UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

TRABAJO FIN DE MÁSTER

Procesado avanzado de señales multicanal procedentes de sensores industriales

Máster Universitario en Ingeniería de Telecomunicación

Autora: GÁMIZ PÉREZ, Sara

Tutor: RAMOS CASTRO, Daniel
Dpto. de Tecnología Electrónica de las Comunicaciones

FEBRERO 2018

Procesado avanzado de señales multicanal procedentes de sensores industriales

Autora: GÁMIZ PÉREZ, Sara

Tutor: RAMOS CASTRO, Daniel

Grupo Audias - Audio, Data Intelligence And Speech
Dpto. Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Febrero de 2018

Procesado avanzado de señales multicanal procedentes de sensores industriales

Sara Gámiz Pérez

Palabras clave: Redes neuronales, Filtro de Wiener, clasificación de eventos, combinación multicanal, señales industriales, señal multicanal, sensores.

Resumen

En este proyecto se describe un sistema de detección y clasificación de eventos, cuya finalidad es la de encontrar irregularidades presentes en piezas industriales, de manera que se aseguren unos estándares de calidad establecidos. Para ello se hará uso de señales multicanal que han sido adquiridas mediante un conjunto de sensores industriales. Se ha comprobado que las señales presentan correlación en cuanto a la aparición de eventos, de modo que se complementan, lo que ayudará a la distinción entre eventos. Además, se ha propuesto la utilización de técnicas de procesado avanzado de señal, en particular se usarán Redes Neuronales ya que han tenido un gran auge, en los últimos tiempos, sobre todo en el campo del reconocimiento de patrones.

Sin embargo, la escasez de señales en la Base de Datos y la presencia de ruido en las señales procedentes de sensores, han supuesto dificultades a las que se ha tenido que hacer frente. Las redes neuronales necesitan una gran cantidad de datos para su entrenamiento, por lo que se ha implementado un generador de señales sintéticas, de las mismas características que las reales, cuya finalidad es aumentar la Base de Datos y tener un entorno más robusto. Por otro lado, se ha generado un sistema de reducción de ruido, también basado en redes neuronales, orientado a la clasificación de eventos.

El uso de señales multicanal y la combinación entre el sistema de reducción de ruido y el sistema de detección y clasificación de eventos, dan lugar a un sistema final que proporciona una alta precisión. Esto se comprobará mediante la aplicación de una serie de medidas de rendimiento, las cuales se utilizarán para comparar los resultados obtenidos con resultados procedentes de sistemas clásicos anteriores. En particular, como línea base para la reducción de ruido se utilizarán técnicas de filtrado adaptativo y, para la detección de eventos, se usará un sistema anterior basado en una arquitectura *front-back-end*.

Cabe destacar que los algoritmos descritos en este trabajo se han desarrollado con distintas herramientas. El procesado de las señales, previa entrada a las rede neuronales, así como el procesado de la salida de la red, se ha realizado con $Matlab^{TM}$. Por otro lado, la implementación de las redes neuronales se ha desarrollado en Python, mediante el uso de la librería Keras.

Advanced processing of multichannel signals from industrial sensors

Sara Gámiz Pérez

Keywords: Neuronal Networks, Wiener Filter, events' classification, Multi-Channel combination, industrial signals, multi-channel signal, sensors.

Abstract

In this project we describe a detection and event classification's system, which goal is to find some anomalies given in industrial pieces in order to assure the established quality standards. For this purpose, we will use multi-channel signals that have been adquired by a set of industrial sensors. This decision has been taken because we have checked that these signals have a correlation in the occurrence of events, so they complement themselves. This property will help to distinguish between different events. Moreover, we propose the utilization of some advanced processing signals techniques, in particular the Neuronal Networks due to its high growth in recent times, mainly in fields such as patterns' recognition.

Nevertheless, the lack of signals in the database and the presence of noise in the captured sensors' signals have supposed some difficulties that we had to face. The neuronal networks need a great set of data for their training, so we have implemented a synthetic signal generator with the same original signals' characteristics. Its goal is to increase the database to obtain a more robust system. On the other hand, we have created a noise reduction system, which is also based on neuronal networks, especially oriented to events' classification.

The use of multichannel signals and the combination between the noise reduction system and the detection and classification system lead to a final system with a real high precision. This feature will be checked by using a set of performance meassures, comparing the obtained results with those which come from some classical noise reduction systems. Particulary, as a noise reduction baseline, we will use some techniques based on adaptative filtering and as an events detection baseline, a previous system based on a front-back-end architecture.

It is important to highlight that the described algorithms in this project have been developed with different tools. The stage of signal processing, previous to neuronal network entry, and the processing of the network output have been made using $Matlab^{TM}$. On the ohter hand, the neuronal network implementation has been developed using Python, concretely by Keras' library.

Agradecimientos

En primer lugar me gustaría mencionar a mi familia, y en especial a mis padres. Quiero agradecerles el esfuerzo que han realizado desde que decidí estudiar una ingeniería hace casi 6 años, ya que sin su ayuda, la realización de este Trabajo Fin de Máster no habría sido posible.

A mis hermanos, porque juntos formamos un gran equipo. Laura, te has convertido en mi mayor confidente y Francisco, eres capaz de proporcionar esa alegría tan necesaria en los momentos de más angustia.

Por supuesto, a ti Alex, ya que has sido mi gran apoyo y has sabido decirme las palabras adecuadas en el momento adecuado. Has estado siempre ahí en mis momentos de agobio y de estrés, que no son pocos. Además, contigo, los días infinitos en la biblioteca y los fines de semana de estudio han sido más amenos.

Agradecer también la oportunidad que me ha proporcionado el grupo Audias de realizar este Trabajo Fin de Máster con ellos, y en especial a mi tutor Dani. Gracias por confiar en mi, haber tenido paciencia y haberme explicado todo lo que he preguntado. También agradecer la inestimable colaboración de los demás miembros del laboratorio y sobre todo, gracias a ti Adri, que supiste integrarme y hacer que en poco tiempo fuera una más de vosotros.

A mis compañeros de clase, porque nos hemos dado cuenta de lo importante que es ayudarse y compartir. Y no me puedo olvidar de mi segunda familia, mis amigas. A algunas os conozco desde siempre y a otras desde que hace seis años nos cruzamos en un Colegio Mayor que cambiaría nuestras vidas. Desde entonces se han convertido en personas fundamentales y perfectas para olvidarse un rato del Máster y divertirse.

A todos vosotros, muchas gracias.

Índice general

Agradecimientos **VII**

Lista de figuras **X**

1. Introducción **1**
1.1. Motivación . . . 1
1.2. Objetivos . . . 3
1.3. Planificación del proyecto . . . 3
1.4. Organización de la memoria . . . 5

2. Estado del arte **7**
2.1. Estado del arte en reconocimiento de patrones . . . 7
2.2. Estado del arte en reducción de ruido . . . 9
2.3. Redes Neuronales . . . 10
2.3.1. Introducción . . . 10
2.3.2. Estructura de las Redes Neuronales . . . 10
2.3.3. Algoritmo *Forward Propagation* . . . 11
2.3.4. Algoritmo *Backpropagation* . . . 14
2.3.5. Entrenamiento . . . 21
2.4. Filtrado de Wiener . . . 23

3. Entorno experimental **27**
3.1. Análisis de la Base de Datos . . . 27
3.2. Entorno simulado . . . 30
3.3. Herramientas . . . 33
3.4. Medidas de rendimiento . . . 34
3.4.1. SNR (*Signal to noise ratio*) . . . 34
3.4.2. Curva DET . . . 35
3.4.3. Accuracy . . . 36
3.4.4. Matriz de confusión . . . 37

4. Diseño y desarrollo del sistema **39**
4.1. Arquitectura del sistema . . . 39
4.2. Subsistema preliminar de detección y clasificación de eventos . . . 40
4.2.1. Extracción de características . . . 41
4.2.2. Creación y entrenamiento de la red neuronal . . . 45
4.2.3. Decisión . . . 48

4.3. Subsistema de reducción de ruido para clasificación de eventos 50
4.3.1. Extracción de características 51
4.3.2. Creación y entrenamiento de la red neuronal 53
4.3.3. Reconstrucción de la señal 54
4.4. Combinación multicanal 55
4.4.1. Extracción de características 56
4.4.2. Creación y entrenamiento de la red neuronal 57
4.4.3. Decisión 58

5. Resultados **59**
5.1. Subsistema de reducción de ruido 59
5.2. Sistema de clasificación de eventos 64

6. Conclusiones y trabajo futuro **71**
6.1. Conclusiones 71
6.2. Trabajo futuro 73

Referencias bibliográficas **75**

Índice de figuras

2.1. Esquema de un sistema de reconocimiento de patrones. 7
2.2. Estructura de una red neuronal con D neuronas en las capas de entrada y salida, y dos capas ocultas con M neuronas cada una de ellas. 11
2.3. Representación gráfica de las funciones de activación. 14
2.4. Representación gráfica del cálculo de δ_j para una unidad j. 16
2.5. Esquema del Filtro de Wiener. 23

3.1. Señal multicanal, con la correspondiente señal de material X y señal densidad, que presenta los cuatro tipos de eventos posibles. Se han señalado los eventos presentes utilizando las etiquetas de Ground Truth. 29
3.2. Señal multicanal, con la correspondiente señal de material X y señal densidad, que únicamente presenta eventos de tipo gap. Se han señalado los eventos presentes utilizando las etiquetas de Ground Truth. 29
3.3. Comparativa de la SNR obtenida tras filtrar 17 señales reales con Wiener y utilizando los 113 modelos de ruido. 31
3.4. Comparativa entre señal multicanal real, con corrección de deriva, y señal sintética con presencia de todos los tipos de eventos. 32
3.5. Comparativa entre señal multicanal real, con corrección de deriva, y señal sintética con presencia de eventos tipo gap. 32
3.6. Representación de puntuaciones obtenidas para la clase target y para la clase nontarget, así como del umbral óptimo de decisión. 35
3.7. Representación de una curva DET y del valor EER correspondiente. . . . 36
3.8. Ejemplo de una matriz de confusión con cuatro clases distintas: ruido, desviaciones, gaps y transiciones. 37

4.1. Diagrama de bloques del sistema de detección y clasificación de eventos. . 40
4.2. Diagrama de bloques del sistema de detección y clasificación de eventos inicial. 41
4.3. Comparativa entre la señal de material X y su derivada. 42
4.4. Comparativa entre los *ratios inter-intra* obtenidos para cada evento y para cada una de las características extraídas de las señales. 43
4.5. Comparativa entre una señal de material X real y las características que la representan. 44
4.6. Comparativa entre una señal de material X sintética y las características que la representan. 45

4.7. Evolución del porcentaje de acierto obtenido para cada evento en función del número de neuronas utilizadas en las capas ocultas, así como evolución del tiempo de entrenamiento. 46
4.8. Evolución del porcentaje de acierto obtenido en función del número de señales utilizadas en el entrenamiento. 48
4.9. Filtrado de modas. 49
4.10. Matriz de confusión resultante de evaluar el sistema preliminar de clasificación de eventos con 68 señales de material X reales. 50
4.11. Diagrama de bloques del sistema de reducción de ruido. 51
4.12. Representación del error cuadrático medio (MSE) entre 10 señales sintéticas y las correspondientes 151 señales reconstruidas, obtenidas mediante la utilización de un número diferente de características seleccionadas, poniendo las restantes a 0. 52
4.13. Evolución de los resultados obtenidos (SNR, EER y tiempo de procesamiento) tras evaluar distintos modelos de red neuronal para reducción de ruido que varían en cuanto al número de capas ocultas y el número de neuronas en cada capa. 53
4.14. Diagrama de bloques del sistema de detección y clasificación de eventos. . 55
4.15. Diagrama de bloques del sistema de reducción de ruido. 55
4.16. Representación de una señal de material X y una señal densidad, utilizadas en el sistema de detección de eventos, así como de las características que las representan. 56
4.17. Evolución del porcentaje de acierto obtenido para cada evento tras evaluar distintos modelos de red neuronal que varían en cuanto al número de capas ocultas y el número de neuronas en cada capa. 57

5.1. Comparativa entre una señal densidad real, así como la posición de sus eventos, y las correspondientes señales tras el filtrado con la red neuronal y con Wiener. 60
5.2. Comparativa de SNR para distintas señales reales y las correspondientes señales filtradas con Wiener y con la red neuronal. 61
5.3. Comparativa de curvas DET para el conjunto de señales reales y señales filtradas con Wiener y con redes neuronales. 62
5.4. Comparativa de las características de las señales ruidosa, limpias con Wiener y limpias con redes neuronales. 63
5.5. Comparativa entre la clasificación obtenida con el sistema implementado, con el Sistema Línea Base y el Ground Truth. Se muestra la señal de material X ya que en ella se ven reflejados todos los eventos. 64
5.6. Comparativa entre la clasificación obtenida con el sistema implementado, con el Sistema Línea Base y el Ground Truth. Se muestra la señal de material X ya que en ella se ven reflejados todos los eventos, aunque en esta señal sólo aparecen eventos de tipo gap. 65
5.7. Representación de las Curvas DET para los eventos Ruido y Transiciones, cuando se aplica el sistema de detección de eventos basado en redes neuronales y el Sistema Línea Base. 66

5.8. Representación de las Curvas DET para los eventos Desviaciones y Gaps, cuando se aplica el sistema de detección de eventos basado en redes neuronales y el Sistema Línea Base. 67
5.9. Matriz de confusión obtenida tras evaluar el sistema de clasificación de eventos desarrollado con 68 señales reales. 68
5.10. Matriz de confusión obtenida tras evaluar el Sistema Línea Base [19] [20] con 68 señales reales. 68
5.11. Comparativa entre el *accuracy* obtenido con el sistema de clasificación que usa redes neuronales, el obtenido con el sistema basado en una arquitectura *front-back-end* y la línea base. 69

Capítulo 1

Introducción

En este primer capítulo se presenta el contenido que se va a desarrollar a lo largo de esta memoria. En primer lugar, se detalla la motivación que lleva a realizar el estudio que se describe, así como se especifica en qué marco se realiza este Trabajo Fin de Máster. A continuación, se exponen los objetivos que fueron fijados antes de comenzar con el proyecto y finalmente, se describe brevemente la organización de la memoria.

1.1. Motivación

El control de calidad, en los procesos de fabricación de piezas industriales, es de vital importancia para asegurar que se cumple con todas las garantías de fabricación y con los estándares de calidad establecidos. Este proceso, suele incluir tareas tales como la detección de imperfecciones en dichas piezas o la medida de la concentración de determinados materiales que las componen.

Estas tareas se pueden realizar mediante el análisis de señales multicanal, que reflejan el estado y la composición de dichas piezas. La adquisición de las señales se realiza a través del paso de las piezas industriales por un conjunto de sensores de distinta naturaleza, los cuales realizan mediciones. Por lo tanto, si una pieza sufriera algún defecto o imperfección, debería verse reflejado en la señal obtenida, mediante una serie de eventos aparentes.

En estudios previos, se ha comprobado que la utilización de una única señal no es suficiente para tener una información completa. Por lo tanto, en este proyecto se utilizarán dos tipos de señales, de manera que se complementen y mediante las cuales se consiga alcanzar los objetivos marcados. Estos objetivos no sólo se centran en la detección de dichos eventos, sino que se pretende clasificarlos de modo que se conozca el tipo de imperfección existente, ya que no todas afectan de la misma manera a las piezas.

Sin embargo, estas señales pueden llegar a presentar ruido no deseado o incluso no tener un comportamiento visiblemente claro, lo que dificulta los objetivos planteados. Es por esto, que este Trabajo Fin de Máster se enmarca en el contexto del control de calidad en la fabricación de este tipo de piezas y pretende aplicar técnicas de procesado avanzado

de señal y reconocimiento de patrones, como son las redes neuronales. En particular, se utilizarán para detectar y clasificar eventos que ayuden a analizar el estado de dichas piezas, así como para limpiar ruido y mejorar las señales. La elección de estas técnicas se ha tomado en base a mejorar los resultados que proporcionan otra técnicas clásicas más conocidas y menos novedosas en este contexto como son el filtrado adaptativo para reducción de ruido, o arquitecturas *front-back-end* para clasificación de eventos.

De esta manera, se pretende transferir el conocimiento y las técnicas que usualmente se utilizan en el campo de la investigación universitaria, al mundo industrial, de forma que se puedan solventar los inconvenientes planteados. En este caso, el grupo de investigación Audias colabora con la empresa Tecnatom S.A., bajo condiciones de confidencialidad industrial desde el punto de vista del dispositivo generador de las señales, y que será la que nos proporcione la Base de Datos con la que se va a trabajar. El tratamiento y procesado de las señales se puede publicar, a condición de que no se revele la naturaleza del dispositivo que las genera.

Sin embargo, este trabajo puede llegar a ser un tanto desafiante, pues nunca antes se han aplicado técnicas de procesado avanzado se señal a este tipo de señales industriales y por lo tanto, además de su clara componente profesional es también un trabajo de investigación e innovación, en el cual se debe profundizar en el estudio y comportamiento de estas técnicas, de cara a encontrar la forma óptima de aplicarlas.

Los principales problemas que esperamos encontrarnos son dos. En primer lugar, la Base de Datos de la que se dispone no es muy extensa, por lo que habrá que adecuar los algoritmos avanzados propuestos a este hecho (redes neuronales), ya que suelen requerir de cantidades considerables de datos. En segundo lugar, cabe destacar que para la reducción de ruido, se necesita de señales ruidosas y señales limpias para entrenar los algoritmos. La Base de Datos no dispone de este segundo tipo de señales, por lo que se realizará un generador de señales sintéticas, adaptado al tipo de señales de las que se dispone, de modo que se obtendrán señales sintéticas de la misma naturaleza que las reales. Este generador también se utilizará para aumentar la cantidad de datos de los que se dispone para el entrenamiento de los algoritmo. Asimismo, se realizarán pruebas con señales reales, con variabilidad similar a la de la aplicación real, para comprobar la capacidad de generalización de los algoritmos propuestos.

Finalmente, destacar que este Trabajo Fin de Máster supondrá la extensión y aplicación, en otros campos, de los conocimientos adquiridos durante el Máster de Ingeniería de Telecomunicación. En particular se profundizará sobre técnicas avanzadas de procesado de señal las cuales se han mencionado en la asignatura Procesado Avanzado de Señales Multimedia. Además se emplearán otros conocimientos como son los adquiridos en otras asignaturas como Teoría de la Información para Comunicaciones o Proyectos en Ingeniería de Telecomunicación.

1.2. Objetivos

El objetivo principal de este Trabajo Fin de Máster es el de mejorar el rendimiento del procesamiento de señales multicanal, procedentes de sensores industriales, mediante la aplicación de estrategias avanzadas de procesado de señal. En particular, se pretende crear un sistema capaz de detectar y clasificar los distintos eventos que aparecen en las señales, registradas por los sensores tras el paso de una pieza industrial, y que ayudarán a comprobar la existencia de imperfecciones y el cumplimiento de los estándares de calidad establecidos.

Como ya se mencionó en la motivación, este sistema es multicanal, ya que hace uso de dos tipos de señales distintas, las cuales no son ideales, pues pueden presentar ruido no deseado o no tener un comportamiento visiblemente claro. Así pues, para el correcto cumplimiento del objetivo principal, ha sido necesario diseñar e implementar dos subsistemas, cada uno de los cuales trabajará con un tipo de señal distinta:

- Subsistema de detección y clasificación de eventos, mediante la utilización de técnicas avanzadas de reconocimiento de patrones, como son las basadas en redes neuronales.

- Subsistema de reducción del nivel de ruido de las señales disponibles, a partir de la aplicación de redes neuronales como algoritmo avanzado de procesado de señales.

Finalmente, estos dos subsistemas se unirán para tener un sistema multicanal final.

1.3. Planificación del proyecto

La realización de este Trabajo Fin de Máster se divide en cuatro fases. La primera de ellas está dedicada al estudio de los conceptos que serán necesarios para el correcto desarrollo del trabajo. La segunda de ellas se centra en la implementación de dos sistemas de procesado avanzado de señal. En la tercera fase se evaluarán los resultados que se obtienen al aplicar los algoritmos sobre señales reales y sintéticas. Finalmente, la cuarta y última fase se centrará en la redacción de la memoria. A continuación se detallarán las tareas contenidas en cada fase del proyecto:

Plan de trabajo (45 horas, del 25/09/2017 al 10/10/2017)

- Análisis del objetivo principal y planteamiento de los objetivos específicos.
- Organización del plan de trabajo en función de los objetivos impuestos.
- Estudio de la Base de Datos con la que se va a trabajar. Se analizará el tipo de señales y sus características para comprender el problema a tratar.

- Instalación y preparación de las herramientas con las que se trabajará. En este caso se utilizará Matlab y Keras, una librería de Python que proporciona la creación de modelos Deep Learning sobre otras librerías como son TensorFlow, Theano o CNTK.
- Estudio y documentación sobre técnicas avanzadas de procesado de señal tanto para detección de eventos como para reducción de ruido. Esta tarea se enfocará en aquellas técnicas más novedosas como son las redes neuronales.

Implementación de los algoritmos (130 horas, del 11/10/2017 al 08/12/2017)

- Análisis, diseño e implementación de diferentes aproximaciones para el pre-procesado y la extracción de características de las señales multicanal disponibles. Tanto en el caso de clasificación de eventos, como en el caso de reducción de ruido, este pre-procesado tendrá la finalidad de representar, de forma óptima, las señales con un conjunto de características de cara a poder entrenar las redes neuronales con ellas.
- Generación de un sistema de creación de señales sintéticas, con el objetivo de probar los algoritmos implementados en entornos realistas de simulación.
- Creación de un sistema de clasificación de eventos utilizando redes neuronales y reconocimiento de patrones.
- Creación de un sistema de reducción de ruido utilizando técnicas de procesado avanzado de señal como son las redes neuronales.

Análisis de resultados (50 horas, del 11/12/2017 al 11/01/2017)

- Comprobación del correcto funcionamiento del programa con las señales sintéticas implementadas y los señales reales proporcionadas.
- Realización de pruebas de identificación y clasificación de eventos, mediante la utilización del sistema creado anteriormente y comparación con aproximaciones existentes basadas en detectores *front-back-end*.
- Realización de pruebas de evaluación y de rendimiento en limpieza de ruido, utilizando el sistema de reducción de ruido creado, basado en técnicas avanzadas de procesado de señal, y comparación con técnicas clásicas basadas en filtrado adaptativo.
- Corrección de errores que produzcan resultados inesperados o anomalías en el funcionamiento de los algoritmos.

Redacción de la memoria (75 horas, del 12/01/2017 al 12/02/2017)

- Redacción de la memoria en la que se incluyen las bases teóricas necesarias para la correcta comprensión del trabajo realizado, los algoritmos implementados, el análisis de los resultados, así como las conclusiones extraídas tras la realización del proyecto.

1.4. Organización de la memoria

Esta memoria refleja el trabajo, de investigación e implementación, que se ha realizado. A continuación se describe, de forma resumida, los capítulos que la componen:

Capítulo 1: Introducción. Se presenta la estructura de la memoria, la motivación y el contexto en el que se enmarca el Trabajo Fin de Máster, así como se fijan unos objetivos que se deben llevar a cabo.

Capítulo 2: Estado del Arte. Se presenta un breve Estado del Arte sobre la evolución de las técnicas de detección y clasificación de eventos, así como de las técnicas de reducción de ruido. Además, se detalla el fundamento teórico de las Redes Neuronales.

Capítulo 3: Entorno experimental. Presenta los conceptos fundamentales para la comprensión de las señales con las que se trabaja, así como se detalla el sistema generador de señales sintéticas implementado. Finalmente se comentarán las herramientas utilizadas para la realización de este trabajo y se mostrarán las medidas de rendimiento que se utilizarán para evaluar los resultados obtenidos.

Capítulo 4: Diseño y desarrollo del sistema. Se describirán los pasos que se han llevado a cabo para la realización de cada subsistema implementado, así como del sistema final. Además se describirá el proceso de investigación realizado para la elección de distintos parámetros del sistema.

Capítulo 5: Resultados. En este capítulo se aplicarán los sistemas implementados sobre señales reales de la Base de Datos. Además se aplicarán sobre estas señales otras técnicas clásicas de reducción de ruido y detección de eventos. Finalmente se compararán los resultados obtenidos.

Capítulo 6: Conclusiones y vías futuras. Se realizará una valoración global del trabajo realizado, en el cual se comprobará si se han cumplido los objetivos fijados. Además se detallarán los inconvenientes encontrados durante el periodo de estudio e implementación de los algoritmos. Finalmente, se mencionarán algunas vías de trabajo futuro.

Capítulo 2

Estado del arte

En este capítulo se detallará el Estado del Arte que sirve de base para la correcta comprensión de este Trabajo Fin de Máster. En primer lugar se mencionarán algunas técnicas que se han usado a lo largo del tiempo, para el reconocimiento de patrones y la reducción de ruido. A continuación se dará un fundamento teórico de las Redes Neuronales, ya que éstas son la base de los sistemas implementados en este trabajo, así como del Filtrado de Wiener.

2.1. Estado del arte en reconocimiento de patrones

El reconocimiento de patrones es una rama del aprendizaje automático que se centra en encontrar regularidades en los datos. Es decir, no es más que la asignación de una etiqueta a un valor de entrada dado.

El esquema de un sistema de reconocimiento se representa en la Figura 2.1. En ella, el bloque *extractor de características* es, como su propio nombre indica, el encargado de encontrar las características que se pueden utilizar para discriminar entre clases. Un buen conjunto de características es aquel que representa a eventos de diferentes clases, con valores de características distintos.

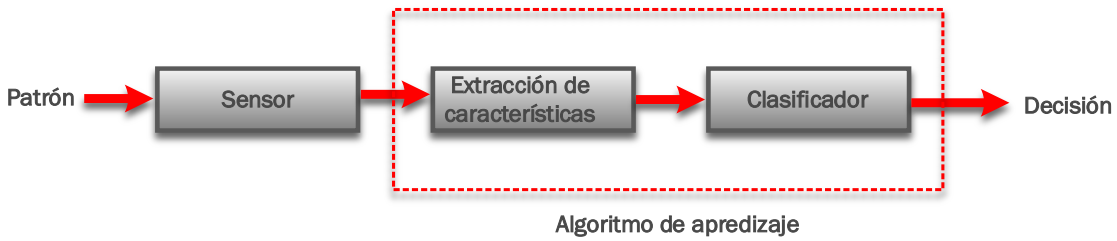

Figura 2.1: Esquema de un sistema de reconocimiento de patrones.

Por otro lado, el *clasificador* se encarga de dividir el espacio de características en distintas regiones, cada una correspondiente a una clase. Para crear este bloque, existen diferentes algoritmos que dependen del tipo de salida (decisión) y de si el aprendizaje es supervisado (disponibilidad de datos de entrenamiento etiquetados) o no supervisado (no hay datos etiquetados disponibles). Además, estos algoritmos pueden ser generativos o discriminativos, en función de si se centran en modelar características o por el contrario, se centran en optimizar una función de coste sobre las etiquetas de clasificación. En este trabajo, disponemos de datos de entrenamiento etiquetados, por lo que a continuación, se mencionarán las técnicas más comunes en aprendizaje supervisado.

- **Linear Discriminant Analysis (LDA)**. Es un método utilizado en estadística y en reconocimiento de patrones para encontrar una combinación lineal de características que separe dos o más clases de objetos. Sobre todo se suele usar para la reducción de la dimensionalidad.

- **Clasificador Bayesiano**. Utiliza el teorema de Bayes. El aprendizaje se puede ver como el proceso de encontrar la hipótesis más probable, dado un conjunto de entrenamiento y un conocimiento a priori sobre la probabilidad de cada hipótesis.

- **Estimación k-NN (k-Nearest Neighbors)**. Es un método no paramétrico que sirve para estimar la función de densidad que predice el valor de un dato para una clase. La idea se basa en que el nuevo dato se clasificará en la clase más frecuente de sus k vecinos más próximos.

- **Máquinas de vectores soporte**. Es un algoritmo para encontrar clasificadores lineales en espacios transformados. Buscan encontrar fronteras de decisión que separen cada una de las clases del problema.

