Trabajo de Fin de Máster
"Máster Universitario en Microelectrónica:
Diseño y Aplicaciones de Sistemas Micro/Nanométricos"

APLICACIÓN DE SISTEMAS NEUROMÓRFICOS CON APRENDIZAJE PROFUNDO PARA SISTEMAS DE COMUNICACIÓN BASADOS EN RADIO COGNITIVA

Autor:
Virginia Zúñiga González

Tutores:
José Manuel de la Rosa Utrera
Luis A. Camuñas Mesa

Sevilla, 4 de septiembre de 2020

Agradecimientos

Después de un intenso período de trabajo y un amplio aprendizaje, no solo en el campo de la microelectrónica sino también a nivel personal, este proyecto me da paso a finalizar el máster el cual me abre las puertas para continuar mi recorrido profesional e investigador. El esfuerzo que todo ello ha supuesto, queda justificado por la superación personal, académica y por el apoyo que he recibido de amigos, familiares y profesores.

A mi principal apoyo, mi madre, quien siempre me ha apoyado en todas mis decisiones y ayudado en todo lo posible. Gracias, por educarme tal como lo has hecho y sobre todo de ser mi gran referente. Al fin y al cabo, yo también estoy orgullosa de todo el esfuerzo que has invertido para que, tanto mi hermana como yo, tuviéramos una buena educación. Además has estado siempre, en los momentos de agobio aportándome consejos, comprensión y compañía y en las buenas noticias siendo siempre motivo de celebración.

A tí, papá, por toda tu ayuda y esfuerzo para facilitarme estos años de estudio los cuales gracias a ti han sido posible. Me has mostrado estar orgulloso de mí por cada aprobado y por cada meta superada, siendo esto de gran entusiasmo para mí.

A mi hermana, ella me hace más ameno los ratos libres con risas, paseos y momentos de relax. Sé que siempre será un gran apoyo.

A todos mis compañeros del despacho PA-34 o, mejor dicho, amigos que han hecho que mi experiencia en el IMSE sea una de las mejores que haya podido tener. Ellos han hecho posible que además de crecer académicamente, haya adquirido más confianza en mí misma y me lleve tantos y tantos buenos momentos y que de seguro, habrá muchos más.

Por último, agradecer a mis tutores José Manuel de la Rosa y Luis Camuñas Mesa quienes a parte de ser buenos profesores son unas personas grandiosas las cuales admiro por sus trayectorias en el mundo de la investigación. Siempre consideraré haber tenido suerte de poder contar con ellos ya que, de lo contrario, no habría aprendido de forma exuberante en tan solo un año.

Resumen

Una de las principales limitaciones para la implantación práctica de dispositivos IoT (*Internet Of Things*) se encuentra en la banda del espectro electromagnético empleado para comunicaciones, el cual ha de ser compartido entre un número creciente de dispositivos que procesan una gran cantidad de información. La tecnología denominada Radio Cognitiva (RC) permite hacer un uso más eficiente del espectro electromagnético, modificando de forma dinámica los parámetros de transmisión y recepción de los dispositivos en función de la información sensada del entorno en términos de interferencias, ocupación de la banda, nivel de la batería, etc.

Este trabajo explora el concepto de un sistema de radio cognitiva con perspectiva a poder ser implementable con técnicas de digitalización altamente programables y adaptativas usando un *Software Defined Radio* (SDR), junto con un procesamiento de la información asistido mediante técnicas de aprendizaje automático basados en procesamiento neuromórfico. Es decir, esta Inteligencia Artificial (IA) va a enfocarse en una de las tecnologías clave de hoy en día conocida como aprendizaje profundo o en inglés *Deep Learning* (DP) que, a su vez, presenta diferentes estructuras lógicas o redes neuronales que se asemejan a la organización del sistema nervioso de los mamíferos. En concreto se usa la red *Long Short-Term Memory* (LSTM) para la predicción en series temporales del nivel de ocupación para distintos anchos de banda del espectro electromagnético.

El concepto del sistema radio cognitivo es implementado mediante una red LSTM con la que se obtiene una predicción futura de las ondas electromagnéticas y que haciendo uso de esta información, se podrá decidir que ancho de banda es el más adecuado modificando los parámetros necesarios de un filtro analógico/RF paso de banda.

Palabras clave: *Internet Of Things*, radio cognitiva, espectro electromagnético, ocupación de la banda, *Software Defined Radio*, Inteligencia Artificial, aprendizaje profundo, *Long Short-Term Memory*, predicción futura, filtro analógico/RF paso de banda.

