.9: Diagrama esquemático de la arquitectura del modelo CNN-LSTM.

4.5.4. Modelo CNN-LSTM-p

La cuarta configuración, cuya representación esquemática se observa en la Figura 4.10, es la más compleja de todas, ya que presenta una red neuronal híbrida paralela que combina capas CNN y capas LSTM en rutas separadas, diseñada para aprovechar las fortalezas de ambas arquitecturas en el procesamiento de datos secuenciales y temporales.

Al igual que en el resto de configuraciones, esta arquitectura comienza con una capa de entrada que acepta datos con una forma específica definida por "input_shape". A partir de esta entrada, la red se divide en dos rutas paralelas: La primera ruta, denominada ruta CNN, aplica tres capas convolucionales consecutivas: la primera capa convolucional tiene 32 filtros con un tamaño de kernel de 3, la segunda tiene 64 filtros con un tamaño de kernel de 3, y la tercera tiene 128 filtros con un tamaño de kernel de 5. Todas estas capas convolucionales utilizan la función de activación ReLU y un padding causal. Estas capas convolucionales son responsables de la extracción de características locales de la secuencia de entrada. La segunda ruta, denominada ruta LSTM, aplica dos capas LSTM consecutivas directamente a la entrada: la primera capa LSTM tiene 64 unidades, mientras que la segunda tiene 128 unidades. Ambas capas LSTM utilizan la función de activación tanh para la salida y la función de activación sigmoide para las puertas recurrentes. Además, ambas capas LSTM están configuradas para que devuelvan la secuencia completa de salidas en cada paso de tiempo. Estas capas LSTM permiten al modelo capturar dependencias a largo plazo en los datos de entrada.

Después de estas rutas paralelas, las salidas de ambas rutas se combinan mediante una capa de suma (Add), fusionando así las características extraídas por las capas convolucionales y las capas LSTM. Esta salida combinada se procesa a través de una capa LSTM adicional con 64 unidades, que ayuda a integrar la información de ambas rutas. Tras esta capa LSTM final, se aplica una capa flatten para convertir la salida multidimensional en un vector unidimensional. Luego, se incluye una capa de regularización Dropout con una tasa del 50 %, que se utiliza para prevenir el sobreajuste al desactivar aleatoriamente la mitad de las neuronas durante el entrenamiento, mejorando así la capacidad de generalización del modelo. Finalmente, el modelo culmina con una capa densa que tiene una única neurona con una función de activación lineal, proporcionando una única predicción continua.

Esta arquitectura paralela CNN-LSTM permite al modelo beneficiarse simultáneamente de la capacidad de las CNNs para extraer características locales y de la habilidad de las

LSTMs para capturar dependencias a largo plazo, potencialmente mejorando su rendimiento en tareas complejas de procesamiento de series temporales.

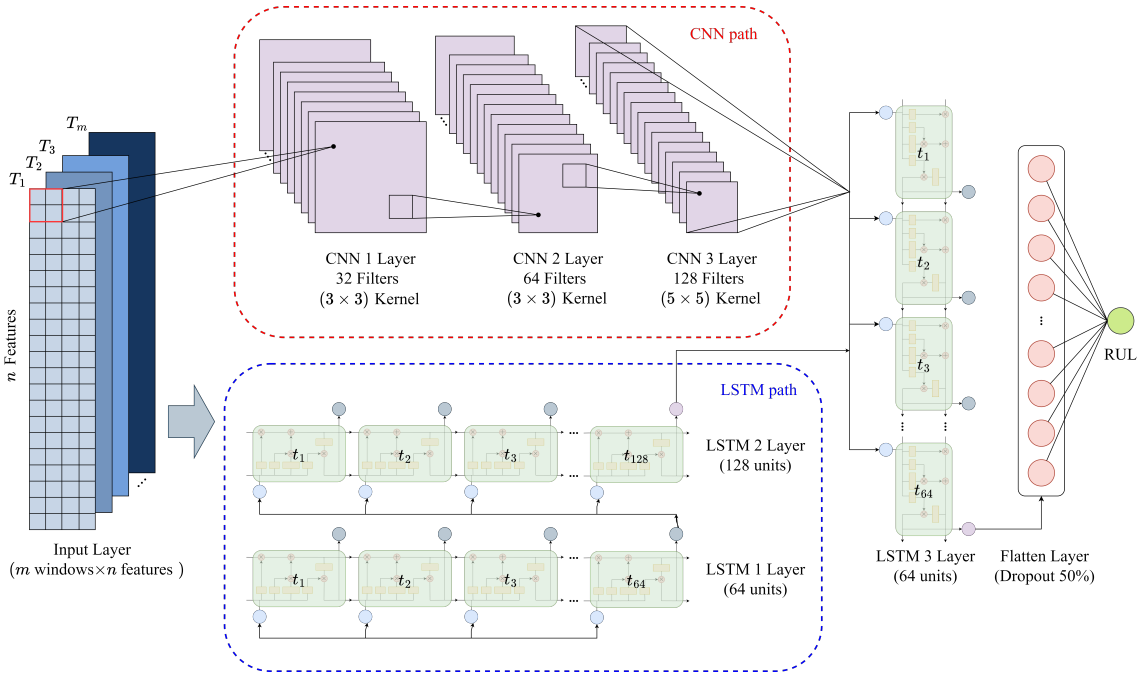


Figura 4.10: Diagrama esquemático de la arquitectura del modelo CNN-LSTM-p.

4.5.5. Dropout

El dropout es una técnica de regularización que consiste en desactivar aleatoriamente una fracción de las unidades (neuronas) de una capa durante el proceso de entrenamiento. Esto fuerza al modelo a no depender de unidades específicas, promoviendo una representación más robusta y generalizable de los datos.

En las arquitecturas presentadas, se aplicó un dropout con una tasa $p = 0.5$ después de la capa de Flatten en cada modelo. La elección de este valor se basa en su efectividad comprobada en la literatura para reducir el sobreajuste en tareas de predicción de series temporales y en modelos con múltiples capas profundas [94]. Esta implementación asegura que cada unidad tiene una probabilidad del 50 % de ser desactivada en cada iteración del entrenamiento, lo que contribuye significativamente a la robustez del modelo.

4.5.6. Optimizadores

El proceso de optimización de los modelos se centra en la selección de optimizadores, funciones de pérdida y esquemas de ajuste de la tasa de aprendizaje para asegurar una convergencia eficiente y efectiva.

Los optimizadores constituyen el mecanismo por el cual las redes neuronales ajustan sus parámetros para minimizar la función de pérdida. Para la optimización de los modelos, se empleó el optimizador Adam, el cual combina las ventajas de dos métodos de optimización populares: AdaGrad y RMSProp, ajustando los parámetros de aprendizaje adaptativamente para cada parámetro. Este optimizador es especialmente efectivo en problemas de aprendizaje profundo debido a su capacidad para manejar grandes cantidades de datos y altos niveles de ruido [43].

4.5.7. Funciones de Pérdida

Las funciones de pérdida cuantifican la discrepancia entre las predicciones de los modelos y los valores reales, guiando así el proceso de optimización. La función de pérdida seleccionada para entrenar los modelos es el Mean Squared Error (MSE). Esta función mide el promedio de los errores al cuadrado entre las predicciones del modelo y los valores reales, penalizando fuertemente los errores grandes. El MSE es adecuado para tareas de regresión como la predicción del RUL debido a su interpretación clara y su relación directa con el rendimiento del modelo [15].

4.5.8. Learning Rate Scheduling

Se implementó el callback ReduceLROnPlateau para ajustar la tasa de aprendizaje durante el entrenamiento. Este callback reduce la tasa de aprendizaje en un factor de 0.2 si no se observa una mejora en la métrica de validación (val_loss) durante 25 épocas consecutivas, con una tasa de aprendizaje mínima establecida en 1×10^{-6}. Esta técnica es fundamental para evitar que el modelo se estanque en mínimos locales y asegurar una convergencia más suave y efectiva.

4.6. Entrenamiento de los Modelos Predictivos

El proceso de entrenamiento de los modelos se diseñó para maximizar la eficiencia y la efectividad, incorporando diversas técnicas para monitorear y mejorar el rendimiento durante el entrenamiento.

4.6.1. Tamaño de Lote

Se utilizó un tamaño de lote de 128. La elección de este valor se basa en un balance entre la estabilidad del gradiente y la eficiencia computacional[60]. Un tamaño de lote moderado permite aprovechar las capacidades de paralelización de los procesadores modernos, a la vez que proporciona una estimación más estable del gradiente, lo que puede conducir a una mejor convergencia del modelo.

4.6.2. Número de Épocas

Una época representa una iteración o pasada completa a través de todo el conjunto de datos de entrenamiento. El número de épocas influye directamente en cuánto tiempo se entrena el modelo y puede afectar su capacidad de generalización [28]. El número de épocas se fijó en 250, permitiendo al modelo iterar suficientemente sobre los datos de entrenamiento para aprender patrones complejos sin riesgo significativo de sobreajuste, especialmente considerando la regularización aplicada. La monitorización continua del rendimiento de validación garantiza que el proceso de entrenamiento se detenga antes si se alcanza una convergencia adecuada antes del límite de épocas.

4.6.3. Callbacks

Se utilizaron varios callbacks durante el entrenamiento:

- **ModelCheckpoint:** Este callback guarda el mejor modelo basado en la métrica de validación (val_loss), asegurando que siempre se retiene la versión del modelo con mejor desempeño en el conjunto de datos de validación [96].

- **CSVLogger:** Registra las métricas de entrenamiento y validación en un archivo CSV para facilitar el análisis posterior.

- **ReduceLROnPlateau:** Ajusta la tasa de aprendizaje en respuesta a la falta de mejora en la métrica de validación. Esto permite que el modelo ajuste sus pesos de manera más fina a medida que se acerca a un óptimo [91].

4.7. Evaluación de los Modelos Predictivos

4.7.1. Métricas de Evaluación

Para cuantificar el desempeño y la precisión de la estimación del RUL de los distintos modelos de deep learning, se definen las siguientes métricas:

1. **Error Absoluto Medio (MAE):** Corresponde a la media de la diferencia absoluta entre los valores predichos y los valores reales. El MAE es útil porque proporciona una medida directa de la magnitud media del error en las predicciones, sin considerar la dirección del error. En la ecuación 4.73, un menor valor indica un mejor desempeño en la predicción del RUL.

$$(\text{MAE})_{\text{RUL}} = \frac{1}{N} \cdot \sum_{i=1}^{N} \left| \text{RUL}_i - \widehat{\text{RUL}}_i \right| \tag{4.73}$$

   El valor del MAE se encuentra en la misma magnitud de la variable objetivo, y su valor se puede interpretar como la distancia absoluta promedio entre el valor real y el valor predicho.

2. **Raíz del Error Cuadrático Medio (RMSE):** Corresponde a la raíz cuadrada del error cuadrático medio o MSE. EL RMSE es útil para interpretar el error de predicción en las mismas unidades que la variable objetivo. En la ecuación 4.74, un menor valor indica un desempeño superior en la predicción del RUL.

$$(\text{RMSE})_{\text{RUL}} = \sqrt{\frac{1}{N} \cdot \sum_{i=1}^{N} \left(\text{RUL}_i - \widehat{\text{RUL}}_i \right)^2} \tag{4.74}$$

   La diferencia principal entre el MAE y el RMSE radica en cómo tratan a los errores de diferente magnitud; el MAE trata a todos los errores de manera lineal, sin importar su

magnitud, mientras que el RMSE penaliza más los errores grandes debido al término cuadrático en su cálculo. Debido a lo anterior, el valor del RMSE siempre es igual o mayor que el MAE para el mismo conjunto de predicciones. Además, la diferencia entre MAE y RMSE sirve como indicador de la variabilidad en la magnitud de los errores, ya que una gran diferencia sugiere que hay presencia de algunos valores extremos muy grandes en las predicciones, y una pequeña diferencia indica que los errores son más consistentes en magnitud en toda la distribución.