- **Redes neuronales**. Es la técnica que se va a utilizar en este trabajo para la detección y clasificación de eventos. Por ello se ha realizado un estudio teórico que se presenta en la sección 2.3.

- **Clasificador Gaussiano de Máxima Verosimilitud**. Asume que los datos siguen una función de distribución normal para asignar la probabilidad de que un evento cualquiera pertenezca a cada una de las clases. Se dispone de un sistema de clasificación basado en este método [19] [20]. La finalidad de este proyecto es mejorar los resultados que se obtienen con dicho sistema, mediante la utilización de redes neuronales. En ese sentido, el clasificador gaussiano ML se utiliza como línea base, ya que es la mejor aproximación que se había logrado antes de comenzar este TFM. Por lo tanto, finalmente se compararán los resultados con ambos sistemas, de modo que se compruebe si se ha conseguido el objetivo.

2.2. Estado del arte en reducción de ruido

Las señales industriales de la Base de Datos presentan una gran cantidad de ruido procedente de los sensores que las registran. Este ruido es independiente de dichas señales y puede llegar a ocasionar una degradación de la señal o incluso la inteligibilidad de la misma. En particular, este ruido hace que algunos eventos de interés no queden visibles, por lo que se pierde información de interés.

Uno de los objetivos de este Trabajo Fin de Máster es la creación de un sistema de reducción de ruido, por lo que previamente a la implementación de los algoritmos, se van a analizar las técnicas más conocidas. Sin embargo, cabe destacar, que no hay mucha investigación en el ámbito industrial, por lo que tendremos que indagar en otro campo, que tenga cierta similitud con nuestras señales y en el que sí haya habido publicaciones, como es el de la limpieza de ruido en voz.

Las primeras investigaciones en el filtrado de señales de voz mono canal se realizaron hace más de 40 años. En 1958, Manfred R. Schroeder proponía una implementación analógica de substracción espectral. 15 años más tarde, Boll, reinventó este método pero en el dominio digital. Casi al mismo tiempo, Jae S. Lim y Alan V. Oppenheim, tras un análisis de las técnicas existentes hasta el momento, demostraron que la reducción de ruido no era sólo útil en la mejora de la calidad de las señales de voz, sino que mejoraba la calidad y la inteligibilidad de la codificación lineal predictiva (LPC). A partir de ahí numerosas técnicas han ido surgiendo, hasta que a día de hoy podemos dividirlas en tres conjuntos diferenciados por las características y requerimientos que se le imponen: filtrado adaptativo, substracción espectral y métodos basados en modelos.

Las técnicas basadas en filtrado utilizan un filtro lineal de forma que al pasar la señal ruidosa a través él, la componente correspondiente al ruido queda atenuada. Los algoritmos más representativos incluyen el filtrado de Wiener en el dominio del tiempo y en el de la frecuencia y el filtrado de Wiener paramétrico. Por otro lado, los métodos de substracción espectral, tratan la reducción de ruido como un problema de estimación espectral. Entre los algoritmos más conocidos se encuentra el estimador MMSE (*Minimun-Mean-Square-Error*), el estimador ML (*Maximum-Likelihood*) y el estimador MAP (*Maximun a Posteriori*). Finalmente, los métodos basados en modelos, tratan la reducción de ruido como un problema de estimación paramétrica. En este grupo destacamos técnicas como LP-Kalman (*Linear Predition*) y técnicas estadísticas como son los Modelos Ocultos de Markov [1].

El desarrollo que ha experimentado la Inteligencia Artificial ha permitido llevarlo al campo de reducción de ruido, hasta el punto de que se han establecido métodos basados en Redes Neuronales, que mapean la señal ruidosa a una señal libre de ruido. En cuando a la arquitectura de red óptima, se tienen distintas soluciones. En [2] se creó una red neuronal de mejora de habla con arquitectura ADALINE (*Adaptative Linear Neuron*), en [3] se decidió que para reconocimiento automático de habla la red neuronal más efectiva debia ser una red profunda y recurrente, finalmente en [4] se utiliza el Perceptrón Multicapa para reducción de ruido.

En este trabajo se va a generar un sistema de reducción de ruido con Redes Neuronales, basado en el modelo descrito en [5], al cual se le hará modificaciones, para adaptarlo al problema en cuestión. Además, se van a comparar los resultados obtenidos con Redes Neuronales, con resultados obtenidos con un filtrado adaptativo. En particular se va a utilizar filtrado de Wiener, diseñado por Pascual Scalart et al. [6], el cual se explica en la sección 2.4.

2.3. Redes Neuronales

2.3.1. Introducción

Como se dice en [7], "Una red neuronal es un sistema de computación, hecho por un gran número de elementos simples, elementos de proceso muy interconectados, los cuales procesan información por medio de su estado dinámico como respuesta a entradas externas"(Hech-Nielsen).

Se pueden ver las redes neuronales como una aproximación a la inteligencia artificial, la cual es la ciencia que permite realizar mediante máquinas operaciones que se consideran propias del ser humano, como el aprendizaje a partir de datos. Es decir, son una forma de emular ciertas características propias de los seres humanos como son la capacidad de memorizar y asociar hechos. Se caracterizan por su habilidad de obtener resultados de datos complicados e imprecisos y se utilizan para extraer patrones o detectar secuencias difíciles de apreciar por otras técnicas computacionales.

El inicio de las redes neuronales aritificiales cae en los años 40', cuando Mc Coullloc y Pitts crearon el primer modelo de neurona artificial. Sin embargo, no fue hasta 1958 cuando Rosenblatt introdujo el primer modelo neuronal, el perceptrón, el cual se caracterizaba por tener dos capas, entrada y salida. En 1960 surgió el primer modelo multicapa y en 1974, Paul Werbos introduce la idea básica del algoritmo de aprendizaje *backpropagation*. Finalmente en 1986, Rummelhart, Hinton y Williams publicaron el algoritmo general *backpropagation* para un perceptrón multicapa [8]. A partir de entonces creció la investigación y el desarrollo de las redes neuronales, hasta que en la actualidad son cada vez más los trabajos que se realizan, las publicaciones e incluso las aplicaciones nuevas que surgen.

2.3.2. Estructura de las Redes Neuronales

Las redes neuronales están constituidas por una serie de neuronas interconectadas en capas. El número de capas varía en función de la topologia, por ejemplo el perceptrón está formado únicamente por una capa de entrada y una de salida, sin embargo, las redes neuronales multicapa están formadas por una *capa de entrada*, una o varias *capas ocultas* y la *capa de salida*. Un ejemplo de la arquitectura de una red neuronal multicapa se puede ver en la Figura 2.2.

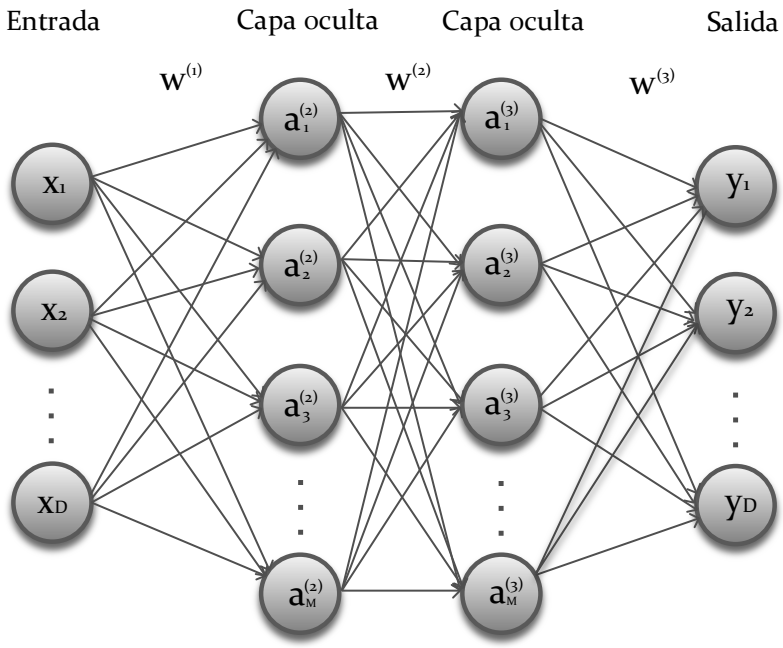

Figura 2.2: Estructura de una red neuronal con D neuronas en las capas de entrada y salida, y dos capas ocultas con M neuronas cada una de ellas.

En cuanto a los elementos básicos de las redes neuronales se tiene las entradas, los pesos y las salidas:

- Las entradas son las que llevan la información del exterior. Estos datos serán los procesados a través de la red neuronal para lograr una salida.
- Los pesos son los únicos que cambian durante el aprendizaje, es decir, son coeficientes que se van adaptando. Existe un peso para cada una de las conexiones existentes en la red. Se dice que el aprendizaje de una red neuronal es el proceso mediante el cual se modifican los pesos en respuesta a una información de entrada [9].
- Cada elemento de procesamiento de la red, tiene asociada una salida única que será la que se transfiere a las neuronas vinculadas. Las salidas están asociadas a las funciones de activación, las cuales se estudiarán en la sección 2.3.3.

2.3.3. Algoritmo *Forward Propagation*

El algoritmo, en el cual se basan las redes neuronales, y mediante el que cada una de las neuronas que conforman la red adquiere un valor determinado, se denomina *Forward Propagation*. Se puede describir como una serie de transformaciones funcionales, en las cuales el valor que adquiere cada neurona, en una capa determinada, es el resultado de una operación lineal, en la que participan las neuronas de la capa anterior, y una operación no-lineal para la cual se usan las funciones de activación.

Supongamos que tenemos un modelo como el de la Figura 2.2 con dos *capas ocultas* las cuales constan de M neuronas. Denotaremos x_i a cada una de las D neuronas de entrada, $w^{(k)}$ a la matriz de pesos que controla el mapeo de la capa k a la capa $k+1$, $w^{(0)}$ a los *bias*, $a_j^{(k)}$ a la activación de la unidad j en la capa j y $g(\cdot)$ a la función de activación.

El valor que adquiere cada una de las activaciones de las primera capa oculta se calcula con la expresión de la ecuación 2.1.

$$a_j = \sum_{i=1}^{D} w_{ji}^1 x_i + w_{j0}^1 \tag{2.1}$$

A continuación, para obtener las salidas de cada una de las neuronas de la primera capa oculta, se transforma cada activación usando una función diferencial no lineal de la siguiente forma $z_j = g(a_j)$, la cual se denomina función de activación.

Estos últimos valores calculados, se combinan de forma lineal para dar las activaciones de la segunda capa oculta. En este caso, la siguiente capa es la de salida, y puesto que se tienen K neuronas en dicha capa, la expresión queda de la siguiente manera:

$$a_j = \sum_{j=1}^{M} w_{kj}^2 z_j + w_{k0}^2 \tag{2.2}$$

Finalmente, se transforman estas activaciones para dar las salidas de la red $y_k = g(z_k)$. Cabe destacar que la elección de la función de activación está determinada por la naturaleza de los datos [10].

Funciones de activación

Las funciones de activación representan de forma simultánea la salida de la neurona y su estado de activación, es decir, son las que determinan si la neurona está activa o no, en cuyo último caso la señal no se propagaría hacia el resto de neuronas. Estas funciones se caracterizan debido a que se suelen considerar deterministas y en la mayor parte de los modelos, son monótonas crecientes y continuas. A continuación se mencionan algunos tipos de funciones de activación existentes, las cuales se representan en la Figura 2.3.

- **Lineal**. Como su propio nombre indica, se trata de una función lineal que devuelve un valor proporcional a la entrada. Proporciona un rango de salida comprendido entre $-\infty$ y $+\infty$.

$$g(x_i) = cx_i \tag{2.3}$$

- **Sigmoide**. Función no lineal con rango de salida comprendido entre 0 y 1. Se suele utilizar en problemas de clasificación binaria.

$$g(x_i) = \frac{1}{1 + e^{-x_i}} \tag{2.4}$$

- **Tangh**. Función no lineal que devuelve valores en el rango -1 a 1. Se trata de una función sigmoide escalada.

$$g(x_i) = \frac{2}{1 + e^{-2x_i}} - 1 \tag{2.5}$$

- **ReLU (Rectifier Linear Unit)**. Función no lineal que devuelve como salida la propia entrada en el caso de que ésta sea positiva, y si no, devuelve el valor 0. Esto quiere decir que la salida está en el rango 0 y $+\infty$.

$$g(x_i) = max(0, x_i) \tag{2.6}$$

- **Softmax**. Función no lineal cuyo rango de salida está comprendido entre 0 y 1. Se suele utilizar en problemas de clasificación multiclase y devuelve lo que se considera la probabilidad a posteriori de cada una de las clases del problema.

$$g(x_i) = \frac{e^{x_i}}{\sum_i e^{x_i}} \tag{2.7}$$

- **Softplus**. Función no lineal cuyo rango de salida está comprendido entre 0 y $+\infty$. Surge como una alternativa a las funciones tradicionales Sigmoide y Tangh.

$$g(x_i) = ln(1 + e^{x_i}) \tag{2.8}$$

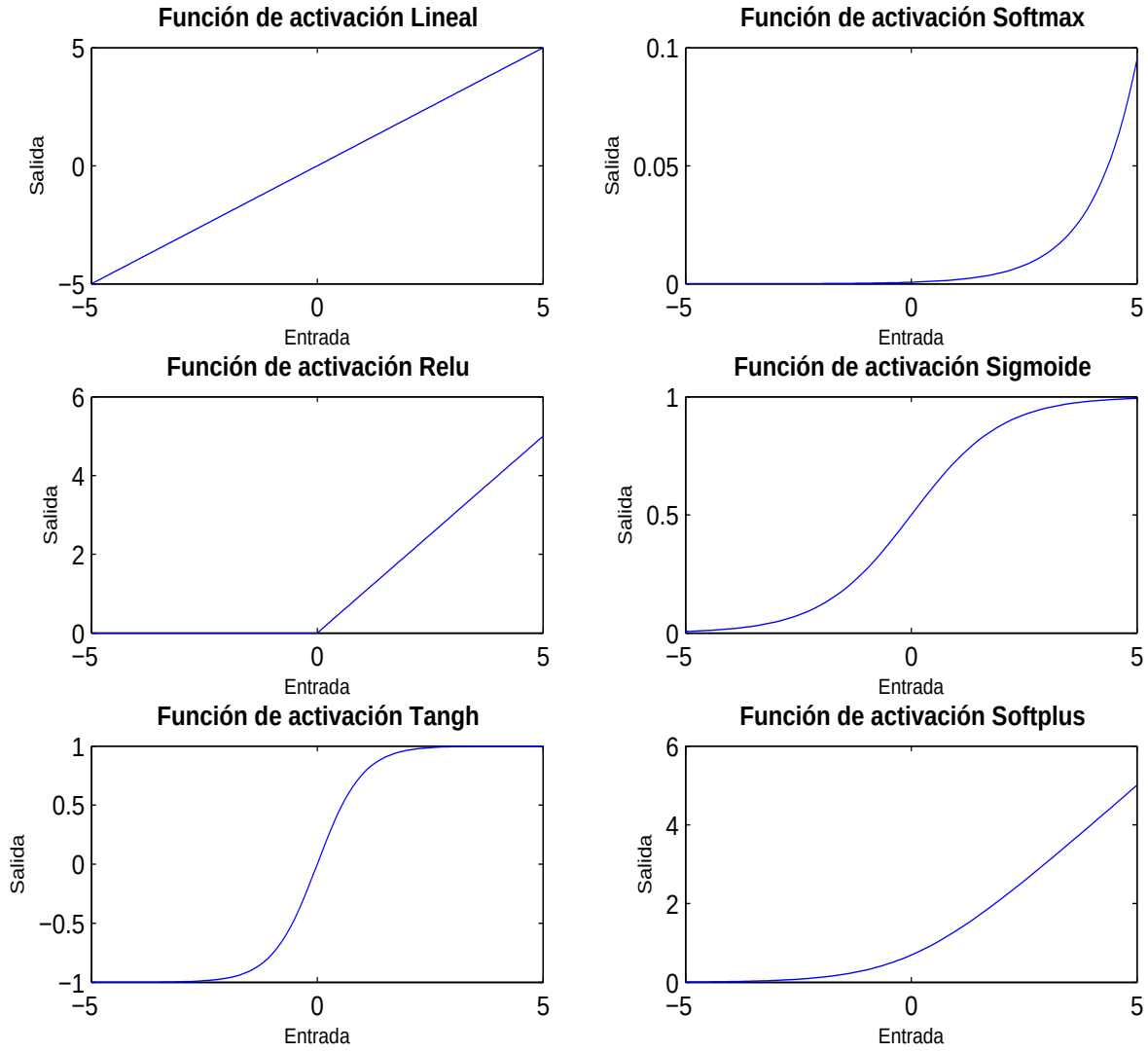

Figura 2.3: Representación gráfica de las funciones de activación.

2.3.4. Algoritmo *Backpropagation*

El término *Backpropagation* se suele utilizar para describir al algoritmo mediante el cual se entrena una red neuronal. El objetivo es obtener la combinación de pesos que minimizan el error entre los valores que predice el modelo y los valores reales, es decir, conseguir aquellos pesos que hacen que la red proporcione el rendimiento deseado.

Este algoritmo se basa en modificar los pesos de la red, de forma iterativa, siguiendo una cierta regla de aprendizaje construida a partir de la optimización de una función de error o coste que mide la eficiencia de la red. La ecuación 2.9 indica cómo se produce la actualización de los pesos en dicho algoritmo. Se puede observar que la variación de

los pesos, en cada iteración, es proporcional al gradiente de una función error o también llamada función de coste $E(w)$. Además, en dicha ecuación, aparece una variable η la cual se refiere a un parámetro llamado tasa de aprendizaje, el cual se explicará posteriormente.

$$w^{\tau+1} = w^{\tau} - \eta \nabla E(w^{\tau}) \tag{2.9}$$

Este algoritmo consta de una serie de pasos. En primer lugar, se inicializan los pesos de la red de forma aleatoria. A continuación se realiza la propagación *fordward*, explicada anteriormente, en la cual la entrada se propaga por las distintas capas de la red neuronal hasta llegar a la capa de salida. El tercer paso consiste en comparar la salida obtenida con la red neuronal y_k, con la salida esperada t_k, obteniendo un error (ecuación 2.10). Para ello se utiliza una función coste. De aquí en adelante, se denomina δ_j^l al error del nodo j en la capa l.

$$\delta_k = y_k - t_k \tag{2.10}$$

A partir del tercer paso, el algoritmo *Backpropagation* debe encargarse de ir modificando y ajustando los pesos de la red, de forma iterativa, de manera que se minimice el error obtenido.

Consideramos E_n como una función error que toma la siguiente forma:

$$E_n = \frac{1}{2} \sum_k (y_{nk} - t_{nk})^2 \tag{2.11}$$

donde $y_{nk} = y_k(x_n, w)$. Por otro lado, el gradiente de la función error con respecto a un peso determinado, se calcula como indica la ecuación 2.12.

$$\frac{\partial E_n}{\partial w_{ji}} = (y_{nj} - t_{nj}) x_{ni} \tag{2.12}$$

Recordando el algoritmo *forward propagation* se puede observar la relación existente entre los pesos y las funciones de activación, de manera que el error E_n depende del peso w_{ji} mediante la entrada sumada de a_j. Por lo tanto, aplicando la regla de la cadena de las derivadas parciales, la expresión 2.12 queda de la siguiente forma:

$$\frac{\partial E_n}{\partial w_{ji}} = \frac{\partial E_n}{\partial a_j} \frac{\partial a_j}{\partial w_{ji}} \tag{2.13}$$

como $\dfrac{\partial a_j}{\partial w_{ji}} = z_i$, si llamamos $\delta_j = \dfrac{\partial E_n}{\partial a_j}$ se tiene que:

$$\frac{\partial E_n}{\partial w_{ji}} = \delta_j z_i \tag{2.14}$$

La ecuación 2.14 indica que la derivada requerida, se obtiene multiplicando únicamente el valor del error δ_j por la transformación de la función de activación $z_i = g(a_i)$.

Para calcular el error δ_j de las capas ocultas, se utiliza la regla de la cadena para las derivadas parciales, como muestra la ecuación 2.15. El cálculo del error en la unidad j depende del sumatorio de los errores de las k unidades de la capa siguiente. La figura 2.4 muestra cómo se produce la propagación del error.

$$\delta_j = \frac{\partial E_n}{\partial a_j} = \sum_k \frac{\partial E_n}{\partial a_k} \frac{\partial a_k}{\partial a_j} \tag{2.15}$$

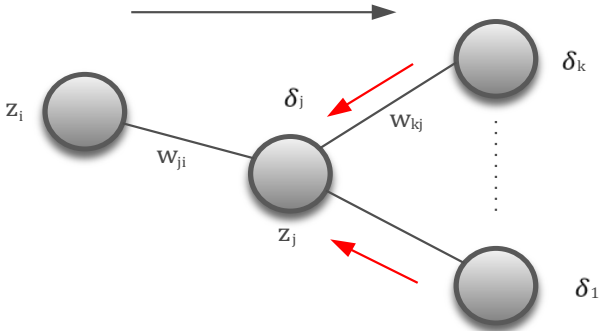

Figura 2.4: Representación gráfica del cálculo de δ_j para una unidad j.

Sustituyendo la definición de δ, dada en 2.14, en la expresión 2.15 y haciendo uso de las expresiones de activación $a_j = \sum_i w_{ji} z_i$ y salida $z_j = g(a_j)$ de una neurona, se puede obtener la ecuación final para el algoritmo *Backpropagation* 2.16 [10]. En dicha ecuación $g'(\cdot)$ corresponde a la derivada de la función de activación.

$$\delta_j = g'(a_j) \sum_k w_{kj} \delta_k \tag{2.16}$$

Volviendo a la ecuación 2.9, en la que se determinaba que la actualización de los pesos dependía del gradiente de la función de error $\nabla E(w)$ y combinándola con la ecuación 2.14, se tiene que la variación de los pesos, de una conexión que va desde una capa de la red hacia la siguiente, se calcula como:

$$\Delta w_{ji}^{\tau+1} = \Delta w_{ji}^{\tau} - \eta \delta_j z_i \tag{2.17}$$

Como ya se ha comentado, el término η corresponde a la tasa de aprendizaje. A mayor tasa, el proceso será más rápido. Sin embargo, si es demasiado alta puede dar lugar a oscilaciones en torno a un máximo local.

Funciones coste

La función de error o coste es una de las partes más importantes en el entrenamiento de una red neuronal, mediante el algoritmo *backpropagation*, ya que define la forma en la que una red va a aprender. Esto supone que la elección de la función de coste es crucial para conseguir que el modelo de red neuronal entrenado alcance el rendimiento que se desea.

Esta función mide la relación que existe entre la predicción de la red, es decir la salida proporcionada, y la que se quería obtener. Se trata de un valor positivo que disminuye a medida que aumenta el ajuste del modelo a los datos. A continuación se citan las funciones de coste más utilizadas así como se introduce su expresión matemática [11].

- **Mean Squared Error**

  La función *Mean Squared Error* (MSE) es comúnmente utilizada en regresión lineal como medida de rendimiento. Su expresión matemática consiste en minimizar la suma cuadrática de la distancia de cada punto de la línea de regresión. Es decir, busca que la salida de la red sea lo más parecida posible al valor deseado.

$$E = \frac{1}{n}\sum_{i=1}^{n}(y^{(i)} - \widehat{y}^{(i)})^2 \tag{2.18}$$

  El término n se refiere al número de dimensiones o muestras del vector con el que se trabaja, $y^{(i)}$ representa el vector objetivo o deseado y finalmente, $\widehat{y}^{(i)}$ representa al vector predicho por la red.

- **Mean Squared Logarithmic Error**

  Esta función también se usa para medir diferencias entre lo real y lo predicho. Se suele utilizar cuando no se desea penalizar grandes diferencias en los valores pronosticados y reales cuando éstos son números muy grandes.

$$E = \frac{1}{n}\sum_{i=1}^{n}(log(y^{(i)} + 1) - log(\widehat{y}^{(i)} + 1))^2 \tag{2.19}$$

  En la expresión anterior término n se refiere al número de dimensiones o muestras del vector con el que se trabaja, $y^{(i)}$ representa el vector de observaciones y finalmente, $\widehat{y}^{(i)}$ representa las predicciones de la red.

- **Mean Absolute Error**

  Mide el promedio de los errores en un conjunto de predicciones, sin considerar su dirección. Es decir, utiliza las diferencias absolutas entre los valores predichos y las observaciones.

$$E = \frac{1}{n}\sum_{i=1}^{n}|y^{(i)} - \widehat{y}^{(i)}| \tag{2.20}$$

Igual que las expresiones de las funciones de coste anteriores, el término n se refiere al número de dimensiones o muestras del vector con el que se trabaja, $y^{(i)}$ representa el vector objetivo o deseado y finalmente, $\widehat{y}^{(i)}$ representa al vector predicho por la red.

La diferencia entre el Mean Square Error (MSE) y el Mean Absolute Error (MAE) es que debido al cuadrado, los errores grandes tienen más influencia en el MSE (ecuación 2.18), por lo que el MAE (ecuación 2.20) es más robusto a valores atípicos.

- **Cross-entropy**

  Esta función se utiliza principalmente en problemas de clasificación. Tiene una estructura diferente a las expresiones mostradas anteriormente, ya que trabaja con distribuciones de probabilidad. En este caso, el término n corresponde al número de clases del problema, $y^{(i)}$ representa la distribución real obtenida a partir de los datos, y el término, $\widehat{y}^{(i)}$ a la distribución de probabilidad generada por el modelo.

$$E = -\frac{1}{n}\sum_{i=1}^{n}[y^{(i)}log(\widehat{y}^{(i)}) + (1 - y^{(i)})log(1 - \widehat{y}^{(i)})] \tag{2.21}$$

Optimizadores

Como se ha comentado, el aprendizaje de las redes neuronales consiste en un proceso de búsqueda de los pesos de la red de manera que se minimice la función de error. Sin embargo, esta función no es una función lineal, de modo que no se dispone de algoritmos sencillos para encontrar el mínimo y habrá que utilizar métodos, basados en el espacio de parámetros, que se vayan aproximando de forma iterativa. En particular, los métodos más usados son los que utilizan aproximaciones numéricas basadas en propiedades diferenciales de la función de error, como son los métodos basados en Descenso por Gradiente.

El Descenso por Gradiente es un método iterativo que se basa en construir un punto $w^{\tau+1}$ a partir de w^{τ}. La aproximación más simple para aplicar esta técnica es la expresada en la ecuación 2.22.

$$w^{\tau+1} = w^{\tau} - \eta\nabla E(w^{\tau}) \tag{2.22}$$

En la ecuación 2.22, $w^0, w^1, ..., w^{\tau}$ corresponde a la sucesión de parámetros y η es un valor positivo conocido como *learning rate* o tasa de aprendizaje, que puede fijarse a priori o calcularse mediante un proceso de optimización unidimensional a lo largo de la dirección de entrenamiento por cada una de las iteraciones. La elección del valor de

este parámetro η, es fundamental para alcanzar un buen entrenamiento y que el modelo sea eficiente. Cabe destacar que para valores de *learning rate* pequeños, el entrenamiento de la red neuronal será mas fiable, pero la convergencia hasta llegar a los pesos óptimos será lenta. Por otro lado, para valores grandes, los cambios en los pesos de cada iteración pueden ser tan elevados que el entrenamiento podría no converger.

En cuanto al movimiento, el vector se mueve en la dirección de mayor tasa de disminución de la función de error. Tras cada actualización, el gradiente se vuelve a evaluar para un nuevo vector de pesos. Se debe tener en cuenta que la función error se define con respecto al conjunto de entrenamiento y que en cada paso se requiere que se procese todo o parte de dicho conjunto para evaluar ∇E [12].

Existen una serie de algoritmos basados en Descenso del Gradiente, ampliamente utilizados en las redes neuronales. Estos algoritmos se diferencian en como usan el *learning rate*. Los más comunes se presentan a continuación.