Índice general

I Prolegómeno 1

1. Introducción 3

1.1. Motivación . . . 3

1.2. Objetivos y Alcance . . . 4

1.3. Estado del arte . . . 5

1.4. Estructura de la memoria y material adicional . . . 10

II Marco teórico 13

2. Inteligencia artificial en sistemas de Radio Cognitiva 15

2.1. Radio Cognitiva. ¿Qué es y a dónde quiere llegar? . . . 15

2.2. Evaluación y medidas del espectro de ocupación . . . 16

2.2.1. Cálculo de la ocupación espectral . . . 16

2.2.2. Parámetros a considerar en la medición del espectro . . . 18

2.2.3. Dispositivos para la medida del espectro de ocupación . . . 20

2.3. Inteligencia artificial para sistemas de comunicación inalámbrica . . . 20

2.3.1. El problema de aprendizaje para la predicción del espectro electromagnético 22

2.3.2. Predicción de series temporales . . . 24

2.3.3. Redes Neuronales para la predicción de series temporales. . . . 25

3. LSTM para la predicción de series temporales 31

3.1. Arquitectura de las redes LSTM . . . 31

III Desarrollo, simulaciones y conclusiones 37

4. Predicción de series temporales con redes LSTM usando MATLAB® 39

4.1. Arquitectura de la red LSTM en MATLAB® . . . 39

4.2. Entrenamiento de la red LSTM en MATLAB® 41
4.3. Implementación de la red LSTM enfocada en el caso de estudio 42

5. Sistema de selección de banda para un Filtro *Bandpass* mediante redes neuronales LSTM 49

5.1. Planteamiento del sistema de la selección de banda para un Filtro *Bandpass* gobernado por la predicción dada por una red neuronal LSTM 50
5.2. Señal de entrada. Datos de entrenamiento y de test 52
5.3. Entrenamiento de la red LSTM 54
5.4. Bucle de predicción de la banda ocupacional para los distintos canales de comunicación 54
5.5. Resultados obtenidos 56

6. Conclusiones y Mejoras 63

6.1. Conclusiones 63
6.2. Mejoras y trabajos futuros 63

Índice de figuras

1.1. Evolución de la capacitación nacional para el desarrollo de radios militares definidas por software. 5

1.2. Evolución hacia las redes móviles 5G como una clave para habilitar dichos dispositivos a todas las aplicaciones industriales. 6

1.3. Ventajas y desventajas de los detectores MIMO propuestos hasta el momento[1]. . 7

1.4. Reconstrucción por distorsión no-lineal por autointerferencia basadas en una red neural para sistemas *full-duplex* [1]. 9

2.1. Esquema conceptual de un sistema radio cognitiva 16

2.2. Representación de la división en frecuencia del espectro electromagnético mediante asignación por canales. 17

2.3. Definición del rango de frecuencias para múltiples canales adyacentes en un ancho de banda completo a estudiar. Se muestra en (a) un correcto ajuste y separación entre canales mientras que en (b) la anchura de banda de medición es demasiado amplia [2]. 19

2.4. Sobrecarga del receptor durante la medición de la señal RF por un alto nivel de ocupación en el canal 4 [2]. 19

2.5. Esquema representativo de un sistema inteligente como una caja negra que recibe información del entorno y actúa frente a esta [3]. 21

2.6. Esquema conceptual del aprendizaje automático como una subcategoría de la inteligencia artificial. A su vez, el aprendizaje profundo se considera una forma de aprendizaje automático. 22

2.7. Redes neuronales convolucionales para el reconocimiento de objetos en imágenes. . 26

2.8. Imágenes o *frames* correlacionados en una secuencia de vídeo. 27

2.9. Las RNN operando en el tiempo. 27

2.10. RNN *many-to-many* con capa oculta y totalmente recurrente. 28

2.11. Arquitectura realimentada de las RNN con dos entradas y dos salidas. 28

2.12. Arquitectura de las neuronas RNN estándar. 29

3.1. Diagrama de una cadena de celdas LSTM repetidas. El estado de la celda como solución a las largas dependencias. 32

3.2. Estructuras de las tres puertas que incluye la arquitectura de una celda LSTM. . . 32

3.3. Diagrama de “capa de la puerta del olvido” de una celda LSTM. 33

3.4. Diagrama de la “capa de la puerta de entrada” y etapa de actualización de una celda LSTM. 34

3.5. Diagrama de la puerta de salida de una celda LSTM. 35

3.6. Arquitectura completa de una celda LSTM. 35

4.1. Diagrama de bloques simple de la arquitectura LSTM para un problema de series temporales por regresión. 40

4.2. Esquema de entradas y salidas de la capa LSTM desenrollada en el tiempo. 40

4.3. Ventana emergente que aparece durante el entrenamiento de la red neuronal donde se aprecia la disminución del error RMSE y la pérdida *loss*. 42

