PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA INFORMÁTICA

Diagnosticador de severidad de fallos de rodamientos bajo velocidades variables utilizando la transformada de Fourier y Deep learning

Sebastián Ignacio López Norambuena

Profesor guía: **Nibaldo Rodríguez Agurto**

Profesor co-referente: **Guillermo Cabrera Guerrero**

Septiembre, 2017

Resumen

El rodamiento es uno de los componentes más utilizados en la maquinaria rotativa. Las fallas en estos elementos son muy comunes y tienen un gran impacto en la cadena de producción de una planta. Por lo tanto, es importante estudiar la tecnología de diagnóstico de fallas en rodamientos e incluirla en los planes de mantenimiento. Una señal de vibración perteneciente a un rodamiento, transporta información dinámica sobre el estado estado de salud de este; así que puede ser utilizada para identificar y clasificar fallas en la estructura de los rodamientos. Para automatizar este proceso, la ingeniería ha propuesto algoritmos de *machine learning* que combinados con técnicas de extracción de características; entregan un diagnóstico óptimo sobre el estado de los rodamientos. Para mejorar el rendimiento de estos modelos cuando la dimensión de los datos de entrada es muy alta, se incluyen técnicas de reducción de dimensionalidad o selección de características. En este documento se propone utilizar esta combinación de técnicas de extracción de características y algoritmos de *deep learning* que reducen la dimensionalidad y realizan un diagnostico de la salud del rodamiento. En la etapa de extracción, se aplicará la transformada rápida de Fourier a una señal de vibración; para obtener la magnitud de los coeficientes y emplearlos como características. Posteriormente una red neuronal profunda compuesta por dos sparse autoencoders y una capa de neuronas con la función de activación softmax, se encargarán diagnosticar la salud del rodamiento.

Palabras Clave: rodamientos, diagnóstico, mantenimiento, señal de vibración, deep learning.

Índice

Lista de Figuras

Lista de Tablas

1 Introducción

La maquinaria rotativa es ampliamente utilizada en muchos campos industriales y en los últimos años el monitoreo de condiciones de estas máquinas ha cobrado especial importancia en el mantenimiento moderno. Esto se explica porque existe la necesidad de reducir los tiempos de inactividad no programados y los costos asociados al mantenimiento, de manera que se pueda mantener la competitividad corporativa [1]. Con el monitoreo de condiciones se pueden detectar tempranamente fallas potencialmente desastrosas, reduciendo el tiempo de inactividad total de la máquina y de operaciones enteras.

Las técnicas de mantenimiento predictivo han demostrado ser estrategias eficaces para reducir fallas inesperadas en la maquinaria. El monitoreo de vibraciones es la tecnología de mantenimiento predictivo más utilizada debido a la cantidad significativa de información que porta la señal sobre la salud de la maquinaria [2]. Considerando que los rodamientos son un componente frecuentemente encontrado en la maquinaria rotativa y que pueden causar alrededor del 40-50% [10] de todos los desperfectos, muchas investigaciones se centran en el monitoreo de vibraciones para estos componentes.

Los rodamientos giran constantemente en condiciones ambientales severas tales como alta temperatura, velocidad de rotación variable y grandes cargas, por lo que presentan una alta frecuencia de rotura [3]. Realizar un diagnóstico temprano en estos elementos es difícil, la señal de vibración tiene un comportamiento no lineal, no estacionario y además presenta ruido [4]. Este comportamiento de la señal es consecuencia de la compleja estructura mecánica del rodamiento y sus duras condiciones de trabajo.

En general el diagnóstico de fallas está compuesto por dos etapas: la primera etapa consiste en extraer características útiles desde los datos recolectados con alguna herramienta de procesamiento de señales, y la segunda etapa es la clasificación de fallas basado en las características obtenidas en el paso anterior. Obtener las características es la etapa clave en el diagnóstico de fallas para maquinaria rotativa [5] porque reduce la dimensión de los datos, afecta directamente el reconocimiento de patrones y por lo tanto el desempeño del clasificador. Los métodos convencionales de extracción de características incluyen técnicas en el dominio del tiempo, dominio de la frecuencia y tiempo frecuencia.

Diversos autores utilizan estadísticos basados en la forma de la onda en el dominio del tiempo, tales como el valor pico, la media cuadrática, el factor de cresta, el índice de curtosis, la asimetría y la entropía [9]. Los principales métodos en el dominio del tiempo corresponden a: el calculo del espectro energético de Fourier [6], el espectro de potencia [4] y el análisis cepstral. Respecto a las técnicas de tiempo-frecuencia suelen ser utilizadas la transformada de Fourier de tiempo corto, la distribución Wigner-Ville y la transformada de paquetes wavelet. También es posible encontrar otros métodos basados en procesamiento de señales como la descomposición modal empírica (EMD) [7], funciones modales intrínsecas (IMF), la transformada wavelet discreta (DWT), la transformada wavelet (WT) [16] y la transformada Hilbert-Huang (HHT) [8].

Respecto a los clasificadores inteligentes utilizados en estas investigaciones, podemos encontrar redes neuronales artificiales (ANN), máquinas de vectores de soporte (SVM) [19], sistemas adaptativos de inferencia neurodifusa [10] y otros. Estas herramientas reciben un entrenamiento con las características obtenidas de la señal, que ajusta diversos parámetros de sus algoritmos, y que hace posible que aprenda patrones ocultos en los datos. Una vez realizado el entrenamiento, el clasificador está listo para que, con un nuevo conjunto de atributos de la señal sea capaz de distinguir si el componente analizado tiene desperfectos o no.

Una de las principales limitaciones que poseen los trabajos previamente descritos, es que sus clasificadores están limitados al problema de estimación de parámetros y cuando son muchos, se requiere de un mayor volumen de datos para el entrenamiento, empeorando el desempeño del algoritmo. Además, estas investigaciones solo clasifican la señal en 4 estados, que corresponden a desperfectos en el anillo interno, anillo externo, bola y saludable. Como consecuencia de esto, se deja de lado en la clasificación la severidad de las fallas en los componentes del rodamiento.

En función de las investigaciones anteriormente descritas, es que se propone un modelo de diagnóstico de fallas para rodamientos con la transformada de Fourier como técnica de extracción de caracterizaras de la señal, y una red neuronal profunda como clasificador. A diferencia de los instrumentos de clasificación mencionados previamente, la red neuronal profunda utiliza *sparse* autoencoders que reducen la dimensión de las características a la vez que realizan un pre entrenamiento para la capa que realiza la clasificación, lo que mejora el tiempo de entrenamiento y la exactitud de los resultados. Además, se propone la importante adición de las severidades de falla a cada componente del rodamiento, mejorando el nivel de detalle en el que la red neuronal profunda diagnostica las fallas.

2 Definición de objetivos

En esta sección se definen los objetivos general y específicos que persigue la ejecución de este proyecto.

2.1 Objetivo general

Desarrollar un diagnosticador de severidad de fallos de rodamientos, bajo velocidades variables utilizando la transformada de Fourier discreta.

2.2 Objetivos específicos

Los objetivos específicos de este proyecto son los siguientes:

1. Descomponer señales de vibración con la transformada de Fourier discreta.
2. Diseñar e implementar un algoritmo de *deep learning* con sparse autoencoders.
3. Evaluar la exactitud del diagnosticador con 4 estados de salud del rodamiento.
4. Evaluar la exactitud del diagnosticador con 10 estados de salud del rodamiento.

3 Plan de trabajo

Este plan de trabajo utiliza los objetivos específicos como una base, debido a que estos explican las tareas que se deben llevar a cabo para cumplir el objetivo general. Esta segunda entrega, contempla la implementación del modelo propuesto y la evaluación del rendimiento de este. Para la próxima entrega se habrá pulido el marco teórico y la redacción de la investigación en general, sobre todo la discusión de resultados y el estado del arte.

4 Marco Teórico

Esta sección tiene como objetivo presentar antecedentes, conceptos, definiciones y la importancia de las herramientas y técnicas que se utilizarán en el desarrollo de esta investigación. Comenzando por los antecedentes de la utilización de enfoques en el dominio de la frecuencia, seguido del aprendizaje profundo y finalmente el sustento matemático de ambos.

En la literatura especializada existen antecedentes que sirven como base para esta investigación, un ejemplo de estos es [20], donde se utiliza la transformada de Fourier para extraer características de una señal y usar la energía contenida en el espectro de frecuencias como entrada a 3 clasificadores: una red neuronal artificial, autoencoders apilados y una red con *extreme learning machine*. Este ejemplo corresponde a un excelente antecedente debido a que implementa la misma técnica de extracción de características y clasificador que este trabajo, con la salvedad que sus autoencoders no incluyen la dispersión de los nodos en la capa oculta, variación denominada *sparse* autoencoder y que si se usa en esta investigación.