- **SGD**

  El algoritmo *Stochastic Gradient Descent*, corresponde al clásico procedimiento de descenso de gradiente estocástico para actualizar los pesos de la red de forma iterativa en función de los datos de entrenamiento (ecuación 2.22).

  Este método utiliza un *learning rate* fijo que se puede seleccionar de forma empírica. Para ello, se podría comenzar con valores en torno a 0.1 e ir disminuyendo de forma exponencial a valores más bajos 0.01, 0.001, etc. El objetivo es conseguir que el algoritmo converja, pero siempre teniendo en cuenta la velocidad del entrenamiento. Sin embargo, la elección de una tasa de aprendizaje adecuada es difícil, pues como se ha mencionado anteriormente, valores demasiado pequeños pueden hacer que la convergencia del algoritmo sea muy lenta, mientras que valores grandes pueden dificultar la convergencia y hacer que la funcion de coste fluctúe alrededor de un mínimo o incluso diverja.

  Por este motivo, aunque es un optimizador que suele funcionar bien, la necesidad de fijar un *learning rate* ha hecho que se estudien otras propuestas, las cuales se mencionan a continuación.

- **Adagrad**

  Es un algoritmo de optimización que adapta el *learning rate* a los parámetros, desarrollando grandes actualizaciones a los menos frecuentes y menores actualizaciones a los más frecuentes. Por este motivo, es adecuado para tratar con datos dispersos.

  Adagrad adapta el *learning rate* a cada parámetro de la red, en función de los gradientes vistos previamente. Esto se basa en dividir el gradiente actual por la suma de los gradientes previos lo que hace que si el gradiente es muy grande, el parámetro η disminuye y viceversa.

$$g_{\tau+1} = g_\tau + \nabla E(w_\tau)^2 \tag{2.23}$$

$$w_{\tau+1} = w_\tau - \frac{\eta \nabla E(w_\tau)^2}{\sqrt{g_{\tau+1} + \epsilon}} \tag{2.24}$$

- **RMSprop**

  El *Root Mean Square Propagation* es un método basado en *learning rate* adaptativo. La única diferencia entre este método con el anterior es que el término g_τ se calcula mediante el promedio de la degradación exponencial y no mediante la suma de gradientes. De esta forma se consigue solucionar lo que se denomina como el *vanishing gradient problem*.

  Este problema consiste en lo siguiente. Como ya se ha comentado, cada peso se actualiza proporcionalmente al gradiente de la función de error en cada iteración. En algunas ocasiones, métodos como el Adragad pueden ocasionar que para aquellos parámetros que ocurren más frecuentemente, se tengan gradientes tan pequeños que se impide que el peso cambie de valor. En el peor de los casos este hecho puede detener por completo el entrenamiento de la red [13].

$$g_{\tau+1} = \gamma g_\tau + (1-\gamma) \nabla E(w)^2 \tag{2.25}$$

$$m_{\tau+1} = \gamma m_\tau + (1-\gamma) \nabla E(w) \tag{2.26}$$

  En la ecuación 2.25 el término g_τ se refiere al momento de segundo orden de ΔE y en la ecuación 2.26 el término m_τ se refiere al momento de primer orden. La expresión final se recoge en la ecuación 2.27.

$$w_{\tau+1} = w_\tau - \frac{\eta \nabla E(w)^2}{\sqrt{g_{\tau+1} - m_{\tau+1}^2 + \epsilon}} \tag{2.27}$$

- **Adam**

  Este algoritmo utiliza los beneficios de los algoritmos Adagrad y RMSProp. Al contrario que el algoritmo SGD que mantiene una tasa de aprendizaje única para todas las actualizaciones de los pesos, este método adapta el *learning rate* de cada parámetro individual a partir de las estimaciones de los momentos primero (ecuación 2.28) y segundo de los gradientes (ecuación 2.29).

  El procedimiento que realiza es el siguiente. En primer lugar se calcula el gradiente en el instante actual. A continuación se actualiza la estimación de primer y segundo orden. Se calculan las estimaciones de primer orden corregidas (ecuación 2.30) y de segundo orden (ecuación 2.31). Finalmente se actualizan los parámetros como indica la ecuación 2.32.

$$m_{\tau+1} = \gamma_1 m_\tau + (1-\gamma_1)\nabla E(w_\tau) \tag{2.28}$$

$$g_{\tau+1} = \gamma_2 g_\tau + (1-\gamma_2)\nabla E(w_\tau)^2 \tag{2.29}$$

$$\widehat{m}_{\tau+1} = \frac{m_{\tau+1}}{1-\gamma_1^{\tau+1}} \tag{2.30}$$

$$\widehat{g}_{\tau+1} = \frac{g_{\tau+1}}{1-\gamma_2^{\tau+1}} \tag{2.31}$$

$$w_{\tau+1} = w_\tau - \frac{\eta \widehat{m}_{\tau+1}}{\sqrt{\widehat{g}_{\tau+1}} + \epsilon} \tag{2.32}$$

En [14] los autores, además de presentar esta técnica, citan alguna de sus ventajas. En particular, este algoritmo destaca por ser eficiente computacionalmente, tener pocos requisitos de memoria y ser adecuado en problemas con muchos datos o parámetros. Además, se caracteriza debido a que soluciona problemas como son el *vanishing gradient problem* y el *exploding gradient problem*. Este último consiste en que, en ocasiones, el valor del gradiente, utilizado en la actualización de los pesos, es tan grande que puede hacer que los pesos adquieran un valor demsiado elevado, produciéndose un *overflow* o desbordamiento en la red y por consiguiente obteniéndose valores *NaN*.

2.3.5. Entrenamiento

Una vez que se han introducido las Redes Neuronales y se ha explicado el mecanismo de aprendizaje, se debe comentar cómo realizar un entrenamiento eficiente y efectivo que proporcione buenos resultados. A continuación se analizarán cada uno de los puntos a tener en cuenta cuando se va a proceder a crear y entrenar un modelo de Red Neuronal.

- **Datos de entrenamiento**

  El conjunto de datos de entrenamiento debe ser lo más grande posible ya que la Redes Neuronales necesitan de una gran cantidad de datos para lograr un buen aprendizaje. Algunas soluciones que se ofrecen, si el conjunto no es lo suficientemente grande es generar datos sintéticos, redimensionar datos o incluso agregar ruido. Por otro lado, hay que asegurarse que el conjunto de datos es el adecuado, eliminando datos dañados (textos cortos, imágenes distorsionadas, etiquetas de salida espurias, características con muchos valores nulos, etc.).

- **Estructura de la red**

  El número de neuronas de la capa de entrada está determinado por el número de datos de entrada. Por ejemplo, si se desea entrenar una red de modo que cada dato de entrada está determinado por 10 características, el número de neuronas de la capa de entrada será 10.

  Por otro lado, en cuanto al número de neuronas de la capa de salida, dependerá de la aplicación. Por ejemplo, para modelos de reducción de ruido, el número de neuronas de la capa de salida será el mismo que el de la capa de entrada. Sin, embargo para problemas de clasificación, el número de neuronas de la capa de salida será tal como clases hay.

  La elección del número capas y unidades ocultas de la red es otro de los aspectos clave para obtener buenos resultados. El número de neuronas ocultas determina la capacidad de aprendizaje de la red neuronal. Como lo que se desea es evitar el sobreajuste, la elección será tal que la red rinda de forma adecuada. Para conseguir el número adecuado se suele evaluar el rendimiento de varias arquitecturas y elegir en función de los resultados obtenidos. Por otro lado, en cuanto al número de capas ocultas, se ha demostrado que para la mayoría de los problemas prácticos basta con utilizar un par de capas ocultas.

- **Elección de los pesos iniciales**

  Se suele inicializar los pesos con valores aleatorios. En cuanto a la elección del rango en el que estén comprendido dichos valores, hay que tener en cuenta otros parámetros. Por ejemplo, si se inicializan los pesos con valores demasiado grandes y se utilizan funciones de activación de tipo Sigmoide, la salida que proporcionan dichas funciones estará saturada. Por otro lado, si los valores de los pesos son demasiado pequeños, los gradientes también serán demasiado pequeños.

  La elección de los valores iniciales de los pesos es realmente importante para alcanzar una convergencia eficiente. Es por esto que ha habido numerosas investigaciones al respecto. En particular Glorot y Bengio demostraron que la mejor opción es la de utilizar una distribución uniforme [15].

- **Tasas de aprendizaje**

  La elección de un valor de tasa de aprendizaje o *learning rate* (η), influye en el cambio de los pesos en cada iteración. Valores demasiado grandes puede dar lugar a inestabilidades en la función error lo que provoca que no se logre la convergencia. Valores demasiado pequeños puede ocasionar una muy baja velocidad en la convergencia lo que puede acabar en el estancamiento en un mínimo local. Es por esto que se suele elegir $\eta = 0.1$ [16].

  En este campo también ha habido una gran cantidad de investigación sobre métodos de optimización, que ha dado lugar a métodos basados en tasa de aprendizaje

adaptativa. Estos métodos, que se mencionaron en la sección 2.3.4, evitan la elección manual de una tasa de aprendizaje inicial y hacen que los modelos converjan sin problemas.

- **Elección del *batch* y del número de épocas**

  Se denomina época al proceso en el cual el conjunto de datos de entrenamiento, pasa por la red *hacia adelante* y *hacia atrás*. Sin embargo, no se pueden utilizar todos los datos de entrenamiento a la vez, por lo que este conjunto se suele dividir en lotes llamados *batchs*. En este caso, se cumplirá una época cuando todos los *batchs* hayan pasado por la red. Es por esto que el número de iteraciones necesarias para entrenar la red se calcula multiplicando el número de épocas por el número de *batchs* necesarios para completar una época.

  En cuanto al número de épocas necesarias en el entrenamiento de una red, no hay una respuesta única. Como es lógico no basta con una única época pues la actualización de los pesos no será suficiente. Sin embargo, tampoco se pueden utilizar demasiadas épocas pues se puede llegar a sobreajustar el modelo. El número de épocas necesario está relacionado con la diversidad de los datos.

2.4. Filtrado de Wiener

El filtro de Wiener, es uno de los filtros lineales óptimos más conocidos. En su forma más general, el modelo sobre el que se basa el filtro, consiste en una señal $d(n)$ a la cual se le ha añadido ruido $v(n)$ obteniéndose una señal $x(n)$ y un filtro lineal de respuesta $w(n)$. Tal y como vemos en la Figura 2.5, este filtro es alimentado por $x(n)$ y produce a su salida $\widehat{d}(n)$ que no es más que una estimación de la señal deseada $d(n)$. Esta técnica se basa en la minimización del error cuadrático medio (MSE) entre la señal de interés y la estimada.

El proceso consiste en determinar el filtro que obtiene mejor estimación $\widehat{d}(n)$.

$$e(n) = d(n) - \widehat{d}(n) = d(n) - w(n) * x(n) \tag{2.33}$$

siendo $w(n)$ la respuesta al impulso del filtro de longitud L, la cual será determinada en función de las características de la señal.

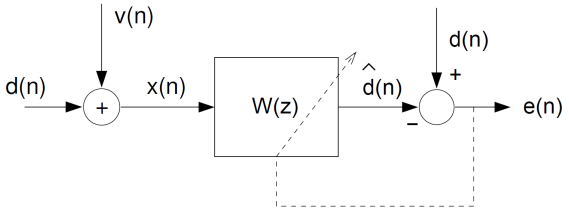

Figura 2.5: Esquema del Filtro de Wiener.

Como ya se ha comentado, el principio para obtener la mejor limpieza, es obtener el filtro que minimiza el MSE, $J(w) = E\left\{e^2(n)\right\}$. Para obtenerlo, se debe definir previamente el filtro que no altera la señal $w_1 = [1, 0, ..., 0]^T$. A continuación se calcula el MSE (2.34):

$$\begin{aligned} w_0 &= argmin_w J(w) \\ J(w_0) &< J(w_1) \end{aligned} \tag{2.34}$$

Desarrollando la ecuación 2.34, se obtiene la expresión 2.35 en la cual R_d se refiere a la matriz de correlación de la señal observada y r_{dx} al vector de correlación cruzada entre la señal observada y la señal limpia de ruido.

$$\begin{aligned} E\left\{d(n) \cdot d^T(n)\right\} \cdot w_0 &= E\left\{d(n) \cdot x(n)\right\} \\ R_d \cdot w_0 &= r_{dx} \end{aligned} \tag{2.35}$$

Puesto que el ruido y la señal deseada están incorreladas, se puede expresar el vector de correlación cruzada de la siguiente forma:

$$r_{dx} = E\left\{d(n) \cdot d(n)\right\} - E\left\{v(n) \cdot v(n)\right\} = r_{dd} - r_{vv} \tag{2.36}$$

De este modo se obtiene la expresión final de la estimación del filtro.

$$w_0 = R_y^{-1} \cdot r_{dd} - R_y^{-1} \cdot r_{vv} \tag{2.37}$$

Sin embargo, el cálculo del inverso de la matriz de corrrelación no es una tarea sencilla. En ocasiones es imposible de calcular o tiene un coste computacional muy elevado. Por este motivo, han surgido distintas técnicas basadas en solucionar este problema. En [17] se propone un filtro de Wiener adaptativo que se basa en adaptar el filtro a la señal del problema en función de sus estadísticos locales. Asume que el ruido tiene media cero y varianza σ_v^2 por lo que el espectro de potencia queda:

$$P_v(w) = \sigma_v^2 \tag{2.38}$$

Considerando un pequeño segmento en el cual la señal es estacionaria, se tendría que $x(n)$ se puede modelar como indica la expresión 2.39.

$$x(n) = m_x + \sigma_x w(n) \tag{2.39}$$

donde m_x e la media del segmento, σ_x es la desviación estándar y $w(n)$ es una unidad de varianza del ruido.

Dentro de ese pequeño segmento se puede aproximar la respuesta al impulso del filtro de Wiener como indica la expresión 2.40.

$$w(n) = \frac{\sigma_d^2}{\sigma_d^2 + \sigma_v^2}\delta(n) \tag{2.40}$$

donde σ_d se irá actualizando a medida que se tengan muestras, mientras que σ_v es constante. Será esta actualización la que haga que el filtro se adapte a los estadísticos de la señal.

Se ha demostrado que el filtrado anterior puede alterar los eventos, disminuyendo su amplitud, por lo tanto se ha buscado otra técnica que no altere la señal. En particular, en este trabajo se va a utilizar el filtro de Wiener propuesto por Pascual Scalart et al. en [6], al cual se le han realizado algunas modificaciones.

La estimación del ruido se basa en un factor de supresión a corto plazo *Short-Term Suppresion Factor*. Como esta técnica supone que las componentes espectrales son estadísticamente independientes, dicho factor se ajusta de forma automática en función de la *SNR a posteriori* de cada frecuencia.

Si se tiene que $x(t) = d(t) + v(t)$, entonces se definen $D_k = A_k e^{j\alpha_k}$, B_k, $X_k = R_k e^{jvk}$ como la k-ésima componente espectral de la señal, del ruido y de las observaciones ruidosas en el intervalo $[0, T]$.

Esta técnica, divide en primer lugar la señal en tramas. A continuación, calcula, la *SNR a posteriori* como se indica en la 2.41.

$$SNR_{post}(w_k) = \frac{|X_k|^2}{E[|B_k|^2]} \tag{2.41}$$

El siguiente paso es utilizar la SNR calculada anteriormente para obtener una *SNR a priori* que se define de la siguiente forma (2.42):

$$SNR_{prio}(w_k) = (1-\beta)P[SNR_{post}(w_k) - 1] + \beta\frac{|D^{t-1}|}{E[|B_k|^2]} \tag{2.42}$$

donde β es un parámetro, que según Pascual Scalart [6] debe tomar el valor 0.98. Este proceso, se realiza de forma iterativa, para cada unas de las tramas en las que se ha dividido la señal. Como se puede ver, la segunda parte de la ecuación 2.42, se refiere a la trama anterior, por lo que esta técnica va calculando la SNR_{prio} en función de la trama de entrada y de la trama anterior, por lo que se va ajustando a los propios datos de la señal y a su evolución con el tiempo.

Finalmente, queda la función de transferencia del filtro como indica la ecuación 2.43.

$$H(w) = \frac{SNR_{prio}(w)}{1 + SNR_{prio}(w)} \tag{2.43}$$

Se puede estimar la señal sin ruido como la multiplicación entre la señal ruidosa y la función de transferencia del filtro, en el dominio de la frecuencia (ecuación 2.44), o la convolución en el dominio del tiempo (ecuación 2.45).

$$\widehat{D}(w) = H(w) \cdot X(w) \tag{2.44}$$

$$\widehat{d}(t) = h(t) * x(t) \tag{2.45}$$

Capítulo 3

Entorno experimental

En este capítulo se analizará la Base de Datos con la que se va a trabajar. Además se desarrollará el entorno generador de señales sintéticas que se ha implementado. Finalmente, se elegirán las herramientas en las que se realizará el trabajo y se diseñarán las medidas de rendimiento que se van a utilizar para evaluar el sistema final.

3.1. Análisis de la Base de Datos

Este Trabajo Fin de Máster se enmarca en el contexto del control de calidad en la fabricación de piezas industriales. El objetivo es detectar características presentes en estas piezas, para lo cual se utilizará un *array* de sensores que miden distintas propiedades. El análisis de las señales proporcionadas por dicho *array* será el que nos permita encontrar anomalías en las piezas industriales.

Los sensores registrarán señales con el paso de la pieza por el *array*. En particular, este trabajo está centrado en dos tipos de señales, una denominada señal densidad y la otra denominada señal de material X. En este último caso el sensor mide la concentración de un determinado material, que por motivos de confidencialidad por parte de la empresa Tecnatom, S.A se denominará material X.

La Base de Datos de la que se dispone, está formada por 181 señales multicanal, resultantes tras el paso de 4 piezas industriales por los distintos sensores. En específico, se tienen 30 inspecciones de la pieza número 1, 10 inspecciones de la pieza número 2, 10 inspecciones de la pieza número 3 y 17 inspecciones de la pieza número 4. Sin embargo, cabe destacar que no todas las inspecciones son iguales, pues aunque presentan una estructura idéntica con los mismos eventos, tanto la amplitud como las muestras no son iguales. Por otro lado, las 113 señales restantes no presentan ningún tipo de evento, por lo que serán empleadas para caracterizar el ruido asociado a cada sensor.

De cada una de las inspecciones se dispone de la señal densidad, de la señal correspondiente al material X, así como de otras señales que no se utilizarán en este trabajo. La señal densidad proporciona una medida inversa a la densidad de la pieza, es decir, tiene

una amplitud elevada en los puntos donde la pieza presenta menor densidad y viceversa. Por otro lado, la señal de material X, mide la concentración de dicho material en la pieza. Se caracteriza porque puede presentar una deriva creciente o decreciente que hace que se incremente o decrezca el nivel medio de la señal con el paso de las muestras. Además, esta señal presenta una fuerte componente frecuencial interferente, la cual provoca que la señal tenga el aspecto de una onda periódica escalonada. Este comportamiento puede ser debido a la interferencia electromagnética de los motores del sistema de actuación.

Puesto que el objetivo del trabajo e la detección y clasificación de los distintos tipos de eventos presentes en las señales de densidad y de material X, es importante realizar un estudio de dichos eventos, de manera que se permita conocerlos y distinguirlos con facilidad. Por ello, a continuación se presentan las principales características de estos eventos así como se muestra un ejemplo de cada uno de ellos en las Figuras 3.1 y 3.2.

- **Ruido**

  Es introducido por el propio sensor que capta la señal tras el paso de la pieza industrial, esto significa que cada una de las señales con la que se trabaja presenta un ruido de distintas características. El ruido existente en la señal densidad presenta un alto valor de varianza. Además tiene una gran potencia, lo que hace que en ocasiones algunos de los eventos presentes en esta señal no sean distinguibles (Figura 3.2). Por otro lado, el ruido visible en la señal de material X tiene poca potencia, por lo que no supondrá ningún inconveniente para el cumplimiento de los objetivos marcados.

- **Desviaciones**

  Son variaciones positivas o negativas visibles, únicamente, en la señal de material X. Este evento ayuda a la detección de las piezas que presentan desviaciones con respecto a la concentración de material X que debe tener según las especificaciones.

- **Transiciones**

  Corresponde a las zonas de paso entre regiones con diferente concentración del material X. Las transiciones pueden ser ascendentes, si se pasa de una zona de baja concentración a una zona de alta concentración, o descendentes si se pasan de zonas altas a zonas bajas. Este evento también es únicamente visible en la señal de material X.

- **Gaps**

  Es un evento visible tanto en la señal densidad, en forma de grandes variaciones positivas, como en la señal de material X, en forma de pequeñas variaciones negativas. Sirve para detectar roturas en las piezas o huecos entre sus componentes.

  Los gaps presentes en la señal de material X, pueden ser confundidos con pequeñas desviaciones negativas, por lo que la distinción de estos dos eventos se realiza mediante el uso de la señal densidad (Figura 3.1).

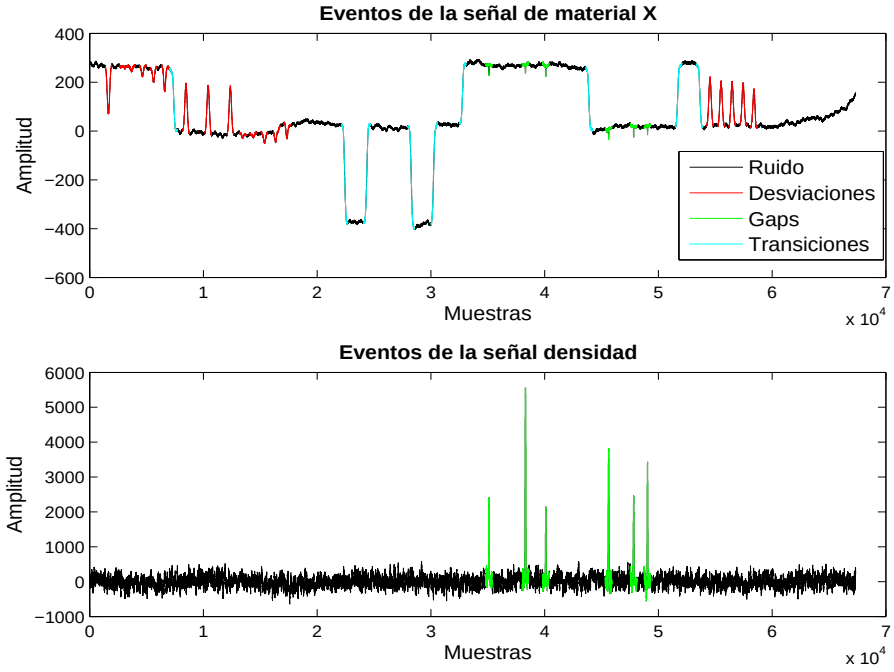

Figura 3.1: Señal multicanal, con la correspondiente señal de material X y señal densidad, que presenta los cuatro tipos de eventos posibles. Se han señalado los eventos presentes utilizando las etiquetas de Ground Truth.

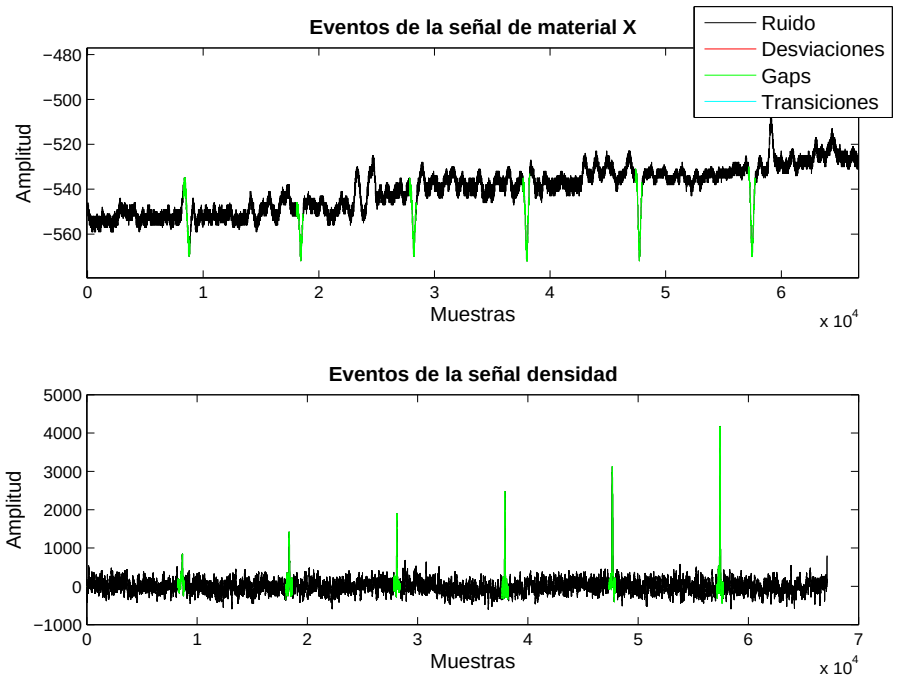

Figura 3.2: Señal multicanal, con la correspondiente señal de material X y señal densidad, que únicamente presenta eventos de tipo gap. Se han señalado los eventos presentes utilizando las etiquetas de Ground Truth.

3.2. Entorno simulado

Los algoritmos supervisados, en particular los basados en Redes Neuronales, necesitan una gran cantidad de datos para ser entrenados. Sin embargo, la Base de Datos de la que se dispone es limitada, pues no contiene suficientes señales. Por este motivo se ha implementado un entorno que simule señales multicanal, tanto de densidad como del material X, que sean lo más parecidas posible a las reales.

El primer paso es realizar un estudio de las características de los eventos que se disponen. En concreto, se analizará la relación existente entre la altura y la anchura de las señales en las zonas en las que existe un evento de tipo gap, transición o desviación. Para ello se recogerá el valor de la anchura y la altura de cada uno de los eventos de las señales de la Base de Datos. A continuación, se realiza *clustering* sobre estos datos para agruparlos y tener un número determinado de anchuras y alturas posibles. Seguidamente, se ha generado una función de distribución acumulada para cada anchura, que muestra la relación existente altura/anchura así como la frecuencia de aparición. Esta función ayudará a que el entorno proporcione más prioridad a aquellos conjuntos anchura-altura que más veces han aparecido en los datos reales.

En cuanto al funcionamiento del entorno, se recibe como entrada el valor del número de muestras de la señal, el número de eventos que aparecerán y la potencia del ruido a añadir. A continuación, de forma aleatoria, separa los eventos y elige qué tipo de evento se situará en cada posición. El siguiente paso, consiste en calcular el ancho y alto correspondiente a cada uno de los eventos, para lo cual utiliza los datos generados en el estudio de las características mencionado anteriormente. Con estos valores se simulan los eventos y se van añadiendo a la señal final. Por último se realiza la adición de ruido sintético.

Para la generación de cada evento, se elige en primer lugar, y de forma aleatoria, el valor de la anchura (dentro del rango posible). Por otro lado, el valor del alto, se obtiene basándose en las funciones de distribución, ya que estas reflejan cuales son las alturas más probables para cada uno de los anchos posibles. Este procedimiento hace que los eventos generados se parezcan estadísticamente lo máximo posible a los eventos reales. Cabe destacar la aleatoriedad existente en los datos elegidos, de manera que no se generen todos los eventos iguales.

Modelado de ruido sintético

Para la adición de ruido sintético se dispone de 113 señales en la base de datos. Estas señales son de calibración por lo que no presentan ningún tipo de evento. Sin embargo, tras diversas pruebas se ha llegado a la conclusión de que no todas las señales representan bien al ruido. Por ello se ha realizado un estudio para seleccionar aquellos modelos de ruido, que mejor representan al ruido presente en las señales reales disponibles y por lo tanto aquellos que finalmente se utilizarán en el entorno generador de señales sintéticas.

El estudio realizado ha hecho uso del Filtro de Wiener [6] que se introdujo de forma teórica en la sección 2.4. Cabe mencionar que este filtro realiza limpieza de la señal

utilizando el propio ruido existente en ella. Por este motivo, se ha propuesto realizar limpieza de ruido con Wiener para 17 señales densidad distintas y con los 113 modelos de ruido disponibles y almacenar la SNR de la señal resultante. Teóricamente, analizando la evolución de la SNR para cada señal y modelo de ruido utilizado, se obtendrán los modelos que más se asemejan al ruido real existente.