4.4. Estructura de la red neuronal implementada en Matlab® indicando una sola unidad o neurona de entrada y salida. 43

4.5. Gráficas de los pesos y sesgos aprendidos de la red LSTM. Se comprueba que estos son valores fijos por lo que no cambian con las predicciones. 44

4.6. Gráficas del estado de la celda y estado oculto para cada unidad oculta en dos predicciones diferentes con la misma red LSTM entrenada. 45

4.7. Gráficas de los pesos y sesgos aprendidos de la red LSTM. Se comprueba que estos son valores fijos por lo que no cambian con las predicciones. 47

5.1. Diagrama de bloques del receptor basado en radio cognitiva usado como caso de estudio. 50

5.2. (a) Ilustración del espectro de frecuencia con n canales. (b) Representación de la señal de ocupación en el tiempo para los distintos canales. (c) Diagrama de bloques del sistema propuesto. 51

5.3. Evolución en el tiempo de la potencia para una señal de un canal especifico que sigue el estándar WLAN (frecuencia de 2.412 GHz con un ancho de banda de 5 MHz) [2]. 53

5.4. *Dataset* para el sistema de selección de bandas. En azul claro se representan los datos para el entrenamiento de las redes LSTM (DataTrain). Las demás señal, representan los datos de validación o test para cada uno de los cuatro canales usados. 54

5.5. Diagrama de flujo de la implantación software del selector de bandas mediante la predicción de series temporales usando redes neuronales LSTM. 57

5.6. Conjunto de datos test o de validación usados para la predicción del selector de banda ocupacional. 58

5.7. Resultados de simulación obtenidos para los cuatro canales donde las señales (a)-(d) representan las medidas de la ocupación de canal en el tiempo y (e) muestra la banda seleccionada. 59

5.8. Gráfica de datos de test para el canal 4 y la predicción realizada en la transición de (a) bajada y (b) subida. 61

5.9. Conjunto de las cuatro funciones de transferencia asociadas al filtro BP cuando cada canal es seleccionado. 62

Índice de cuadros

2.1. Ejemplo de un *dataset* de un problema de series temporales. 23

2.2. Ejemplo de un *dataset* de un problema de series temporales convertidos para aplicar aprendizaje supervisado. 24

4.1. Valores RMSE de las predicciones sin y con actualización de la red neuronal para distintas ejecuciones del mismo ejemplo de Matlab® 46

5.1. Información de la predicción para la ocupación de los canales usando los primeros 1000 muestras o pasos de tiempo. Muestra el tipo de transición, la subida supone la ocupación del canal y la bajada la desocupación de este. También, la muestra actual en la que se predice el cambio y la muestra o paso de tiempo en la que realmente se sobrepasa el umbral que marca esta transición. La resta de las dos anteriores refleja el número de pasos con los que se esta anticipando la predicción. 59

5.2. Media de pasos de tiempo con los que se predice la transición de estado de la ocupación de los canales. La subida supone la próxima ocupación del canal y la bajada la desocupación de este. 60

5.3. Tiempos de ejecución para diferentes procesos del selector de banda implementado en Matlab®. 62

Parte I

Prolegómeno

Capítulo 1

Introducción

1.1. Motivación

Nos encontramos en la mayor revolución tecnológica que ha conocido la humanidad. De hecho, las primeras décadas del siglo XXI serán recordadas por la expansión de las tecnologías de la información y las comunicaciones (TIC), y de los dispositivos y aplicaciones de las mismas, como teléfonos móviles, tablets, ordenadores personales, etc. que nos permiten acceder a la información a través de internet de una forma ubicua y con velocidades de conexión cada vez mayores.

Este desarrollo se debe en gran medida a la Microelectrónica la cual ha evolucionado en los últimos 50 años de una forma exponencial según la conocida ley de Moore. Una de las consecuencias de esta evolución tecnológica es un alto grado de conectividad con el entorno, dando lugar a lo que se ha denominado Internet de las cosas o IoT (*Internet of Things*). IoT comprende la interconexión de miles de millones de entidades ciberfísicas, con una estructura híbrida software/hardware, capaces de comunicarse entre ellas sin necesidad de intervención humana y tienen un sinfín de aplicaciones. La teleasistencia sanitaria personalizada, las operaciones bursátiles automatizadas, las redes energéticas inteligentes, la robotización en procesos industriales, o los vehículos autónomos, son solo algunos ejemplos del uso de IoT, que está cada vez más presentes en nuestras vidas.