Otro ejemplo de un antecedente es [21] en donde se postula que el análisis espectral o dominio de la frecuencia, es el enfoque más utilizado para detectar fallas en los rodamientos. Dado que los enfoques en el dominio del tiempo solo pueden detectar cuándo ocurre una falla en el rodamiento, mientras que el análisis espectral además puede determinar en qué componente del rodamiento se encuentra el desperfecto. También se explica que la transformada rápida de Fourier es el algoritmo computacionalmente eficiente para convertir una señal en el dominio del tiempo a componentes de frecuencia discretos.

En cuanto a los antecedentes de redes neuronales profundas podemos encontrar a [9], donde proponen la utilización de una red neuronal profunda sin autoencoders para diagnosticar una señal de vibración de un rodamiento. Los autores no utilizan métodos de extracción de características, y solo dividen en segmentos la señal original para entregársela a la red neuronal profunda. El argumento de los autores es que se debe mantener la coherencia temporal de la señal, pero la cantidad de datos que recibe la red es enorme, resultando en grandes tiempos de entrenamiento y una gran capacidad de cómputo.

A continuación se encuentran los conceptos y definiciones matemáticas de las herramientas que se utilizarán en esta investigación, comenzando por la transformada de Fourier y sus variantes, seguido del *deep learning* (o aprendizaje profundo), los autoencoders y su variante *sparse* autoencoder.

4.1 Transformada de Fourier

La transformada de Fourier es una herramienta matemática extremadamente poderosa, por lo que es utilizada en diversos campos de la ciencia y la ingeniería. Puede ser vista como la transformación de una señal en un dominio (típicamente tiempo o espacio) a otro dominio, el dominio de la frecuencia. Al aplicar la transformada de Fourier a una señal se obtienen sus componentes fundamentales en frecuencia que a menudo revelan estructuras ocultas, que en el dominio del tiempo, no son posibles de observar.

Como consecuencia de lo anterior, el análisis de Fourier está en la base de muchas teorías de la ciencia y juega un papel fundamental en el diseño práctico de la ingeniería. El alcance actual de la influencia del análisis de Fourier está indicado por una lista parcial de científicos e ingenieros que la utilizan [12]:

- Los ingenieros de audio utilizan técnicas de Fourier, en parte porque el oído parece ser sensible al comportamiento del dominio de frecuencia.
- Los estadísticos y probabilistas caracterizan y calculan las distribuciones de probabilidad usando transformadas de Fourier. Las transformadas de Fourier de las funciones de covarianza se utilizan para caracterizar y estimar las propiedades de los procesos aleatorios.
- Los radioastrónomos utilizan la transformada de Fourier para formar imágenes a partir de datos interferométricos recogidos por arreglos de antenas.
- Los cristalógrafos encuentran las estructuras cristalinas usando transformadas de Fourier de los patrones de difracción de rayos X.

- Los ingenieros biomédicos recogen datos en el dominio de la frecuencia y luego los transforman inversamente con Fourier para obtener imágenes de resonancia magnética.

4.1.1 Transformada de Fourier continua

La transformada de Fourier continua es resultado de la generalización de las series de Fourier. Puede aplicarse prácticamente a cualquier función, no requiere que la función sea periódica, y para datos discretos, puede ser evaluada rápidamente usando una técnica moderna llamada transformada de Fourier Rápida (FFT). La transformada de Fourier de una función $f(t)$ se define [11] como:

$$F(\omega) = \int_{-\infty}^{\infty} f(t)e^{-i\omega t}dt \tag{1}$$

Donde ω es igual a la frecuencia angular $2\pi f$, y $F(\omega)$ es una función continua de valores complejos. Una vez que se ha determinado una transformada de Fourier, la función original $f(t)$ puede ser recuperada a partir de la transformada de Fourier inversa:

$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(\omega)e^{i\omega t}d\omega \tag{2}$$

4.1.2 Transformada de Fourier discreta

La transformada discreta de Fourier (DFT) (Ecc. 3), como lo indica su nombre, opera sobre un número finito de datos y es discreta en ambos dominios, el del tiempo y el de la frecuencia. La DFT trabaja sobre un bloque de N muestras de la secuencia $x(n)$ y entrega N componentes de frecuencia indexadas mediante la variable entera k.

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{-j(2\pi/N)kn} \quad , 0 \leq k \leq N-1 \tag{3}$$

La transformadas discreta de Fourier inversa, o IDFT está dada por la Ecc. 4, la cual recibe N componentes de frecuencia y entrega N muestras de la secuencia $x(n)$. La fórmula es muy similar a la de la DFT, se distingue en el signo del exponente de e y en la presencia del factor de normalización $1/N$. Ambas diferencias son necesarias para que se cumpla la condición de la Ecc. 5.

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)e^{j(2\pi/N)kn} \quad , 0 \leq n \leq N-1 \tag{4}$$

$$x(n) = IDFT[DFT[x(n)]] \tag{5}$$

4.1.3 Transformada de Fourier de tiempo corto

Consideremos una señal $x(t)$ y supongamos que es estacionaria cuando se ve a través de una ventana $w(t)$, cuya duración es T, centrada en la posición de tiempo τ, la transformada de Fourier de la señal en ventana $x(t)$ es la transformada de Fourier de corto tiempo [15] (STFT):

$$X(\tau,\omega) = \int_{-\infty}^{\infty} x(t)w(t-\tau)e^{-j\omega t}dt \tag{6}$$

Esta transformada mapea la señal desde el dominio del tiempo en un plano de tiempo-frecuencia conjunto (τ, f). Para la implementación, la versión discreta de la Ecc. 6 debe utilizarse como:

$$X(k) = \sum_{i=n-\frac{N}{2}}^{i=n+\frac{N}{2}} x(i)w(i-n)e^{-j\frac{2\pi i}{N}} \tag{7}$$

Donde n y k $(1 \leq k \leq N)$ son el tiempo discreto y la frecuencia respectivamente, y N es la longitud de la ventana

4.2 Deep learning

En los años recientes un nuevo enfoque de *machine learning* denominado *deep learning*, ha llamado la atención de investigadores de múltiples áreas debido a una serie de ventajas respecto a otros enfoques. El *deep learning* ha tenido un gran impacto en trabajos actuales de reconocimiento de voz, visión artificial y procesamiento del lenguaje natural, debido a la efectividad empírica de éste respecto a enfoques alternativos en este ámbito. El desarrollo de esta tecnología fue potenciado en gran medida por la creciente capacidad de cómputo de los equipos, y por el uso de los inmensos volúmenes de datos que son generados todos los días por la humanidad.

Esta ola de interés por el *deep learning*, se centra principalmente en los perceptrón multicapa profundo (deep multilayer perceptron), las redes convolucionales profundas (deep convolutional network) y las redes neuronales recurrentes (recurrent neuronal network) de acuerdo a [22]. Sin embargo, la mayoría de los métodos de *deep learning* se basan en el perceptrón multicapa profundo, porque lo utilizan como el bloque esencial de construcción. El *deep learning* formula el problema subyacente como una arquitectura en forma de red, en la que su capa de salida define una función de costos (también llamada función de perdida) que es necesaria para aprender.

El desarrollo del *deep learning* está relacionado con uso de técnicas de aprendizaje no supervisado como los diferentes tipos de autoencoder que existen para realizar un pre entrenamiento de las capas ocultas. Y es que uno de los principales propósitos del aprendizaje no supervisado es producir buenas representaciones de los datos [24] que puedan ser usadas para la detección, reconocimiento, predicción o visualización. Las buenas representaciones eliminan variaciones irrelevantes de los datos de entrada, mientras que se preserva la información que es útil para la tarea a realizar.

El aprendizaje no supervisado tiene la habilidad de crear jerarquías de características profundas (deep features hierarchies) mediante el apilado de módulos no supervisados uno encima de otro, habilidad que es frecuentemente utilizada en el *deep learning*. Un módulo no supervisado

de un nivel en la jerarquía se alimenta con los vectores de representación producidos por el nivel inferior. Las representaciones de nivel superior captan las dependencias de alto nivel entre las variables de entrada, mejorando así la capacidad del sistema para capturar las regularidades subyacentes en los datos.

4.2.1 Autoencoders

Un autoencoder es una red neuronal artificial no supervisada de tres capas que es entrenada para replicar los datos de entrada en la salida. El entrenamiento de los autoencoder es no supervisado en el sentido de que no requiere datos etiquetados para aprender a replicarlos en la salida. Este tipo especial de red neuronal es utilizada para pre entrenar las capas ocultas de las redes neuronales profundas, de manera que encuentran pesos sinápticos que agilizan la búsqueda de mínimos locales para la función de costos, mejoran la capacidad de generalización en la tarea de clasificar y reducen la dimensión de los datos de entrada a medida que se transforman en las capas ocultas.