En la Figura 3.3 se han representado los resultados obtenidos. Podemos ver la evolución de la SNR obtenida tras limpiar las 17 señales con los 113 modelos de ruido. En primer lugar, se han descartado los 30 primeros modelos, debido a que son los que peores resultados proporcionan. A continuación, de los restantes se han eliminado aquellos que presentan una caída abrupta en la SNR obtenida. Finalmente se ha elegido un conjunto de 50 con los cuales se ha generado un *super-modelo* de ruido y que será el que se utilice para la generación de señales sintéticas.

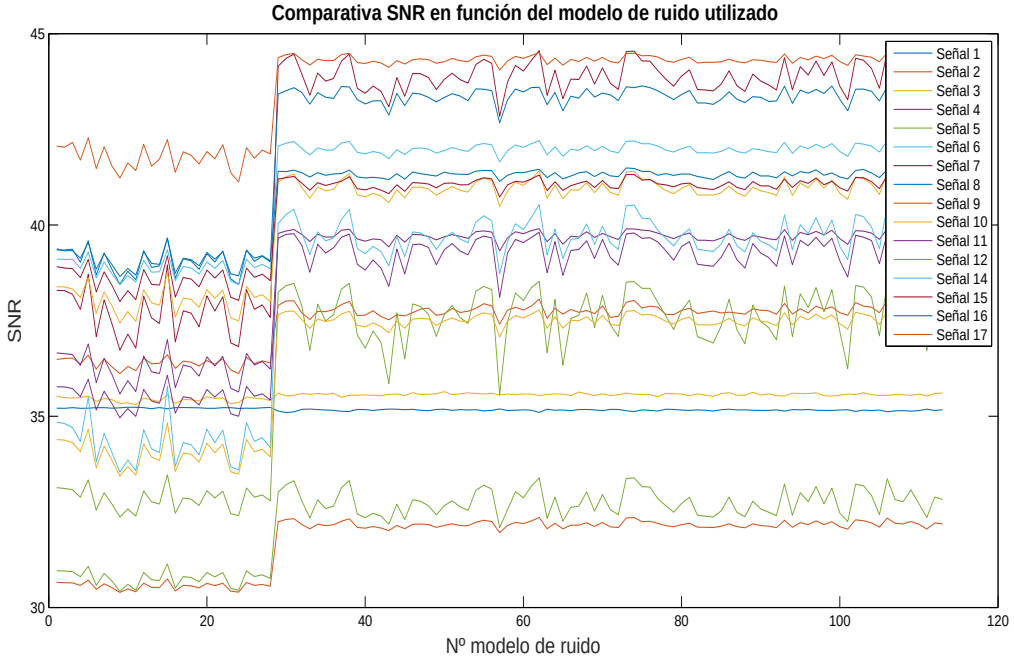

Figura 3.3: Comparativa de la SNR obtenida tras filtrar 17 señales reales con Wiener y utilizando los 113 modelos de ruido.

En la Figura 3.4, se representa una comparativa entre una señal multicanal real y una señal multicanal sintética. Cabe destacar que esta señal presenta eventos tipo ruido, desviaciones y transiciones. Además, es muy similar a la real pues se ha obligado a que presente los mismos tipos de eventos, y de forma aproximada en las mismas posiciones. Esto se ha hecho así con el fin de analizar el buen funcionamiento del sintetizador. Por otro lado, la Figura 3.5 muestra una comparativa entre una señal sintética y una real que únicamente presentan eventos tipo gaps.

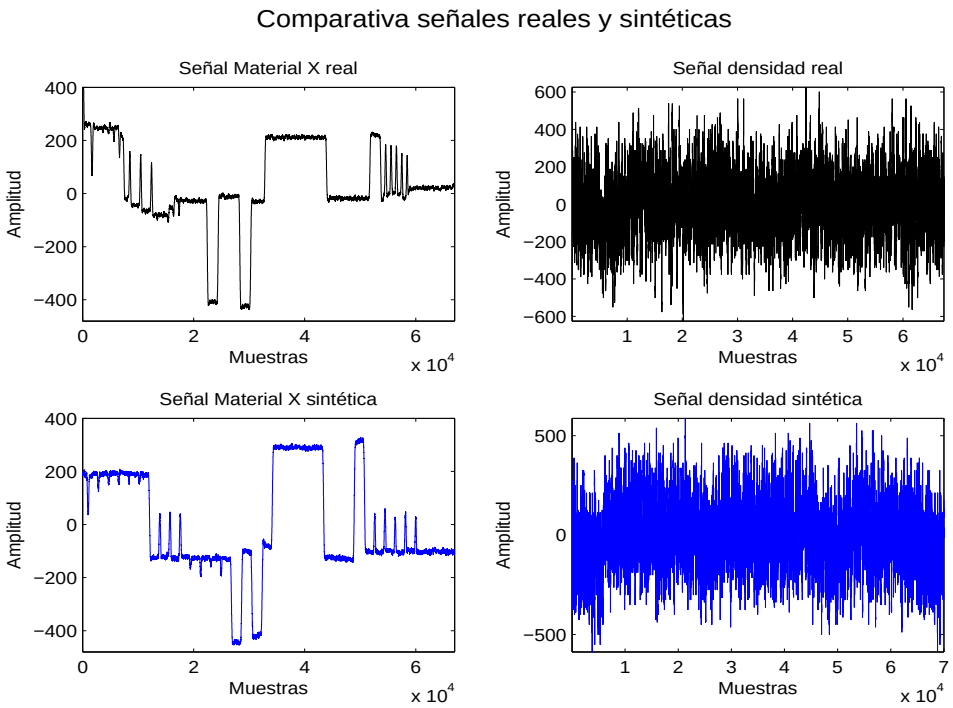

Figura 3.4: Comparativa entre señal multicanal real, con corrección de deriva, y señal sintética con presencia de todos los tipos de eventos.

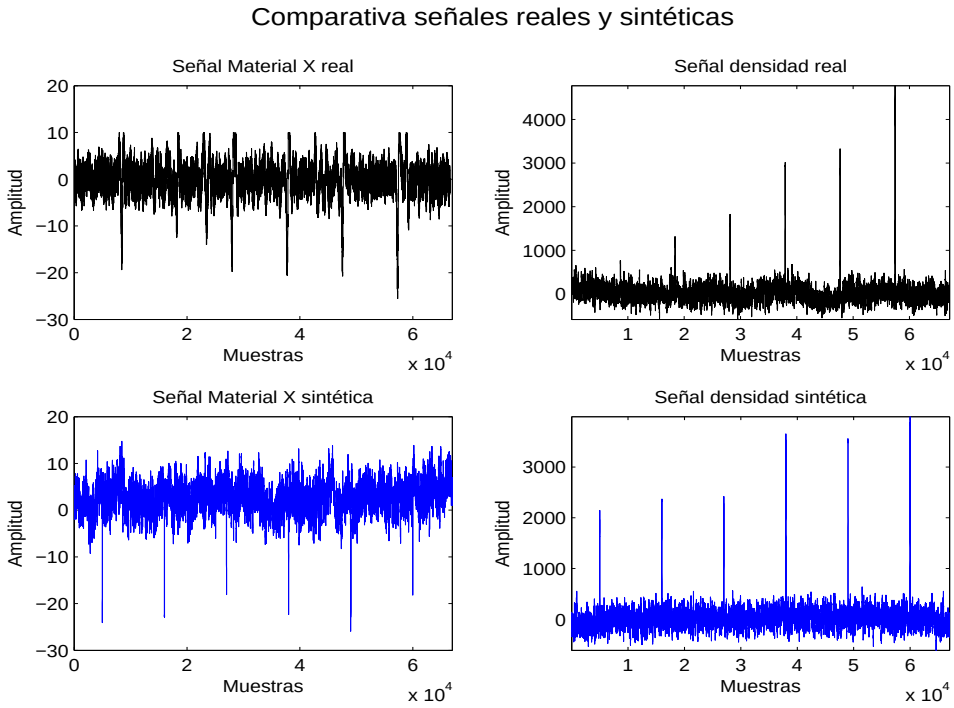

Figura 3.5: Comparativa entre señal multicanal real, con corrección de deriva, y señal sintética con presencia de eventos tipo gap.

Cabe destacar, que como se puede observar en las Figuras 3.1 y 3.2 las señales reales suelen presentar una deriva que no es más que una caída o un incremento, no deseado, en el nivel medio de la señal, que se adquiere a medida que aumenta el número de muestras. Esta deriva, no ha sido añadida a las señales sintéticas, pues como se demostrará en la sección 4.2.1, dicha deriva no afecta en el sistema final implementado. Por este motivo, en las Figuras 3.4 y 3.5 se han mostrado las señales reales tras aplicarles corrección de deriva, para que la comparativa, con las señales sintéticas, fuera más justa. Al margen de esto, ambas figuras muestran que las señales sintéticas y las reales mantienen bastante semejanza, con respecto a la forma de los eventos e incluso al ruido presente e ellas.

3.3. Herramientas

En esta sección se describen las herramientas utilizadas en la implementación de este Trabajo Fin de Máster.

Keras

Keras es una API de redes neuronales de alto nivel lanzada en 2016, escrita en Python y capaz de ser ejecutada sobre TensorFlow, CNTK o Theano. Permite la creación de prototipos fácil y rápida, admite redes convolucionales y recurrentes y se puede ejecutar sobre CPU o GPU.

Se ha utilizado sobre Theano para crear los distintos modelos de redes neuronales de reducción de ruido y detección de eventos. Estos modelos han sido entrenados y evaluados mediante diversos *scripts* generados en Python.

Matlab

Matlab que es una abreviatura de *MATrix LABoratory* es una herramienta de software matemático que ofrece un entorno de desarrollo intregado (IDE) con lenguaje de programación propio (lenguaje M).

Esta herramienta software se ha utilizado para el procesado de los datos de entrada y de salida de la Red Neuronal. Es decir, se han implementado algunas funciones que han transformado las señales de la Base de Datos en las características que se usan como entrada de la red neuronal. Por otro lado, con respecto a los datos de salida, en el caso de reducción de ruido se ha implementado una función que transforma las características de salida de la red en una señal. En el caso de detección y clasificación de eventos se ha utilizado Matlab para procesar los *scores* que devuelve la red neuronal. Además se ha utilizado para realizar estudios sobre la elección de diversos factores o parámetros de los sistemas y se han creado algoritmos para evaluar el rendimiento de los sistemas implementados.

3.4. Medidas de rendimiento

Uno de los aspectos más importantes en la implementación de un sistema, es la evaluación de la eficiencia de las técnicas desarrolladas. En este caso, se van a utilizar una serie de medidas de rendimiento mediante las cuales se comprobará la eficacia de los algoritmos de limpieza de ruido y la precisión en la detección y clasificación de los eventos de las señales.

A continuación, se comentan las técnicas que se han utilizado para evaluar los sistemas implementados en este Trabajo Fin de Máster.

3.4.1. SNR (*Signal to noise ratio*)

Se define Relación Señal/Ruido a una proporción existente entre la potencia de una señal y la potencia del ruido que la corrompe.

$$SNR = \frac{S}{N} \tag{3.1}$$

Esta relación es buena para determinar si la limpieza de ruido realizada ha sido eficaz. Una limpieza correcta supondría que la relación señal/ruido (SNR) antes de la limpieza es menor que la que se obtiene después de la limpieza.

El cálculo de la SNR es sencillo, pues se dispone de señales etiquetadas, cuyas etiquetas muestran el comienzo y el fin de cada evento. Para calcular este parámetro se debe estimar, en primer lugar, la potencia de las zonas de la señal que únicamente presentan ruido (ecuación 3.2).

$$N = \sum_{n} x(n)^2 \tag{3.2}$$

A continuación, se debe calcular la potencia de la señal, para lo cual se necesita disponer de zonas de la señal en las que no haya ruido. Suponiendo que la señal y el ruido son independientes, se puede calcular la potencia de la señal como la diferencia entre la potencia de la señal ruidosa y la potencia del ruido 3.3. Finalmente, se calcula la SNR, la cual queda representada en la ecuación 3.4.

$$X = S + N = \sum_{k} x(k)^2 \tag{3.3}$$

$$SNR = \frac{S}{N} = \frac{X - N}{N} = \frac{X}{N} - 1 \tag{3.4}$$

3.4.2. Curva DET

El sistema implementado de detección y clasificación de eventos, da como salida, para cada evento de entrada, una puntuación o *score* a cada una de las clases posibles. En este caso, puesto que se dispone de 4 tipos de eventos, se obtendrán 4 *scores*. Esta puntuación será proporcional a la certeza que tenga el detector sobre qué tipo de evento es.

El objetivo del sistema es que, si la entrada es un evento tipo ruido, se puntúe con un mayor valor a la clase ruido (*target*) y con valores pequeños a las clases restantes (*nontarget*). Por lo tanto, se pretende aumentar la distancia entre los valores *score* de los *target* y de los *nontarget*, de manera que se pueda utilizar un umbral de decisión.

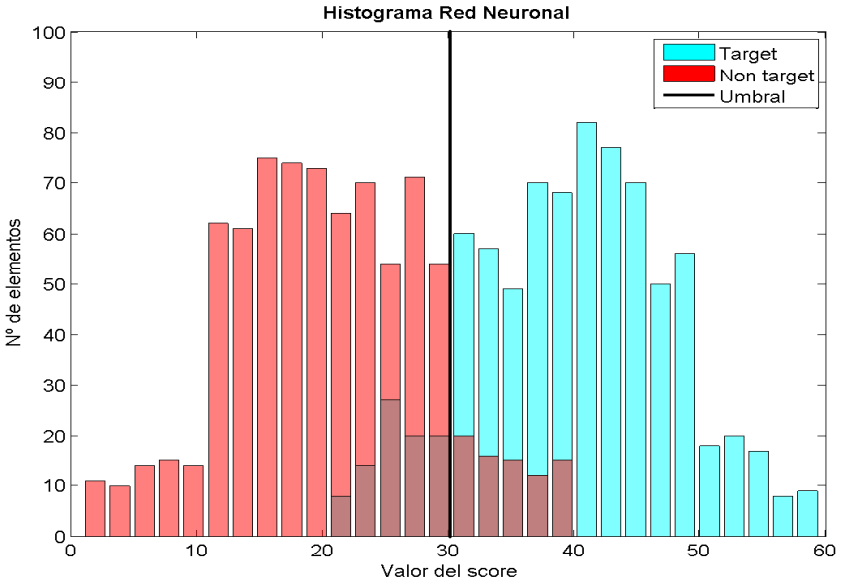

Figura 3.6: Representación de puntuaciones obtenidas para la clase target y para la clase nontarget, así como del umbral óptimo de decisión.

A la hora de clasificar, utilizando dicho umbral, se pueden cometer dos tipos de errores:

- Falsa aceptación: Corresponde a muestras que son clasificadas como un tipo de evento, cuando en realidad no lo son. La tasa de falsa aceptación aumenta, a medida que disminuye el umbral de clasificación, pues se clasifica como evento, muestras que tienen bajos *scores*.
- Falso rechazo: Corresponde a muestras que no han sido clasificadas como evento, cuando si deberían haberlo sido. La tasa de falso rechazo aumenta, a medida que aumenta el umbral de clasificación, pues se necesitan de *scores* muy altos para clasificar una muestra como evento.

La curva DET es una buena forma de representar la tasa de falso rechazo frente a la tasa de falsa aceptación, para todos los tipos de umbrales posibles. Es decir, mediante

esta curva se puede ver como se modifican dichas tasas mediante la evolución del valor del umbral utilizado.

En la diagonal principal de dicha curva, se encuentran los puntos de la gráfica donde la tasa de falsa aceptación es igual a la tasa de falso rechazo. La intersección de las curvas que representan a las tasas con la diagonal, representa el porcentaje de EER (*Equal Error Rate*). Este valor es el que determina el rendimiento del sistema detector.

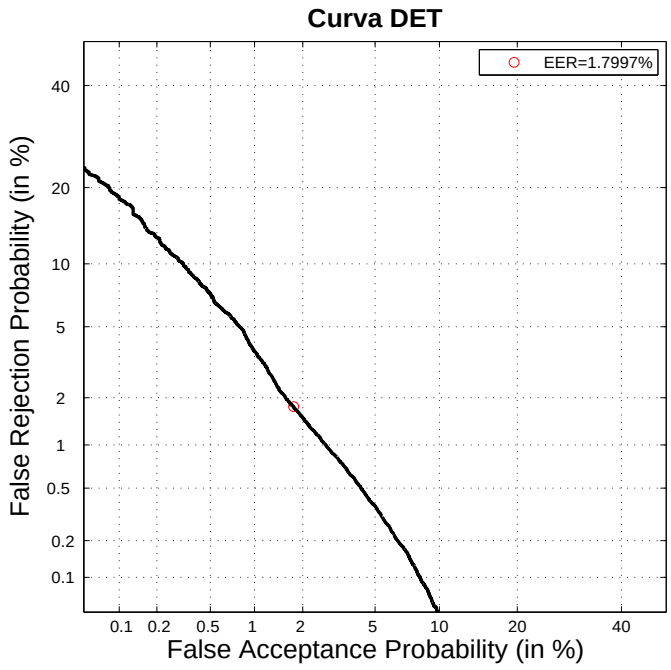

Figura 3.7: Representación de una curva DET y del valor EER correspondiente.

3.4.3. Accuracy

Además de las curvas DET, existen otras medidas de rendimiento en los sistemas de clasificación. En particular, se va a comentar el concepto de *accuracy* o precisión, el cual mide el número de targets que se han clasificado correctamente respecto a número total de elementos de los que se dispone.

Cabe destacar que esta medida no siempre da información relevante, pues supongamos que se tienen 100 muestras, 90 correspondientes a ruido y 10 correspondientes a otro evento. Si se clasifican 9 de las 10 muestras de evento, como evento, se tiene una precisión del 99 %. Sin embargo, si de las 10 muestras correspondientes a evento, no se clasifica ninguna como evento, se tendría una precisión del 90 %. Se trata de un valor bastante alto, cuando en realidad se está perdiendo todos los eventos.

Es por esto que se necesita otro concepto llamado *línea base* el cual indica la precisión, si se supone que todas las clasificaciones son iguales al evento que más veces aparece. En

el caso del ejemplo, cuando se acierta 9/10 eventos, se tendría un *accuracy* del 99 % sobre una línea base de 90 %.

3.4.4. Matriz de confusión

En el campo del aprendizaje automático y específicamente el problema de la clasificación estadística, una matriz de confusión es un diseño de tabla específico que permite la visualización del rendimiento de un algoritmo, generalmente un aprendizaje supervisado. Las filas representan las clases reales del problema, mientras que las columnas representan a las predicciones realizadas. Por otro lado, dentro de cada celda, se representa el porcentaje de veces en las cuales las muestras de una clase determinada se han clasificado como la propia clase o como las demás clases del problema.

Cabe destacar la importancia de la diagonal de esta matriz ya que es la que realmente muestra el rendimiento del sistema, pues proporciona información acerca del porcentaje de clasificación correcta para cada clase.

En la Figura 3.8 se puede un ejemplo de una matriz de confusión. Los resultados muestran que, por ejemplo, el evento transición se ha clasificado un 5 % de las veces como evento tipo ruido y como evento desviación, un 10 % como evento tipo gap y un 80 % como evento transición.

Además, se puede ver que la matriz se ha representado en escala de grises, que van desde el blanco para porcentajes bajos, hasta el negro para porcentajes altos. Por todo eso, se concluye que un sistema óptimo, en cuanto al rendimiento, es aquel cuya matriz de confusión presente altos porcentajes en la diagonal, representados con colores oscuros, y porcentajes bajos en el resto de celdas, representadas con colores claros.

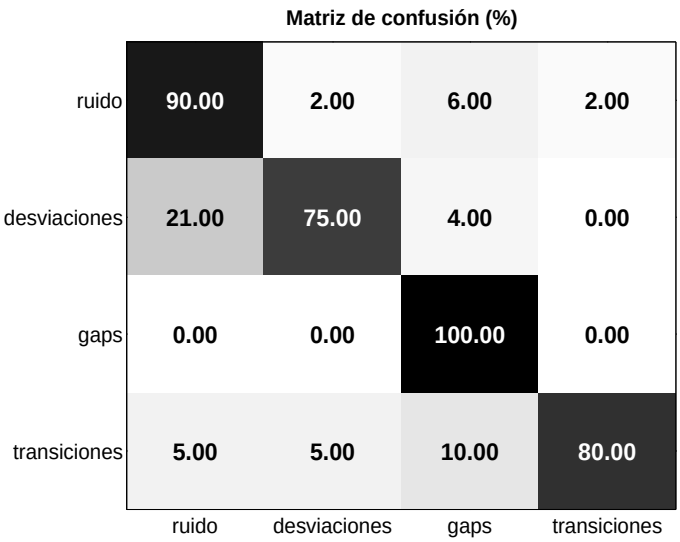

Figura 3.8: Ejemplo de una matriz de confusión con cuatro clases distintas: ruido, desviaciones, gaps y transiciones.

Capítulo 4

Diseño y desarrollo del sistema

El principal objetivo de este capítulo es mostrar cuál ha sido la evolución del trabajo desarrollado. Para ello se profundizará sobre cada uno de los sub-sistemas que componen el sistema final, así como de los bloques de los que están formados. En particular, se comentará el funcionamiento de cada bloque, así como las técnicas de procesado avanzado de señal que se han utilizado. Además, se realizarán estudios teóricos que ayuden a seleccionar algunos parámetros a usar, de manera que se elijan aquellos que optimicen el rendimiento del sistema.

4.1. Arquitectura del sistema

La realización de un estudio de las señales que conforman la Base de Datos hace prever que la utilización de únicamente la señal de material X no será suficiente para alcanzar el objetivo. Se ha llegado a esta conclusión debido a que, aunque dicha señal contiene todos los tipos de eventos, las características de éstos pueden llevar a la confusión, por ejemplo entre desviaciones negativas y gaps. Por este motivo se ha realizado un enfoque multicanal, utilizando, además de la señal correspondiente al material X, la señal densidad, la cual únicamente refleja los eventos tipo gaps. De esta manera, combinando ambas señales, todos los eventos serán completamente distinguibles.

Sin embargo, las señales densidad presentan un ruido que puede dificultar la percepción de algunos eventos, sobre todo de aquellos cuya amplitud sea muy pequeña. Por este motivo se ha tenido la necesidad de implementar dos subsistemas distintos. El primero de ellos realiza reducción de ruido a las señales registradas por el sensor densidad y el segundo se encarga de la detección y clasificación de eventos. La combinación de ambos subsistemas dará lugar al sistema multicanal final que se representa en la Figura 4.14.

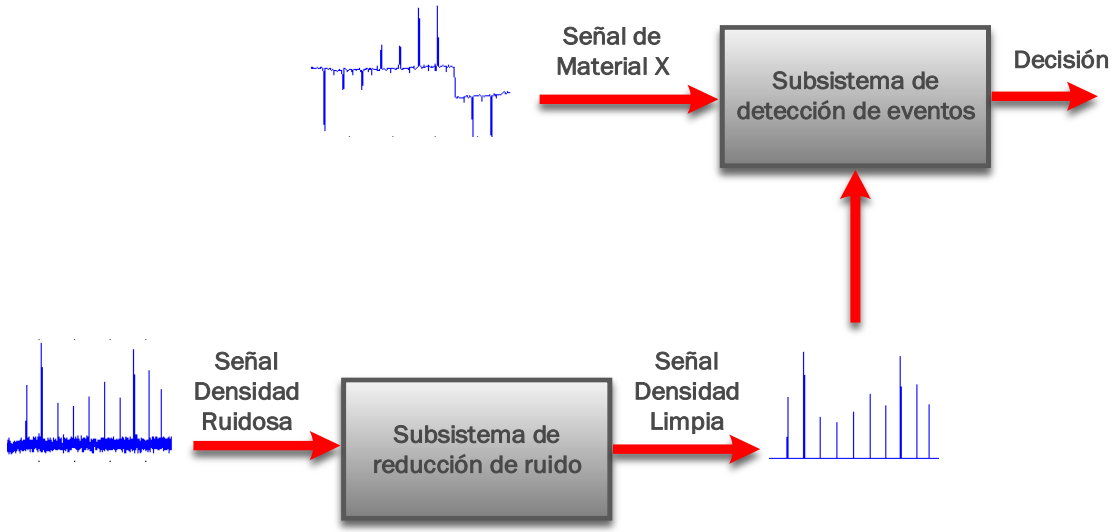

Figura 4.1: Diagrama de bloques del sistema de detección y clasificación de eventos.

Cada uno de los subsistemas consta de una serie de etapas las cuales van, desde la realización de un pre-procesado de la señal de entrada, hasta la salida, ya sea la señal densidad limpia de ruido, en el subsistema de reducción de ruido, o la clasificación final de cada evento, en el subsistema de detección de eventos. En las siguientes secciones se mostrará el diagrama de bloques de cada uno de ellos, así como se describirán las características más significativas.

4.2. Subsistema preliminar de detección y clasificación de eventos

Para lograr el objetivo principal de este trabajo, que es generar un sistema de detección y clasificación de eventos, se propuso utilizar la señal correspondiente al material X, registrada por los sensores en cada inspección de una pieza. Esta decisión preliminar fue tomada, debido a que como se ha podido comprobar en la Figura 3.1, esta señal refleja cada uno de los eventos posibles.

La Figura 4.2 muestra el diagrama de bloques del sistema preliminar implementado. Éste recibe como entrada la señal de material X, la cual pasa al bloque extractor de características. Dentro de este bloque la señal es filtrada y segmentada en distintas tramas. A continuación se genera un conjunto de características que representan a la señal y que son la salida de este bloque. El segundo bloque es una red neuronal que se entrenará para que aprenda qué conjunto de características corresponde a cada evento. Este bloque generará unos *scores* que serán la entrada al último bloque, el decisor, el cual tras analizar y procesar la entrada, es capaz de proporcionar a la salida los tipos de eventos correspondientes.

Figura 4.2: Diagrama de bloques del sistema de detección y clasificación de eventos inicial.

4.2.1. Extracción de características

La elección de las características que se van a utilizar no es una tarea arbitraria, pues deben ser lo suficientemente buenas como para representar la señal y lo suficientemente discriminatorias entre los distintos tipos de eventos que se pretenden detectar.

Antes de comenzar con la extracción de características, se debe hacer un pre-procesado a la señal de entrada. La primera parte consiste en un filtrado paso bajo Chebyshev de tipo 2, el cual reducirá parte del ruido que presentan las señales reales.

La segunda parte consiste en generar la derivada de la señal. Esta decisión se ha tomado debido a que, como se puede ver en la Figura 4.3, las señales reales contienen una deriva. Puesto que no se desea que se vea reflejada dicha deriva en las características, se ha decidido realizar la derivada a la señal, de manera que se consigue que no se vean distintas zonas en ella, sino que esté centrada en 0. En la Figura 4.3 se muestra una comparativa entre una señal de material X y su derivada. Cabe destacar que los diferentes eventos siguen siendo visibles en la señal derivada.

Cuando se tiene preparada la señal, se realiza una segmentación en tramas. Para ello se utilizará un enventanado, que no puede ser mediante la utilización de ventanas demasiado grandes pues se tendría pocas tramas que representen a cada evento, ni con ventanas muy pequeñas, pues en ese caso, las tramas que representen a cada evento no serán lo suficientemente discriminatorias. Se han utilizado, por tanto, ventanas de tipo rectangular de 600 muestras con un solapamiento del 90 %. Estos valores han sido elegidos tras diversas pruebas y debido a las diferentes propiedades de las piezas industriales con las que se trabaja. En este trabajo nos hemos basado en los estudios preliminares desarrollados en [19], aunque se realizará una selección posterior, como se explicará más adelante

En cuanto a la extracción de características en sí, se ha optado por utilizar las componentes espectrales de la señal ya que dan información la amplitud de la señal, de la velocidad de variación e incluso de su orientación. Por ello, para obtener las características se va a aplicar la FFT (*Fast Fourier Transform*) a cada una de las tramas, de las cuales se obtendrá el módulo y la fase. Además, se pretende reforzar dichas características, mediante la adicción de los coeficientes delta [19]. Estos coeficientes aseguran que

en las características de cada trama, esté presente información de las tramas adyacentes, es decir, estos coeficientes proporcionan información de la evolución temporal de las características en cada uno de los eventos.