Una implementación eficiente de IoT requiere el desarrollo de numerosos avances científico-tecnológicos que permitan obtener dispositivos electrónicos seguros y eficientes, tanto en coste como en consumo de energía. Además se precisa que estén dotados de una cierta inteligencia y autonomía, de forma que puedan tomar decisiones en tiempo real y que sean robustos a las condiciones del medio en que se desenvuelven. Para ello es necesario el desarrollo de tecnologías habilitadoras que hagan viable la construcción de un puente sólido entre el medio físico (real) y su versión virtualizada (digital).

Uno de los principales cuellos de botella para la implementación práctica de IoT es la saturación de las bandas del espectro electromagnético por las que se propagan las ondas radioeléctricas que portan la información transmitida por muchos aparatos electrónicos. Esto ha motivado la investigación y desarrollo de tecnologías que permitan hacer un uso más eficiente y sostenible del espectro electromagnético con el crecimiento exponencial de dispositivos interconectados. Una de esas tecnologías es la denominada Radio Cognitiva o CR (de *Cognitive Radio*). En esencia, la radio cognitiva se basa en la convergencia de tecnologías de comunicación y de computación que permiten ajustar de forma autónoma, y transparente para el usuario, los parámetros de transmisión y recepción de los dispositivos electrónicos, en función de la información que sensan del entorno radioeléctrico en el que se utilizan. Sin embargo, los microprocesadores empleados en dispositivos convencionales resultan ineficientes para realizar tareas de IA como las requeridas en sistemas CR. Esto ha motivado la investigación de alternativas computacionales, como son los sistemas neuromórficos.

1.2. Objetivos y Alcance

El objetivo de este trabajo es aportar soluciones eficientes energéticamente que permitan aumentar el nivel de integración de IoT, así como su grado de imbricación en la sociedad. Para ello se propone un sistema de radio cognitiva con perspectiva a poder ser implementable con técnicas de digitalización altamente programables y adaptativas mediante un *Software Defined Radio* (SDR), junto con un procesamiento de la información asistido mediante técnicas de aprendizaje automático basados en procesamiento neuromórfico. Este sistema permite identificar de forma automática la banda óptima del espectro electromagnético en la que se pueda transmitir mejor información, maximizando así la cobertura, minimizando el efecto de las interferencias e incrementando la durabilidad y la vida útil de la batería, entre otras muchas ventajas.

El propósito de este documento es el estudio de los conceptos más relevantes y necesarios para la aplicación de un sistema radio cognitivo y, una demostración de conceptos a nivel sistemático convergiendo un *Software Defined Radio* con inteligencia artificial. Las tareas a desarrollar son:

- **Investigación profunda de sistemas Radio Cognitivos o RC:** Se estudia los trabajos científicos más recientes y aplicaciones comerciales, se detalla la definición, evolución y hacia donde se dirige este área de investigación.
- **Análisis de los conceptos más importantes de un espectro electromagnético:** Para entender bien el problema que se presenta es necesario conocer cómo las ondas electromagnéticas se propagan y qué es necesario considerar en su evolución temporal.
- **Elección de una red neuronal acorde al problema que se presenta:** Existen varias redes neuronales caracterizadas por el tipo de neurona, topología y aprendizaje. Cada una de ellas presenta ventajas y desventajas frente a las demás y son más eficaces para un tipo de datos o problemas determinados. Se especificará cual es la red neuronal a usar razonadamente tras un estudio comparativo.
- **Propuesta sistema radio cognitivo:** Se detallará una aplicación concreta en la que se incluye la red neuronal elegida anteriormente para la predicción anticipada de la evolución de las ondas electromagnéticas. Se propone como ejemplo de aplicación un filtro analogico/RF paso de banda y se modificará sus parámetros o coeficientes del biquad para así seleccionar el ancho de banda más adecuado predicho. Esta aplicación es demostrada en software usando la herramienta de Matlab® y *Deep Learning Toolbox*®

El alcance se limita a lo comentado con antelación, el proyecto se basa en una prueba de conceptos debido a que es una tecnología punta y muy innovadora a día de hoy. Sin embargo, hay un interés creciente por integrar la inteligencia artificial en el hardware de los dispositivos. Por ello este proyecto es un primer paso a un próximo estudio más profundo de los sistemas radio cognitivos. En un futuro, se espera poder incorporar procesamiento neuromórfico en chips de comunicaciones que hagan posible la realización de dispositivos IoT/5G más eficientes, gracias al uso de la radio cognitiva.