3. **Error Porcentual Absoluto Medio (MAPE):** El error porcentual absoluto medio proporciona una forma significativa de evaluar la precisión de la predicción, sobre todo cuando es crucial comprender el error relativo, ya que esto permite comparar el desempeño de modelos en diferentes conjuntos de datos o escalas, lo que no siempre es posible con el MAE o el RMSE. En la ecuación 4.75, un menor valor indica un desempeño superior en la predicción del RUL.

$$(\text{MAPE})_{\text{RUL}} = \frac{1}{N} \cdot \sum \left| \frac{\text{RUL}_i - \widehat{\text{RUL}}_i}{\text{RUL}_i} \right| \times 100\,\% \tag{4.75}$$

   El MAPE es fácilmente interpretable, ya que expresa el error como un porcentaje del valor real. Sin embargo, el MAPE puede ser problemático cuando los valores reales son cercanos a cero, ya que puede resultar en porcentajes de error muy grandes o indefinidos.

4. **Coeficiente de Determinación ($\mathbf{R^2}$):** El coeficiente de determinación es una medida estadística que indica la proporción de la variabilidad en la variable dependiente que es explicada por el modelo. En la ecuación 4.76, un valor de R^2 más cercano a 1 indica un desempeño superior en el ajuste del modelo para la predicción del RUL.

$$\left(\text{R}^2\right)_{\text{RUL}} = 1 - \frac{\sum_{i=1}^{N} \left(\text{RUL}_i - \widehat{\text{RUL}}_i\right)^2}{\sum_{i=1}^{N} \left(\text{RUL}_i - \overline{\text{RUL}}\right)^2} \tag{4.76}$$

   donde $\overline{\text{RUL}}$ es la media de los valores reales.

Las métricas de las ecuaciones 4.73 hasta 4.76 son usadas de manera frecuente para evaluar el rendimiento y la precisión de modelos predictivos en tareas de regresión. En estas

ecuaciones, N representa el tamaño de los datos, $\widehat{\text{RUL}}_i$ es la vida útil remanente predicha por los modelos, y RUL_i es la vida útil remanente real.

4.7.2. Tablas y Figuras

Para cada uno de los escenarios definidos en la sección 4.4, se realizaron las siguientes evaluaciones:

- **Tabulación de Resultados:**
  Se presentan tablas que muestran los resultados de predicción utilizando las métricas de evaluación definidas en la sección anterior (MAE, RMSE, MAPE y R^2). Estas tablas comparan el desempeño de los modelos en los conjuntos de entrenamiento y prueba. Esta comparación permite observar cómo varía el desempeño de los modelos al pasar de datos de entrenamiento a datos de prueba, proporcionando información sobre la capacidad de generalización y la presencia de sobre-ajuste.

- **Gráficos de Dispersión:**
  Se incluyen gráficos de dispersión que muestran la relación entre el RUL predicho y el RUL real para cada modelo en el conjunto de datos de prueba. Estos gráficos facilitan la visualización de la precisión de las predicciones y la identificación de posibles tendencias o discrepancias entre los valores predichos y reales.

Capítulo 5

Resultados y Discusión

5.1. Resultados de la Simulación

En la sección 4.1 se definieron todas las ecuaciones y supuestos que describen la evolución en el tiempo y el comportamiento del sistema de la bomba tríplex. La Figura 5.1 muestra los resultados de la simulación del comportamiento normal tanto del caudal como de la velocidad angular de la bomba. Estos resultados de simulación ofrecen una confirmación visual del funcionamiento esperado del sistema bajo condiciones operacionales normales, respaldando de esta manera el modelo desarrollado.

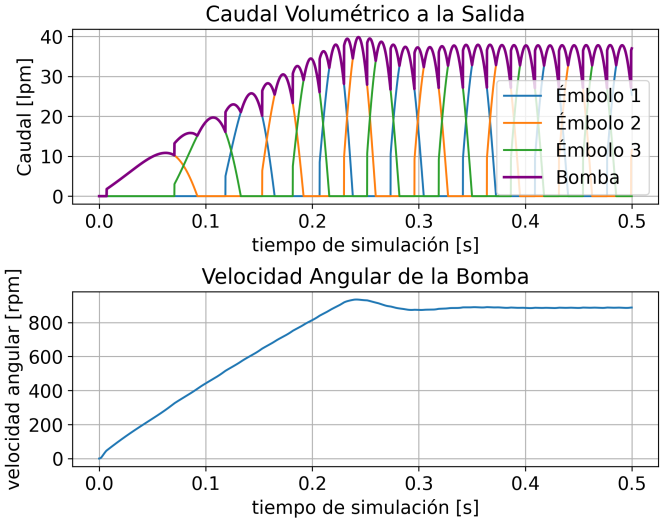


Figura 5.1: Caudal volumétrico a la salida de cada émbolo y caudal total de la bomba. No se han habilitado fallas.

Además del comportamiento normal sin fallas, según lo descrito en la sección 4.1.2, el modelo simulado considera condiciones de deterioro debido a la presencia de tres modos de falla: WornBearing, LeakSeal, y BlockedIn. Las Figuras 5.2 y 5.3 muestran el efecto en el caudal y la velocidad angular de la bomba debido a los modos de falla LeakSeal y BlockedIn respectivamente.

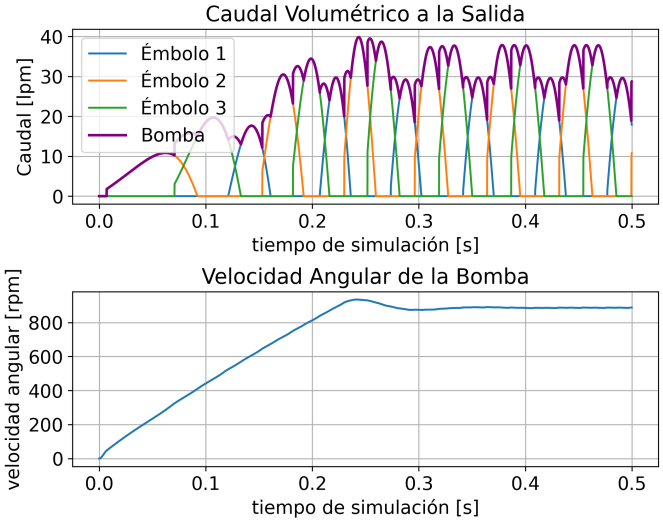


Figura 5.2: Caudal y la velocidad angular de la bomba cuando el fallo de fuga de fluido por desgaste en los sellos está activado en el émbolo 1.

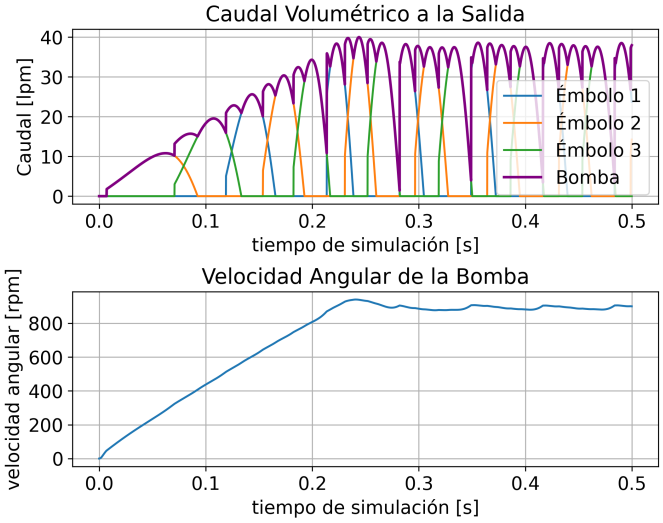


Figura 5.3: Caudal y la velocidad angular de la bomba cuando el fallo de entrada bloqueada está activado en la válvula de retención correspondiente al émbolo 1.

En ambos casos se observa que estas condiciones afectan significativamente el perfil del caudal de salida, provocando una disminución en la eficiencia de bombeo del cilindro afectado, lo que conlleva a una reducción del caudal total de salida y a un perfil de caudal más irregular. Esto de debe a que la presencia de una fuga de fluido en uno de los cilindros perturba el equilibrio entre los émbolos, resultando en un caudal de salida más pulsante y menos estable, mientras que el bloqueo en la válvula de retención de entrada de uno de los cilindros causa períodos de flujo reducido ya que el cilindro afectado no es capaz de aspirar fluido correctamente durante su ciclo de bombeo.

Por otro lado, dado que el modelo simulado considera que los componentes del sistema están sujetos a un proceso de degradación según lo definido en la sección 4.1.3, las Figuras 5.4, 5.5 y 5.6 muestran el efecto en el caudal de la bomba y la corriente eléctrica del motor debido a un incremento paulatino en el deterioro de los sellos en las juntas, el bloqueo progresivo en las válvulas de retención de entrada, y el aumento en el desgaste de los rodamientos del eje cigüeñal de la bomba. En cada caso se observa un empeoramiento en el perfil del caudal de salida y una disminución adicional en la eficiencia de bombeo. Se observa cómo el desgaste de los rodamientos del eje cigüeñal de la bomba (WornBearing) es el modo de falla que mayores efectos genera en el perfil de la corriente eléctrica consumida por el motor.

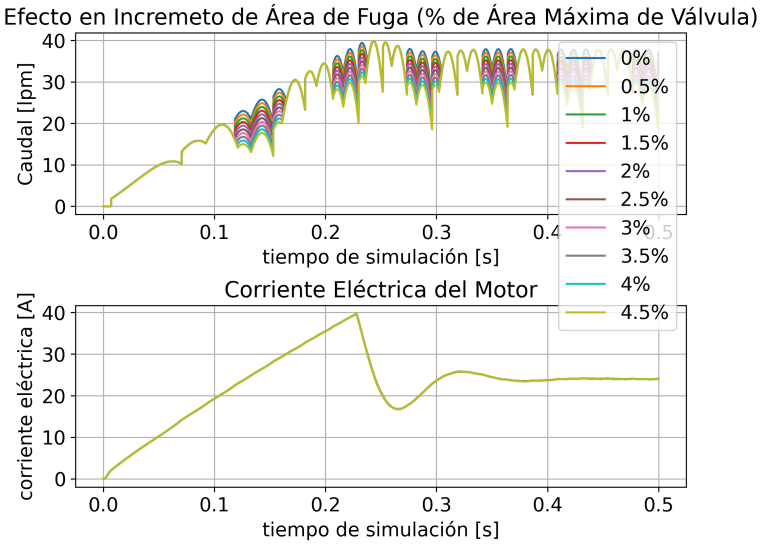


Figura 5.4: Efecto en el caudal de la bomba y en la corriente demandada debido al incremento del área de fuga de fluido en la junta debido al desgaste de los sellos en el émbolo 1.

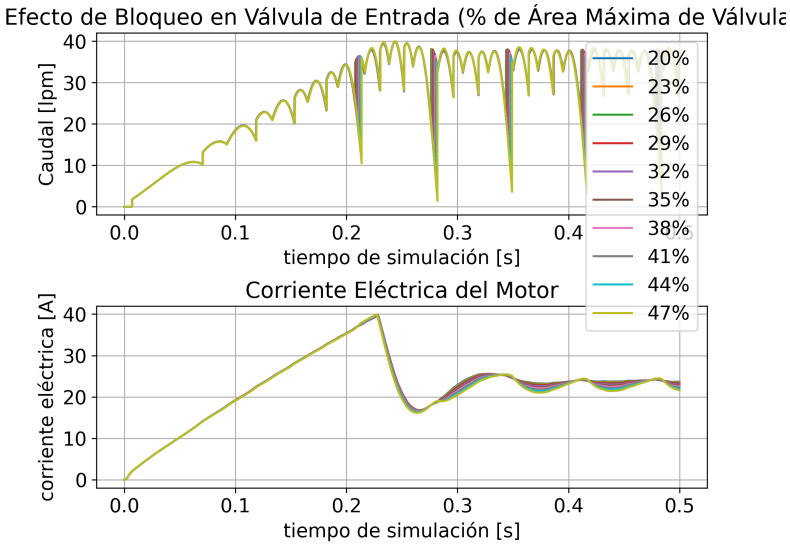


Figura 5.5: Efecto en el caudal de la bomba y en la corriente demandada debido al incremento en el bloqueo en la válvula de retención de entrada del émbolo 1.