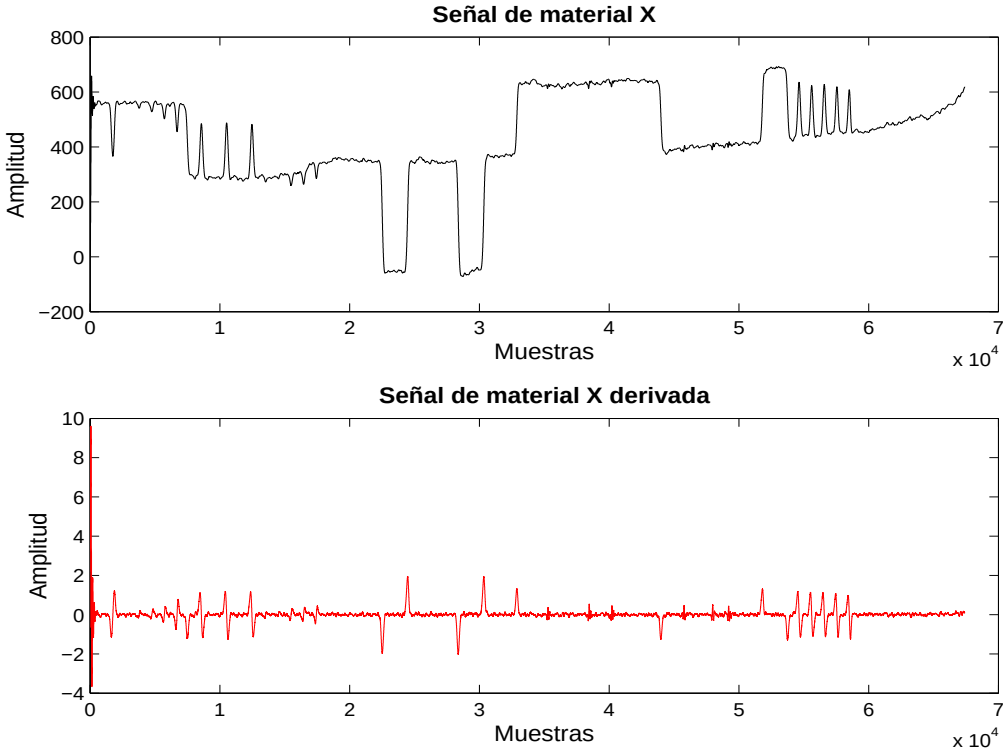

Figura 4.3: Comparativa entre la señal de material X y su derivada.

El uso del módulo y la fase de la FFT, así como de los coeficientes delta, da lugar a un conjunto de características demasiado grande. Esto puede dar lugar a que la red neuronal tenga muchas más componentes y por lo tanto sea lenta computacionalmente, e incluso puede ocasionar problemas de *overfitting*. Por este motivo, se ha realizado un estudio en el que se determina qué características son las que dan más información y por lo tanto, son más discriminatorias entre eventos.

Este estudio hace uso de la varianza intraclase e interclase para cada uno de los eventos. Lo óptimo es que la varianza intraclase sea lo más pequeña posible, pues eso significará que las características representan de manera óptima a dicho evento, así como la varianza interclase sea lo mayor posible, lo que significa que las características son lo suficientemente discriminatorias para diferenciar eventos.

Una vez se han calculado dichas varianzas, se necesita un valor que las represente, el cual se utilizará para la elección de las características. Este valor se denominará *ratio inter-intra* y se obtiene tal y como indica la ecuación 4.1. Cabe destacar que las características seleccionadas serán aquellas que presenten un ratio mayor.

$$ratio = \frac{\sigma^2_{interclase}}{\sigma^2_{intraclase}} \tag{4.1}$$

Para llevar a cabo la selección de características, se han utilizado las 68 señales de la base de datos. El primer paso ha sido descomponer las señales en tramas y obtener el módulo de la FFT de dichos segmentos así como los coeficientes delta. A continuación se han utilizado las etiquetas para dividir los conjuntos de características en función del evento al que corresponden. Seguidamente se han calculado los valores de varianza interclase e intraclase para cada característica y tipo de evento. Finalmente se han obtenido los *ratios inter-intra*.

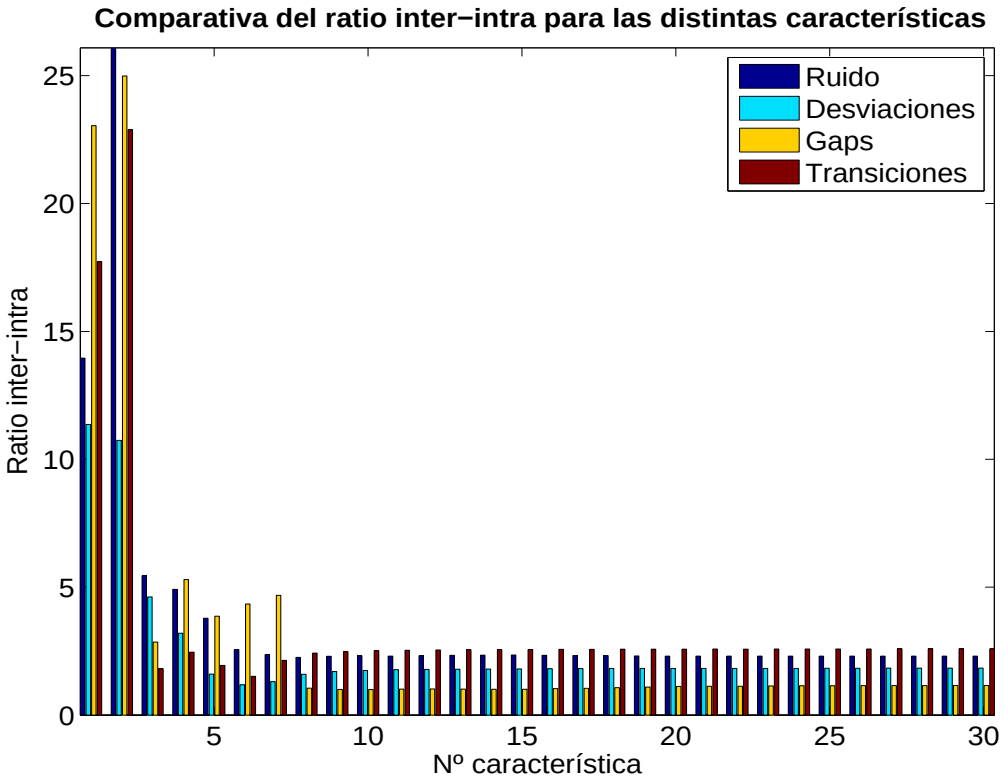

Figura 4.4: Comparativa entre los *ratios inter-intra* obtenidos para cada evento y para cada una de las características extraídas de las señales.

La elección del módulo de la FFT y no de la fase, se ha tomado en base a que la fase no es lo suficiente discriminatoria. Por ejemplo, la clase desviaciones agrupa tanto desviaciones positivas como desviaciones negativas lo que implica que dentro de dicha clase, la fase presentará una alta varianza. Esto mismo también ocurre en la clase transiciones. Por este motivo, se ha decidido que la fase no aportará información de interés que ayude a la distinción entre eventos.

En la Figura 4.4 se han representado las 30 primeras características obtenidas. Analizando dicha figura, se ha decidido utilizar las 15 primeras características con sus respectivos coeficientes delta, ya que son las que mayor valor de ratio ofrecen. Esto significa que cada una de las tramas en las que se descompone la señal, estará representada por un conjunto de 30 características.

Cabe destacar que estos resultados se han obtenido mediante la utilización de señales reales de la Base de Datos, sin embargo, el entrenamiento de las redes se realizará con señales sintéticas. Por lo tanto, uno de los aspectos de interés es comprobar cómo de semejantes son las características de las señales reales y de las sintéticas.

Para ello, se ha implementado una señal sintética lo más parecida posible a una de las señales reales, intentando que presente los mismos tipos de eventos y en la mismas posiciones. A continuación, se ha realizado una representación de ambas señales, la real y la sintética, frente a las 5 primeras características seleccionadas ya que son las que presentan mayor amplitud. Esta comparativa, que se puede ver en las Figuras 4.5 y 4.6, muestra el gran parecido existente entre estas señales y entre las características que las representan.

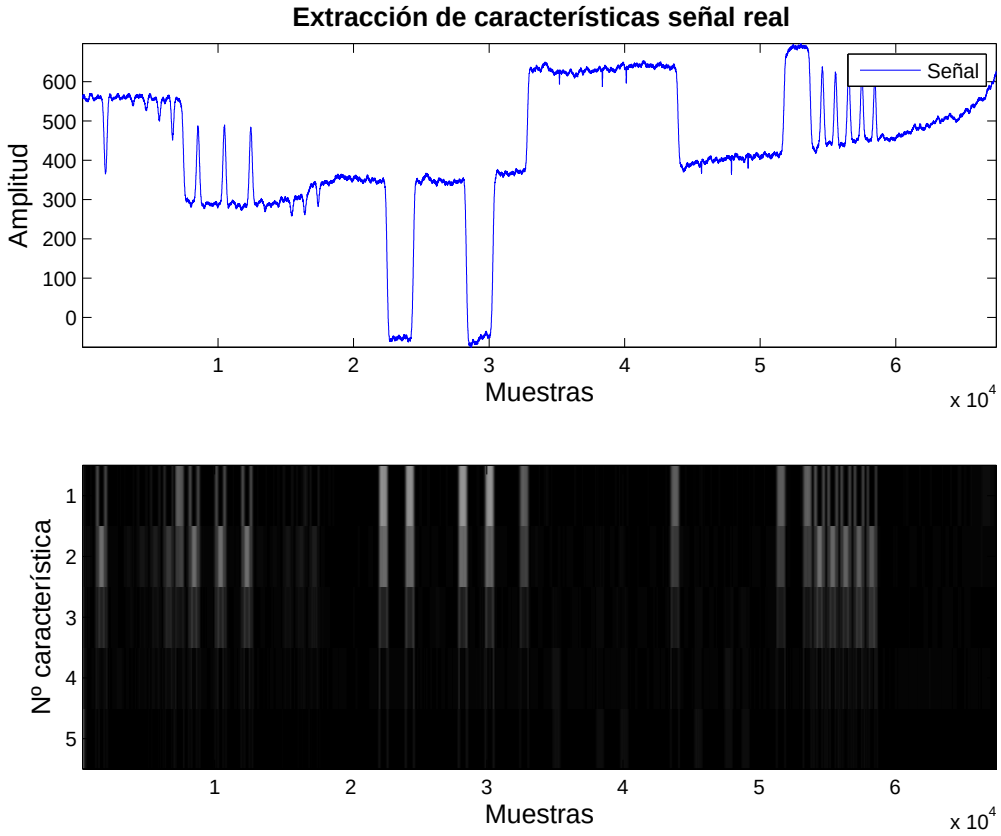

Figura 4.5: Comparativa entre una señal de material X real y las características que la representan.

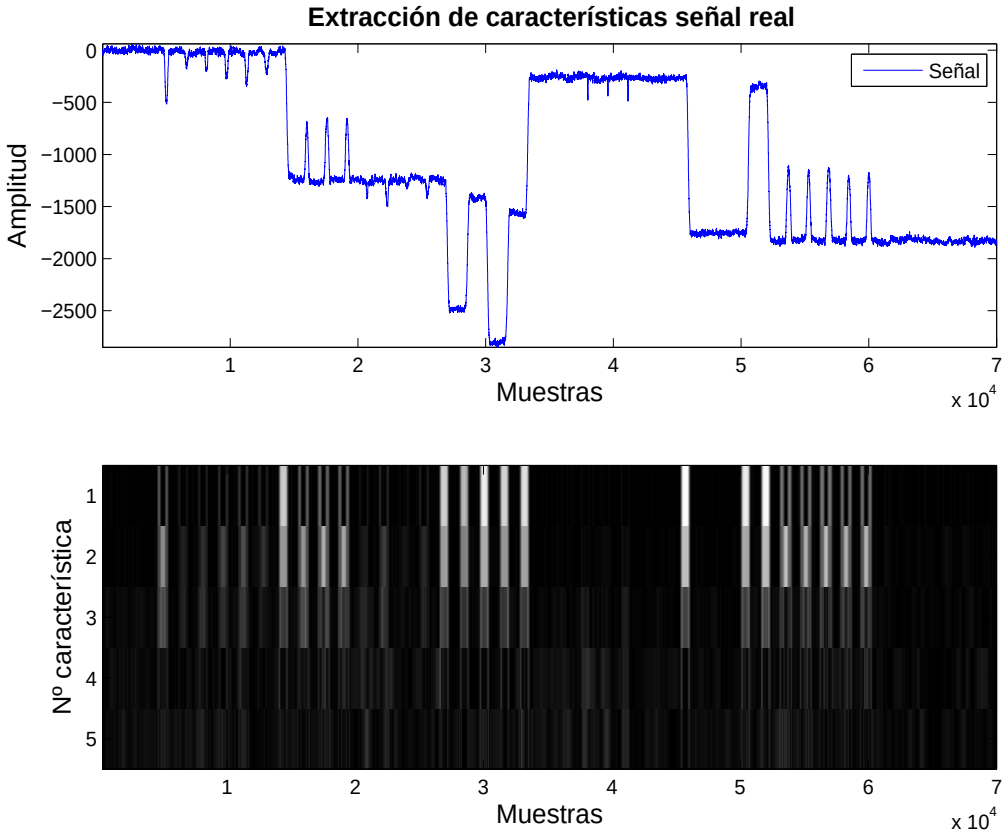

Figura 4.6: Comparativa entre una señal de material X sintética y las características que la representan.

4.2.2. Creación y entrenamiento de la red neuronal

El segundo bloque del sistema preliminar de detección de eventos, está formado por una red neuronal. Se ha utilizado esta técnica ya que en los últimos tiempos, las redes neuronales se han convertido en una de las técnicas de reconocimiento de patrones más famosas. En el Estado del Arte se realizó un estudio teórico de los conceptos fundamentales que hay que conocer para entender lo suficientemente bien el modelo de red implementado, por lo que este apartado estará centrado en la elección de los principales parámetros que describen dicho modelo de red neuronal.

En primer lugar, cabe destacar que se ha representado a cada una de las tramas en las que se segmenta la señal, con 30 características, las cuales serán la entrada de la red neuronal. Por lo tanto, la estructura de la red debe constar de 30 neuronas en la capa de entrada. En cuanto a la capa de salida, se va a diseñar la red para que la salida sea un *score* o puntuación a cada una de las clases del problema, es decir, dada una trama de entrada se dará una puntuación a cada tipo de evento posible. Por este motivo, la capa

de salida consta de 4 neuronas, una por cada tipo de evento: ruido, desviación, transición o gap.

Teniendo en cuenta esta finalidad del modelo de red se han escogido las funciones de activación a utilizar. La escogida para la capa de entrada y las capas ocultas ha sido la ReLu, así como la elegida para la capa de salida es la Softmax, ya que es muy útil para problemas multiclase, pues esta función devuelve un *score* para cada clase posible.

Para finalizar con la estructura de la red neuronal, falta por determinar el número de capas ocultas, así como el número de neuronas en cada capa. La elección de estos parámetros se realiza de forma empírica, tras numerosos entrenamientos de la red y la posterior evaluación. Teniendo en cuenta la cantidad de datos de la que se dispone, se ha decidido utilizar únicamente dos capas ocultas. Además, este sistema es preliminar por lo que no se ha considerado realizar ningún estudio empírico con respecto al número de capas.

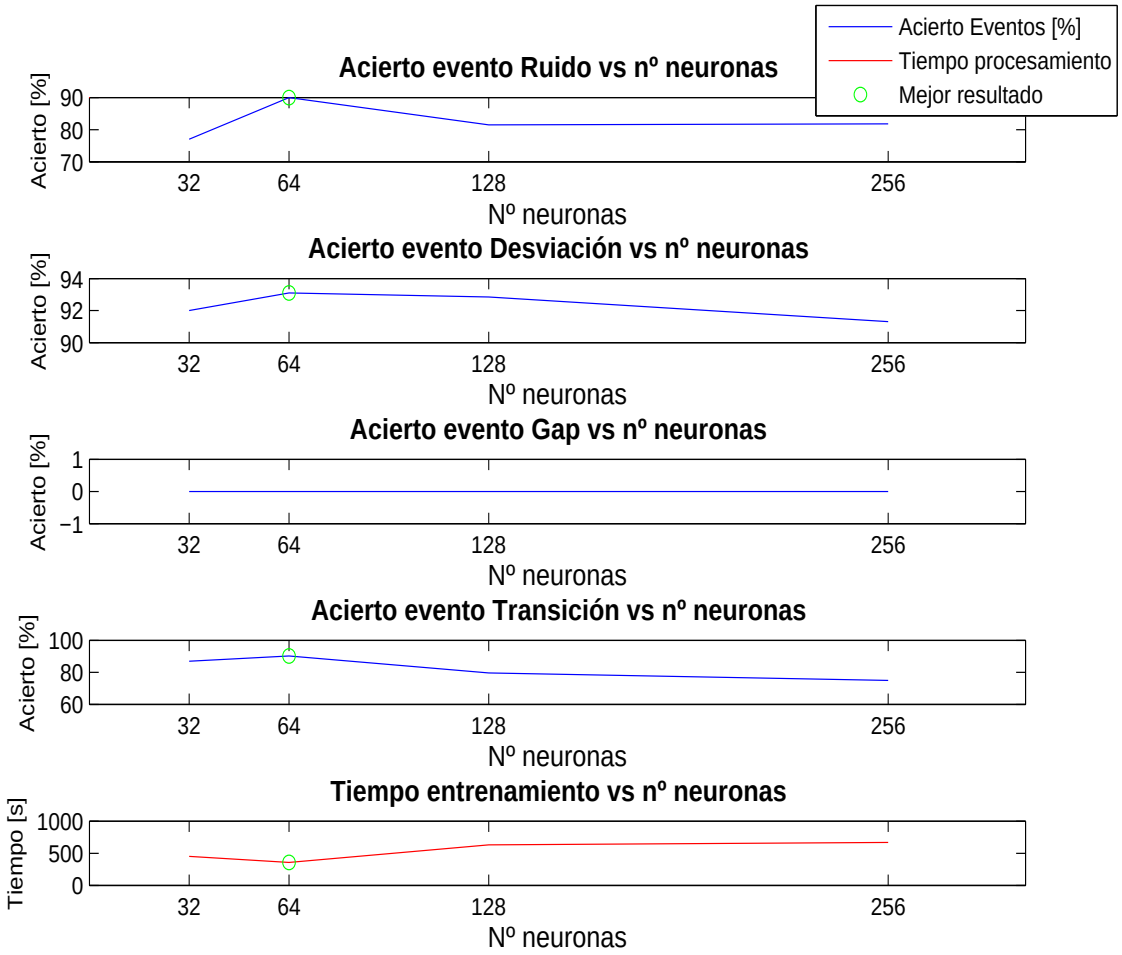

Figura 4.7: Evolución del porcentaje de acierto obtenido para cada evento en función del número de neuronas utilizadas en las capas ocultas, así como evolución del tiempo de entrenamiento.

Con respecto al número de neuronas, se ha representado en la Figura 4.7 cómo evoluciona el porcentaje de acierto para cada tipo de evento detectado, en función del número de neuronas que forman las capas ocultas. Cabe destacar que en este caso lo que se ha denominado acierto corresponde al porcentaje de tramas de la señal que se han clasificado de forma correcta. Además, se ha representado el tiempo de entrenamiento necesario para obtener el modelo final. Este dato es importante ya que se quiere lograr un alta eficiencia computacional, es decir, se desea obtener buenos resultados y en el menor tiempo posible.

Viendo estos resultados se puede determinar que el modelo que mejor funciona es el que tiene 64 neuronas, pues proporciona más de un 90 % de acierto para los eventos ruido, desviación y gap. Así mismo, este modelo es el que menos tiempo de computación ha necesitado para su entrenamiento.

Además llama la atención que para todos los modelos entrenados se produce un acierto del 0 % en el evento tipo gap. Esto significa que este sistema, que utiliza únicamente la señal de material X, no es capaz de detectar eventos de esta clase. Este hecho lleva a que la suposición realizada desde el principio, de que con una única señal no basta para lograr una clasificación óptima, es la correcta. Por lo tanto, el sistema final, además de utilizar la señal de material X, usará la señal densidad ya que ésta refleja los eventos tipo gaps, que son los únicos que no es capaz de detectar este sistema.

Para el entrenamiento de la red neuronal mediante el algoritmo *backpropagation* se debe elegir tanto la función de coste a utilizar, como el optimizador, además de otros parámetros como son el tamaño del *batch* o el número de épocas. Para ello se ha buscado información acerca de otros modelos de red neuronal que hayan funcionado bien en un problema de clasificación similar a este. Puesto que la aplicación de este tipo de técnicas en el mundo industrial no es muy frecuente se ha recurrido al campo de voz.

Finalmente se ha decidido utilizar como función de coste la *Categorial cossentropy* debido a que se tiene un problema multiclase. Por otro lado, se ha usado como optimizador el *SGD (Stochastic Gradient Descent)* con *learning rate* de valor 0.001, el cual se ha elegido de forma empírica tras comprobar que con este, el algoritmo convergía en un tiempo razonable. Además, se ha utilizado un Batch de tamaño 100, se ha activado la función *Dropout* al 50 %, la cual para cada época utiliza la mitad de las neuronas de cada capa, y la función *Early Stopping* que utiliza un conjunto de los propios datos de entrenamiento para evaluar el modelo en cada iteración, de manera que hace que la red pare de entrenar cuando el resultado de evaluar dichos datos converja [18].

Por otro lado, cabe destacar que las redes neuronales necesitan de una gran cantidad de datos de entrenamiento, por lo que se ha realizado un estudio para analizar como mejora el rendimiento de ellas a medida que aumentan dichos datos. La Figura 4.8 muestra la variación del *accuracy* obtenido a medida que se aumenta el número de señales utilizadas para el entrenamiento. Cabe destacar que la evaluación en este caso se ha realizado con 50 señales sintéticas. Puesto que se ha comprobado que a partir de 300 señales los datos no mejoran significativamente y la red se vuelve más lenta computacionalmente, se ha

decidido trabajar con 300 señales sintéticas, de aquí en adelante, para entrenar los distintos modelos.

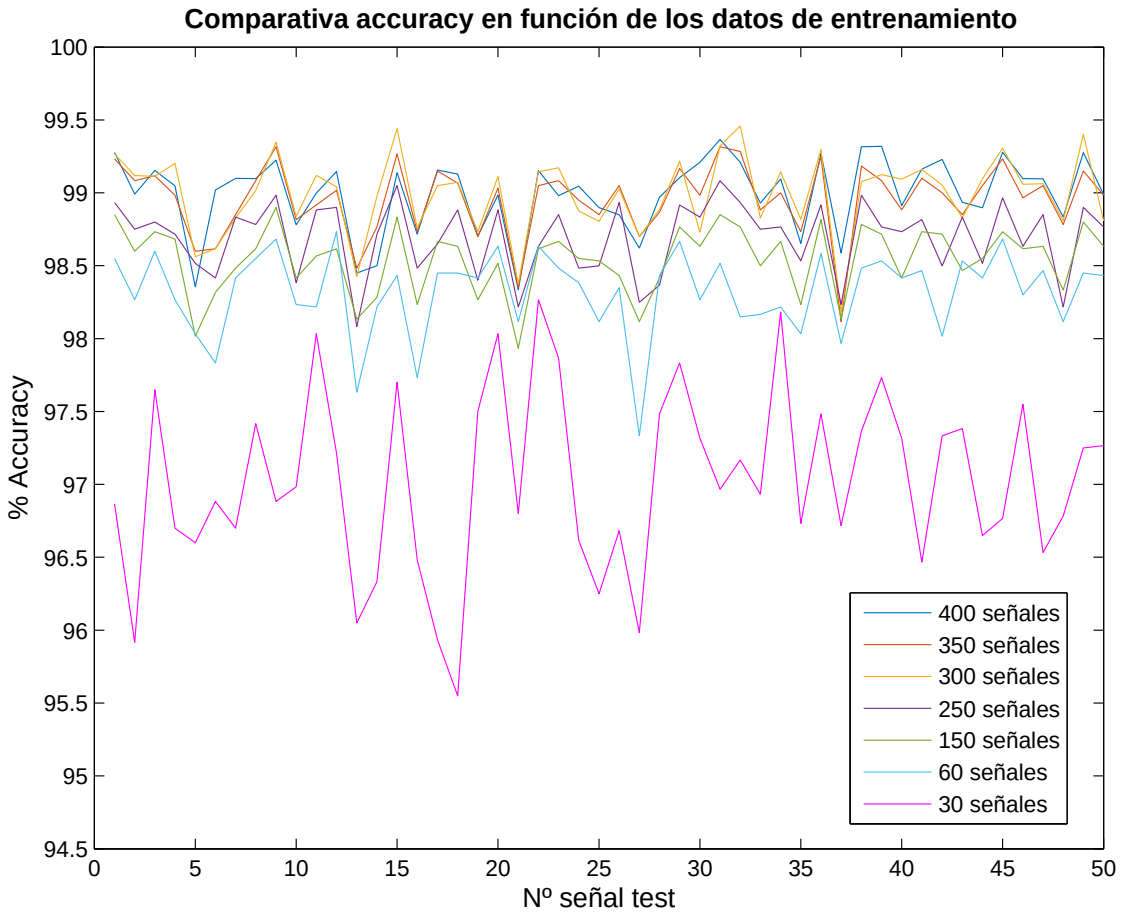

Figura 4.8: Evolución del porcentaje de acierto obtenido en función del número de señales utilizadas en el entrenamiento.

Una vez que se ha entrenado la red neuronal, se generará un modelo y se almacenará. Para la evaluación se cargará este modelo, al cual se le pasará como entrada las 30 características que representan a cada trama de la señal y éste devolverá una puntuación o *score* para cada una de las clases.

4.2.3. Decisión

La clasificación final se realiza tras un procesado de los *scores* que devuelve la red neuronal. El primer paso de este procesado, es realizar una primera decisión en función de qué clase ha obtenido mayor puntuación.

Esta primera decisión se realiza trama a trama, sin embargo es importante tener en cuenta el contexto en el que está situada cada una de ellas. Es decir, si hay un conjunto de tramas consecutivas detectadas, por ejemplo, como evento ruido, y aparece entre ellas una correspondiente a evento desviación, lo más lógico es que haya sido un error de la red y corresponda esa trama también a ruido.

Para tener en cuenta esta información se ha utilizado un *filtro de modas*. Este filtro va recorriendo las decisiones tomadas para cada trama, y compara si es la misma que la que se obtendría cogiendo la trama correspondiente como central y un contexto (N tramas anteriores y N posteriores) y realizando la moda. Si el resultado es distinto, se considera como un error de la red y se cambia. Se puede ver este esquema en la Figura 4.9. En ella se ve que se han detectado dos tramas correspondientes a evento ruido, una trama correspondiente a evento desviación y posteriormente dos tramas correspondientes a evento ruido, lo más lógico es que la trama central también se deba clasificar como evento ruido. Tras este filtrado se genera la clasificación final.