1.3. Estado del arte

Se espera que la industria de la comunicación crezca continuamente y de forma exuberante, lo que es clave para impulsar la innovación y la productividad de otros sectores económicos, como el transporte, la atención de la salud, la agricultura, las finanzas, los servicios, la electrónica de consumo, etc. Por ejemplo, el actual brote de la enfermedad coronavirus (COVID-19) muestra de manera impresionante lo vital que es la industria de la información para la sociedad [1] o como se muestra en la Figura 1.1 obtenida del portal web de "Tecnología e Innovación del Ministerio de Defensa" de España, existe una evolución exponencial en el interés de aplicar Radio Software y Radio cognitiva también en este sector[1]. Para mantenerse al día con estas tendencias, los sistemas de comunicación inalámbrica deben sostener los requisitos extremos de las redes y dispositivos emergentes.

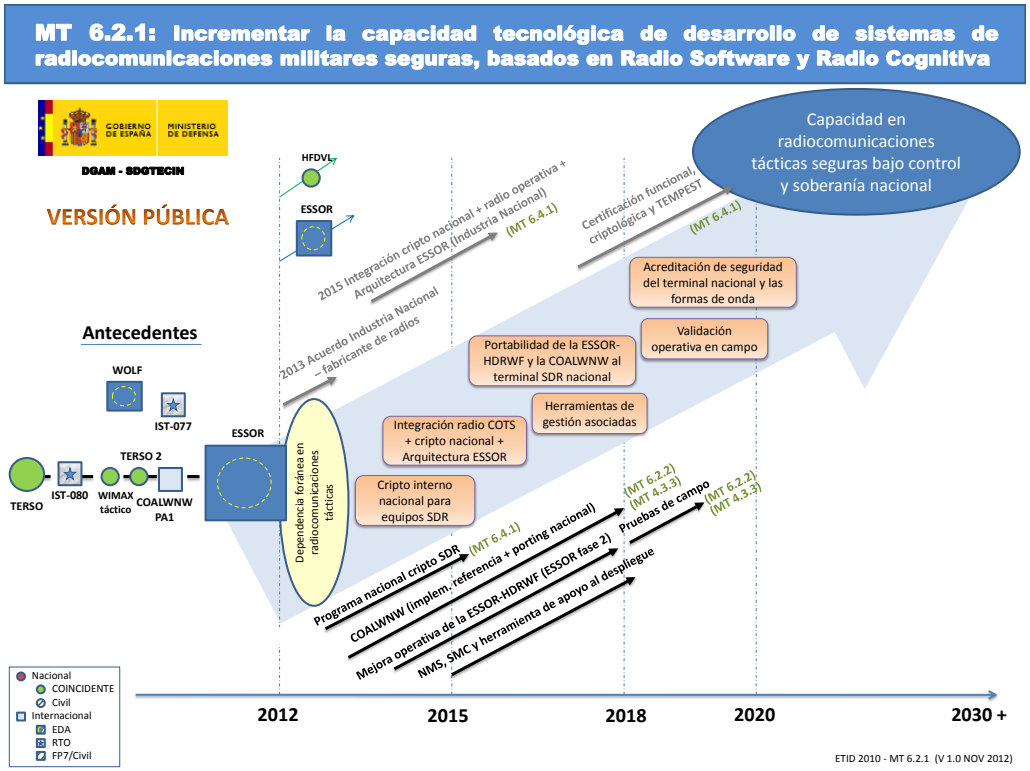


Figura 1.1: Evolución de la capacitación nacional para el desarrollo de radios militares definidas por software.

Una de esas tecnologías en constante desarrollo son las comunicaciones móviles. Hace poco más de un par de décadas, los terminales móviles eran simplemente teléfonos inalámbricos, cuya única funcionalidad era la transmisión de voz (primera generación o 1G), a la que se añadió posteriormente la transmisión de SMS en la segunda generación (2G), con velocidades de transmisión de unos pocos de kilobits por segundo (kb/s). Con el desarrollo del 3G, los móviles pasaron a ofrecer servicios multimedia y conexión a internet de banda ancha con velocidades acceso de varios Megabits/s (Mb/s). En la actualidad, la mayoría de las redes operan con terminales móviles de cuarta generación (4G), que permiten alcanzar velocidades hasta centenares de Mb/s, y ya se empieza a implantar la red 5G, que permiten alcanzar velocidades de Gigabits/s (Gb/s) y latencias a nivel del milisegundo. Esta quinta generación y sus próximas versiones como el B5G tienen como objetivo conectar decenas de miles de millones de dispositivos inalámbricos pero todavía queda un gran número de problemas relacionados con la comunicación inalámbrica. Las redes necesitarán

[1]Para más información sobre la hoja de Ruta de la Meta Tecnológica 6.2.1. "Radio Software y Radio Cognitiva" del Ministerio de Defensa de España acudir a la siguiente página web: http://www.tecnologiaeinnovacion.defensa.gob.es/es-es/Contenido/Paginas/detallepublicacion.aspx?publicacionID=93

transferir cantidades mucho mayores de datos a velocidades mucho más altas obligando a cumplir con las tendencias de eMBB (banda ancha móvil mejorada), mMTC (comunicaciones masivas de tipo máquina), URLLC (comunicaciones ultra fiables de baja latencia) y garantizar calidad del servicio o QoS (*Quality-of-service*) entre otras [4] (ver Figura 1.2). Aunque aún se estudia el uso de la quinta generación para aplicaciones que conecten no solo las personas sino también vehículos, sensores, agentes robóticos, etc; los expertos hablan ya de una sexta generación (6G) que se orientará directamente hacia este nuevo paradigma de los sistemas inteligentes e IoT [5].