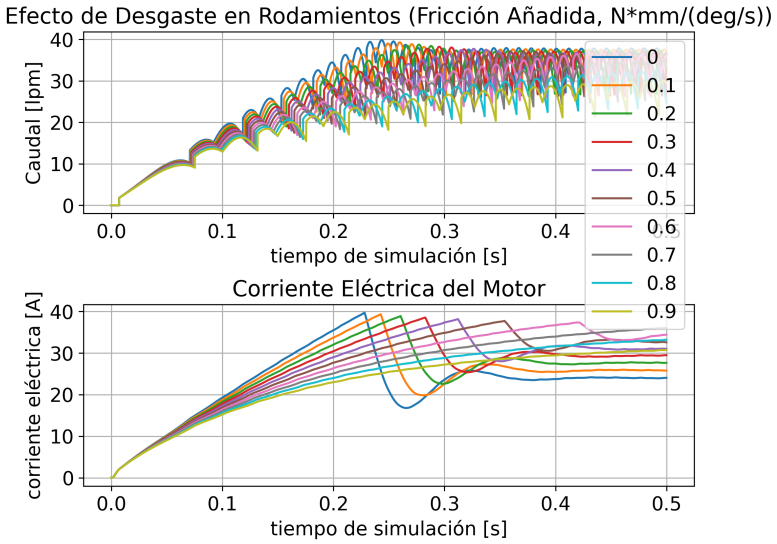


Figura 5.6: Efecto en el caudal de la bomba y en la corriente demandada debido al incremento en el desgaste de los rodamientos del eje cigüeñal de la bomba.

Luego, como se definió en la sección 4.1.3, el momento de aparición de las fallas (t_s) y el tiempo de fin de la vida útil (t_{EOL}) fueron generados de manera aleatoria para cada bomba simulada, según un proceso estocástico considerando distribuciones de probabilidad normal y Weibull. El cambio de estado de salud desde una condición de operación normal hasta la aparición de la falla para las 14 bombas simuladas se muestra en la Figura 5.1.

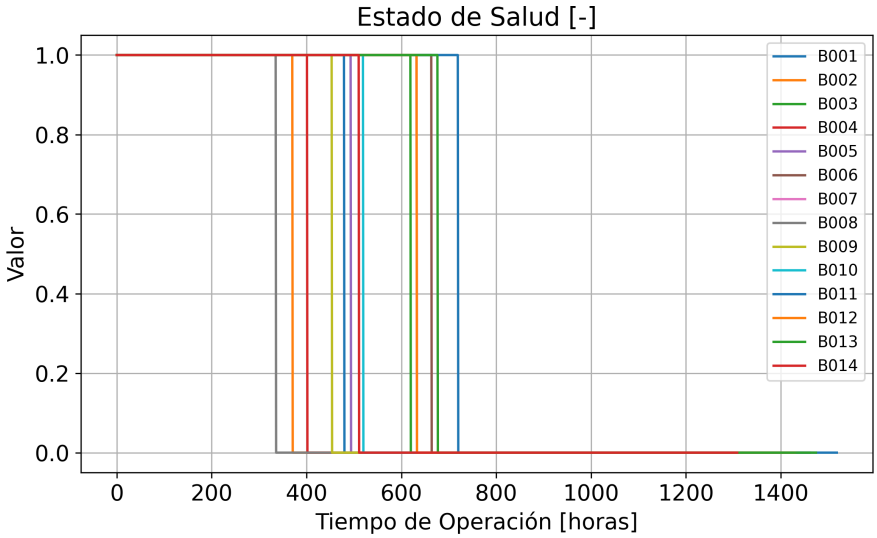


Figura 5.7: cambio de estado.

Según lo establecido en la sección 4.2.3, el sistema simulado de la bomba genera X variables en total, de las cuales sólo la corriente del motor $iMotor$, la velocidad angular del motor $wMotor$, los caudales de entrada y salida de la bomba Q_{in} y Q_{out}, y las presiones de entrada y salida de la bomba P_{in} y P_{out} corresponden a variables medibles del sistema, representando las mediciones de los sensores reales del sistema. El resto de las variables representan sensores virtuales cuya información es usada por el modelo para obtener los perfiles de operación.

Luego, la Figura 5.8 muestra una comparación entre una porción de la señal de corriente eléctrica del motor obtenida por la simulación (señal continua), luego haberle agregado ruido de sensores (señal con ruido), y finalmente, luego haber discretizado la señal mediante un ZOH (señal discreta).

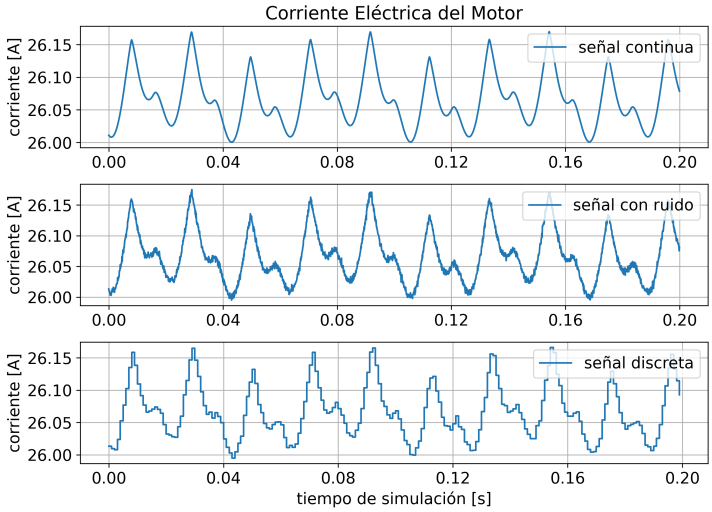


Figura 5.8: Comparación de la señal continua simulada de la corriente eléctrica del motor vs. la señal con ruido y la señal discretizada mediante ZOH.

Finalmente, la Figura 5.9 presenta una muestra de 0.3 segundos de duración del comportamiento normal sin fallas y estacionario de una bomba tríplex, dónde las señales de los sensores son discretas y presentan ruido.

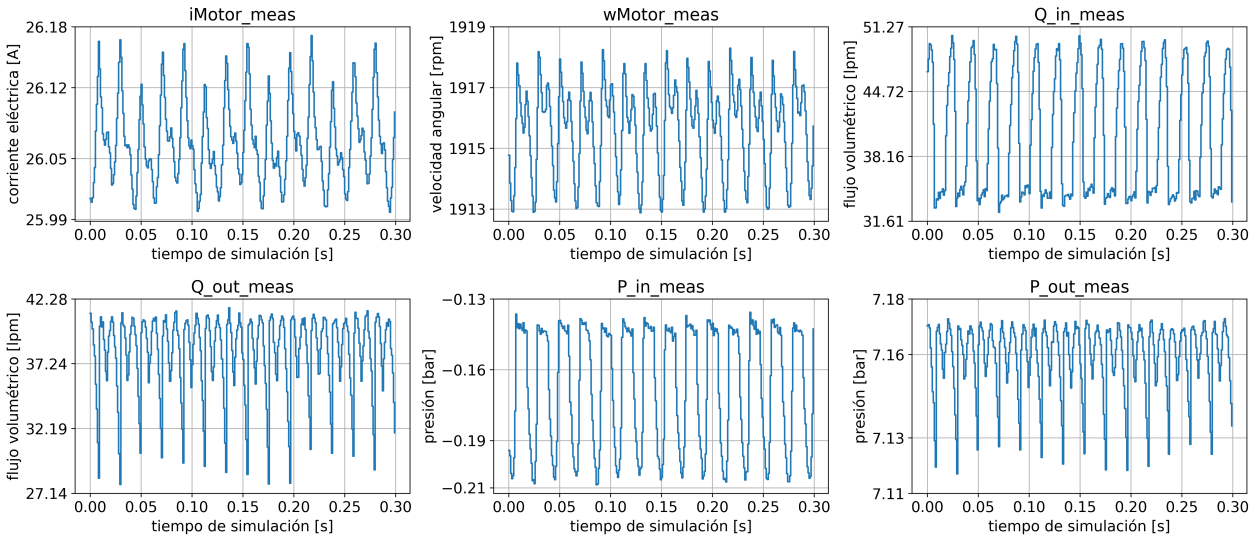


Figura 5.9: Comportamiento normal sin fallas de una bomba tríplex, con señales de sensores discretas y con ruido.

Por su parte, la Tabla 5.1 muestra el momento de aparición de las fallas (t_s), el tiempo de fin de la vida útil (t_{EOL}), el modo de falla, y la cantidad de muestras de sensores generadas para cada una de las unidades simuladas.

Tabla 5.1: Tabla resumen de unidades simuladas.

Unidad	t_s [horas]	t_{EOL} [horas]	Modo de Falla	N° Muestras
B001	479	1279	WornBearing	1279
B002	370	1169	LeakSeal	1169
B003	619	1418	BlockedIn	1418
B004	401	1200	WornBearing_BlockedIn	1200
B005	493	1292	WornBearing_LeakSeal	1292
B006	663	1462	BlockedIn_LeakSeal	1462
B007	632	1431	WornBearing_BlockedIn	1431
B008	335	1135	WornBearing_LeakSeal	1135
B009	453	1252	WornBearing_LeakSeal_BlockedIn	1252
B010	519	1318	WornBearing	1318
B011	719	1519	LeakSeal	1519
B012	632	1431	BlockedIn	1431
B013	676	1475	BlockedIn_LeakSeal	1475
B014	510	1309	WornBearing_LeakSeal_BlockedIn	1309

5.2. Desempeño de los Modelos Predictivos

Las tablas y figuras que se presentan a continuación muestran los resultados en el desempeño de los cuatro modelos de deep learning basados en diferentes arquitecturas (CNN, LSTM, CNN-LSTM, y CNN-LSTM-p) definidos en la sección 4.5, en la estimación del RUL, evaluando aspectos como la precisión en las predicciones, la capacidad de generalización y la robustez frente a diferentes conjuntos de datos.

La Tabla 5.2 muestra la precisión de las predicciones de los cuatro modelos de deep learning en los datos del conjunto de entrenamiento del escenario 1, según las métricas de evaluación definidas en la sección 4.7.1. El MAE para el modelo CNN fue de 18.72 horas, mientras que el RMSE fue de 28.98 horas, el MAPE fue del 16.38 %, y el coeficiente de determinación R^2 fue igual a 0.9961. Para el modelo LSTM, el MAE fue de 33.24 horas, el RMSE fue de 42.34 horas, el MAPE obtenido fue del 33.97 % y el coeficiente de determinación R^2

fue igual a 0.9918. El modelo CNN-LSTM obtuvo un MAE de 50.66 horas, un RMSE de 65.45 horas, un MAPE del 22.34 % y un coeficiente de determinación R^2 de 0.9809. Por último, el modelo CNN-LSTM-p obtuvo un MAE de 28.43 horas, mientras que el RMSE fue de 41.35 horas, el MAPE fue del 9.45 %, y el coeficiente de determinación R^2 fue igual a 0.9921.

Tabla 5.2: Comparación de precisión de la predicción de los modelos en el set de entrenamiento en el escenario 1.