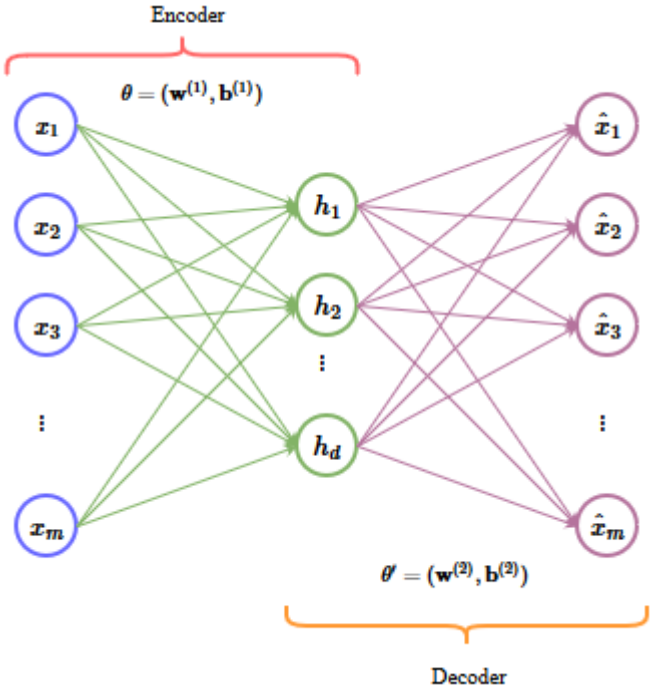

Figura 1: Diagrama que ilustra la estructura de un autoencoder formado por la red encoder (lado izquierdo) y la red decoder (lado derecho) con sus respectivos parámetros de entrenamiento.

Es posible identificar dos estructuras en la red autoencoder, por una parte el codificador en el lado izquierdo de la Fig. 1, que mapea los datos de entrada desde un espacio de alta dimensión en códigos de un espacio de baja dimensión. Por otra parte, que se encuentra al lado derecho de la Fig. 1, reconstruye los datos de entrada a partir de los códigos obtenidos por la capa oculta llevándolos a su dimensión original [13]. Ahora que está claro cuál es el encoder y el decoder, podemos definir matemáticamente ambas estructuras.

Dadas las muestras de entrenamiento $\mathbf{x} = (x_1, x_2, ..., x_n)$, donde la muestra $x_n \in \mathbb{R}^m$, se define la red encoder como una función codificadora f_θ (Ecc. 9) cuya tarea es representar estas muestras en un vector con códigos de menor dimensión denominado $\mathbf{h} = (h_1, h_2, ..., h_n)$, en donde $h_n \in \mathbb{R}^d$ y $d < m$. La matriz de pesos sinápticos de la función encoder corresponde a $\mathbf{w}^{(1)} = (w_1^{(1)}, w_2^{(1)}, ..., w_m^{(1)})$ con $w_m^{(1)} \in \mathbb{R}^d$, mientras que $\mathbf{b}^{(1)} = (b_1^{(1)}, b_2^{(1)}, ..., b_d^{(1)})^T$ representa el arreglo de *biases* pertenecientes a las neuronas de la capa oculta. Los pesos sinápticos y los *biases* conforman θ, el conjunto de parámetros a minimizar en la función de costos que será detallada más adelante.

$$\mathbf{h} = f_\theta(\mathbf{x}) \tag{8}$$

$$f_\theta(\mathbf{x}) = s_f(\mathbf{w}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}) \tag{9}$$

$$\theta = (\mathbf{w}^{(1)}, \mathbf{b}^{(1)}) \tag{10}$$

La red decoder se define como una función de reconstrucción denotada como $g_{\theta'}$ (Ecc. 12), dicha función mapea el vector $\mathbf{h}$ desde el espacio de baja dimensión en el que se encuentra, de vuelta al espacio de alta dimensión original realizando así la reconstrucción. Los datos reconstruidos se denotan mediante $\hat{\mathbf{x}} = (\hat{x}_1, \hat{x}_2, ..., \hat{x}_n)$ (Ecc. 11), en donde $\hat{x}_n \in \mathbb{R}^m$. El primer parámetro que compone a θ' perteneciente al decoder, es la matriz de pesos sinápticos de salida $\mathbf{w}^{(2)} = (w_1^{(2)}, w_2^{(2)}, ..., w_m^{(2)})$ con $w_m^{(2)} \in \mathbb{R}^d$. Mientras que el segundo, es el arreglo de *biases* $\mathbf{b}^{(2)} = (b_1^{(2)}, b_2^{(2)}, ..., b_d^{(2)})^T$.

$$\hat{\mathbf{x}} = g_{\theta'}(\mathbf{h}) \tag{11}$$

$$g_{\theta'}(\mathbf{h}) = s_g(\mathbf{w}^{(2)}\mathbf{h} + \mathbf{b}^{(2)}) \tag{12}$$

$$\theta' = (\mathbf{w}^{(2)}, \mathbf{b}^{(2)}) \tag{13}$$

Los conjuntos de parámetros θ y θ' de las redes encoder y decoder respectivamente, son aprendidos simultáneamente en la tarea de reconstruir en la mayor medida posible los datos originales de entrada. En otras palabras, es deseable obtener el menor error de reconstrucción (Ecc. 14) posible a través de las n muestras de entrenamiento, utilizando el error cuadrático medio (MSE). Este estimador es la función de costo estándar para los autoencoder y muchos otros algoritmos de *machine learning* [22].

$$J_{mse}(\theta, \theta') = \frac{1}{2n} \sum_{i=1}^{n} \|x_i - \hat{x}_i\|^2 \tag{14}$$

El proceso de aprendizaje del autoencoder puede ser visto como un problema de optimización porque tiene como objetivo encontrar los conjuntos de parámetros θ y θ' que minimizan el error de reconstrucción. La función de costos que debemos optimizar (Ecc. 16), también llamada función de pérdida, se compone por el error de reconstrucción (Ecc. 14) y otro término

llamado *weight decay* (Ecc. 15), que penaliza los pesos de entrada y salida para evitar el sobre entrenamiento u *overfitting*.

$$J_{weight}(\theta, \theta') = \frac{\lambda}{2} \sum_{l=1}^{2} \sum_{i=1}^{s_l} \sum_{j=1}^{s_{l+1}} (w_{ji}^{(l)})^2 \quad s_1 = m, s_2 = d, s_3 = m. \tag{15}$$

El término *weight decay* de acuerdo a [23], fuerza a algunos de los pesos sinápticos a tomar valores cercanos a 0, mientras que permite a otros pesos a mantener sus relativos grandes valores. Algunos pesos sinapticos tienen muy poca influencia en el desempeño de la red neuronal y debido a que tienen una pobre capacidad de generalización, provocan que la red se sobre entrene. En la Ecc. 15, β comúnmente parámetro de ajuste de pesos, es el encargado de penalizar a los pesos sinápticos.

$$\phi_{AE}(\theta, \theta') = J_{mse}(\theta, \theta') + J_{weight}(\theta, \theta') \tag{16}$$

Ahora que hemos definido la función de costo (Ecc. 16) de los autoencoder estándar, generalmente se utiliza el algoritmo *backpropagation* como en muchas otras técnicas de *machine learning* con el fin de actualizar sus pesos sinápticos y *biases* y así disminuir en la mayor medida posible el error de reconstrucción. Este algoritmo calcula el gradiente a la función de costos para encontrar direcciones que permiten actualizar los pesos y *biases* iniciales, de manera que con cada iteración se acerquen más al mínimo global de la función. Una vez hallados los parámetro óptimos, la capa oculta **h** del autoencoder es capaz de representar a los datos de entrada en una dimensión menor que la original sin perder información relevante para el reconocimiento de patrones.