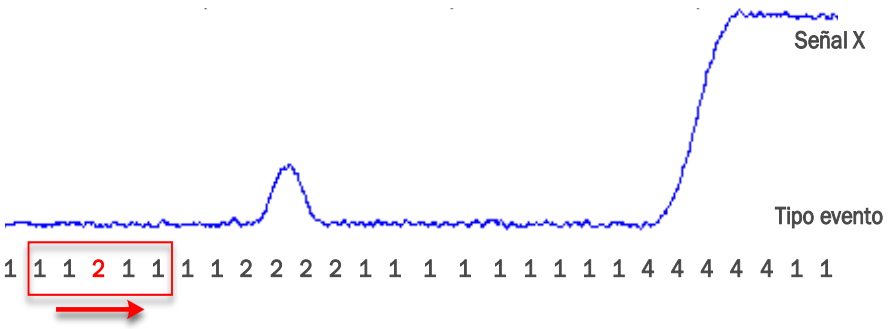

Figura 4.9: Filtrado de modas.

Para analizar la bondad de este sistema preliminar se ha realizado una evaluación de él con las 68 señales de test de las que se dispone en la Base de Datos. La Figura, 4.10 muestra la matriz de confusión de los resultados obtenidos. En ella podemos ver que tanto los eventos tipo ruido, desviación y transición obtienen un porcentaje de acierto bastante elevado. Sin embargo, los eventos tipo gaps se confunden, en su mayoría, con ruido y con desviaciones.

Al igual que se comprobaba en la Figura 4.7, estos resultados vuelven a demostrar la necesidad de usar la señal densidad además de la señal de material X para lograr el objetivo planteado. Se llega a la conclusión de que utilizando únicamente la señal de material X, este sistema no detecta los eventos gap. Sin embargo, mediante la utilización de la señal densidad, en la cual únicamente aparecen los eventos gap, este problema será solucionado.

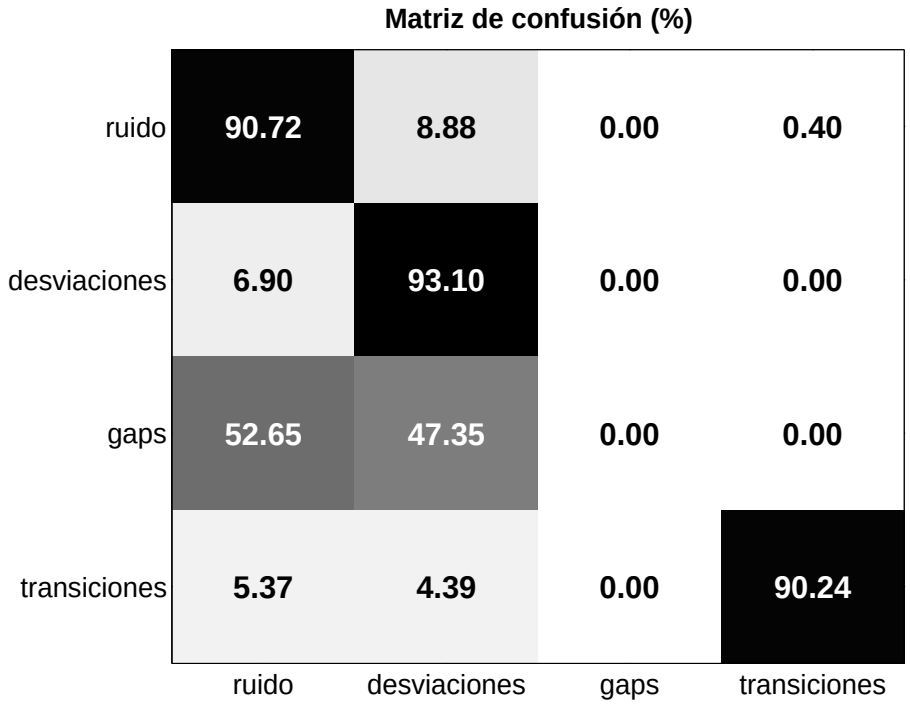

Figura 4.10: Matriz de confusión resultante de evaluar el sistema preliminar de clasificación de eventos con 68 señales de material X reales.

4.3. Subsistema de reducción de ruido para clasificación de eventos

La señal densidad presenta un fuerte ruido que hace que algunos de los eventos no queden visibles. Por este motivo, ha sido necesaria la implementación de un sistema de reducción de ruido, centrado en la clasificación de eventos. Esto abre una nueva vía de investigación en este Trabajo Fin de Máster, mediante la cual se implementará un sistema de reducción de ruido basado en redes neuronales y se compararán los resultados obtenidos con los que se obtendrían tras utilizar una técnica clásica de reducción de ruido como es el Filtrado de Wiener.

La Figura 4.11 muestra el diagrama de bloques del sistema de reducción de ruido. Se recibe como entrada, la señal de densidad, la cual pasa al bloque extractor de características. Dentro de este bloque la señal es filtrada y segmentada en distintas tramas. A continuación, se genera un conjunto de características que representan a la señal ruidosa y que son la salida de este bloque. El segundo bloque es una red neuronal que se entrenará para que aprenda qué conjunto de características limpias corresponde a cada uno de los conjuntos de características ruidosas que recibe como entrada. La salida de dicha red neuronal se pasará a un último bloque que se encarga de la reconstrucción.

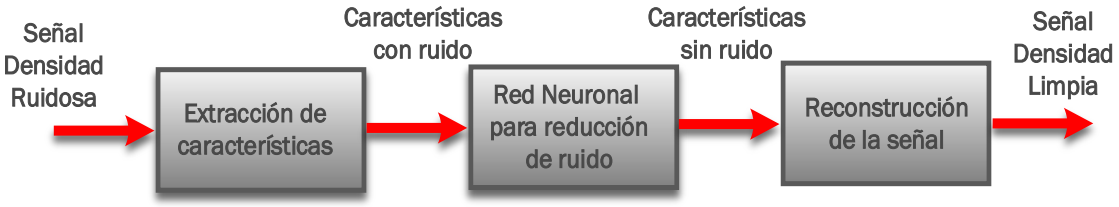

Figura 4.11: Diagrama de bloques del sistema de reducción de ruido.

4.3.1. Extracción de características

Previamente a la creación de un sistema, se debe diseñar y por lo tanto determinar, cuáles serán los parámetros de entrada. La etapa de diseño es esencial para que un proyecto sea efectivo y para que el sistema cumpla con las expectativas esperadas. Puesto que en este caso, la finalidad del sistema es la de limpiar una señal ruidosa, las características elegidas deben ser óptimas para esta finalidad. A lo largo del tiempo, han sido muchas las técnicas que han utilizado el espectro de la señal para reducción de ruido, por lo que en este sistema, también se usará.

Para la obtención de características se debe realizar un enventanado de la señal, mediante la utilización de ventanas rectangulares de N=300 muestras, las cuales estarán solapadas un 60 %. La elección de estos valores ha sido de forma empírica tras diversas pruebas, hasta encontrar aquellos que ofrecían mejores resultados.

Una vez que se tiene la señal segmentada en tramas, se calcula la DFT (*Discrete Fourier Transform*) a cada uno de intervalos enventanados. A continuación, se divide el espectro en módulo y en fase y se almacenan. Cabe descatar que aunque la entrada de la red neuronal consistirá en el módulo de la DFT, se debe almacenar la fase para su utilización en la reconstrucción final de la señal.

Por lo tanto, el sistema de reducción de ruido será tal que el bloque extractor de características generará un conjunto de características correspondientes al módulo del espectro de la señal densidad ruidosa. Éstas, serán la entrada de la red neuronal, la cual devolverá las características limpias de ruido. Finalmente en el último bloque se utilizarán las características correspondientes a la fase de la señal, que han sido previamente almacenadas, junto con la salida de la red neuronal, para la reconstrucción final de la señal.

Una vez que se ha decidido utilizar el módulo de la DFT, se va a determinar el número de características necesario. Como se han utilizado ventanas de longitud 300 para realizar el segmentado de la señal, y puesto que el módulo de la DFT es simétrico, las características que, en principio sería necesario pasar a la red son 151. Sin embargo, se ha realizado un estudio para comprobar si son todas absolutamente necesarias o, si por el contrario, hay un conjunto de ellas con las cuales se obtendría el rendimiendo deseado.

Para ello se han utilizado 10 señales sintéticas y sus correspondientes características en módulo y en fase. A continuación se ha ido reconstruyendo cada una de las señales, utilizando el conjunto completo de características en fase y variando el número de características en módulo. Es decir, la primera señal reconstruida se ha realizado con únicamente una característica correspondiente al módulo, mientras que la última se ha reconstruido con 151.

De esta manera se ha generado un conjunto de señales reconstruidas para las cuales se ha obtenido el MSE (*Mean Square Error*) existente con respecto a la señal original. En la figura 4.12 se ha representado dicho MSE, tras realizar esta comparativa con 10 señales sintéticas de distintas características. Como se puede ver, a partir de la utilización de 15 características, el error permanece prácticamente constante. Es por esto que se ha reducido de 151 a 15 características utilizadas, ya que esta reducción supondrá una mayor eficiencia, pues el modelo de red neuronal generado tendrá un número menor de parámetros, por lo que tanto el entrenamiento como la evaluación tendrá un menor coste computacional.

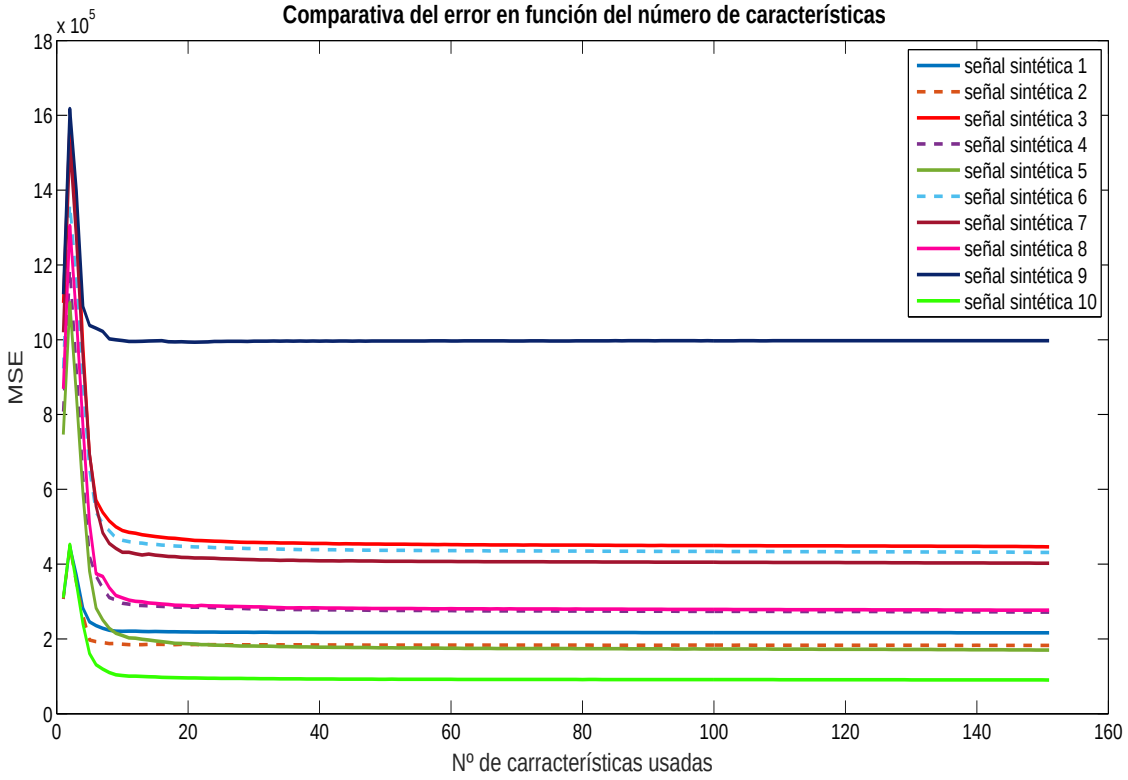

Figura 4.12: Representación del error cuadrático medio (MSE) entre 10 señales sintéticas y las correspondientes 151 señales reconstruidas, obtenidas mediante la utilización de un número diferente de características seleccionadas, poniendo las restantes a 0.

4.3.2. Creación y entrenamiento de la red neuronal

El segundo bloque del sistema de reducción de ruido consiste en una red neuronal, cuya finalidad es, dado un conjunto de características ruidosas que representan a una señal, devolver las características correspondientes limpias de ruido. A continuación de detalla el grafo utilizado así como se describen los parámetros de la red neuronal. Cabe destacar que nunca se había utilizado redes neuronales para reducción de ruido en este tipo de señales industriales por lo que, como se ha dicho anteriormente, se ha tomado como base el sistema de [5].

Puesto que se ha decidido representar cada trama de la señal con un conjunto de 15 características, el modelo de red neuronal debe tener 15 neuronas en la capa de entrada y 15 neuronas en la capa de salida. En cuanto a las capas ocultas, [5] indica el uso de tres capas ocultas con 2048 neuronas cada una de ellas. Sin embargo, su capa de entrada consta de 257 neuronas por lo que esto lleva a la conclusión de que es posible que para 15 neuronas en la capa de entrada no sea necesario un grafo tan grande. Por este motivo se ha realizado un estudio empírico, en el cual se han entrenado distintos modelos cambiando el número de capas ocultas y el número de neuronas en cada una de ellas. Este estudio se ve representado en la Figura 4.13.

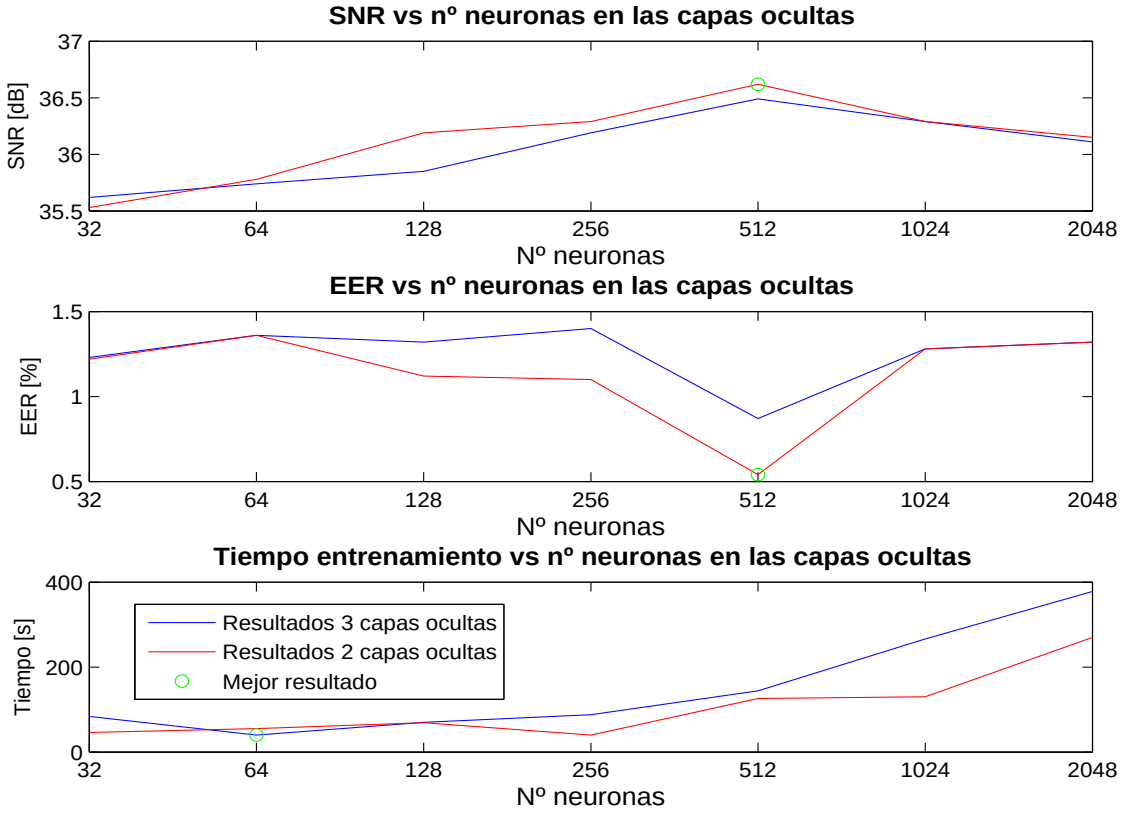

Figura 4.13: Evolución de los resultados obtenidos (SNR, EER y tiempo de procesamiento) tras evaluar distintos modelos de red neuronal para reducción de ruido que varían en cuanto al número de capas ocultas y el número de neuronas en cada capa.

En particular se han creado y entrenado 14 modelos de red neuronal distintos. Siete de ellos constan de 2 capas ocultas y varían en el número de neuronas en cada capa que va desde 32 a 2048 en potencias de 2. Los siete restantes, constan de tres capas ocultas y varían de igual modo el número de neuronas. Cada uno de los modelos ha sido entrenado con 300 señales sintéticas y evaluado con las 68 señales reales de la Base de Datos. Tras la evaluación, se ha calculado la SNR de cada una de las señales limpias y se ha realizado la media para cada uno de los 14 modelos. Además, se ha obtenido el tiempo de computación que ha tardado el algoritmo y se ha calculado el EER. Como se puede ver en la Figura 4.13 los mejores resultados se obtienen para el grafo que consta de 2 capas ocultas y 512 neuronas en cada capa, por lo que esta será la arquitectura finalmente utilizada.

El siguiente paso consiste en decidir las funciones de activación. En este caso, se ha utilizado la función de tipo ReLu para la capa de entrada y para las ocultas y la función Linear para la capa de salida. Esta última ha sido elegida pues no se quiere obtener puntuaciones o *scores*, sino que se desea que la red devuelva las propias características modificadas para que no presenten ruido. También por este motivo, se ha decidido utilizar la función coste *MSE* para el algoritmo *backpropagation* del entrenamiento de la red, pues tras recibir un conjunto de características de entrada y un conjunto de características deseado, se ajustará para que la salida sea lo más parecida al segundo conjunto.

Otros parámetros a destacar en el entrenamiento de la red neuronal es el optimizador, el *batch* y el número de épocas. En cuanto al optimizador se ha decidido utilizar el *Adam* debido a que se comenzó utilizando el SGD, puesto que en el modelo preliminar de detección de eventos había dado buenos resultado. Sin embargo en este modelo producía el *exploding gradient problem*, el cual el optimizador *Adam* solucionaba. Por otro lado, el tamaño del *batch* elegido ha sido 256 ya que se ha comprobado que en otras aplicaciones similares ha dado buenos resultados [5]. Finalmente se han activado las funciones *Dropout* al 50 % y *Early Stopping*, para que el algoritmo pare automáticamente cuando el coste obtenido haya convergido.

Una vez que se ha entrenado la red neuronal, se generará un modelo y se almacenará. Para la evaluación, como ya se ha comentado, se cargará este modelo, se le pasará como entrada las 15 características ruidosas que representan a cada trama de la señal y éste devolverá las 15 características limpias.

4.3.3. Reconstrucción de la señal

La salida de la Red Neuronal corresponde con los 15 primeros valores de la DFT de la señal limpia de ruido. Para reconstruir la señal original se deben poner los restantes coeficientes a 0 y aplicar la simetría. A continuación se utilizan los valores correspondientes a la fase de la DFT y se realiza la DFT inversa para cada conjunto de características, lo que devolverá cada trama. Una vez que se obtienen todas las tramas de la señal se reconstruirá la señal, teniendo en cuenta el enventanado que se realizó y el solapamiento utilizado. Finalmente se le aplica un filtrado paso bajo a la señal para reducir el efecto provocado por la fase ruidosa de la señal.

4.4. Combinación multicanal

El sistema final implementado consiste en la combinación de dos subsistemas, uno de ellos encargado de realizar reducción de ruido a la señal densidad y otro que tras recibir una señal multicanal, se encarga de realizar la detección y clasificación de los eventos que aparecen en ella. En la Figura 4.14 se representa el diagrama de bloques de este sistema definitivo [1].

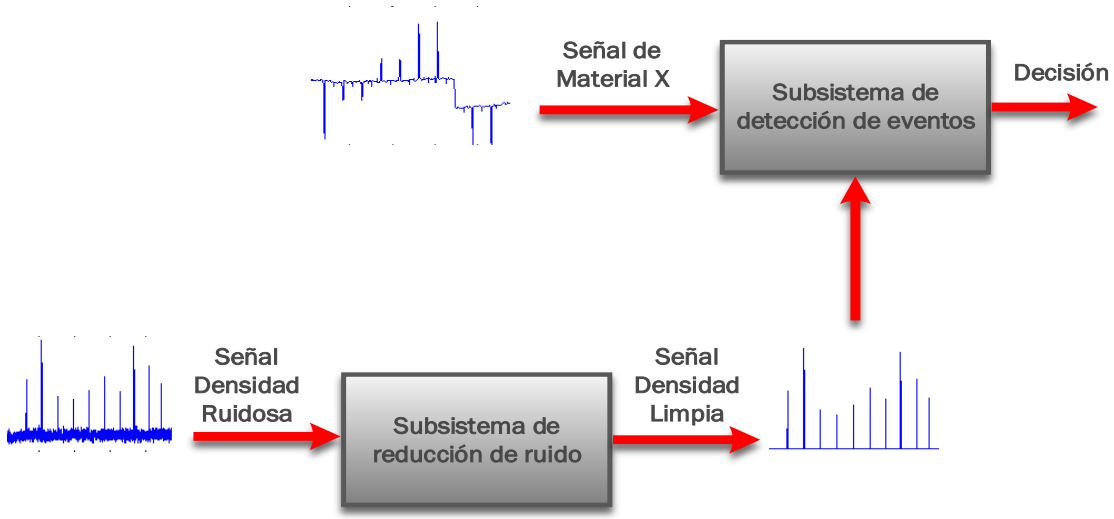

Figura 4.14: Diagrama de bloques del sistema de detección y clasificación de eventos.

En el apartado 4.2, se mostró el subsistema de detección y clasificación de eventos preliminar. Sin embargo, este subsistema sufre conversiones al realizar la combinación multicanal, es decir, al utilizar las señales de material X y las señales de densidad. En este apartado se detallarán los cambios que se ha realizado con respecto al subsistema preliminar, se indicará la finalidad de cada uno de los bloques de los que está formado, así como se describirán sus características más significativas. El diagrama de bloques de este sistema se representa en la Figura 4.15.

Figura 4.15: Diagrama de bloques del sistema de reducción de ruido.

[1] Por motivos de conveniencia se repite la figura

4.4.1. Extracción de características

La combinación multicanal debe utilizar tanto las características correspondientes a la señal de material X como las características de la señal densidad. La primera decisión tomada ha sido la de la utilización del mismo procedimiento de extracción de características de la señal de material X para la señal de densidad. Este procedimiento fue descrito en el apartado 4.2.1.

El bloque extractor de características realizará un filtrado a la señal de material X, pero no a la señal densidad pues ya viene limpia de ruido del subsistema anterior. A continuación se realizará una segmentación en tramas para cada una se las señales y la obtención de un conjunto de 30 características para cada trama y para cada señal. Este conjunto consta de 15 características correspondientes a los primeros coeficientes del módulo de la FFT y los coeficientes delta. Por lo tanto, esto supone que la salida del extractor de características será un conjunto de 60 características, 30 de ellas representan a la señal densidad limpia y las otras 30 restantes representan a la señal de material X. La Figura 4.16 demuestra que esta decisión tomada es correcta pues el conjunto de características utilizado representará de forma óptima a las señales con las que se trabaja.

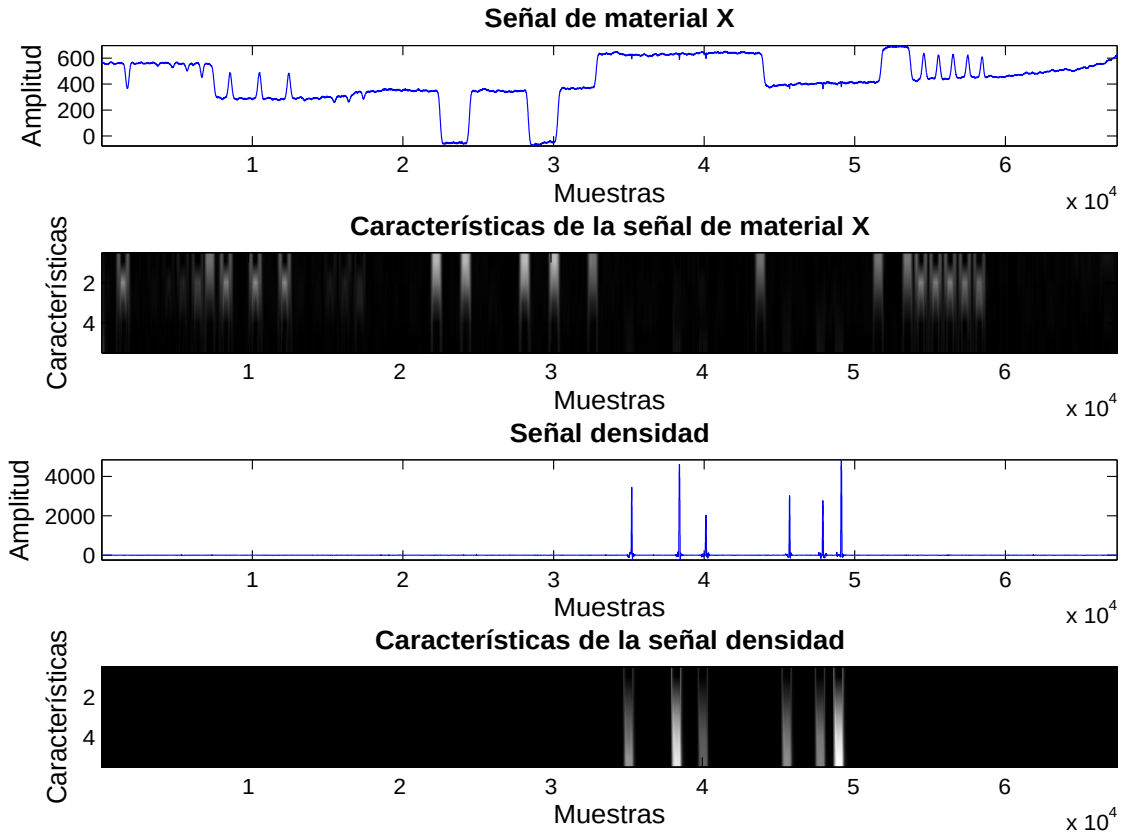

Figura 4.16: Representación de una señal de material X y una señal densidad, utilizadas en el sistema de detección de eventos, así como de las características que las representan.

4.4.2. Creación y entrenamiento de la red neuronal

Una vez que se ha comprobado que el conjunto de características utilizado es el óptimo, se debe crear el grafo de la red neuronal. Puesto que se tienen 60 características, la red neuronal constará de 60 neuronas en la capa de entrada. En cuanto a la capa de salida, como la finalidad de esta red será la de detección y clasificación de eventos, se tendrán cuatro neuronas, cada una en representación a un tipo de evento.

La elección del número de capas ocultas, así como del número de neuronas en cada capa, se realizará de forma empírica, igual que para los subsistemas anteriores. Este procedimiento ha llevado al entrenamiento de un total de 14 modelos de red neuronal y su posterior evaluación. Cabe destacar que el entrenamiento se ha realizado con 300 señales sintéticas y la evaluación con las 68 señales reales.

La Figura 4.17 muestra los resultados obtenidos para cada uno de los modelos generados. En particular, se ha representado el porcentaje de acierto para cada uno de los cuatro eventos de los que se dispone, así como el tiempo de procesamiento que ha tardado cada uno de los modelos. Siendo el acierto el porcentaje de tramas clasificadas correctamente.