Conjunto de Entrenamiento					
Escenario	Modelos	MAE	RMSE	MAPE	R^2
1	CNN	18.72	28.98	16.38	0.9961
	LSTM	33.24	42.34	33.97	0.9918
	CNN-LSTM	50.66	64.45	22.34	0.9809
	CNN-LSTM-p	28.43	41.35	9.45	0.9921

A partir de estos resultados, se aprecia como el modelo con mejor desempeño general en el conjunto de datos de entrenamiento del escenario 1 fue el modelo CNN, ya que presenta los menores valores para las métricas MAE y RMSE, y además, posee el mayor valor para el coeficiente de determinación R^2. Sin embargo, el modelo CNN-LSTM-p obtuvo un MAPE menor al del modelo CNN, por lo que este último tiene un peor desempeño en términos relativos que el modelo CNN-LSTM-p, lo que significa que en promedio, las predicciones del modelo CNN están más alejadas de los valores reales en términos porcentuales que las predicciones hechas por el modelo CNN-LSTM-p. De todos modos, el modelo CNN demuestra ser superior que el resto de los modelos en términos absolutos, donde además, junto con el modelo LSTM, presenta la menor diferencia entre los valores obtenidos de las métricas MAE y RMSE. Esto es importante ya que una diferencia pequeña entre RMSE y MAE, significa que los errores grandes no están influyendo significativamente en los valores de la métrica RMSE, lo que indica una distribución de errores relativamente uniforme. Si la diferencia es grande, esto indica la presencia de errores grandes u outliers que están teniendo un impacto considerable en el valor del RMSE. Por lo tanto, en términos prácticos, un modelo con una menor diferencia entre los valores obtenidos de las métricas MAE y RMSE es un modelo con errores más estables, consistentes y relativamente más pequeños.

En la Tabla 5.3 se muestra la comparación del desempeño en términos de la precisión de la predicción de los cuatro modelos de deep learning evaluados en el conjunto de datos de prueba del escenario 1. El MAE para el modelo CNN fue de 57.76 horas, con un RMSE de 83.88 horas, un MAPE del 47.34 %, y un coeficiente de determinación R^2 igual a 0.9604. Para el modelo LSTM, el MAE fue de 98.51 horas, el RMSE de 117.43 horas, el MAPE fue de un 70.91 %, y su coeficiente de determinación R^2 fue igual a 0.9225. El modelo CNN-LSTM obtuvo un MAE de 84.18 horas, un RMSE de 103.33 horas, con un MAPE del 35.90 %, y un coeficiente de determinación R^2 igual a 0.9400. Finalmente, el modelo CNN-LSTM-p obtuvo un MAE de 60.25 horas, un RMSE de 87.82 horas, con un MAPE del 22.49 % y un coeficiente de determinación R^2 igual a 0.9566.

Tabla 5.3: Comparación de precisión de la predicción de los modelos en el set de prueba en el escenario 1.

Conjunto de Prueba					
Escenario	Modelos	MAE	RMSE	MAPE	R^2
1	CNN	57.76	83.88	47.34	0.9604
	LSTM	98.51	117.43	70.91	0.9225
	CNN-LSTM	84.18	103.33	35.90	0.9400
	CNN-LSTM-p	60.25	87.82	22.49	0.9566

Ahora bien, al evaluar los mismos modelos con el conjunto de prueba del escenario 1, es decir, con datos que fueron vistos durante el entrenamiento, se observa como se mantiene la tendencia mostrada en la Tabla 5.2, es decir, el modelo CNN nuevamente es el que obtuvo el mejor rendimiento general, pues sus valores para las métricas MAE y RMSE son las menores, y su valor para el coeficiente de determinación R^2 es el mayor. Sin embargo, el modelo CNN también presenta el mismo problema mostrado en su desempeño en el conjunto de datos de entrenamiento del escenario 1, pues el valor del MAPE obtenido no es el más bajo de entre los modelos evaluados, lo que significa que en promedio, las predicciones del modelo CNN están más alejadas de los valores reales en términos porcentuales que las predicciones hechas por los modelos CNN-LSTM y CNN-LSTM-p. Por otro lado, en este caso, el modelo CNN presenta la segunda mayor diferencia absoluta entre los valores obtenidos de las métricas MAE y RMSE, es decir, junto con el modelo CNN-LSTM-p, el modelo CNN es uno del modelos con errores menos estables y relativamente más grandes en el conjunto de datos de prueba del escenario 1.

Al comparar las tablas 5.2 y 5.3, se puede extraer información sobre la degradación del rendimiento que presentan los modelos al pasar desde los datos de entrenamiento hasta el conjunto de datos de prueba. En este contexto, la Tabla 5.4 muestra como el modelo CNN-LSTM presenta la menor degradación en su rendimiento al comparar su desempeño sobre los datos de entrenamiento y los datos de prueba, tanto en términos absolutos como en términos relativos. Esto significa que, aún cuando el modelo CNN muestra un mejor rendimiento en ambos conjuntos de datos, su robustez y capacidad de generalización es inferior al modelo CNN-LSTM bajo las mismas condiciones de entrenamiento definidas y consideradas para el escenario 1.

Tabla 5.4: Degradación de la precisión de predicción de los modelos entre los conjuntos de entrenamiento y prueba en el escenario 1.

Degradación del Rendimiento					
Escenario	Modelos	MAE	RMSE	MAPE	R^2
1	CNN	+39.04 (209 %)	+54.90 (189 %)	+30.96 (189 %)	-0.0357 (-4 %)
	LSTM	+65.27 (196 %)	+75.09 (177 %)	+36.94 (109 %)	-0.0693 (-7 %)
	CNN-LSTM	+33.52 (66 %)	+38.88 (60 %)	+13.56 (61 %)	-0.0409 (-4 %)
	CNN-LSTM-p	+31.82 (112 %)	+46.47 (112 %)	+13.04 (138 %)	-0.0355 (-4 %)

Por otro lado, los gráficos de la Figura 5.10 entregan una comparación visual del desempeño predictivo de los cuatro modelos evaluados en las distintas unidades del conjunto de datos de prueba del escenario 1. En los gráficos, el eje horizontal muestra las horas de operación de las bombas, mientras que el eje vertical representa los valores del RUL en horas. La curva de color naranja corresponde al modelo CNN, mientras que la curva de color verde corresponde al modelo LSTM, la de color rojo al modelo CNN-LSTM, y la curva de color púrpura representa las predicciones del modelo CNN-LSTM-p. La linea segmentada de color azul representa los valores “Reales” del RUL.

Al examinar los gráficos de las cuatro unidades pertenecientes al conjunto de datos de prueba del escenario 1 (B005, B008, B009 y B014), se observa que los cuatro modelos predictivos evaluados muestran una capacidad general relativamente buena para seguir la tendencia del RUL real. Esto refuerza el hecho de que todos los modelos obtuvieron valores bastante altos para el coeficiente de determinación R^2 (Tabla 5.3), pues en general, un valor por encima de R^2 superior a 0.8 es considerado aceptable para modelos de regresión [31]

[39]. Por otro lado, en todas las unidades, se observa que prácticamente todos los modelos logran captar el cambio de pendiente presente en la curva del RUL real, el cual indica la aparición de algún modo de falla subyacente.

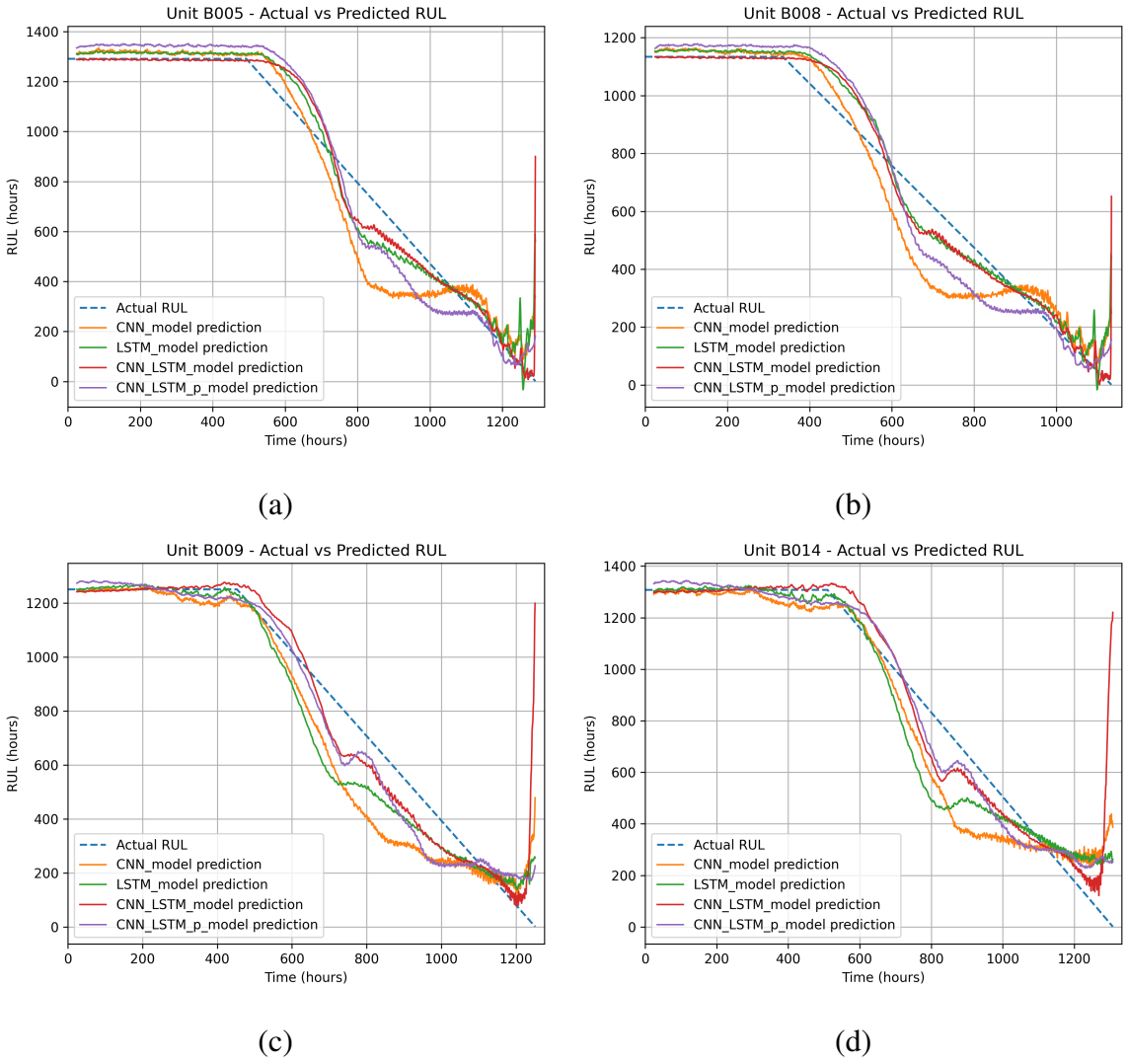


Figura 5.10: Resultados de la predicción en las unidades de prueba en el escenario 1: (a) Modelo CNN, (b) Modelo LSTM, (c) Modelo CNN-LSTM, (d) Modelo CNN-LSTM-p.

Ahora bien, respecto a la precisión apreciable de los distintos modelos predictivos en las diferentes fases de vida de las unidades del conjunto de datos de prueba del escenario 1, se observa como en la fase inicial, donde el RUL se mantiene constante debido a lo discutido en la sección 4.3.6, los modelos muestran un buen ajuste en general. Sin embargo, se ob-

servan algunas fluctuaciones menores, especialmente en las unidades B009 y B014. Por otro lado, en las unidades B005 y B008 todos los modelos a excepción del modelo CNN-LSTM tienden a sobrestimar los valores predichos para el RUL. Luego en la fase de transición, el punto de inflexión donde comienza el deterioro o la propagación de la falla es capturado por todos los modelos, aunque con variaciones en la precisión. En la unidad B005, por ejemplo, el modelo CNN-LSTM-p parece detectar este cambio más temprano que los demás modelos, mientras que el modelo CNN-LSTM tiende a reconocer el cambio de pendiente de manera más tardía. Durante la fase de pendiente negativa pronunciada, se observan diferencias más notables entre los modelos. En general, el modelo CNN-LSTM-p parece seguir más de cerca la curva real en la mayoría de las unidades, especialmente en B008 y B014. Sin embargo, hacia el final de la vida de todas las unidades, se observan cambios rápidos y pronunciados en las predicciones de RUL, especialmente en las últimas 200-300 horas de operación.