4.2.2 Sparse autoencoders

Podemos definir al *sparse* autoencoder como una extensión del autoencoder estándar, que obliga a sus unidades de la capa oculta a mantener una baja activación media. Creando así características dispersas que logran una mejor generalización por parte del clasificador [25]. El *sparse* autoencoder impone un término de penalización a las unidades de la capa oculta, forzando a que la activación media de cada unidad sea cercana al parámetro de dispersión ρ que generalmente toma un valor muy pequeño cercano a 0.

$$\phi_{SAE}(\theta, \theta') = \phi_{AE}(\theta, \theta') + J_{sparse}(\theta, \theta') \tag{17}$$

La restricción se hace efectiva agregando el término de penalización a la función de costos del autoencoder estándar ϕ_{AE} (Ecc. 16), resultando en una nueva función de costos ϕ_{SAE} (Ecc. 17).

$$J_{sparse}(\theta, \theta') = \beta \sum_{j=1}^{d} KL(\rho || \hat{\rho}_j) \tag{18}$$

$$KL(\rho || \hat{\rho}_j) = \rho \log \frac{\rho}{\hat{\rho}_j} + (1 - \rho) \log \frac{1 - \rho}{1 - \hat{\rho}_j} \tag{19}$$

$$\hat{\rho}_j = \frac{1}{n}\sum_{k=1}^{n} h_{j,k} \tag{20}$$

J_{sparse} corresponde al término incorporado que impone la restricción de dispersión, en donde β controla el peso de este nuevo término y $KL(\rho||\hat{\rho}_j)$ (Ecc. 19) es la divergencia Kullback-Leibler entre una variable aleatoria Bernoulli con media ρ y una variable aleatoria Bernouilli media $\hat{\rho}_j$. La divergencia KL es una función estándar para medir la diferencia entre dos distribuciones de probabilidad diferentes. Definiremos $\hat{\rho}_j$ (Ecc. 20) como la probabilidad media de activación en la j-ésima unidad de la capa oculta $\mathbf{h}$ a través las n muestras, y ρ la probabilidad de activación deseada ($\rho \ll 1$) [14].

Una vez que hemos definido el término de dispersión y lo hemos incorporado a la función de costos del autoencoder (Ecc. 21), aplicaremos el método del gradiente conjugado para actualizar los pesos y *biases*. El método del gradiente conjugado pertenece a una clase de métodos de optimización de segundo orden conocidos colectivamente como métodos de dirección conjugada [23]. La motivación para utilizar este método es que converge mas rápido hacia un mínimo que el descenso del gradiente, y a la vez se evitan los requisitos de computación y almacenamiento del hessiano, como lo requiere el método de Newton.

$$\phi_{SAE}(\theta,\theta') = \frac{1}{2n}\sum_{i=1}^{n}\|x_i - \hat{x}_i\|^2 + \frac{\lambda}{2}\sum_{l=1}^{2}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}(w_{ji}^{(l)})^2 + \beta\sum_{j=1}^{d} KL(\rho||\hat{\rho}_j) \tag{21}$$

Para simplificar la notación del método definiremos Θ (Ecc. 22) como un arreglo que contiene los parámetros de la función de costos. Según la Ecc. 23, este método toma los parámetros iniciales y los actualiza sumándoles una dirección multiplicada por una tasa de aprendizaje denominada η. El arreglo que contiene las direcciones (Ecc. 24) se obtiene utilizando la dirección previa multiplicada por el parámetro conjugado γ más el residuo (Ecc. 25). El arreglo gradiente (Ecc. 26) está compuesto por las derivadas parciales de la función de costos, a la vez que la derivada parcial $\frac{\partial \phi_{SAE}}{\partial \theta}$ se compone de las derivadas parciales de pesos y *biases* (Ecc. 27).

$$\Theta = (\theta, \theta') \tag{22}$$

$$\Theta(n+1) = \Theta(n) + \eta s(n) \tag{23}$$

$$s(n+1) = r(n+1) + \gamma s(n) \tag{24}$$

$$r(n+1) = -\nabla\phi_{SAE}(\Theta(n+1)) \tag{25}$$

$$\nabla\phi_{SAE} = \left(\frac{\partial\phi_{SAE}}{\partial\theta}, \frac{\partial\phi_{SAE}}{\partial\theta'}\right) \tag{26}$$

$$\frac{\partial\phi_{SAE}}{\partial\theta} = \left(\frac{\partial\phi_{SAE}}{\partial\mathbf{w}^{(1)}}, \frac{\partial\phi_{SAE}}{\partial\mathbf{b}^{(1)}}\right) \tag{27}$$

5 Experimento y resultados

5.1 Datos experimentales

La base de datos de vibraciones de rodamientos que será utilizada para entrenar y probar a la red neuronal profunda fue creada por el Bearing Data Center of Case Western Reserve University [26], una reconocida Universidad de Estados Unidos de América. Los datos fueron recolectados de un sistema mecánico con un motor bajo cuatro diferentes cargas con una frecuencia de muestreo de 12 y 48 kHz, sin embargo, esta investigación solo utilizará la de 12 kHz. Los instrumentos empleados en este laboratorio de vibraciones corresponden a: un motor de 2 hp a la izquierda del banco de pruebas (Fig. 2), un transductor/codificador de torque en el centro y un dinamómetro a la derecha.

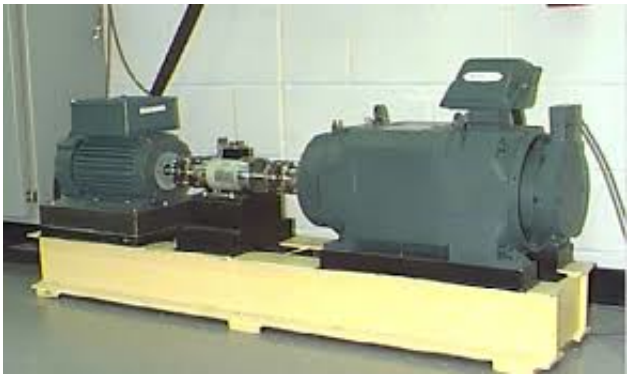

Figura 2: Ilustración del experimento realizado por la Case Western Reserve University [26]

Las mediciones fueron realizadas tanto en el *drive end* (DE) como en el *fan end* (FE) bajo cuatro condiciones de salud: normal (N), falla en el anillo externo (OF), falla en el anillo interno (IF) y falla en la bola (RF). Los tres tipos de fallas fueron inducidas a través de descargas eléctricas, generando piquetes de 0.007 in, 0.014 in y 0.021 in (pulgadas) en cada tipo de falla. Además, el rodamiento observado fue sometido a 4 velocidades con sus cargas correspondientes: 1797 - 0 hp, 1772 - 1 hp, 1750 - 2 hp y 1730 - 3 hp, por lo que los datos pueden ser organizados en conjuntos de datos separados por velocidad/carga como se detallará en la siguiente sección.

El motivo para utilizar esta base de datos es que sirve como punto de comparación con investigaciones que realizan diagnóstico de fallas a rodamientos. Permitiendo sacar conclusiones acerca de los tiempos de entrenamiento, desempeño del diagnóstico y ventajas y desventajas del modelo propuesto. Además, al tener tipos de fallas con severidades, es posible diagnosticar un problema de mayor complejidad y llevar al límite la capacidad de diagnóstico del modelo.

5.2 Construcción de los datasets

Con el objetivo de probar la capacidad de diagnósticos del modelo, se utilizarán 5 conjuntos de datos separados por velocidad/carga, tipo de falla y severidad. En su interior hay 10 carpetas con las señales de vibración, por ejemplo: para el tipo de falla anillo interno habrá 3 carpetas con las señales de vibración de las tres severidades de la falla. El *dataset* E, es especial debido

Dataset	Carga (hp)	Cantidad de muestras	Tipo de falla	Diámetro de falla (in)	Etiqueta
A/B/C/D/E	0/1/2/3/0-3	58/58/58/58/232	RF	0.007	1
		58/58/58/58/232	RF	0.014	2
		58/58/58/58/232	RF	0.021	3
		58/58/58/58/232	IF	0.007	4
		58/58/58/58/232	IF	0.014	5
		58/58/58/58/232	IF	0.021	6
		58/58/58/58/232	OF	0.007	7
		58/58/58/58/232	OF	0.014	8
		58/58/58/58/232	OF	0.021	9
		58/58/58/58/232	N	N/A	10

Table 1: Descripción de los data sets del experimento.

a que tiene las 4 velocidades juntas, por lo que en las 10 carpetas se encuentran las señales de vibración por tipo de falla y severidad para las 4 velocidades; resultando en el *dataset* más grande y complejo de diagnosticar.

Las señales utilizadas para construir los conjuntos de datos corresponden a las que fueron medidas en el *drive end* (DE) del sistema Fig. 2. La principal razón para utilizar estas señales es que el acelerómetro con el que se tomaron las muestras se encontraba cerca del rodamiento observado, por lo que las señales son menos afectadas por la atenuación y el ruido. En la Tabla 1 se aprecia cada uno de los 5 conjuntos de datos, con 58 muestras por señal consecuencia de un tamaño de segmentación igual a 2048 obtenido de un largo proceso de *tunning*.

Como se explicaba en párrafos anteriores, el *dataset* E es el más grande porque reúne las 5 cargas y es que mas nos interesa de las pruebas porque los rodamientos en un sistema real trabajan bajo velocidades y cargas variables. Lo que implica que la señal de vibración resultante es no lineal y no estacionaria, de manera que el diagnóstico de fallas se torna complejo como se explicaba en la introducción de este trabajo. Otro *dataset* relevante es el A, porque tiene carga 0 y hace que la señal que es modulada en amplitud refleje en forma tenue los fallos incipientes en los componentes del rodamiento.