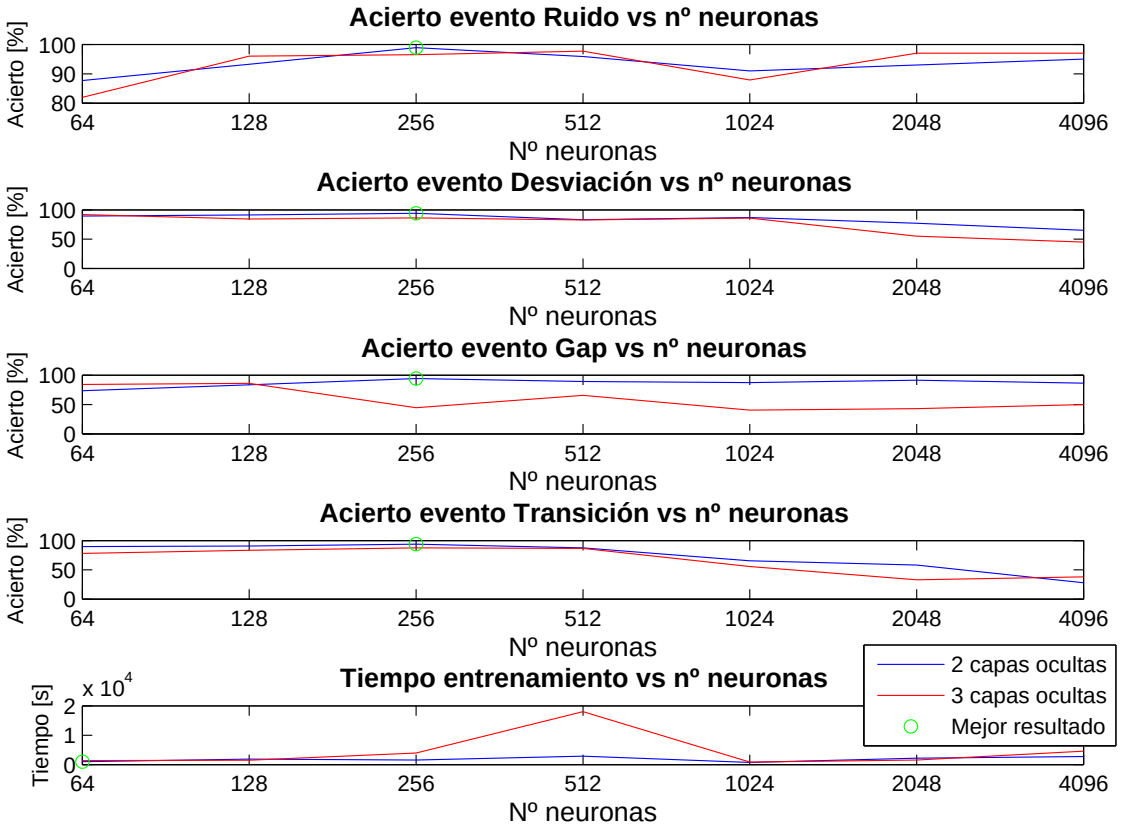

Figura 4.17: Evolución del porcentaje de acierto obtenido para cada evento tras evaluar distintos modelos de red neuronal que varían en cuanto al número de capas ocultas y el número de neuronas en cada capa.

Cabe destacar, por los mismos motivos que en el sistema preliminar de detección de eventos, se han tomado una serie de decisiones. La función de activación escogida para la cada de entrada y las dos capas ocultas ha sido la ReLu. La capa de salida tiene función de activación Softmax. Se usará la función coste *Categorial cossentropy* y como optimizador el *SGD (Stochastic Gradient Descent)*. Además el Batch tendrá tamaño 100.

Los 14 modelos se han entrenado con la función *Early Stopping* activa, la cual hace que el algoritmo pare cuando el resultado de evaluar la función de coste converja. Viendo los resultados, cabe destacar que a medida que aumenta el número de capas ocultas y el número de neuronas en cada capa, el porcentaje de acierto para cada clase empeora. Esto puede significar que el algoritmo no llega a converger y la red no llega a aprender lo que debería. Finalmente, se ha decidido utilizar un grafo con 2 capas ocultas y con 256 neuronas en cada capa. Esta arquitectura, además de proporcionar un porcentaje de acierto alto para cada clase del problema, tiene un tiempo de computación es bajo, lo cual hace que el modelo sea eficiente.

Finalmente, cabe mencionar que se activará la función *Dropout* al 50 %. Esta función es muy útil para evitar el *overfitting* o sobreentrenamiento, ya que en cada iteración del algoritmo *backpropagation* del entrenamiento, únicamente activa la mitad de las neuronas. Además, también se ha comprobado que el sistema final no está sobreentrenado puesto que se entrena con datos sintéticos y se evalúa con datos reales.

4.4.3. Decisión

El bloque decisor será el mismo que el que se utilizó en el subsistema preliminar de detección de eventos (apartado 4.2.3).

En este bloque, se realiza la clasificación final tras un procesado de los *scores* que devuelve la red neuronal. Este procesado incluye una primera decisión, trama a trama, en la que se decide el evento correspondiente como aquel que tiene un mayor *score*. Posteriormente se aplica un filtrado de modas para tener en cuenta el contexto en el que está situada cada trama.

En este punto, se tendría una clasificación de a qué tipo de evento corresponde cada una de las tramas en las que se ha dividido la señal. Sin embargo, para tener una mayor visibilidad se realiza finalmente una transformación muestra a muestra, de modo que el bloque decisor devuelve una señal con el mismo número de muestras que la señal de entrada. Esta señal toma valores del 1 al 4 en función del tipo de evento al que clasifica cada una de las muestras, correspondiendo el valor 1 al evento ruido, el valor 2 al evento desviación, el valor 3 al evento gap y el valor 4 al evento transición.

Capítulo 5

Resultados

En este capítulo se mostrarán los resultados obtenidos al aplicar el sistema de detección y clasificación de eventos implementado sobre las señales reales de la Base de Datos. Estos resultados se evaluarán con las medidas de rendimiento descritas en la sección 3.4. Además, se compararán, con los que se obtendrían con otros algoritmos del Estado del Arte, de manera que se pueda ver si hay una mejora significativa.

5.1. Subsistema de reducción de ruido

El rendimiento del sistema de detección y clasificación de eventos, depende en gran medida del subsistema de reducción de ruido. Cabe recordar que el sistema final utiliza una señal multicanal compuesta por una señal densidad y una señal de material X. La señal densidad posee un ruido indeseado que hace incapaz la distinción de aquellos eventos de menor amplitud, por lo que se propuso implementar un subsistema de reducción de ruido que facilitara esta tarea. Por lo tanto, el objetivo de este subsistema no es tanto la reducción de ruido, sino hacer que el mayor número de eventos quede visible, de ahí que esté enfocado a la clasificación de eventos.

En el Estado del Arte, se realizó un estudio de la evolución de las técnicas de reducción de ruido existentes para el campo de voz, ya que en el industrial no hay mucha investigación sobre ello. En este estudio se mencionó tanto técnicas clásicas como son el Filtrado de Wiener, como las técnicas más novedosas y más utilizadas en los últimos tiempos como son las Redes Neuronales. Esta última ha sido la utilizada en el subsistema de reducción de ruido implementado, debido a que el Filtrado de Wiener, no es capaz de proporcionar los resultados deseados.

Por lo tanto, en esta sección se demostrará por qué el uso de las Redes Neuronales está en auge y cuales son las ventajas que ofrece frente a las técnicas clásicas. Para ello se mostraran los resultados obtenidos al realizar reducción de ruido a señales reales con el subsistema implementado y con el Filtro de Wiener de Pascal Scalart et al. [6]. En particular se aplicará la reducción de ruido a las 68 señales de densidad presentes en la

Base de Datos. Estas señales muestran un pico de amplitud positiva en aquellos puntos en los que exista evento de tipo gap.

Cabe repasar el funcionamiento de las técnicas utilizadas. El subsistema de reducción de ruido comienza segmentando la señal en tramas. A continuación representa cada trama con un conjunto de características ruidosas, las cuales al pasar por la red neuronal, que ha sido previamente entrenada, quedarán limpias de ruido y por lo tanto se utilizarán para realizar la reconstrucción de la señal. Por otro lado, el Filtro de Wiener recibe tanto la señal densidad como un modelo de ruido que utilizará para realizar el filtrado, su funcionamiento se describió en la sección 2.5. El modelo de ruido utilizado no es más que la unión de las 50 señales de ruido que se seleccionaron en la sección 3.2.

Mediante ambas técnicas se obtendrá un conjunto de 68 señales limpias de ruido. Para comenzar con el análisis de los resultados, se representa en la Figura 5.1 una comparativa entre una de las señales originales y la salida obtenida tras el filtrado con la red neuronal y con Wiener. Además se muestra la posición de los eventos que están presentes en la señal real, para poder compararla con los eventos que aparecen en las señales filtradas.

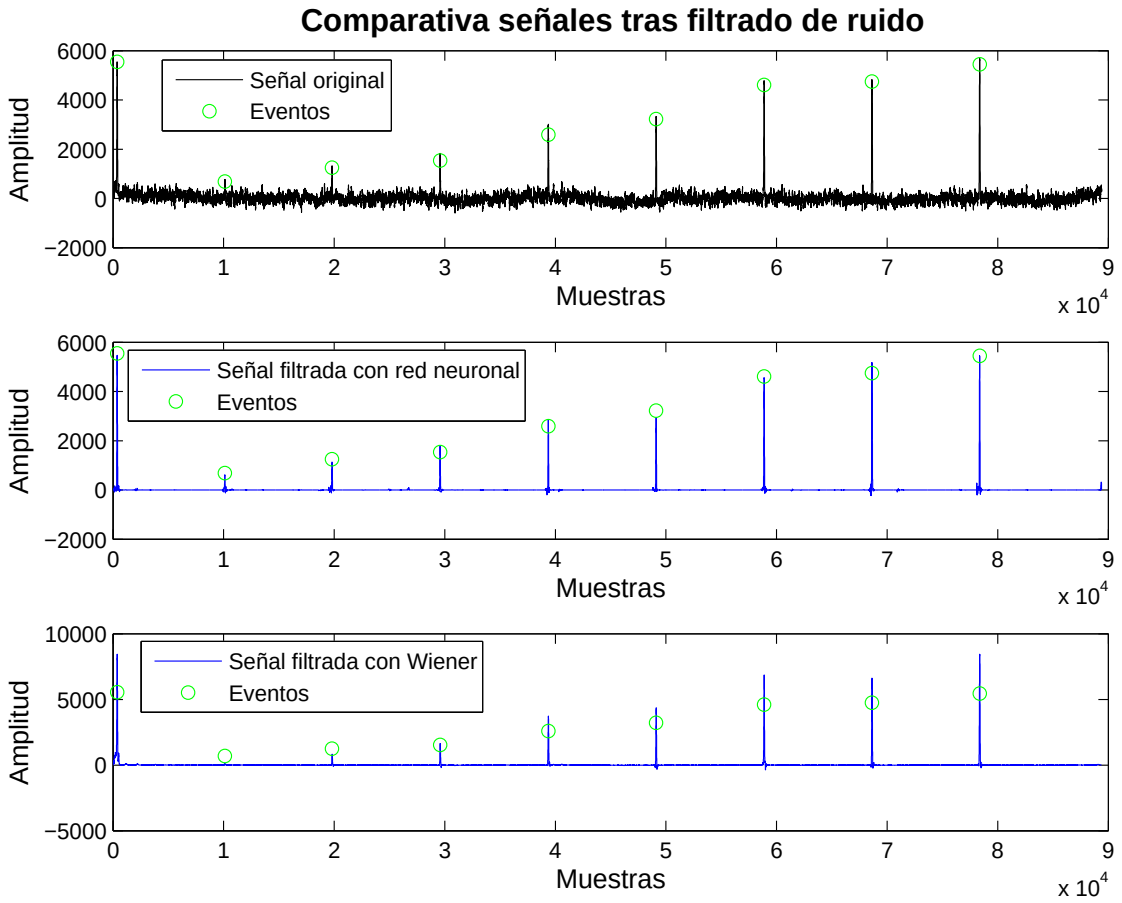

Figura 5.1: Comparativa entre una señal densidad real, así como la posición de sus eventos, y las correspondientes señales tras el filtrado con la red neuronal y con Wiener.

El primer resultado que se aprecia es la impactante reducción de ruido que logra hacer ambas técnicas. Esto significa que los modelos de ruido utilizados para el filtrado de Wiener y para entrenar la red neuronal han sido óptimos. Además, las características utilizadas en el subsistema de reducción de ruido han sido adecuadas, de manera que la red ha conseguido aprender a eliminar el ruido y devolver una señal de valor en torno a 0 en su lugar. Cabe destacar que el entrenamiento de la red neuronal se ha realizado con señales sintéticas limpias y ruidosas cuya adición de ruido es la que utilizaba estos modelos.

Debido a que la evaluación se ha realizado con señales reales, no se dispone de la señal limpia original para comparar los resultados obtenidos. Por ello se han utilizado algunas medidas de rendimiento, como es el caso de la SNR, para analizar los resultados de la limpieza de ruido a todo el conjunto de test. La Figura 5.2 muestra una comparativa entre la relación señal a ruido (SNR) de las señales originales ruidosas, de las señales filtradas con Wiener y de las señales filtradas con la red neuronal. Se puede observar como la gran limpieza de ruido visible en la Figura 5.1 se ve también reflejada en estas gráficas, pues de obtener un valor medio de 24.38dB para las señal originales, se consigue llegar a valores de 36.01dB para las señales filtradas con Wiener y a 36.62dB para las señales cuya reducción de ruido se ha realizado con la red neuronal.

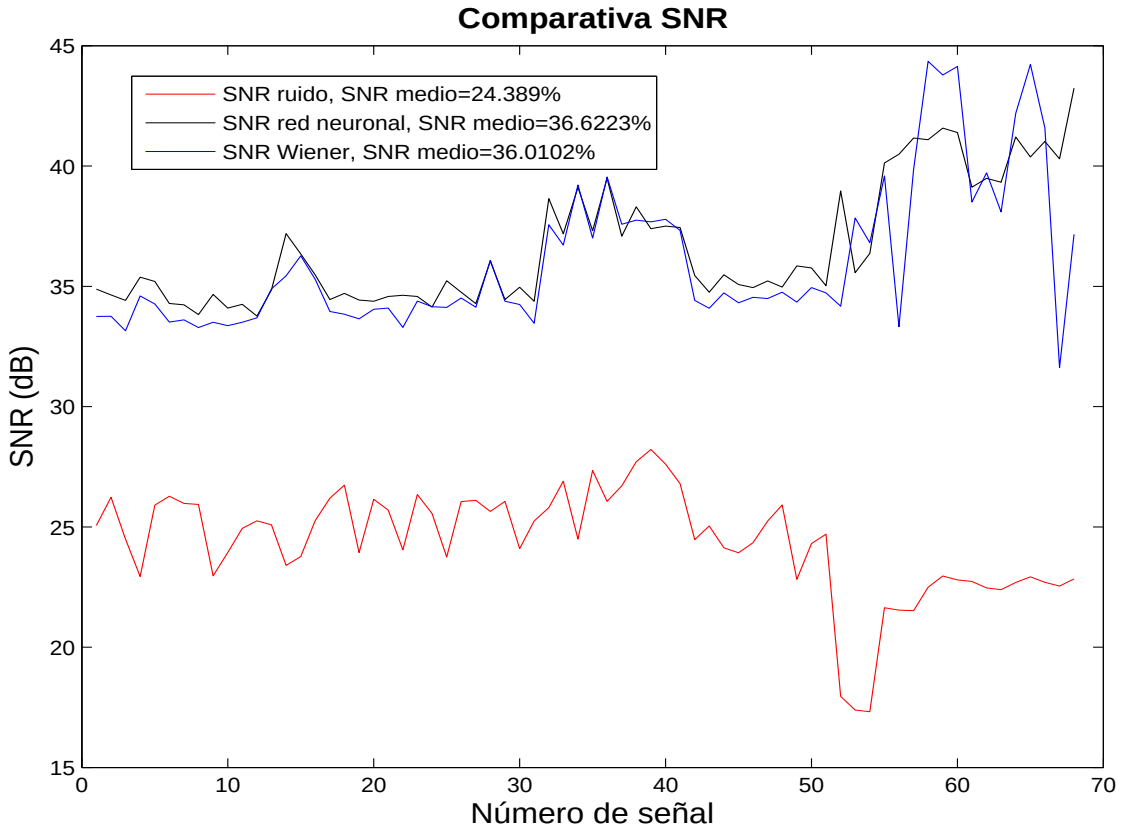

Figura 5.2: Comparativa de SNR para distintas señales reales y las correspondientes señales filtradas con Wiener y con la red neuronal.

Sin embargo, como ya se ha comentado, este subsistema está orientado a la detección de eventos, por lo que se pretende detectar el mayor número de ellos. La comparativa de la SNR realizada anteriormente, no da información sobre ello, es decir no es capaz de determinar qué técnica proporciona mejores resultados, por lo que se ha propuesto la utilización de otra medida de rendimiento, como son las Curvas DET. Estas curvas representan la tasa de falso rechazo frente a la tasa de falsa aceptación, es decir representa la relación existente entre los eventos no detectados siendo reales, y los eventos detectados como reales cuando en realidad no existen.

Volviendo a la Figura 5.1 se puede comprobar que en cuanto a la detección de eventos, la red neuronal proporciona mejores resultados pues es capaz de detectar hasta los eventos más pequeños, como es el caso del primero que es casi indistinguible en la señal ruidosa, pero que está completamente visible en la señal filtrada con la red. Sin embargo, la señal filtrada con Wiener no es capaz de detectar este evento, habiéndolo confundido con ruido. De ahí al gran auge en el que están las redes neuronales ya que son algoritmos inteligentes que son capaces de ver más allá, de lo que veían las técnicas clásicas.

Para comprobar si realmente se obtiene mejores resultados con la red neuronal se han obtenido las Curvas DET para el conjunto de test original, para el conjunto filtrado con Wiener y para el obtenido con redes neuronales. Estas curvas se han representado en la Figura 5.3. El EER es el parámetro que marca qué técnica es la que mejor funciona, en este caso demuestra que la red neuronal es la mejor pues es la que tiene un EER más bajo. Además, es lógico que la limpieza con Wiener es la que peor resultados proporciona, pues como se ha visto que se están perdiendo eventos.

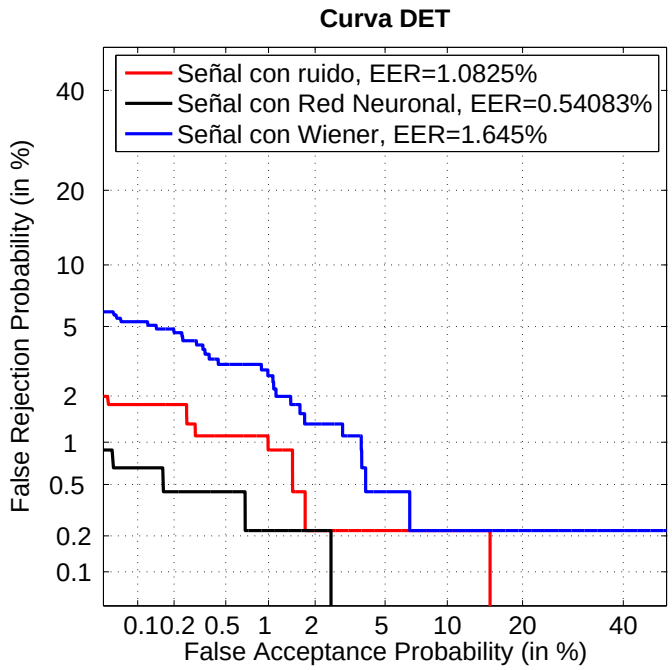

Figura 5.3: Comparativa de curvas DET para el conjunto de señales reales y señales filtradas con Wiener y con redes neuronales.

Una vez que se ha demostrado que la utilización de redes neuronales, con finalidad de reducción de ruido, proporciona mejores resultados que las técnicas clásicas, se va a comprobar si realmente funcionarán dentro del sistema completo de detección de eventos. Cabe destacar que la señal densidad limpia de ruido será la utilizada en el sistema completo para ayudar a la detección de los eventos gaps, los cuales no son completamente diferenciables en la señal de material X. En particular, el sistema de detección final, trabaja con unas características que fueron analizadas en la sección 4.4.1. Por lo tanto, finalmente se quiere comprobar si realmente las características obtenidas con la señal limpia obtenida con la red neuronal aportan más información que las obtenidas con la señal ruidosa o con la señal limpia con Wiener.

La Figura 5.4 muestra una comparativa de las características extraídas de las tres señales con las que se está tratando en esta sección y que se representaron en la Figura 5.1. Se puede ver que estas características reflejan lo que se ha ido comentando a lo largo de la sección, es decir, demuestran que la mejor limpieza de ruido se obtiene con el subsistema implementado, pues evidencian la presencia de 9 eventos en la señal.

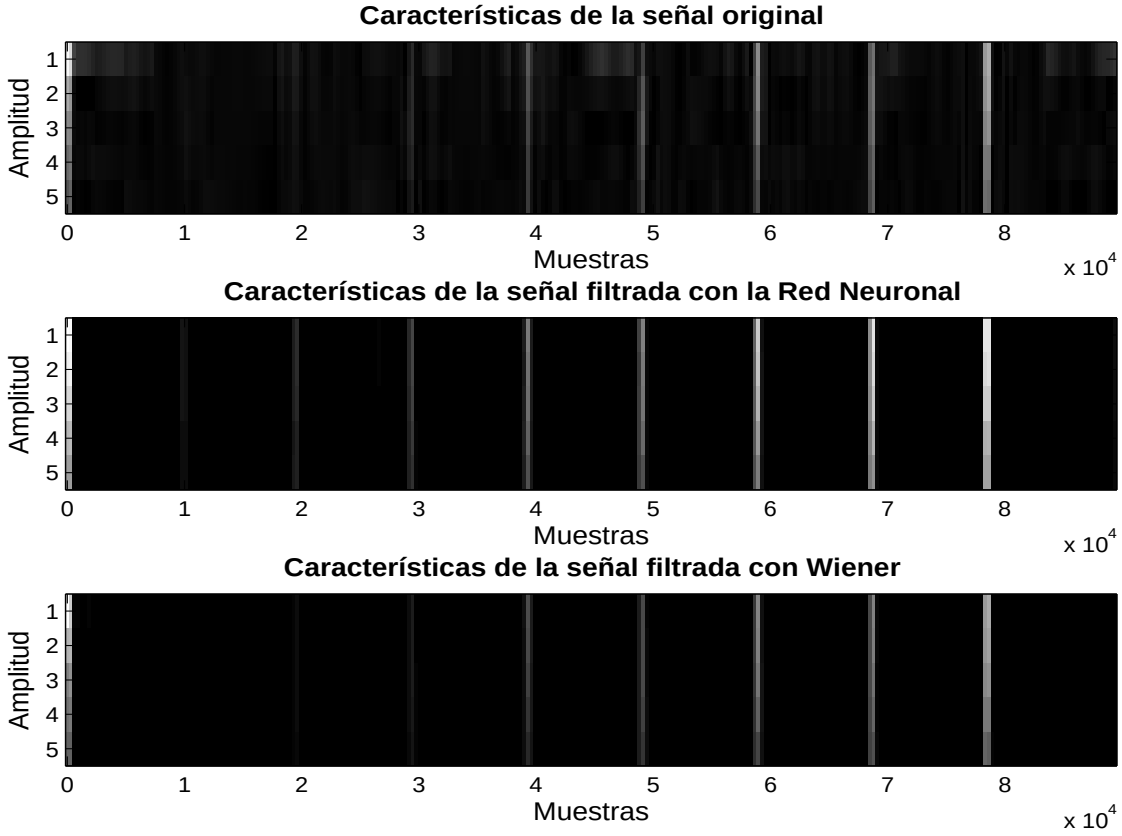

Figura 5.4: Comparativa de las características de las señales ruidosa, limpias con Wiener y limpias con redes neuronales.

5.2. Sistema de clasificación de eventos

En esta sección va a evaluar el rendimiento del sistema final de detección y clasificación de eventos y se va a comparar con el mejor sistema disponible hasta ahora, denominado Sistema Línea Base, que estaba basado en una arquitectura *front-back-end* [19] [20]. Para ello, se van a utilizar las 68 señales multicanal reales de la Base de Datos. En particular, se utilizarán las 68 señales de material X y las 68 señales de densidad resultantes tras aplicarles la correspondiente reducción de ruido. Estas señales están etiquetadas por lo que se conoce el Ground Truth, necesario para la evaluación del rendimiento.

Cabe recordar que se dispone de señales que presentan un máximo de 4 tipos de eventos distintos: ruido, desviaciones, gaps y transiciones. El sistema se encarga de segmentar la señal y de extraer un conjunto de características representativas de cada trama, las cuales serán la entrada de una red neuronal que ha sido previamente entrenada para realizar clasificación de la señal en esos 4 tipos de eventos. Finalmente la salida de la red neuronal pasa a un bloque decisor que devuelve una señal de la misma longitud que la de entrada, donde cada muestra obtiene un valor del 1 al cuatro en función del tipo de evento en el que se ha clasificado.

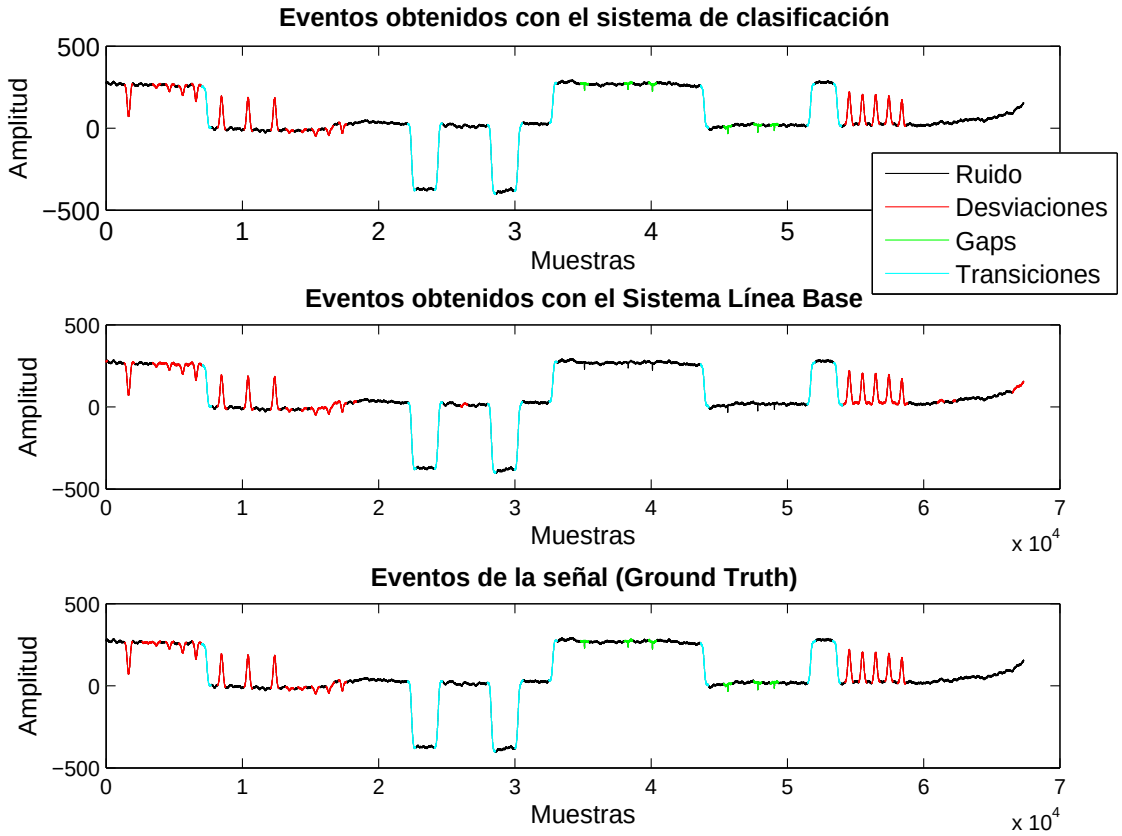

Figura 5.5: Comparativa entre la clasificación obtenida con el sistema implementado, con el Sistema Línea Base y el Ground Truth. Se muestra la señal de material X ya que en ella se ven reflejados todos los eventos.

Para comenzar se va a mostrar la salida del sistema de clasificación de eventos para dos señales de la Base de Datos, las cuales pertenecen a dos inspecciones de dos piezas industriales distintas. Por este motivo, estas señales presentan diferentes características, una de ellas contiene los 4 tipos de eventos y la otra contiene únicamente evento tipo ruido y evento tipo gap. Además, se va a comparar con la salida del sistema de clasificación basado en una arquitectura *Front-end/Back-end*. La representación gráfica de esta comparativa se puede ver en las Figuras 5.5 y 5.6.