La Tabla 5.5 muestra la precisión de las predicciones de los cuatro modelos de deep learning en los datos del conjunto de entrenamiento del escenario 2, según las métricas de evaluación definidas en la sección 4.7.1. El MAE para el modelo CNN fue de 29.52 horas, mientras que el RMSE fue de 39.25 horas, el MAPE fue del 24.24 %, y el coeficiente de determinación R^2 fue igual a 0.9922. Para el modelo LSTM, el MAE fue de 33.43 horas, el RMSE fue de 46.60 horas, el MAPE obtenido fue del 31.81 % y el coeficiente de determinación R^2 fue igual a 0.9890. El modelo CNN-LSTM obtuvo un MAE de 23.50 horas, un RMSE de 40.19 horas, un MAPE del 56.59 % y un coeficiente de determinación R^2 de 0.9918. Por último, el modelo CNN-LSTM-p obtuvo un MAE de 21.38 horas, mientras que el RMSE fue de 27.39 horas, el MAPE fue del 18.65 %, y el coeficiente de determinación R^2 fue igual a 0.9962.

Tabla 5.5: Comparación de precisión de la predicción de los modelos en el set de entrenamiento en el escenario 2.

Conjunto de Entrenamiento					
Escenario	Modelos	MAE	RMSE	MAPE	R^2
2	CNN	29.52	39.25	24.24	0.9922
	LSTM	33.43	46.60	31.81	0.9890
	CNN-LSTM	23.50	40.19	56.59	0.9918
	CNN-LSTM-p	21.38	27.39	18.65	0.9962

A partir de estos resultados, se aprecia como el modelo con mejor desempeño general en el conjunto de datos de entrenamiento del escenario 2 fue el modelo CNN-LSTM-p, ya que presenta los menores valores para las métricas MAE, RMSE y MAPE, y además, posee el mayor valor para el coeficiente de determinación R^2. En modelo CNN-LSTM-p también presenta la menor diferencia entre los valores obtenidos de las métricas MAE y RMSE, lo que significa que los errores grandes no están influyendo tan significativamente en los valores de la métrica RMSE como en los demás modelos, por lo que el modelo CNN-LSTM-p es el que presenta errores más estables, consistentes y relativamente más pequeños.

En la Tabla 5.6 se muestra la comparación del desempeño en términos de la precisión de la predicción de los cuatro modelos de deep learning evaluados en el conjunto de datos de prueba del escenario 2. El MAE para el modelo CNN fue de 50.54 horas, con un RMSE de 70.33 horas, un MAPE del 37.07 %, y un coeficiente de determinación R^2 igual a 0.9772. Para el modelo LSTM, el MAE fue de 42.44 horas, el RMSE de 60.62 horas, el MAPE fue de un 36.38 %, y su coeficiente de determinación R^2 fue igual a 0.9755. El modelo CNN-LSTM obtuvo un MAE de 45.89 horas, un RMSE de 72.84 horas, con un MAPE del 46.80 %, y un coeficiente de determinación R^2 igual a 0.9755. Finalmente, el modelo CNN-LSTM-p obtuvo un MAE de 40.71 horas, un RMSE de 59.51 horas, con un MAPE del 24.67 % y un coeficiente de determinación R^2 igual a 0.9755.

Tabla 5.6: Comparación de precisión de la predicción de los modelos en el set de prueba en el escenario 2.

Conjunto de Prueba					
Escenario	Modelos	MAE	RMSE	MAPE	R^2
2	CNN	50.54	70.33	37.07	0.9772
	LSTM	42.44	60.62	36.38	0.9831
	CNN-LSTM	45.89	72.84	46.80	0.9755
	CNN-LSTM-p	40.71	59.51	24.67	0.9837

Al evaluar los mismos modelos con el conjunto de prueba del escenario 2, es decir, con datos que fueron vistos durante el entrenamiento, se observa como se mantiene la tendencia mostrada en la Tabla 5.5, es decir, el modelo CNN-LSTM-p nuevamente es el que obtuvo el mejor rendimiento general, pues sus valores para las métricas MAE, RMSE y MAPE son las menores, y su valor para el coeficiente de determinación R^2 es el mayor. Sin embargo,

en este caso, el modelo CNN-LSTM-p presenta la segunda menor diferencia absoluta entre los valores obtenidos de las métricas MAE y RMSE, ya que, el modelo LSTM presenta una diferencia ligeramente inferior.

Al comparar las tablas 5.5 y 5.6, se puede extraer información sobre la degradación del rendimiento que presentan los modelos al pasar desde los datos de entrenamiento hasta el conjunto de datos de prueba. En este contexto, la Tabla 5.7 muestra como el modelo LSTM presenta la menor degradación en su rendimiento al comparar su desempeño sobre los datos de entrenamiento y los datos de prueba, tanto en términos absolutos como en términos relativos. Esto significa que, aún cuando el modelo CNN-LSTM-p muestra un mejor rendimiento en ambos conjuntos de datos, su robustez y capacidad de generalización es inferior al modelo LSTM bajo las mismas condiciones de entrenamiento definidas y consideradas para el escenario 2. Adicionalmente, cabe destacar que, aún cuando el modelo CNN-LSTM presentó la mayor degradación tanto absoluta como relativa en las métricas, MAE, RMSE y R^2, el MAPE se redujo al pasar del modelo de entrenamiento al de prueba.

Tabla 5.7: Degradación de la precisión de predicción de los modelos entre los conjuntos de entrenamiento y prueba en el escenario 2.

Degradación del Rendimiento					
Escenario	Modelos	MAE	RMSE	MAPE	R^2
2	CNN	+21.02 (71 %)	+31.08 (79 %)	+12.83 (53 %)	-0.0150 (-2 %)
	LSTM	+9.01 (27 %)	+14.02 (30 %)	+4.57 (14 %)	-0.0059 (-1 %)
	CNN-LSTM	+22.39 (95 %)	+32.65 (81 %)	-9.79 (-17 %)	-0.0163 (-2 %)
	CNN-LSTM-p	+19.33 (90 %)	+32.12 (117 %)	+6.02 (32 %)	-0.0125 (-1 %)

Los gráficos de la Figura 5.11 entregan una comparación visual del desempeño predictivo de los cuatro modelos evaluados en las distintas unidades del conjunto de datos de prueba del escenario 2. En los gráficos, el eje horizontal muestra las horas de operación de las bombas, mientras que el eje vertical representa los valores del RUL en horas. La curva de color naranja corresponde al modelo CNN, mientras que la curva de color verde corresponde al modelo LSTM, la de color rojo al modelo CNN-LSTM, y la curva de color púrpura representa las predicciones del modelo CNN-LSTM-p. La linea segmentada de color azul representa los valores “Reales” del RUL.

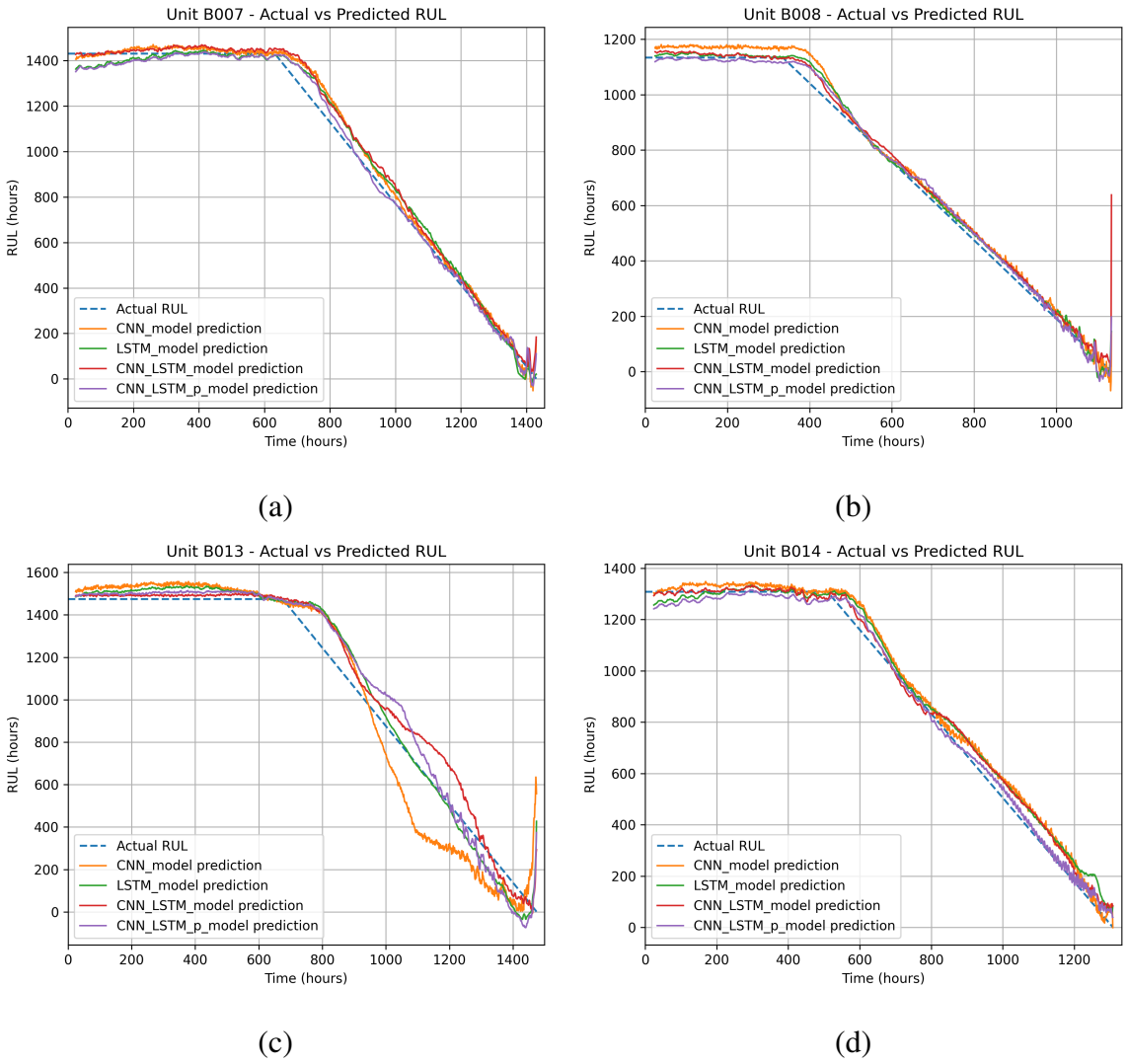


Figura 5.11: Resultados de la predicción en las unidades de prueba en el escenario 2: (a) Modelo CNN, (b) Modelo LSTM, (c) Modelo CNN-LSTM, (d) Modelo CNN-LSTM-p.

Al examinar los gráficos de las cuatro unidades pertenecientes al conjunto de datos de prueba del escenario 2 (B007, B008, B013 y B014), se observa que los cuatro modelos predictivos evaluados muestran una capacidad general relativamente buena para seguir la tendencia del RUL real, aunque el desempeño de estos modelos es notoriamente pero en la unidad B013, en especial en la zona de deterioro. Por otro lado, en todas las unidades, se observa que prácticamente todos los modelos logran captar el cambio de pendiente presente en la curva del RUL real, aunque nuevamente, este cambio de pendiente es más tardío para todos los modelos en la unidad B013.