5.3 Preproceso de los datos

Esta sección, describe cómo las señales de cada *dataset* son cargadas, cortadas, concatenadas y transformadas para poder entrenar al clasificador. El primer paso que es cargar las señales, es realizado por una función en MATLAB que ingresa a cada carpeta del *dataset* para leer las señales y ordenarlas en una matriz de dimensiones determinadas por el largo del segmento y la cantidad de muestras, por ejemplo 2048×58. El tamaño de las señales de vibración es variable en la base de datos original, por lo que es necesario tomar el mínimo de muestras por señal y que corresponde a 120.000 puntos.

La idea de utilizar este mínimo es de aplicarlo como tamaño fijo para todas las señales y así evitar el sesgo, por una señal que tenga más muestras, en el aprendizaje del clasificador. Una vez que todas las señales son cargadas y ordenadas como matrices, es momento de concatenarlas para formar una gran matriz cuyas filas son la cantidad de muestras y las columnas son el tamaño del segmento. Posterior a la unión de las señales, continúa el proceso de aplicar la transformada rápida de Fourier y obtener el espectro de energía como se detalla en la siguiente sección.

5.3.1 FFT y energy spectrum

Una vez obtenida la matriz con las señales de determinado *dataset*, se aplica la transformada rápida de Fourier (FFT) con una eficiente función de MATLAB. Calculados los componentes de frecuencia de las señales, se les aplica valor absoluto y elevan al cuadrado (Ecc. 28) para conseguir la energía del espectro de Fourier. Dado que la transformada de Fourier tiene la propiedad de simetría, es posible tomar la primera parte de la energía del espectro, resultando en 1024 puntos por muestra.

$$E = \frac{1}{2\pi} \int_{-\infty}^{\infty} |F(\omega)|^2 d\omega \tag{28}$$

5.3.2 Naturaleza de la señal de vibración

Como se explicó en la introducción, los rodamientos pueden comenzar a fallar por muchos motivos: un mal diseño o instalación, corrosión, poca lubricación y sobrecarga son algunos de ellos. Durante la operación del rodamiento se generan impulsos de banda ancha cuando la bola pasa por el defecto a una frecuencia determinada por la velocidad del eje, la geometría del rodamiento y la ubicación del defecto. La dificultad de diagnosticar la salud de los rodamientos reside en el hecho de que los patrones de un rodamiento defectuoso se extienden a través de una amplia banda de frecuencias, por lo que pueden ser fácilmente enmascarados por ruido y efectos de baja frecuencia.

La transformada rápida de Fourier (FFT) fue seleccionada como técnica de extracción de características por ser una herramienta de baja complejidad computacional y porque puede llevar la señal original al dominio de la frecuencia. Sin embargo tiene una serie de desventajas en este tipo de problemas en que la señal de vibración (Fig. 3) es tremendamente compleja. Podríamos decir que la desventaja más importante es que, esta técnica es menos eficiente para defectos en el anillo interno (IF), los cuales emiten señales de vibración muy tenues durante la etapa incipiente de falla ya que tienen un mayor número de segmentos de transferencia [8].

En la Fig. 5 se encuentra el diagrama de flujo de los pasos descritos en esta sección, que van desde la carga de las señales, hasta el entrenamiento de la red neuronal profunda y las gráficas de las métricas para el diagnóstico. En el primer bloque del diagrama se encuentra la sucesión de pasos para obtener las características que alimentaran a la capa de entrada del clasificador.

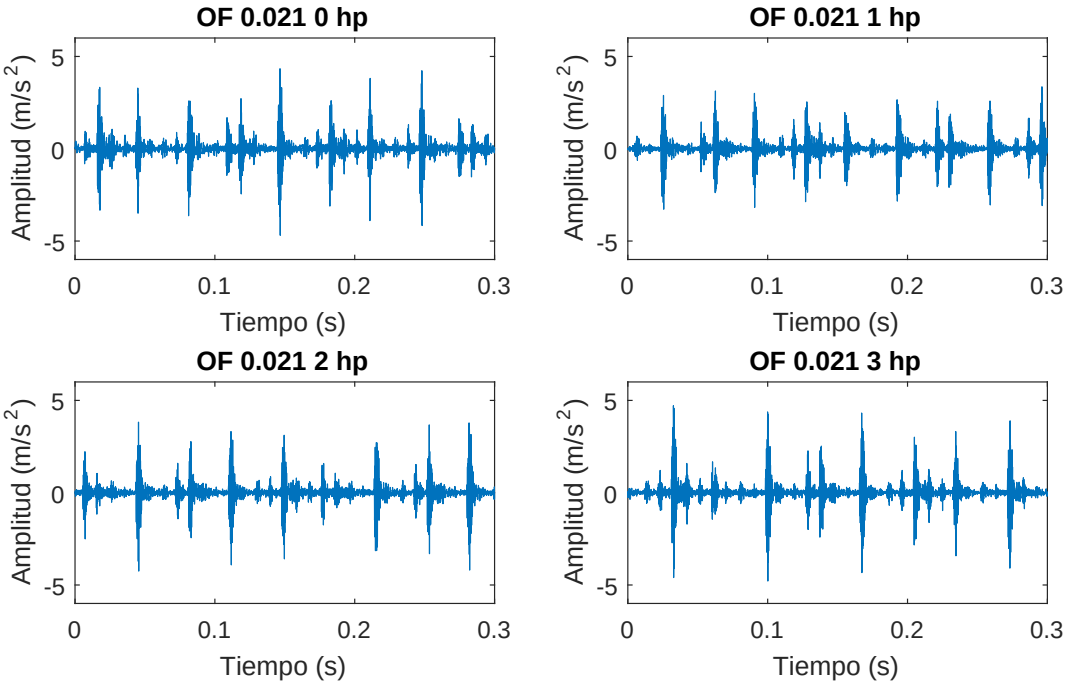

Figura 3: 4 tipos de señal de vibración de un rodamiento, con una carga de 3 hp y 0,54 mm de diámetro de falla.

5.4 Entrenamiento

En esta sección se dará detalle al proceso de entrenamiento de la red neuronal profunda (DNN) y los elementos que la componen. La red utilizada en este modelo, posee una capa de entrada, dos capas ocultas y una capa de salida con la función de transferencia softmax. Las dos capas ocultas son pre entrenadas con dos *sparse* autoencoder (SAE) respectivamente. Estas redes de tres capas permiten encontrar los parámetros de entrenamiento a la vez que reducen la dimensión de los datos de entrada codificandolos en un espacio de menor dimensión.

Una vez que el primer SAE termina de aprender su representación de los datos mediante el algoritmo del gradiente conjugado (Ecc. 26), inicializa con sus parámetros óptimos θ_1 a la primera capa oculta de la DNN como se muestra en la parte izquierda A de la Fig. 4. Luego el segundo SAE, constituido por la primera y la segunda capa oculta de la DNN, aprende un conjunto de parámetros que inicializa a la segunda capa oculta y los códigos de la segunda capa oculta alimentan a la regresión softmax para que sea entrenada.

Una vez obtenidos los parámetros de las capas ocultas, la DNN realiza un *fine tuning* que básicamente es aplicar el algoritmo *backpropagation* a la red entera para mejorar los pesos y *biases* en conjunto. Se utiliza el descenso del gradiente en vez del gradiente conjugado porque los parámetros se encuentran cerca del mínimo global de la función de costos, por lo que no son necesarias demasiadas iteraciones en el proceso.