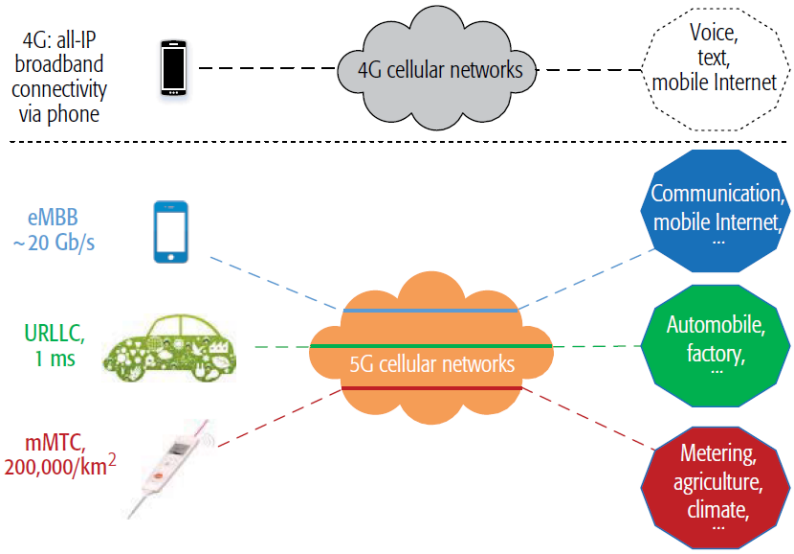


Figura 1.2: Evolución hacia las redes móviles 5G como una clave para habilitar dichos dispositivos a todas las aplicaciones industriales.

Como prueba de las exitosas aplicaciones ya realizadas en campos como visión por computadora, reconocimiento de voz, procesamiento de lenguaje natural, reconocimiento de audio, traducción automática, minería de redes sociales, bioinformática y diseño de compuestos, los investigadores están considerando aplicar técnicas de inteligencia artificial a las comunicaciones inalámbricas y, por tanto, a las comunicaciones móviles. Los resultados existentes a nivel de la capa física han demostrado que la IA puede ayudar a comprender los contenidos inalámbricos, reconocer patrones no revelados, reducir la complejidad e incluso producir resultados superan los enfoques convencionales. Aunque se han nombrado algunas iniciativas relacionadas, algoritmos, implementaciones y optimizaciones no están, lamentablemente, completas. Se espera que el importante potencial de la IA haga avanzar las arquitecturas de redes [6], las soluciones de procesamiento de señales [7], las tecnologías de semiconductores [8], así como la optimización a nivel de sistema [9].

A continuación se hace hincapié en algoritmos, implementaciones y optimizaciones relacionados con la IA para 5G para proporcionar una visión general de los progresos recientes nombrando algunos ejemplos de estudio. Aunque un número de ediciones y sesiones especiales en conferencias clave del IEEE (*Institute of Electrical and Electronics Engineers*) como [10], escuelas y tutoriales aparecieron recientemente sobre este tema emergente, nos centramos solo en aquellos cercanos a las implementaciones de circuitos y sistemas. Se resumen algunas de las últimas publicaciones sobre IA aplicada para la detección y precodificación masiva de múltiples entradas y salidas o MIMO (*Multiple-Input Multiple-Output*), para la codificación del canal, para el procesamiento de otros módulos de banda base, los métodos de aprendizaje automático (ML) que se ocupan de las incertidumbres del modelo y, por último, se examina las futuras direcciones de investigación.