Ahora bien, respecto a la precisión apreciable de los distintos modelos predictivos en las diferentes fases de vida de las unidades del conjunto de datos de prueba del escenario 2, se observa como en la fase inicial, los modelos muestran un buen ajuste en general, aunque hay presencia de ligeras fluctuaciones, las cuales son más marcadas en la unidad B014. Por otro lado, en la unidades B013,todos los modelos tienden a sobrestimar los valores predichos para el RUL, mientras que en las unidades B007 y B014 los modelos LSTM y CNN-LSTM-p tienden a subestimar predichos para el RUL. Luego en la fase de transición, el punto de inflexión donde comienza el deterioro o la propagación de la falla es capturado por todos los modelos, aunque con variaciones en la precisión. En la unidad B007, por ejemplo, el modelo CNN-LSTM-p parece detectar este cambio más temprano que los demás modelos, mientras que el modelo CNN-LSTM tiende a reconocer el cambio de pendiente de manera más tardía. Luego, en la unidad B013 los cuatro modelos reconocen el cambio de pendiente prácticamente en el mismo instante. Finalmente, durante la fase de pendiente negativa pronunciada, el ajuste a la curva original de RUL de todos los modelos en la unidad B008 es casi perfecto, aunque es evidente que las predicciones tienden a sobrestimar los valores del RUL. Lo anterior se repite en las unidades B007 y B014, aunque el ajuste no es tan bueno como en el caso de la unidad B008. Las diferencias más notables entre las predicciones de los distintos modelos y los valores reales del RUL son apreciables en la unidad B013, donde en general, el modelo LSTM parece seguir más de cerca la curva real la mayor parte del tiempo. Sin embargo, hacia el final de la vida de la unidad B013, se observan cambios rápidos y pronunciados en las predicciones de RUL, especialmente en las últimas 50-100 horas de operación. Estas fluctuaciones también se aprecian en las demás unidades, aunque en menor medida en las unidades B007 y B014.

5.3. Discusión de Resultados

El estudio se centró en la evaluación y comparación del desempeño en la estimación del RUL de cuatro modelos de deep learning (CNN, LSTM, CNN-LSTM, y CNN-LSTM-p) en un conjunto de datos sintéticos generados a partir de la simulación de la operación hasta el fallo de una pequeña flota de 14 bombas tríplex, las cuales contaron con la implementación de tres modos de falla principales: WornBearing, LeakSeal y BlockedIn, y la combinación de estos, para generar un total de 7 escenarios de falla distintos. Además, se utilizó un modelo de degradación exponencial basado en [85] y [7], para la propagación de las fallas, y la generación de trayectorias de degradación hasta el fallo.

Respecto al desempeño de los modelos predictivos, los principales hallazgos fueron:

1. En el escenario 1: El modelo CNN mostró el mejor desempeño general en los conjuntos de datos de entrenamiento y prueba, mientras que el modelo CNN-LSTM fue el que presentó la menor degradación de su rendimiento entre los conjuntos de entrenamiento y prueba.
2. En el escenario 2: El modelo CNN-LSTM-p obtuvo el mejor rendimiento general en ambos conjuntos de datos, mientras que el modelo LSTM mostró la menor degradación de rendimiento entre ambos conjuntos.
3. Se observaron diferencias en el desempeño predictivo y la precisión de las modelos en distintas fases de vida de las unidades, dónde destacan las fluctuaciones pronunciadas hacia el final de la vida de las unidades.
4. Todos los modelos mostraron una capacidad general bastante elevada para seguir la tendencia real de los valores del RUL en ambos escenarios, aunque en el escenario 1, dónde los datos de prueba estaba compuestos por datos y modos de falla no vistos durante la fase de entrenamiento de los modelos, existieron desviaciones evidentes en las predicciones, especialmente en la fase de deterioro del ciclo de vida de las unidades.

Los resultados muestran que los modelos son capaces de capturar la tendencia general del RUL bajo diferentes condiciones operativas (diferentes modos de falla), dónde destaca el hecho de que en ambos escenarios los modelos lograron captar el cambio de pendiente del RUL, lo cual indica la capacidad de los modelos de detectar la desviación del comportamiento normal de las distintas unidades, reconociendo en mayor o menor medida el punto de inflexión donde comienza el deterioro. Esto sugiere que los modelos han sido capaces de aprender e identificar patrones sutiles en los datos de entrenamiento, que indican el inicio del proceso de degradación. Lo anterior resulta aún más sorprendente al considerar que los resultados del escenario 1 mostrados en los gráficos de la Figura 5.10 corresponden a las predicciones sobre modos de falla no vistos durante el entrenamiento, ya que el hecho de que los modelos detecten el comienzo del proceso de degradación anormal, implica que estos modelos aprendieron de manera precisa los patrones propios del comportamiento normal en ausencia de fallas, otorgándoles buenas capacidades para la detección temprana de de fallas en las bombas. Además, la variabilidad en el desempeño de los modelos entre las diferentes fases de vida de las bombas indica que existen patrones distintivos en los datos en cada una de estas fase, lo que a su vez sugiere que las características propias de los datos cambian

a medida que la bomba se deteriora. Esto es coherente con la evolución de la degradación esperada de los fallos mecánicos, ya que el modelo simulado de las bombas considera un deterioro con dos fases diferenciadas: una fase de degradación normal lineal, y una fase de degradación anormal exponencial.

Un resultado inesperado fue el alto MAPE del modelo CNN-LSTM en el escenario 2 para el conjunto de datos de entrenamiento, a pesar de tener un buen desempeño en las demás métricas de evaluación. Esto podría deberse a que el modelo es más sensible a errores relativos en ciertos rangos del RUL, posiblemente en valores bajos donde un pequeño error absoluto puede resultar en un gran error porcentual. Además, la mejora del MAPE del modelo CNN-LSTM al pasar del conjunto de entrenamiento al de prueba en el escenario 2 también representa un resultado inesperado. Esto podría indicar que el modelo se adaptó mejor a ciertas características presentes en el conjunto de prueba, o que hubo una distribución más favorable de los valores de RUL en dicho conjunto.

Por otro lado, además de las diferencias en la complejidad y naturaleza de los distintos modelos predictivos evaluados, una posible explicación para las diferencias de desempeño entre estos modelos podría ser la sensibilidad a la inicialización de los pesos o parámetros de la redes neuronales, o a la secuencia específica de datos usada durante las fases de entrenamiento. Repetir el entrenamiento con diferentes semillas aleatorias y además repetir el entrenamiento de los distintos modelos en cada escenario varias veces, podría proporcionar una visión más robusta del desempeño relativo de los modelos, al generar una distribución de los valores predichos del RUL en cada paso de tiempo en lugar de sólo valores puntuales, lo que incluso da la opción de evaluar el desempeño de los modelos en términos probabilísticos. Luego, las diferencias en el desempeño entre los escenarios 1 y 2 podrían deberse no solo a la cantidad de datos de entrenamiento, sino también a la distribución específica de los modos de falla en cada conjunto. Un análisis más detallado de la composición de los conjuntos de datos podría arrojar luz sobre este aspecto.

Sin embargo, los resultados obtenidos cumplen con lo esperado, ya que en ambos escenarios los modelos de deep learning mostraron una buena capacidad para aprender y predecir patrones complejos en datos de series temporales. En particular, como se definió en la sección 4.4, el escenario 1 fue diseñado para simular una configuración realista, en donde los datos históricos de fallas de equipos no están del todo balanceados en el sentido de que contienen muchos más datos de condiciones normales sin fallo y pocos ejemplos de datos con fallas. En

este contexto, lo esperado es que los distintos modelos al menos fueran capaces de detectar el cambio de comportamiento en los equipos al pasar de un estado normal a uno con presencia de una falla. Sin embargo, además de lograr lo anterior, todos los modelos evaluados en el escenario 1 muestran una aproximación relativamente aceptable del RUL, sobre todo cerca del final de la vida las bombas, justo antes de la aparición de las marcadas fluctuaciones finales. De lo anterior se desprende que la capacidad de los modelos para generalizar del conjunto de entrenamiento al conjunto de prueba fue mayor de lo esperado.

Por su parte, para el escenario 2 se planteó la evaluación de todos los modos de falla al usar el argumento del conocimiento experto y el valor de aumentar la calidad de los conjuntos de datos al agregar datos simulados. En este contexto, además de la detección de la aparición de la falla, se esperaba un rendimiento superior de los modelos, sobre todo en el ajuste durante la fase de deterioro. Aunque esto fue así, en general, los resultados del desempeño de los distintos modelos predictivos en la unidad B013 mostraron una evidente diferencia con el resto de unidades, lo que podría indicar que hay diferencias significativas entre los patrones de deterioro de dicha bomba en específico aún cuando su modo de falla subyacente estuvo presente en los datos de entrenamiento de los distintos modelos predictivos, lo que en definitiva dificulta su generalización.

Los resultados de este trabajo tienen implicaciones significativas para la implementación de sistemas de mantenimiento predictivo en bombas tríplex y potencialmente en otros equipos industriales similares. La capacidad de los modelos para predecir el RUL con un alto grado de precisión podría conducir a:

- Mejoras en la planificación del mantenimiento, reduciendo el tiempo de inactividad no programado.
- Optimización de la vida útil de los equipos, evitando tanto el reemplazo prematuro como las fallas catastróficas.
- Reducción de costos operativos y de mantenimiento en industrias que dependen de bombas tríplex.

Estos hallazgos contribuyen a la literatura existente sobre el uso de técnicas de deep learning en el pronóstico de RUL, ya que confirman la efectividad de las arquitecturas CNN y LSTM en este dominio, como se ha visto en estudios previos. Sin embargo, también destacan la importancia de considerar arquitecturas híbridas como CNN-LSTM y CNN-LSTM-p, que

pueden ofrecer ventajas en ciertos escenarios. Por lo tanto, este trabajo aporta nuevas perspectivas en aspectos como la comparación directa de múltiples arquitecturas de deep learning en el contexto específico de la estimación del RUL de bombas tríplex, la evaluación del desempeño de los modelos en diferentes fases de la vida útil de las bombas, proporcionando una comprensión más matizada de sus fortalezas y debilidades, y la identificación de la variabilidad en la degradación del rendimiento entre los conjuntos de entrenamiento y prueba, lo que subraya la importancia de considerar la robustez y la capacidad de generalización en la selección del modelo. Sin embargo, las conclusiones de este trabajo se limitan principalmente al contexto de las bombas tríplex y a los escenarios específicos evaluados. Aunque los resultados son prometedores, se debe tener precaución al generalizar estas conclusiones a otros tipos de equipos o condiciones operativas significativamente diferentes. Algunas de las limitaciones que podrían haber impactado los resultados incluyen:

1. Tamaño del conjunto de datos: Aunque se utilizaron múltiples bombas, un conjunto de datos más grande podría proporcionar resultados más robustos y generalizables.
2. Variabilidad en los modos de falla: La distribución específica de los modos de falla en los conjuntos de datos podría influir en el desempeño relativo de los modelos.
3. Simplificaciones en la simulación: Cualquier simplificación en el modelo de simulación podría limitar la representatividad de los datos con respecto a las condiciones del mundo real.

A pesar de estas limitaciones, los resultados siguen siendo válidos y valiosos para responder a la pregunta de investigación sobre la efectividad comparativa de diferentes arquitecturas de deep learning en la predicción del RUL de bombas tríplex. La consistencia general en el desempeño de los modelos y su capacidad para capturar tendencias importantes del RUL respaldan la validez de los hallazgos.

Capítulo 6

Conclusiones

El presente trabajo de titulación ha logrado abordar de manera efectiva el desafío de predecir la vida útil remanente (RUL) de activos críticos, específicamente bombas tríplex, mediante el uso de arquitecturas de redes neuronales profundas aplicadas a datos sintéticos. Estos datos fueron generados a partir de un modelo de ecuaciones diferenciales que simula trayectorias de degradación hasta el fallo. A través de la investigación y el desarrollo de modelos basados en CNN, LSTM, y combinaciones híbridas (CNN-LSTM y CNN-LSTM-p), se ha demostrado que estas técnicas pueden proporcionar predicciones precisas y confiables en el contexto del mantenimiento predictivo.