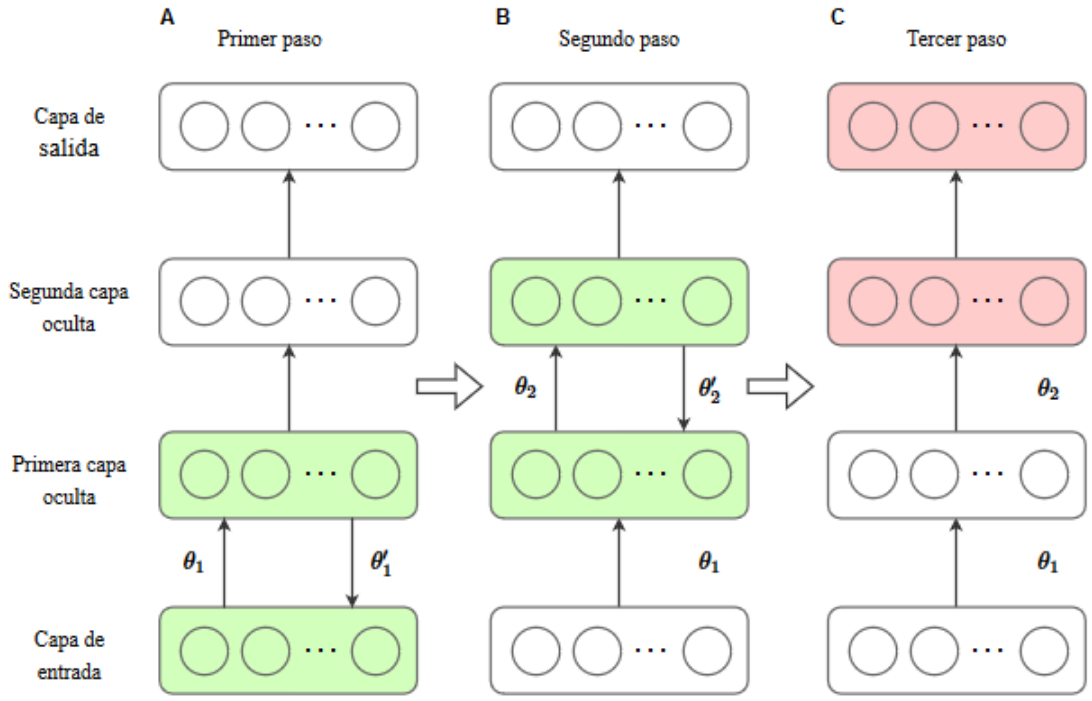

Figura 4: Diagrama que representa el entrenamiento de los *sparse* autoencoder y la asignación de sus parámetros a la DNN

5.4.1 Reordenamiento, muestras de entrenamiento y testing

Tal como lo indica el nombre de la subsección, este apartado explica el funcionamiento del re ordenamiento de las muestras de un *dataset* y la división de las muestras en datos de entrenamiento y de *testing*. Una vez que el proceso de extracción de características termina, el algoritmo codificado en MATLAB procede a reordenar las muestras y separarlas datos de *training* y *testing* con sus respectivas etiquetas. Los porcentajes que serán probados, y que afectan significativamente al modelo son los siguientes: 0.5, 0.6, 0.7 , 0.75 y 0.8.

5.4.2 Ambiente de desarrollo

La red neuronal profunda es construida con la *toolkit* de MATLAB versión R2016a, y considera dos *sparse* autoencoder apilados y una capa de neuronas con la función de transferencia softmax. La CPU utilizada para ejecutar estos algoritmos corresponde a un AMD(R) A10-5750M(TM) funcionando con el sistema operativo Windows 10(TM).

5.5 Evaluando el desempeño

Para evaluar el desempeño del modelo respecto a la calidad del diagnóstico se emplearan métricas que son habitualmente usadas en problemas de clasificación multiclase. Variadas investigaciones usan la validación cruzada para evaluar el aprendizaje del modelo, sin embargo, en este trabajo solo se efectuaran 30 ejecuciones por *dataset* con el fin evaluar el rendimiento del modelo. A estas ejecuciones se les calculará la exactitud por ejecución, la exactitud (Ecc.

32) promedio por clase, el F-score (Ecc. 31) o media armónica por clase y el F-score promedio por ejecución.

las Ecuaciones 29, 30, 31 y 32 son calculadas con: verdaderos positivos (TP), verdaderos negativos (TN), falsos positivos (FP) y falsos negativos (FN) que plasman la calidad de la predicción realizada por el clasificador. La precisión (Ecc. 29) representa el ratio o razón entre los correctamente identificados o verdaderos positivos y todos los positivos, mientras que la sensibilidad (Ecc. 30) representa el ratio o razón entre los correctamente identificados y los negativos. El F_1score es la media armónica de los resultados de clasificación, y se usa este tipo de media en vez de la media aritmética porque es ideal cuando se trabaja con ratios o proporciones.

$$Precisión = \frac{TP}{TP + FP} \tag{29}$$

$$Sensibilidad = \frac{TP}{TP + FN} \tag{30}$$

$$F_1score = \frac{2 \cdot Precisión \cdot Sensibilidad}{Precisión \cdot Sensibilidad} \tag{31}$$

$$Exactitud = \frac{VP + VN}{VP + FP + VN + FN} \tag{32}$$

5.6 Resultados

Realizando las 30 ejecuciones para cada conjunto de datos, la red neuronal profunda se encargó de aprender complejas relaciones no lineales entre las características que le fueron entregadas. Logrando muy buenos resultados en especial para los dos *datasets* más complicados (A y E) de acuerdo a las gráficas de exactitud y F_1score. En las pruebas fueron puestas apruebas distintas topologías, distintos tamaños de segmento, diferentes proporciones de entrenamiento y *testing* y distintos valores para los hiperparámetros. Estos últimos corresponden a los términos que penalizan la función de costo de los *sparse* autoencoder: λ (weight decay), β (spasity regularization) y ρ (sparsity proportion).

Fue realmente complicado encontrar parámetros que mejoraran sustancialmente la exactitud en los *datasets* de mayor interés. A pesar de esto, fue posible hallar un conjunto que diera buenos resultados, para el primer SAE: $\lambda = 0.022$, $\beta = 0.97$ y $\rho = 0.02$ y para el segundo SAE $\lambda = 0.0002$, $\beta = 0.97$ y $\rho = 0.25$. Respecto al tamaño de los segmentos, el mejor resultó en 2048 puntos por segmento tal como se comentaba en la descripción de los *datasets*.

La cantidad óptima de unidades en la primera capa oculta resulto en 102 y para la segunda capa oculta 25, por lo que la red neuronal profunda llevó las características originales desde un espacio de alta dimensión a uno de menor dimensión, omitiendo las variaciones irrelevantes para la detección de patrones. La proporción ideal de muestras para entrenamiento y *testing* fue de 70% y 30% respectivamente. Los resultados de las ejecuciones con estas configuraciones pueden ser observados en las gráficas que a continuación se discutirán.

Dataset	Carga (hp)	Exactitud media global	Desviación estándar	Mínimo	Máximo
A	0	99.8084%	±0.2756	99.4253%	100%
B	1	99.9234%	±0.2495	98.8506%	100%
C	2	100%	±0	100%	100%
D	3	99.9425%	±0.1754	99.4253%	100%
E	0-3	99.9473%	±0.1222	99.4253%	100%

Table 2: Resultados del testing para cada dataset.

Los resultados fueron muy buenos considerando la complejidad del problema Los, el dataset A obtuvo en promedio para las 30 ejecuciones 99.8084% de exactitud con una desviación estándar pequeña, por lo que el modelo entrega un buen diagnostico incluso en velocidad constante sin carga. Sorprendentemente en el dataset C la red neuronal profunda logra reconocer correctamente el estado de salud del rodamiento en cada una de las ejecuciones, superando a los resultados de diagnostico de [6] para ese dataset.

El resultado que mayor satisfacción entrega, es el del dataset E, con un 99.9473 de exactitud promedio casi igualando a los resultados de una investigación a nivel de doctorado como lo es la de [6]. agregar explicacion

La Fig. 6 refleja el comportamiento de la exactitud a través de las ejecuciones revelando las ejecuciones donde baja considerablemente la exactitud y empeora el promedio de la prueba. la Fig. 7 detalla la exactitud promedio para las 10 clases, revelando las clases con mayores dificultades para el clasificador, como la clase 2, 3, 7 y 8 que corresponden a la bola y al anillo externo.

5.7 Conclusión

Como conclusión, la red neuronal profunda compuesta por *sparse* autoencoders apilados puede lograr un desempeño asombroso respecto a la exactitud del diagnostico (a nivel de doctorado), siempre que encuentre una buena configuración de hiperparámetros, los porcentajes de entrenamiento y de *testing* y una buena topología de las capas ocultas. Es por esto que las redes neuronales profundas, muy utilizadas en reconocimiento de imágenes, están ganando territorio en otras áreas de investigación, como lo es el diagnostico de fallas en rodamientos mediante su señal de vibración. Sin embargo, este modelo de monitoreo de condiciones tiene la desventaja de un excesivo tiempo de *tuning* de los hiperparámetros, si bien la técnica de extracción de características es rápida computacionalmente; el proceso de encontrar parámetros para los *sparse* autoencoders es lento y engorroso. Por lo que puede resultar poco alentador calibrar el modelo por el tiempo que demanda encontrar las variables que potencian al clasificador. De esta misma manera, los tiempos de ejecución tampoco son los mejores, el algoritmo del gradiente conjugado necesita bastantes iteraciones (menos que el descenso del gradiente) para acercarse al optimo global de la función de costos. Entonces, las redes neuronales profundas son una excelente herramienta de diagnostico de fallas para este problema muy complejo, pero requiere un sacrificio no menor a la hora de buscar parámetros para los SAE.