Debido a las ventajas en eficiencia tanto espectral como energética y en la calidad de servicio o QoS (*quality-of-service*), la tecnología MIMO es un componente básico para muchos sistemas inalámbricos. Sin embargo, las implementaciones eficientes de la detección y precodificación MIMO siempre deben compensar el rendimiento con la complejidad y a la vez detectar con una exactitud suficiente teniendo en cuenta el número de transmisores y esquemas de modulación de orden

superior que conforme el sistema. En la Figura 1.3 se presenta una tabla resumen de las ventajas e inconvenientes de las diferentes soluciones propuestas hasta ahora para los detectores MIMO. Los algoritmos lineales, como la ecualización basada en el forzamiento cero y el error cuadrático medio mínimo (o MMSE según su nombre en inglés *Minimum Mean Square Error*), y los métodos iterativos lineales, por ejemplo, el Gauss-Seidel (GS) o el descenso por coordenadas tienen una baja complejidad; sin embargo, su rendimiento suele ser inaceptable en condiciones de propagación realistas. Los algoritmos de detección no lineales, como el *Tree-Search*, la cancelación de interferencias y el paso de mensajes, pueden lograr un rendimiento casi óptimo a una complejidad significativamente mayor que los métodos lineales. En particular, los métodos basados en el paso de mensajes suelen sufrir una grave degradación del rendimiento en canales realistas, lo que motiva el desarrollo de soluciones basadas en la Inteligencia Artificial con el fin de mejorar el rendimiento, la complejidad y la robustez. En [11] se propone una arquitectura de red neuronal profunda o *deep neural network* (DNN) adaptada al algoritmo de propagación de creencias (en inglés, *belief propagation algorithm* o BP) dando como solución detectores de paso de mensajes o *message passing detectors* (MPDs) basados en estas redes. Esta publicación presenta diferentes algoritmos que demuestran que el método MPD propuesto con DNN es capaz de lograr un rendimiento superior y mejora la robustez en comparación con otros métodos MPD para una serie de configuraciones de sistema y condiciones de canal. Otro ejemplo se presenta en el artículo de Junkai Liu y Jienan Chen [12] que presenta el sistema de formación de haz híbrido o HB (*Hybrid beamforming*) basado en ondas milimétricas MIMO con conjuntos de antenas a gran escala como una de las tecnologías prometedoras para el futuro de las comunicaciones previéndose clave para el desarrollo de la de quinta generación (5G) y más allá.

Detectors	Pros	Cons
Optimal	- Optimal detection performance	- NP-hard and not practical for large-scale systems
Tree-Search Based	- Near-optimal performance with reduced complexity compared to optimal ones	- Exponential worst-case complexity and impractical for massive systems
Linear	- Good performance with low complexity for small-scale MIMO	- Involves a matrix inversion of complexity $\mathcal{O}(N^3)$ - Limited performance for massive MIMO
Linear Iterative	- Matrix inversion-free with $\mathcal{O}(N^2)$ complexity - Implementation-friendly	- Performance is bounded by MMSE
Message Passing	- Achieves better performance than MMSE with acceptable complexity - Naturally provides soft output, which can further utilized by other baseband blocks - Implementation-friendly and reconfigurable	- Suffers convergence problem due to loopy factor graph - With hard-to-optimize compensation factors
AI Based	- Optimizes factors with learning techniques - Performance improvement over message passing	- High-complexity training stage - Requires re-training for varying channels

Figura 1.3: Ventajas y desventajas de los detectores MIMO propuestos hasta el momento[1].

Por otro lado, los estudios recientes de la decodificación de canales basada en IA han demostrado que el rendimiento de los métodos existentes también puede mejorarse (a menudo de manera significativa). Mientras que algunos resultados utilizan redes neuronales para entrenar los parámetros de escalado de los algoritmos convencionales, otros aplican redes neuronales al posprocesamiento de la salida de los decodificadores, otros intentan decodificar directamente con redes neuronales mediante la decodificación de una sola vez. Dado que tanto los códigos de comprobación de paridad de baja densidad o LDPC (*Low-density parity-check*) como los códigos polares han sido normalizados por 5G, las investigaciones más recientes se centran en estos tipos de códigos. El algoritmo de propagación de creencias o BP es un algoritmo de paso de mensajes, que decodifica de forma iterativa los códigos LDPC mientras proporciona un rendimiento casi óptimo de la tasa de error. Sin embargo, debido a la complejidad bastante alta del algoritmo BP, existen varias simplificaciones. Los códigos polares son capaces de alcanzar asintóticamente la capacidad de Shannon[2] utilizando el algoritmo de decodificación de cancelación sucesiva (*successive cancellation* o SC) a medida que la longitud del código se acerca al infinito. El algoritmo de decodificación SC estima secuencialmente los bits de información basándose en las relaciones logarítmicas de probabilidad o LLR (*log-likelihood ratios*) del canal recibido y en la estructura de codificación. Además del algoritmo de decodificación SC, el algoritmo BP también puede decodificar códigos polares pasando valores LLR basados en

[2]En teoría de la información, el teorema de Shannon-Hartley define la capacidad del canal. esta es una cota superior que establece la máxima cantidad de datos digitales que pueden ser transmitidos o no sin error sobre dicho enlace de comunicaciones con un ancho de banda específico y que está sometido a la interferencia del ruido.