El modelado y la simulación de la degradación de las bombas tríplex permitieron crear un conjunto de datos sintético robusto y representativo, esencial para la evaluación de los modelos predictivos. Sin embargo, es importante destacar que cualquier simplificación en el modelo de simulación podría limitar la representatividad de los datos respecto a las condiciones del mundo real. Los resultados obtenidos indican que las arquitecturas híbridas, particularmente CNN-LSTM y CNN-LSTM-p, ofrecieron un desempeño superior en ciertos escenarios, capturando de manera efectiva las tendencias de degradación y proporcionando estimaciones precisas del RUL. No obstante, se observó que la calidad y cantidad de datos de entrenamiento son factores críticos que pueden afectar significativamente el rendimiento de los modelos. Un conjunto de datos más extenso y variado podría mejorar aún más la robustez y precisión de las predicciones.

Las técnicas de preprocesamiento de datos, como la normalización y la selección de características en los dominios del tiempo y la frecuencia, fueron fundamentales para mejorar

la calidad de los datos y, por ende, la precisión de los modelos. A pesar de esto, el impacto de diferentes enfoques de preprocesamiento en el rendimiento de los modelos sugiere que futuras investigaciones podrían beneficiarse de la exploración de metodologías más avanzadas o alternativas para optimizar este proceso. La implementación de técnicas de validación cruzada en la evaluación de los modelos es una limitación que debe ser abordada en futuros estudios para obtener una estimación más robusta y confiable del rendimiento de los modelos.

Una limitación significativa de este estudio es la generalización de los resultados obtenidos exclusivamente con datos sintéticos. La aplicación de estos modelos a datos reales podría presentar desafíos adicionales, lo que sugiere la necesidad de una validación exhaustiva en entornos operativos reales. Además, la variabilidad en los modos de falla y las condiciones operativas podrían influir en el rendimiento de los modelos, subrayando la importancia de incluir una mayor diversidad de escenarios en futuros estudios.

Basándonos en los hallazgos y limitaciones identificadas en este trabajo, se proponen las siguientes recomendaciones y direcciones para el trabajo futuro:

1. Validación con datos reales:
    - Implementar los modelos desarrollados en entornos industriales reales para evaluar su desempeño con datos operativos de bombas tríplex.
    - Colaborar con empresas del sector para obtener conjuntos de datos históricos que permitan una validación más robusta de los modelos.

2. Mejora de la simulación:
    - Refinar el modelo de simulación para incluir una mayor variedad de condiciones operativas y modos de falla.
    - Incorporar factores ambientales y de mantenimiento en el modelo de degradación para aumentar su realismo.

3. Exploración de arquitecturas avanzadas:
    - Investigar el uso de arquitecturas como Transformers y otras variantes de RNN para capturar relaciones temporales más complejas.
    - Desarrollar modelos híbridos que combinen las fortalezas de diferentes arquitecturas de deep learning.

4. Optimización de hiperparámetros y preprocesamiento:
    - Implementar técnicas avanzadas de búsqueda, como la optimización bayesiana, para la selección óptima de hiperparámetros.
    - Explorar técnicas de preprocesamiento más sofisticadas, incluyendo métodos de selección de características basados en aprendizaje automático.

5. Extensión a otros activos críticos:
    - Adaptar y validar los modelos para su uso en otros tipos de equipos industriales críticos.
    - Investigar la transferibilidad de los modelos entre diferentes tipos de activos.

6. Incorporación de técnicas de explicabilidad:
    - Implementar métodos de interpretación de modelos para aumentar la confianza en las predicciones y facilitar su adopción en la industria.
    - Desarrollar visualizaciones que ayuden a comprender los factores que influyen en las predicciones de RUL.

La implementación de estas recomendaciones y líneas de trabajo futuro no solo abordaría las limitaciones identificadas en el presente estudio, sino que también ampliaría significativamente el alcance y la aplicabilidad de los modelos de predicción de RUL en el mantenimiento predictivo.

En conclusión, este trabajo ha demostrado el potencial de las redes neuronales profundas para la predicción de la vida útil remanente en el contexto del mantenimiento predictivo, proporcionando una base sólida para futuras investigaciones y aplicaciones prácticas en la industria. Los resultados obtenidos son prometedores, pero es necesario continuar explorando y refinando estas técnicas para asegurar su efectividad y aplicabilidad en una amplia variedad de escenarios y condiciones operativas. La investigación ha cumplido con sus objetivos principales, ofreciendo insights valiosos sobre el rendimiento comparativo de diferentes arquitecturas de deep learning en la predicción del RUL para bombas tríplex, y estableciendo una metodología sólida para futuros estudios en este campo crítico del mantenimiento predictivo. La implementación de las recomendaciones propuestas y la continuación del trabajo en las direcciones sugeridas permitirán avanzar significativamente en la aplicación práctica y efectiva de estas tecnologías en entornos industriales reales.

Bibliografía

[1] https://en.wikipedia.org/wiki/Low-pass_filter.

[2] https://chem.libretexts.org/Bookshelves/Physical_and_Theoretical_Chemistry_Textbook_Maps/Supplemental_Modules_(Physical_and_Theoretical_Chemistry)/Physical_Properties_of_Matter/Solutions_and_Mixtures/Ideal_Solutions/Dissolving_Gases_In_Liquids%2C_Henry's_Law.

[3] https://www.fracttal.com/es/mantenipedia/pasos-para-realizar-el-analisis-weibull.

[4] Mayank Agarwal. Back propagation in convolutional neural networks — intuition and code. *Becoming Human: Artificial Intelligence Magazine*, 2017.

[5] Ali Al-Dulaimi, Soheil Zabihi, Amir Asif, and Arash Mohammadi. A multimodal and hybrid deep neural network model for remaining useful life estimation. *Computers in Industry*, 108:186–196, 2019.

[6] Ethem Alpaydin. *Introduction to machine learning*. MIT press, 2020.

[7] Manuel Arias Chao, Chetan Kulkarni, Kai Goebel, and Olga Fink. Aircraft engine run-to-failure dataset under real flight conditions for prognostics and diagnostics. *Data*, 6(1):5, 2021.

[8] B. Armstrong and C.C. de Wit. Friction modeling and compensation. In *The Control Handbook*. CRC Press, 1995.

[9] Nils T Basse. Turbulence intensity and the friction factor for smooth-and rough-wall pipe flow. *Fluids*, 2(2):30, 2017.

[10] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[11] Christopher M Bishop. Pattern recognition and machine learning. *Springer google schola*, 2:1122–1128, 2006.

[12] Y-Lan Boureau, Jean Ponce, and Yann LeCun. A theoretical analysis of feature pooling in visual recognition. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 111–118, 2010.

[13] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[14] Thyago P Carvalho, Fabrízzio A Soares, Roberto Vita, Roberto da P Francisco, João P Basto, and Symone GS Alcalá. A systematic literature review of machine learning methods applied to predictive maintenance. *Computers & Industrial Engineering*, 137:106024, 2019.

[15] Tianfeng Chai and Roland R Draxler. Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature. *Geoscientific model development*, 7(3):1247–1250, 2014.

[16] Junfei Chen, Qian Li, Huimin Wang, and Menghua Deng. A machine learning ensemble approach based on random forest and radial basis function neural network for risk evaluation of regional flood disaster: a case study of the yangtze river delta, china. *International journal of environmental research and public health*, 17(1):49, 2020.

[17] Xiaowu Chen and Zhen Liu. A long short-term memory neural network based wiener process model for remaining useful life prediction. *Reliability Engineering System Safety*, 226:108651, 2022.

[18] Kyunghyun Cho. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[19] François Chollet. Keras. `https://github.com/keras-team/keras`, 2015.

[20] Cyril F Colebrook and Curt M White. Experiments with fluid friction in roughened pipes. *Proceedings of the Royal Society of London. Series A-Mathematical and Physical Sciences*, 161(906):367–381, 1937.

[21] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[22] Mirko Dobrnjac. Determination of friction coefficient in transition flow region for waterworks and pipelines calculation. *Annals of the Faculty of Engineering Hunedoara*, 10(3):137, 2012.

[23] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

[24] Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*, 2016.

[25] Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

[26] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. *Neural computation*, 12(10):2451–2471, 2000.

[27] Hossein Gholizadeh, Richard Burton, and Greg Schoenau. Fluid bulk modulus: Comparison of low pressure models. *International Journal of Fluid Power*, 13(1):7–16, January 2012.

[28] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016. `http://www.deeplearningbook.org`.

[29] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10):2222–2232, 2016.

[30] Sebastian Günther, Jonathan Brandt, Astrid Bensmann, and Richard Hanke-Rauschenbach. Estss—energy system time series suite: a declustered, application-independent, semi-artificial load profile benchmark set. *Energy Informatics*, 7(1):3, 2024.

[31] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

[32] Simon S. Haykin. *Neural Networks and Learning Machines*. Pearson International Edition. Pearson, 2009.

[33] Ahmed Hinchi and Mohamed Tkiouat. Rolling element bearing remaining useful life estimation based on a convolutional long-short-term memory network. *Procedia Computer Science*, 127:123–132, 01 2018.

[34] Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. *Cited on*, 14(8):2, 2012.

[35] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[36] Chang Woo Hong, Changmin Lee, Kwangsuk Lee, Min-Seung Ko, Dae Eun Kim, and Kyeon Hur. Remaining useful life prognosis for turbofan engine using explainable deep neural networks with dimensionality reduction. *Sensors*, 20(22):6626, 2020.

[37] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257, 1991.

[38] Sadiqa Jafari and Yungcheol Byun. A cnn-gru approach to the accurate prediction of batteries' remaining useful life from charging profiles. *Computers*, 12:219, 10 2023.

[39] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, et al. *An introduction to statistical learning*, volume 112. Springer, 2013.

[40] Andrew KS Jardine, Daming Lin, and Dragan Banjevic. A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical systems and signal processing*, 20(7):1483–1510, 2006.

[41] Luis Felipe Jeréz. Detección de anomalías aplicado al mantenimiento predictivo, mediante técnicas de deep learning no supervisado, utilizando autoencoders, long short-term memory y self-attention. 2022.

[42] Samir Khan and Takehisa Yairi. A review on the application of deep learning in system health management. *Mechanical Systems and Signal Processing*, 107:241–265, 2018.

[43] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[44] Patrick Klein and Ralph Bergmann. Generation of complex data for ai-based predictive maintenance research with a physical factory model. In *ICINCO (1)*, pages 40–50, 2019.

[45] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[46] Guoxing Lan, Qing Li, and Nong Cheng. Remaining useful life estimation of turbofan engine using lstm neural networks. In *2018 IEEE CSAA Guidance, Navigation and Control Conference (CGNCC)*, pages 1–5. IEEE, 2018.

[47] A. Lavopa and M. Delera. What is the fourth industrial revolution? Industrial Analytics Platform, January 2021. `https://iap.unido.org/articles/what-fourth-industrial-revolution`.

[48] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[49] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[50] Jay Lee, Hossein Davari, Jaskaran Singh, and Vibhor Pandhare. Industrial artificial intelligence for industry 4.0-based manufacturing systems. *Manufacturing letters*, 18:20–23, 2018.

[51] Jay Lee, Hung-An Kao, and Shanhu Yang. Service innovation and smart analytics for industry 4.0 and big data environment. *Procedia cirp*, 16:3–8, 2014.

[52] Jinhao Lei, Chao Liu, and Dongxiang Jiang. Fault diagnosis of wind turbine based on long short-term memory networks. *Renewable energy*, 133:422–432, 2019.

[53] Jialin Li, Li Xueyi, and David He. A directed acyclic graph network combined with cnn and lstm for remaining useful life prediction. *IEEE Access*, PP:1–1, 05 2019.

[54] Xiang Li, Qian Ding, and Jian-Qiao Sun. Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Engineering & System Safety*, 172:1–11, 2018.

[55] Yuan Li, Huanjie Wang, Jingwei Li, and Jie Tan. A 2-d long short-term memory fusion networks for bearing remaining useful life prediction. *IEEE Sensors Journal*, 22(22):21806–21815, 2022.