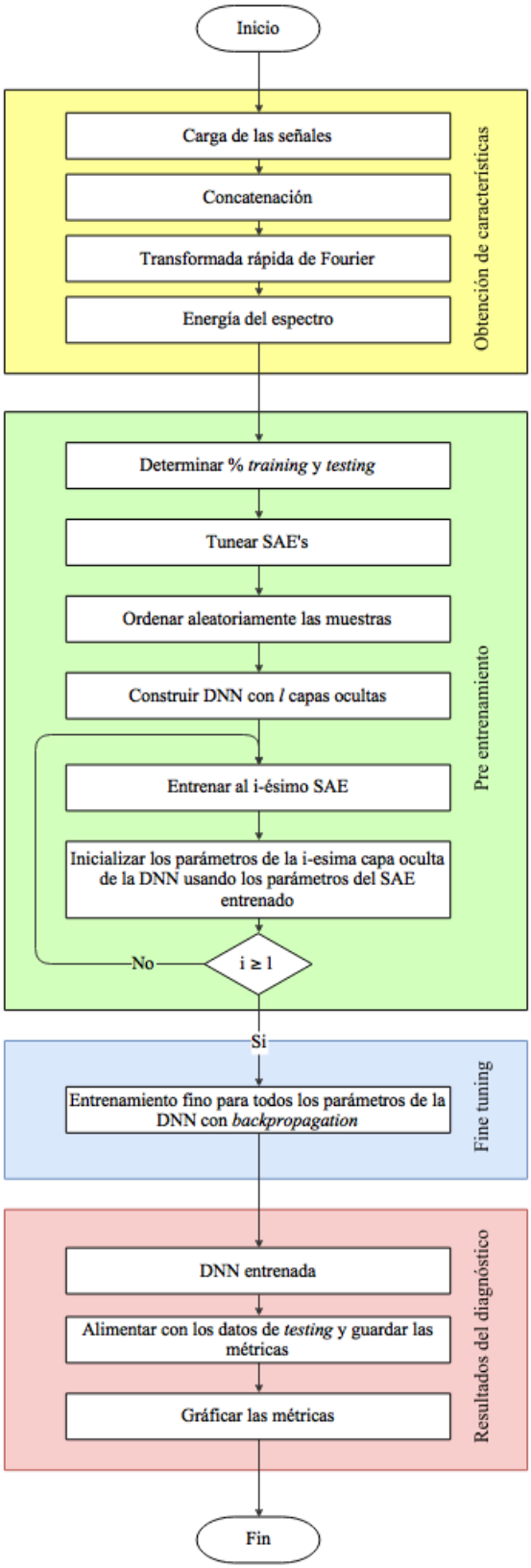

Figura 5: Diagrama de flujo de los procesos que se llevan a cabo para entrenar y probar el modelo propuesto.

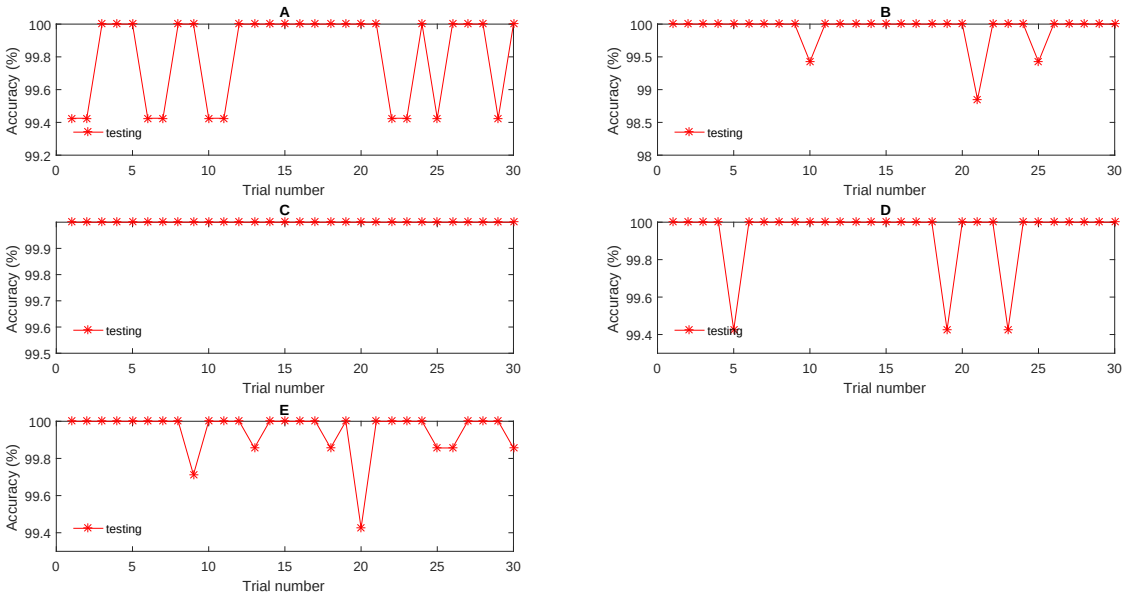

Figura 6: Diagrama con la exactitud del clasificador en 30 ejecuciones.

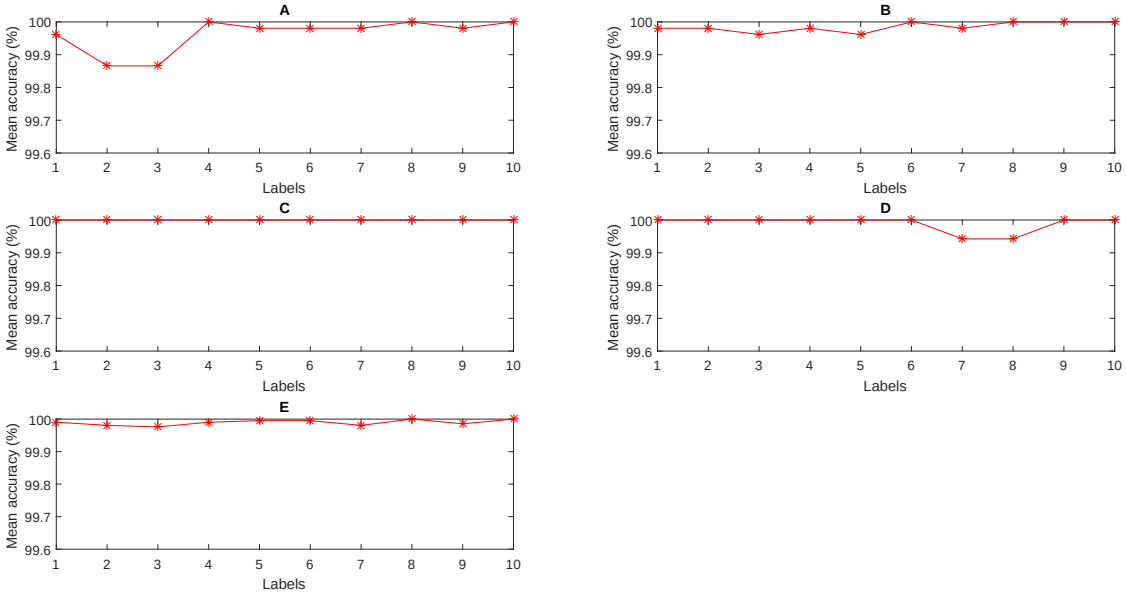

Figura 7: Diagrama con la exactitud promedio de las 30 ejecuciones para cada una de las 10 clases.

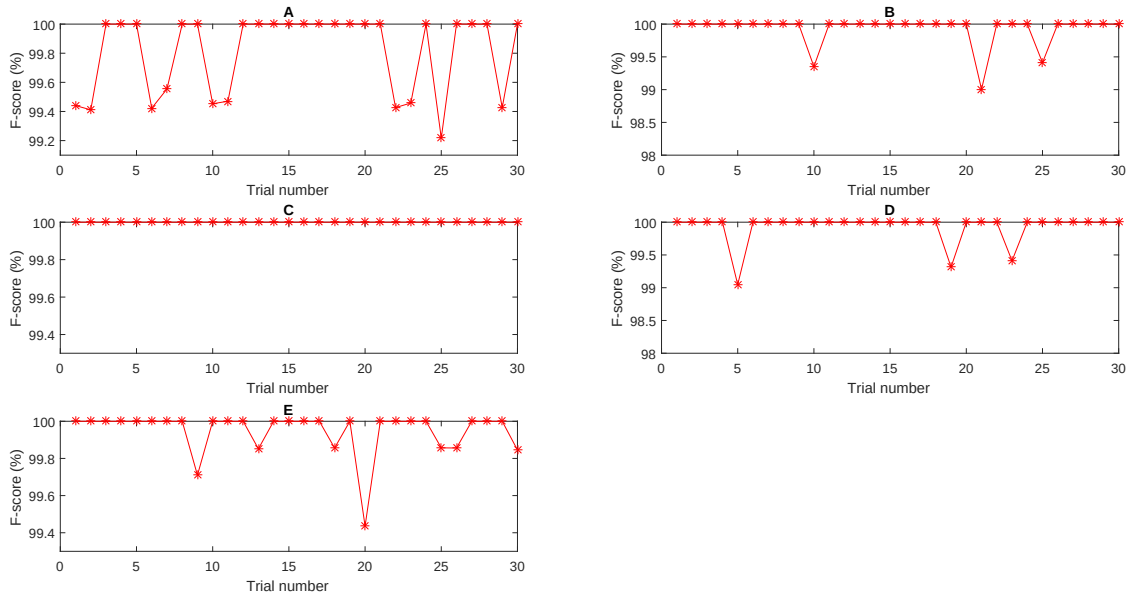

Figura 8: Diagrama con el F_1score de cada una de las 30 ejecuciones.