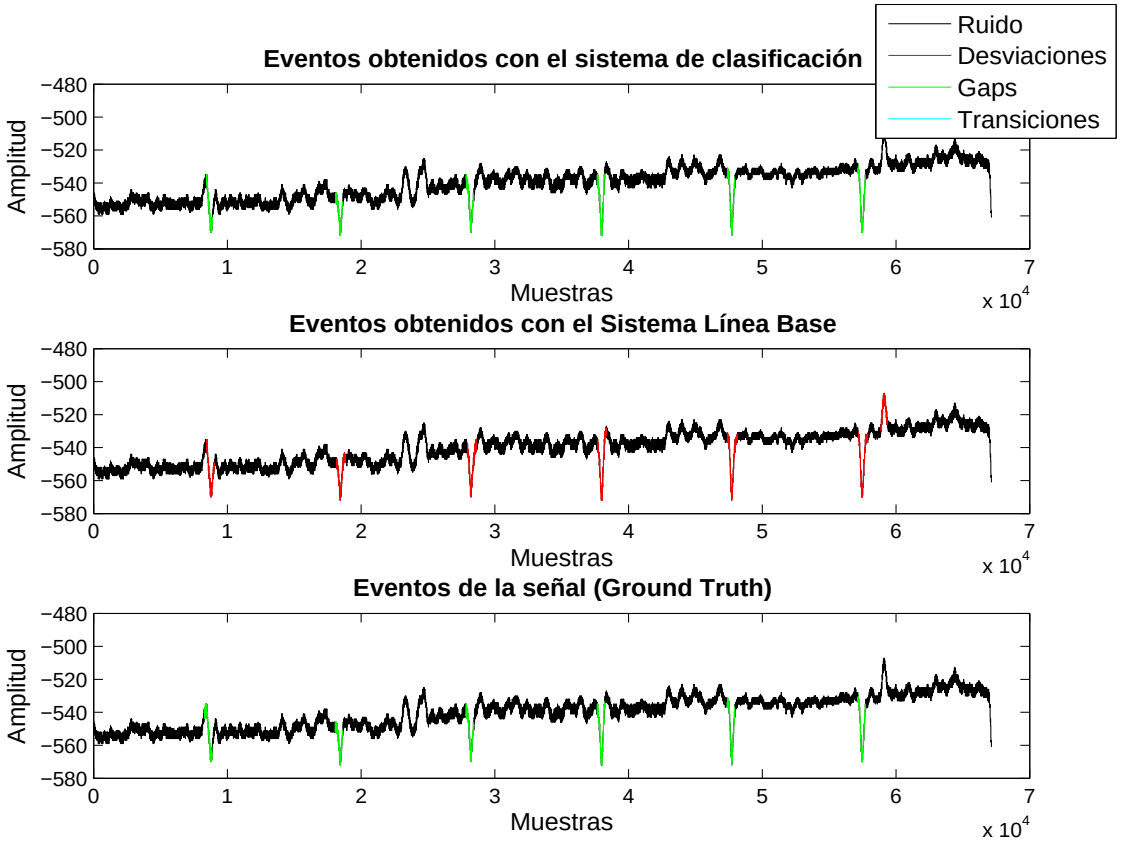

Figura 5.6: Comparativa entre la clasificación obtenida con el sistema implementado, con el Sistema Línea Base y el Ground Truth. Se muestra la señal de material X ya que en ella se ven reflejados todos los eventos, aunque en esta señal sólo aparecen eventos de tipo gap.

Analizando la Figura 5.5 se puede comprobar que el sistema que utiliza redes neuronales es capaz de clasificar todos los eventos que aparecen en la señal, sin embargo, el Sistema Línea Base no es capaz de detectar los eventos tipo gaps, confundiéndolos con ruido. Este comportamiento también es visible en la Figura 5.6, la cual muestra que el sistema usado como base confunde los eventos tipo gaps con desviaciones. Cabe destacar que la elección de estas dos señales se ha hecho en base a poder mostrar la mayor variedad de resultados distintos.

Por lo tanto, se llega a la conclusión de que, en función de la naturaleza y del tamaño del evento tipo gap, el Sistema Línea Base suele confundirlo o con evento tipo ruido o con evento tipo desviación, mientras que la red neuronal es capaz de detectar que corresponde a evento gap. Este problema, que también estaba presente en el sistema preliminar de detección de eventos, es debido a que el sistema basado en la arquitectura *front-back-end*, utiliza únicamente la señal de material X y es justo por este motivo por el que en este trabajo se ha decantado por la utilización de señales multicanal, combinando la señal de material X y la señal densidad, y de redes neuronales.

Para comprobar si estas conclusiones extraídas son correctas, así como ver el rendimiento del sistema para la clasificación de cada uno de los tipos de eventos, se van a utilizar distintas medidas de rendimiento como son las Curvas DET, la matriz de confusión y el *accuracy*.

En primer lugar, se van a analizar los resultados que reflejan las Curvas DET. Las Figuras 5.7 y 5.8 muestran las curvas obtenidas para los cuatro tipos de eventos, tanto en la clasificación con el sistema que usa redes neuronales, como con el sistema basado en una arquitectura *front-back-end*. Observando ambas figuras se puede comprobar que el sistema implementado en este Trabajo Fin de Máster muestra unos resultados bastante mejores que los de sistemas anteriores. Además, llama la atención la curva DET de la Figura 5.8 la cual muestra que el porcentaje de eventos de tipo gap que nunca se clasifican correctamente en el sistema usado como línea base, es muy elevado.

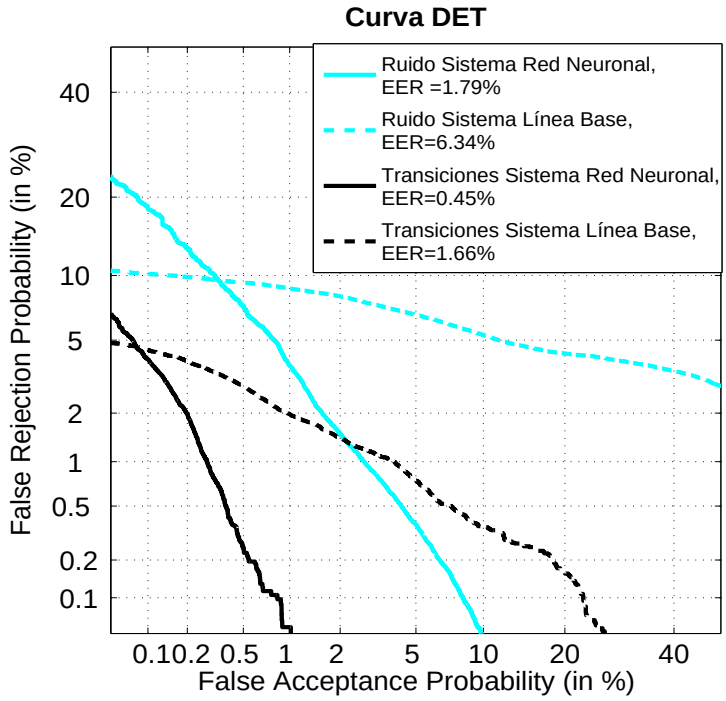

Figura 5.7: Representación de las Curvas DET para los eventos Ruido y Transiciones, cuando se aplica el sistema de detección de eventos basado en redes neuronales y el Sistema Línea Base.

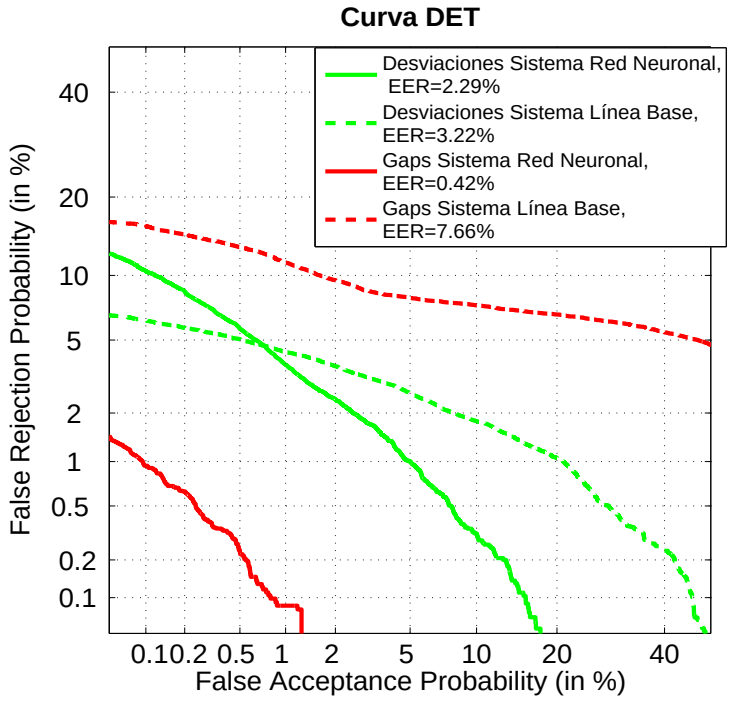

Figura 5.8: Representación de las Curvas DET para los eventos Desviaciones y Gaps, cuando se aplica el sistema de detección de eventos basado en redes neuronales y el Sistema Línea Base.

Cabe destacar que la Curva DET, que muestra el rendimiento del sistema implementado en este trabajo, se ha obtenido con los scores de salida de la red neuronal, es decir, sin pasar por el bloque Decisor. Por este motivo, los resultados no son del todo fiables, pues el bloque Decisor realiza un procesado a dichos scores, con el cual obtiene la clasificación definitiva.

Para comprobar la mejora del sistema mediante la adición de este bloque Decisor, así como para comprobar cuál es el rendimiento definitivo, se va a utilizar una nueva medida de rendimiento, basada en la clasificación final, que se denomina matriz de confusión. Esta matriz, se obtiene comparando el Ground Truth con el vector de salida del bloque Decisor, de modo que calcula el porcentaje de eventos que se ha clasificado correctamente, así como el porcentaje de confusión de un evento con otro. Las Figuras 5.9 y 5.10 muestran respectivamente, a las matrices de confusión correspondientes al sistema que usa redes neuronales y el Sistema Línea Base.

A simple vista se puede comprobar que los resultados del sistema implementado en este trabajo son mucho mejores, pues se consigue detectar un alto porcentaje de todos los eventos, incluidos los gaps, los cuales no son detectados en el otro sistema. Además, se puede comprobar que el bloque Decisor mejora el rendimiento final, pues en la Curva DET se ve que los eventos de tipo ruido y de tipo desviación tienen un mayor EER que los otros eventos, sin embargo en la matriz de confusión, el porcentaje de acierto mostrado es muy alto. Esto es debido a la existencia de desviaciones muy pequeñas que, cuando

el ruido es más pronunciado, suelen confundirse con él. El procesado de los scores que realiza el bloque Decisor, en particular el filtrado de modas, soluciona este problema.

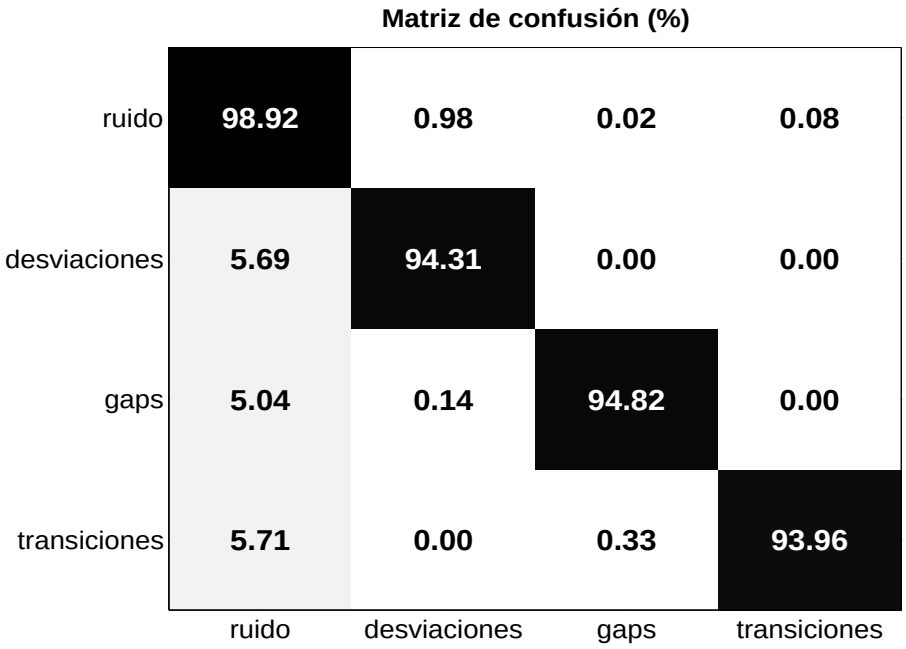

Figura 5.9: Matriz de confusión obtenida tras evaluar el sistema de clasificación de eventos desarrollado con 68 señales reales.

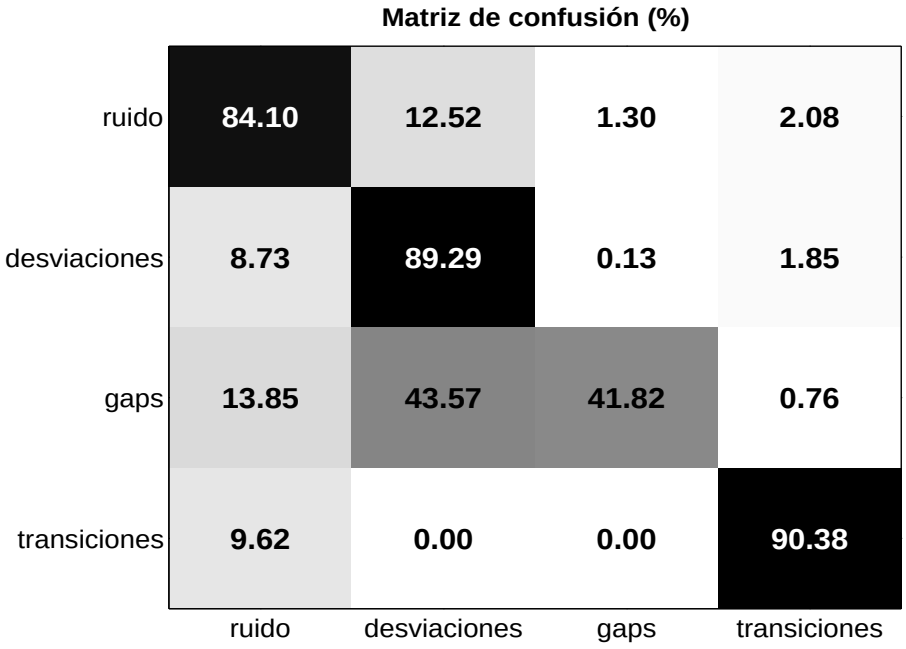

Figura 5.10: Matriz de confusión obtenida tras evaluar el Sistema Línea Base [19] [20] con 68 señales reales.

Para finalizar, se va a utilizar una última medida de rendimiento denominada *accuracy* la cual muestra el porcentaje de acierto para cada una de las señales. Esta medida compara muestra a muestra si la clasificación se ha realizado de forma correcta y calcula el porcentaje de acierto final. Los resultados obtenidos se muestran en la Figura 5.11, en la cual se pueden ver tres gráficas distintas. La gráfica roja corresponde al *accuracy* calculado para cada una de las clasificaciones (de las 68 señales) con el sistema que usa redes neuronales. La gráfica verde es el *accuracy* que se obtiene cuando se clasifica con el sistema *Front-end/Back-end*. Finalmente, la gráfica azul muestra la línea base, que como se comentó en el capítulo de Entorno Experimental, representa al porcentaje de acierto si se clasificaran todas las muestras de la señal como el evento que más veces aparece, en este caso el ruido.

Esta figura muestra de forma obvia que el sistema que mejor funciona es el que utiliza redes neuronales. Sin embargo, aunque se puede ver que la clasificación no es del 100 %, esto no significa que no se hayan detectado correctamente todos los eventos. Este pequeño porcentaje que falta para llegar al 100 % corresponde a las muestras de las intersecciones entre eventos, debido a que en esas zonas es en las que suele haber una mayor confusión.

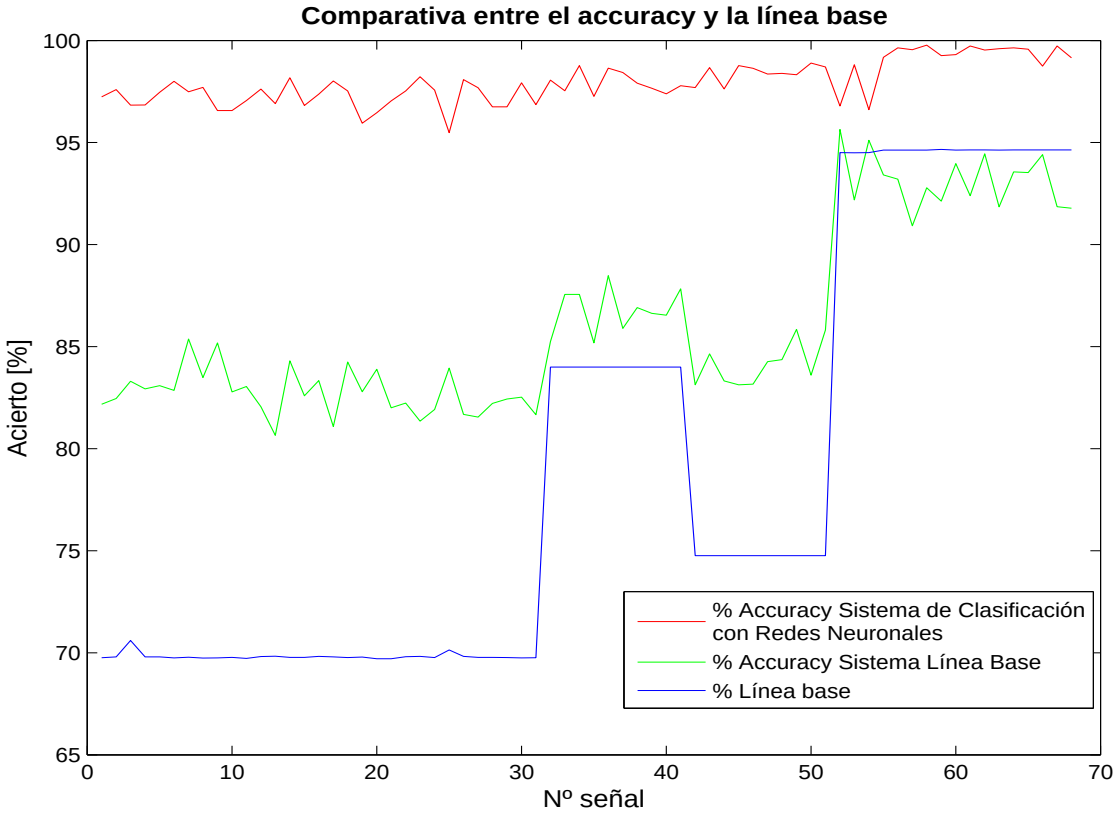

Figura 5.11: Comparativa entre el *accuracy* obtenido con el sistema de clasificación que usa redes neuronales, el obtenido con el sistema basado en una arquitectura *front-back-end* y la línea base.

Capítulo 6

Conclusiones y trabajo futuro

En este capítulo se va a realizar una valoración global del trabajo que se ha desarrollado. En primer lugar se analizará si se han cumplido los objetivos que fueron marcados antes de empezar el estudio, a continuación se detallará cuál ha sido el aprendizaje académico conseguido y finalmente se propondrán algunas vías de desarrollo futuras.

6.1. Conclusiones

El principal objetivo, como ya se ha dicho en varias ocasiones a lo largo de la memoria, ha sido la implementación de un sistema de detección y clasificación de eventos. En trabajos anteriores [19] [20] se implementó un sistema basado en un arquitectura *Front-end/Back-end* que se caracterizó por no conseguir distinguir dos tipos de eventos: gaps y desviaciones negativas. Por lo tanto, el sistema implementado debía mejorar estos resultados.

Para ello se ha comenzado tomando dos decisiones. La primera de ellas consiste en la utilización de señales multicanal, es decir, utilizar dos tipos de señales distintas. Esta elección se ha tomado en base a que las señales que utilizan los sistemas anteriores, señales de material X, no distinguen entre estos eventos, sin embargo existe otra señal llamada señal densidad, que es capaz de mostrar los eventos tipo gaps. De esta manera, combinando ambas señales, se podría conseguir una correcta diferenciación. La segunda decisión ha sido la de utilizar técnicas de procesado avanzado de señal para lograr los objetivos, ya que hasta el momento se habían utilizado técnicas clásicas, en particular se han utilizado las redes neuronales.

Sin embargo, a lo largo del trabajo se han encontrado una serie de obstáculos, los cuales se han ido solucionando. El primer inconveniente ha sido la escasez de señales en la Base de Datos. Los algoritmos basados en redes neuronales necesitan una gran cantidad de datos para el entrenamiento, sin embargo la Base de Datos de la que se dispone, contiene 181 señales, de las cuales únicamente 68 presentan eventos. Por este motivo, se ha tenido la necesidad de implementar un sistema generador de señales sintéticas multicanal,

incluyendo la señal de material X y la señal densidad, y que fueran lo más parecidas posible a las señales reales. Con estas señales sintéticas se entrenarían los modelos de redes neuronales y con la señales reales se realizaría la evaluación.

El segundo inconveniente ha sido la presencia de un gran ruido en las señales de densidad, necesarias para la correcta detección de los eventos tipo gap, el cual hacía que algunos de los eventos no fueran visibles. Esto se solucionó generando un subsistema de reducción de ruido, el cual es parte del sistema final de detección y clasificación de eventos. Este sistema de reducción de ruido también utiliza una red neuronal, la cual ha sido entrenada con señales densidad sintéticas.

Con todo esto, se ha conseguido implementar un sistema de limpieza de ruido que junto a un sistema de clasificación de eventos, han mejorado la precisión en la detección de los eventos más difíciles, como son los gaps del material y por lo tanto se ha conseguido mejores resultados en la tarea de identificación de irregularidades en las piezas industriales. Esto se ha podido demostrar gracias a las medidas de rendimiento realizadas, tanto al sistema implementado en este trabajo, como a otros sistemas previamente implementados en Audias, y que han servido como "Línea Base".

Se quiere también mencionar la gran importancia que ha tenido, para la generación de este sistema, el estudio teórico realizado sobre las redes neuronales. Ha permitido conocer su arquitectura, cómo funcionan y cuales son sus parámetros más característicos. Sin embargo, no hay ninguna regla que defina qué arquitectura debe tener la red neuronal para conseguir un buen funcionamiento, sino que los estudios indican que debe ser de manera empírica. Por este motivo, se ha profundizado en cómo varían los resultados obtenidos en función del número de capas ocultas, así como de neuronas en cada capa, de cara a seleccionar aquel modelo mejor.

Otro aspecto a destacar consiste en que se ha logrado transferir técnicas que normalmente se utilizan para otro tipo de datos, especialmente en señales de voz, al campo industrial. De esta manera se consigue comprobar que determinadas técnicas de tratamiento de señal, son capaces de solucionar los problemas planteados. Por otro lado, destacar que el sistema de detección de eventos se ha implementado de la forma más genérica posible, de modo que puede ser empleado en el futuro para otro tipo de señales.

Para finalizar, resaltar que se ha logrado cumplir los objetivos marcados y que este Trabajo Fin de Máster ha supuesto una ampliación de los conocimientos adquiridos en el máster y un gran aprendizaje académico que engloba cómo enfrentarse e investigar, cuando se tiene que realizar un trabajo con técnicas desconocidas. Al mismo tiempo ha sido muy interesante formar parte de un proyecto en colaboración con una empresa, lo cual es relevante a nivel profesional.

6.2. Trabajo futuro

Como posibles vías de trabajo futuro se tienen los siguientes puntos:

- Aumentar la base de datos de señales reales, de modo que se pueda confirmar el correcto funcionamiento del sistema de detección y clasificación de eventos en un entorno más robusto.
- Perfeccionar el sistema generador de señales sintéticas, mediante la adición de características de un conjunto mayor de señales reales.
- Aplicación de otras técnicas de procesado avanzado de señal, como puede ser el uso de las Redes Neuronales Bayesianas. Estas redes se basan en dar una relación de probabilidad entre la entrada y la salida de la red. Además, son muy útiles para resolver problemas en dominios donde hay escasez de datos de entrenamiento, como una forma de prevenir el sobreajuste.

Referencias bibliográficas

[1] CHEN, Jingdong, et al. Fundamentals of Noise Reduction. En *Springer Handbook of Speech Processing* Springer Berlin Heidelberg, 2008. p. 843-872.

[2] FAH, Liew Ban; HUSSAIN, Aini; SAMAD, Salina Abdul. Speech enhancement by noise cancellation using neural network. En TENCON 2000. *Proceedings* IEEE, 2000. p. 39-42.

[3] MAAS, Andrew L., et al. Recurrent neural networks for noise reduction in robust ASR. En *Thirteenth Annual Conference of the International Speech Communication Association*, 2012.

[4] VYAS, Pankaj Kumar; RAWAT, Paresh; KHATRI, Swati. Back propagation based adaptive noise chancellor. *International Journal of Computer Technology and Electronics Engineering (IJCTEE)*, 2011, vol. 1, p. 14-16.

[5] XU, Yong, et al. A regression approach to speech enhancement based on deep neural networks. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 2015, vol. 23, no 1, p. 7-19.

[6] SCALART, Pascal, et al. Speech enhancement based on a priori signal to noise estimation. En *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on.* IEEE, 1996. p. 629-632.

[7] FLÓREZ, Raquel; FERNÁNDEZ, José M. Las Redes Neuronales Artificiales. Fundamentos teóricos y aplicaciones prácticas.

[8] RUMELHART, David E.; HINTON, Geoffrey E.; WILLIAMS, Ronald J. Learning internal representations by error propagation. California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[9] MATICH, Damián J. Redes Neuronales: Conceptos básicos y aplicaciones. *Cátedra de Informática Aplicada a la Ingeniería de Procesos–Orientación I*, 2001.

[10] BISHOP, Christopher M. Pattern Recognition and Machine Learning. Springer, 2006.

[11] JANOCHA, Katarzyna; CZARNECKI, Wojciech Marian. On Loss Functions for Deep Neural Networks in Classification. *arXiv preprint arXiv:1702.05659.*, 2017.

[12] BISHOP, Christopher M. Neural networks for pattern recognition. Oxford university press, 1995.

[13] KURBIEL, Thomas; KHALEGHIAN, Shahrzad. Training of Deep Neural Networks based on Distance Measures using RMSProp. *arXiv preprint arXiv:1708.01911*, 2017.

[14] KINGMA, Diederik P.; BA, Jimmy. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[15] GLOROT, Xavier; BENGIO, Yoshua. Understanding the difficulty of training deep feedforward neural networks. En *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010. p. 249-256.

[16] LECUN, Yann, et al. Efficient backprop. En *Neural networks: Tricks of the trade.* Springer, Berlin, Heidelberg, 1998. p. 9-50.

[17] ABD EL-FATTAH, M. A., et al. Speech enhancement using an adaptive wiener filtering approach. *Progress in Electromagnetics Research,* 2008, vol. 4, p. 167-184.

[18] GENCOGLU, Oguzhan; VIRTANEN, Tuomas; HUTTUNEN, Heikki. Recognition of acoustic events using deep neural networks. En *Signal Processing Conference (EUSIPCO), 2014 Proceedings of the 22nd European.* IEEE, 2014. p. 506-510.

[19] IGLESIAS, Álvaro. Extracción de características para señales temporales procedentes de sensores industriales. *Universidad Autónoma de Madrid*, 2017.

[20] LABRADOR, Beltrán. Clasificación de eventos en señales temporales procedentes de señales industriales. *Universidad Autónoma de Madrid,* 2017.