[56] Zachary Chase Lipton. A critical review of recurrent neural networks for sequence learning. *arXiv Preprint, CoRR, abs/1506.00019*, 2015.

[57] Mary Ann Lundteigen and Marvin Rausand. Failures and failure classification. `https://www.ntnu.edu/documents/624876/1277046207/SIS+book+-+chapter+03+-+failures+and+failure+classification/36f29566-bd55-4a91-b002-1e17a177c035`, 2024.

[58] Batta Mahesh. Machine learning algorithms-a review. *International Journal of Science and Research (IJSR).[Internet]*, 9(1):381–386, 2020.

[59] Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. Long short term memory networks for anomaly detection in time series. 04 2015.

[60] Dominic Masters and Carlo Luschi. Revisiting small batch training for deep neural networks. *arXiv preprint arXiv:1804.07612*, 2018.

[61] Kamal Medjaher, Noureddine Zerhouni, and Rafael Gouriveau. *From prognostics and health systems management to predictive maintenance 1: Monitoring and prognostics*. John Wiley & Sons, 2016.

[62] Steve Miller. Predictive maintenance in a hydraulic pump. `https://github.com/mathworks/Simscape-Triplex-Pump/releases/tag/23.2.2.6`, 2024. Recuperado March 28, 2024.

[63] Tom M Mitchell and Tom M Mitchell. *Machine learning*, volume 1. McGraw-hill New York, 1997.

[64] R. Keith Mobley. 1 - impact of maintenance. In R. Keith Mobley, editor, *An Introduction to Predictive Maintenance (Second Edition)*, Plant Engineering, pages 1–22. Butterworth-Heinemann, Burlington, second edition, 2002.

[65] Nurhazimah Nazmi, Mohd Azizi Abdul Rahman, Shin-Ichiroh Yamamoto, Siti Anom Ahmad, Hairi Zamzuri, and Saiful Amri Mazlan. A review of classification techniques of emg signals during isotonic and isometric contractions. *Sensors*, 16(8):1304, 2016.

[66] Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.

[67] Jiahe Niu, Chongdang Liu, Linxuan Zhang, and Yuan Liao. Remaining useful life prediction of machining tools by 1d-cnn lstm network. In *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1056–1063. IEEE, 2019.

[68] Jorge Nocedal and Stephen J Wright. *Numerical optimization*. Springer, 1999.

[69] Nicolás Oyharcabal Astorga. Convolutional recurrent neural networks for remaining useful life prediction in mechanical systems. 2018.

[70] Joao Pereira and Margarida Silveira. Unsupervised anomaly detection in energy time series data using variational recurrent autoencoders with attention. In *2018 17th IEEE international conference on machine learning and applications (ICMLA)*, pages 1275–1282. IEEE, 2018.

[71] Alejandro Jorge Pistarelli. 1 - mantenimiento y disponibilidad. In *Manual de mantenimiento ingeniería, gestión y organización*, pages 15–67. Talleres Gráficos R y C, Buenos Aires, 2010.

[72] Alejandro Jorge Pistarelli. 3 - mantenimiento predictivo y proactivo. In *Manual de mantenimiento ingeniería, gestión y organización*, pages 121–145. Talleres Gráficos R y C, Buenos Aires, 2010.

[73] Boris T Polyak. Some methods of speeding up the convergence of iteration methods. *Ussr computational mathematics and mathematical physics*, 4(5):1–17, 1964.

[74] Qinglin Qi and Fei Tao. Digital twin and big data towards smart manufacturing and industry 4.0: 360 degree comparison. *Ieee Access*, 6:3585–3593, 2018.

[75] Somayeh Bakhtiari Ramezani, Logan Cummins, Brad Killen, Richard Carley, Amin Amirlatifi, Shahram Rahimi, Maria Seale, and Linkan Bian. Scalability, explainability and performance of data-driven algorithms in predicting the remaining useful life: A comprehensive review. *Ieee Access*, 2023.

[76] Yongyi Ran, Xin Zhou, Pengfeng Lin, Yonggang Wen, and Ruilong Deng. A survey of predictive maintenance: Systems, purposes and approaches. *arXiv preprint arXiv:1912.07383*, 2019.

[77] Waseem Rawat and Zenghui Wang. Deep convolutional neural networks for image classification: A comprehensive review. *Neural computation*, 29(9):2352–2449, 2017.

[78] BigData Republic. Machine learning for predictive maintenance: where to start? *bigdatarepublic*, 2017.

[79] Persistence Market Research. Global market study on safety sensors and switches: Significant demand for safety sensors and switches for leak detection to be observed in the coming years. https://www.persistencemarketresearch.com/market-research/safety-sensors-and-switches-market.asp, March 2018.

[80] Osborne Reynolds. Xxix. an experimental investigation of the circumstances which determine whether the motion of water shall be direct or sinuous, and of the law of resistance in parallel channels. *Philosophical Transactions of the Royal society of London*, (174):935–982, 1883.

[81] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

[82] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[83] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[84] A. L. Samuel. Some studies in machine learning using the game of checkers. ii—recent progress. *IBM Journal of Research and Development*, 11(6):601–617, 1967.

[85] Abhinav Saxena, Kai Goebel, Don Simon, and Neil Eklund. Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 international conference on prognostics and health management*, pages 1–9. IEEE, 2008.

[86] Dominik Scherer, Andreas Müller, and Sven Behnke. Evaluation of pooling operations in convolutional architectures for object recognition. In *International conference on artificial neural networks*, pages 92–101. Springer, 2010.

[87] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.

[88] Klaus Schwab. *The fourth industrial revolution*. Crown Currency, 2017.

[89] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[90] Brandon Da Silva. Playing super mario bros with proximal policy optimization. *Machine Learning Blog*, 2018.

[91] Leslie N Smith. Cyclical learning rates for training neural networks. In *2017 IEEE winter conference on applications of computer vision (WACV)*, pages 464–472. IEEE, 2017.

[92] Symone Gomes Soares. *Ensemble learning methodologies for soft sensor development in industrial processes*. PhD thesis, 2015.

[93] Society of Automotive Engineers. *SAE JA1011: Evaluation Criteria for Reliability-Centered Maintenance (RCM) Processes*. SAE International, Warrendale, 2 edition, 2009.

[94] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[95] Gian Antonio Susto, Andrea Schirru, Simone Pampuri, Seán McLoone, and Alessandro Beghi. Machine learning for predictive maintenance: A multiple classifier approach. *IEEE transactions on industrial informatics*, 11(3):812–820, 2014.

[96] Keras team. Keras documentation: Modelcheckpoint callback. `https://keras.io/api/callbacks/model_checkpoint/`, 2021.

[97] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio, 2016.

[98] Laurens Van Der Maaten, Eric Postma, Jaap Van den Herik, et al. Dimensionality reduction: a comparative. *J Mach Learn Res*, 10(66-71):13, 2009.

[99] Ali R Vatankhah. Comment on "gene expression programming analysis of implicit colebrook–white equation in turbulent flow friction factor calculation". *Journal of Petroleum Science and Engineering*, 124:402–405, 2014.

[100] Chenyang Wang, Chaolin Yin, Shan Bai, and Yongming Han. Remaining useful life prediction of rolling bearings using correlation coefficient based long short-term memory neural network. In *2022 IEEE 11th Data Driven Control and Learning Systems Conference (DDCLS)*, pages 585–589. IEEE, 2022.

[101] Hai-Kun Wang, Yi Cheng, and Ke Song. Remaining useful life estimation of aircraft engines using a joint deep learning model based on tcnn and transformer. *Computational Intelligence and Neuroscience*, 2021(1):5185938, 2021.

[102] Shunli Wang, Yongcun Fan, Siyu Jin, Paul Takyi-Aninakwa, and Carlos Fernandez. Improved anti-noise adaptive long short-term memory neural network modeling for the robust remaining useful life prediction of lithium-ion batteries. *Reliability Engineering & System Safety*, 230:108920, 2023.

[103] Waloddi Weibull. A statistical distribution function of wide applicability. *Journal of applied mechanics*, 1951.

[104] Sheng Xiang, Yi Qin, Jun Luo, Huayan Pu, and Baoping Tang. Multicellular lstm-based deep learning model for aero-engine remaining useful life prediction. *Reliability Engineering & System Safety*, 216:107927, 2021.

[105] Sheng Xiang, Yi Qin, Caichao Zhu, Yangyang Wang, and Haizhou Chen. Long short-term memory neural network with weight amplification and its application into gear remaining useful life prediction. *Engineering Applications of Artificial Intelligence*, 91:103587, 2020.

[106] Li Da Xu, Eric L Xu, and Ling Li. Industry 4.0: state of the art and future trends. *International journal of production research*, 56(8):2941–2962, 2018.

[107] Muhamad Yani, Si MT Budhi Irawan, S, and MT Casi Setiningsih, ST. Application of transfer learning using convolutional neural network method for early detection of terry's nail. In *Journal of Physics: Conference Series*, volume 1201, page 012052. IOP Publishing, 2019.

[108] Hu Yu, Xiaodong Miao, Zhiwen Xun, Fan Ping, and Ze Wu. Remaining useful life prognosis for rolling bearings based on bi-convlstm. pages 1–5, 10 2022.

[109] Lotfi ZEGHADNIA. *Computation of the pressurized turbulent flow in circular pipe*. PhD thesis, Badji Mokhtar university-Algeria, 2007.

[110] Lotfi Zeghadnia, Jean Loup Robert, and Bachir Achour. Explicit solutions for turbulent flow friction factor: A review, assessment and approaches classification. *Ain Shams Engineering Journal*, 10(1):243–252, 2019.

[111] Xiaoyang Zhang, Xin Lu, Weidong Li, and Sheng Wang. Prediction of the remaining useful life of cutting tool using the hurst exponent and cnn-lstm. *The International Journal of Advanced Manufacturing Technology*, 112:2277–2299, 2021.

[112] Yang Zhang, Paul Hutchinson, Nicholas AJ Lieven, and Jose Nunez-Yanez. Remaining useful life estimation using long short-term memory neural networks and deep fusion. *IEEE access*, 8:19033–19045, 2020.

[113] Yongzhi Zhang, Rui Xiong, Hongwen He, and Michael G Pecht. Long short-term memory recurrent neural network for remaining useful life prediction of lithium-ion batteries. *IEEE Transactions on Vehicular Technology*, 67(7):5695–5705, 2018.

[114] Rui Zhao, Ruqiang Yan, Zhenghua Chen, Kezhi Mao, Peng Wang, and Robert X Gao. Deep learning and its applications to machine health monitoring. *Mechanical Systems and Signal Processing*, 115:213–237, 2019.

[115] Zhibin Zhao, Jingyao Wu, David Wong, Chuang Sun, and Ruqiang Yan. Probabilistic remaining useful life prediction based on deep convolutional neural network. 2020.

[116] Danhua Zhou, Zhanying Li, Jiali Zhu, Haichuan Zhang, and Lin Hou. State of health monitoring and remaining useful life prediction of lithium-ion batteries based on temporal convolutional network. *IEEE Access*, 8:53307–53320, 2020.

[117] Antonio Marín Écija. Proyecto de machine learning en mantenimiento: Cómo enfocarlo para el éxito. Technical report, Data Science Energy - Advanced Analytics, Octubre 2019. https://www.dscenergy.es/presentations.

Capítulo 7

Anexos

En el desarrollo de este trabajo de titulación, la implementación y posterior resolución del sistema de ecuaciones que simula la operación y degradación del sistema de las bombas tríplex se realizó utilizando el lenguaje de programación Python en una serie de notebooks de Jupyter. Todos los notebooks utilizados para la simulación del activo, además de los notebooks utilizados tanto en el preprocesamiento de los datos como en el desarrollo y posterior entrenamiento de los modelos predictivos basados en redes CNN y LSTM se encuentran en el siguiente repositorio de Github:

- `https://github.com/ignaciopablo357/Triplex_Pump_R2F_pred`