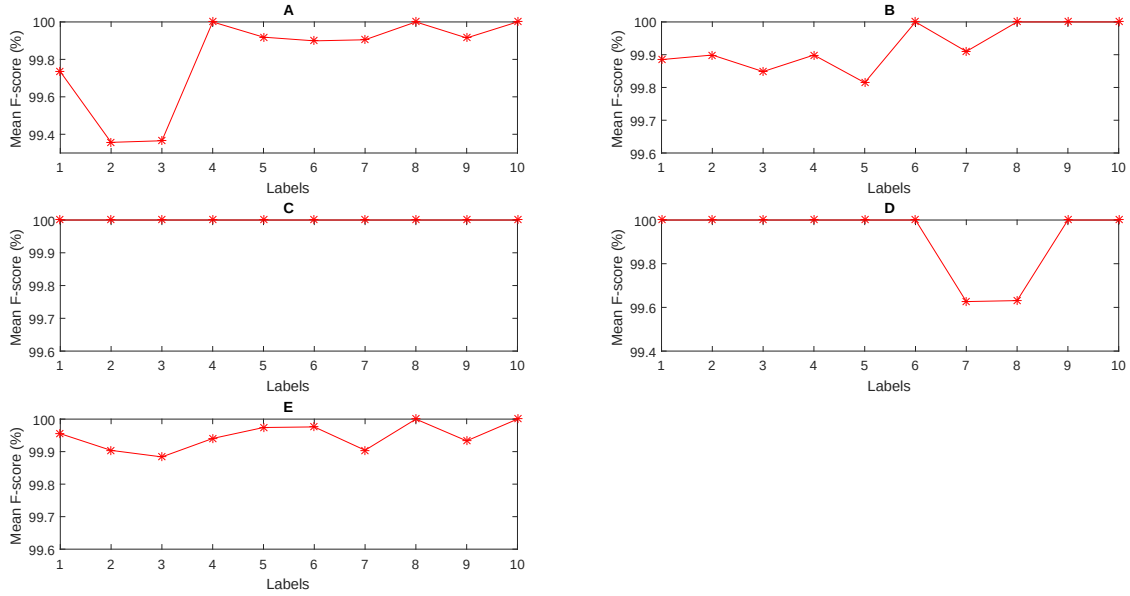

Figura 9: Diagrama con el F_1score promedio de las 30 ejecuciones para cada una de las 10 clases.

Referencias

[1] Manjeevan Seeraa, M.L. Dennis Wonga y Asoke K. Nandic, *Classification of ball bearing faults using a hybrid intelligent model*, Applied Soft Computing, 2017.

[2] Yimin Zhan y Chris K. Mechefske, *Robust detection of gearbox deterioration using compromised autoregressive modeling and Kolmogorov–Smirnov test statistic—Part I: Compromised autoregressive modeling with the aid of hypothesis tests and simulation analysis*, Mechanical Systems and Signal Processing, 2007.

[3] Yunxiao Fu, Limin Jia, Yong Qin y Jie Yang, *Product Function Correntropy and Its Application in Rolling bearing Fault Identification*, Measurement, 2016.

[4] Wei Li, Zhencai Zhu, Fan Jiang, Gongbo Zhou y Guoan Chen, *Fault diagnosis of rotating machinery with a novel statistical feature extraction and evaluation method*, Mechanical Systems and Signal Processing, 2014.

[5] Hong Guo, Lindsay B. Jack, y Asoke K. Nandi, *Feature Generation Using Genetic Programming With Application to Fault Classification*, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), Vol 35, No. 1, 2005.

[6] Yang Yu, YuDejie y Cheng Junsheng, *Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data*, Mechanical Systems and Signal Processing, 2015

[7] Feng Jia, YaguoLei, Jing Lin, Xin Zhou y Na Lu, *A roller bearing fault diagnosis method based on EMD energy entropy and ANN*, Journal of Sound and Vibration 294 269–277, 2006

[8] V.K. Rai y A.R. Mohanty, *Bearing fault diagnosis using FFT of intrinsic mode functions in Hilbert–Huang transform*, Mechanical Systems and Signal Processing 21 2607–2615, 2007

[9] Ran Zhang, Zhen Peng, Lifeng Wu, Beibei Yao y Yong Guan, *Fault Diagnosis from Raw Sensor Data Using Deep Neural Networks Considering Temporal Coherence*, Sensors, 2017.

[10] Issam Attoui, Nadir Fergani, Nadir Boutasseta, Brahim Oudjani y Adel Deliou, *A new time–frequency method for identification and classification of ball bearing faults*, Journal of Sound and Vibration, 2017.

[11] Anthony J. Wheeler y Ahmad R. Ganji, *Introduction to Engineering Experimentation*, Págs. 107-117, tercera edición, Prentice Hall, 2009.

[12] Robert M. Gray y Joseph W. Goodman, *Fourier Transforms: An Introduction for Engineers*, Págs. 53-57, primera edición, Springer, 1995.

[13] Shao Haidong, Jiang Hongkai, Zhao Huiwei y Wang Fuan, *A novel deep autoencoder feature learning method for rotating machinery fault diagnosis*, Mechanical Systems and Signal Processing volumen 95, 2017.

[14] Song-Zhi Su, Zhi-Hui Liu, Su-Ping Xu, Shao-Zi Li y Rongrong Ji, *Sparse auto-encoder based feature learning for humanbody detection in depth image*, Signal Processing, 2014.

[15] Yufeng Zhang, Zhenyu Guo, Weilian Wang, Side He, Ting Lee y Murray Loew, *A comparison of the wavelet and short-time fourier transforms for Doppler spectral analysis*, Medical Engineering & Physics, 2003.

[16] Changting Wang y Robert X. Gao, *Wavelet Transform with Spectral Post-Processing for Enhanced Feature Extraction*, IEEE Instrumentation and Measurement Technology Conference, 2002.

[17] Aurelien Prudhom, Jose Antonino-Daviu, Hubert Razik y Vicente Climente-Alarcon, *Time-frequency vibration analysis for the detection of motor damages caused by bearing currents*, Mechanical Systems and Signal Processing, 2015.

[18] Zhiqiang Chen, Shengcai Deng, Xudong Chen, Chuan Li, René-Vinicio Sanchez y Huafeng Qin, *Deep neural networks-based rolling bearing fault diagnosis*, Microelectronics Reliability, 2017.

[19] P. Konar y P. Chattopadhyay, *Bearing fault detection of induction motor using wavelet and Support Vector Machines (SVMs)*, Applied Soft Computing, 2011.

[20] Wentao Mao, Jianliang He, Yuan Li y Yunju Yan, *Bearing fault diagnosis with auto-encoder extreme learning machine: A comparative study*, Journal of Mechanical Engineering Cience, 2016.

[21] Kanpaj Gupta y M.K. Pradhan, *Fault detection analysis in rolling element bearing: A review*, 5th International Conference of Materials Processing and Characterization, 2016.

[22] Ian H. Witten, Eibe Frank, Mark A. Hall y Christopher J. Pal, *Data Mining: Practical Machine Learning Tools and Techniques-Morgan Kaufmann*, cuarta edición, Morgan Kaufmann, 2016.

[23] Simon Haykin, *Neural Networks and Learning Machines*, tercera edición, Pearson, 2009.

[24] Marc'Aurelio Ranzato, Y-Lan Boureau y Yann LeCun, *Sparse feature learning for deep belief networks*, Adv.NeuralInf.Process.Sys. 1185–1192, 2008.

[25] Erzhu Li, Peijun Du, Alim Samat, Member, Yaping Meng y Meiqin Che, *Mid-Level Feature Representation via Sparse Autoencoder for Remotely Sensed Scene Classification*, IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 10 1068–1081, 2017

[26] Case Western Reserve University Bearing Data Center, *Bearing vibration data set*, http://csegroups.case.edu/bearingdatacenter/pages/12k-drive-end-bearing-fault-data, (visitado el 30-08-2017).