el gráfico de codificación de forma iterativa. En comparación con el algoritmo de decodificación SC, la decodificación BP proporciona un mayor rendimiento y una latencia más corta, pero sufre una degradación sustancial del rendimiento de la tasa de error. Tanto para la decodificación SC como para la decodificación BP, la investigación está trabajando para lograr un equilibrio entre el rendimiento de la tasa de error y la complejidad. Dado que algunos de los mecanismos subyacentes son difíciles de modelar, la inteligencia artificial ha sido recientemente considerada la posibilidad de optimizar este equilibrio. Se puede nombrar el uso de IA para optimizar los decodificadores utilizando redes neuronales, varios de ellos sustituyen directamente el decodificador por una red neuronal para realizar una decodificación de un solo disparo (es decir, no iterativa). En [13], se utiliza un decodificador de red neuronal para decodificar códigos no estructurados como lo es el LDPC y códigos estructurados (o códigos polares). Los resultados de las simulaciones muestran que para ambos, el decodificador de red neuronal o NND (*neural network decoder*) puede aproximarse al rendimiento de la máxima probabilidad *a posteriori* para longitudes cortas (relativamente menores a 1024 bits). Se observa que el algoritmo de decodificación de los códigos estructurados es más fácil de aprender para las redes neuronales. Además varios trabajos de investigación estudian las redes para el post-procesado concatenando una red neuronal a la salida del decodificador del canal para procesar su salida, y retroalimentan la información para su procesamiento iterativo. Esto se hace, por ejemplo, en [14] que propone un algoritmo SC asistido por red de memoria a largo y corto plazo (LSTM) para decodificar los códigos polares. Si el detector de error falla, la salida del decodificador SC será alimentada en una red LSTM para localizar y cambiar el primer bit de error durante el siguiente intento de decodificación SC.

Además de los dos módulos clave comentados, el decodificador de canal y el detector MIMO, el diseño y la aplicación de otros módulos de banda de base, como el estimador de canal, el ecualizador y el detector de acceso múltiple no ortogonal (NOMA), también han utilizado métodos de IA para explotar patrones específicos, especialmente el comportamiento no lineal, que a menudo no pueden manejarse con los métodos convencionales. Centrando la atención en los estimadores de canal, es necesario saber que en los sistemas de comunicación inalámbrica, la detección y precodificación coherentes requieren estimaciones precisas de la función de transferencia del canal. Recientemente, se han adoptado métodos de aprendizaje profundo para la estimación de canales. Inspirado en el procesamiento de imágenes, [15] aplica la red neural aprendida de paso de mensaje aproximado basado en la eliminación de ruido a la estimación de canales, considerando la matriz de canales como una imagen. Se muestra como la red aprende la estructura de los canales y a partir de esto se obtienen resultados que superan a otros métodos de estimación de canales basados en la detección comprimida. En resumen, las técnicas de IA se han explotado para otros módulos de procesamiento de la banda base notándose en una mejora del rendimiento. Sin embargo, debido a la naturaleza de los diferentes módulos, los enfoques de la IA pueden variar. Además, se espera que en futuras investigaciones se diseñe un hardware eficiente.

Hasta ahora se ha ilustrado la forma en que las técnicas de ML pueden emplearse como alternativa a los algoritmos convencionales que no son óptimos. Esas soluciones son interesantes cuando se desconocen los algoritmos teóricamente óptimos o las soluciones más complejas. Una motivación ligeramente diferente, pero igualmente válida, para el uso de las técnicas de ML en las comunicaciones son los escenarios o problemas para los que el modelo de sistema correspondiente en sí mismo no es conocido o es insuficientemente conocido. En esta situación, los algoritmos convencionales solo pueden derivarse sobre la base de supuestos a menudo simplistas que en algunos casos pueden resultar difíciles de realizar o que pueden adolecer de una incertidumbre considerable. En esos casos, la capacidad de aprender un modelo genérico gracias al ML o de entrenar un algoritmo basado en observaciones sin necesidad de conocer explícitamente el modelo de sistema subyacente puede ser una gran oportunidad. A continuación se presentan dos ejemplos de aplicación. En primer lugar se puede nombrar un nuevo enfoque de la comunicación de radio en la que un sistema transmite y recibe al mismo tiempo en la misma banda de frecuencia. La principal dificultad reside en la necesidad de una cancelación altamente eficiente de la auto-interferencia que a menudo está por encima de los 80dB sobre la débil señal recibida. La reconstrucción y la cancelación de esta auto-interferencia debería ser en principio sencilla, ya que la señal transmitida es conocida. Desafortunadamente, los componentes de radiofrecuencia (RF) en un sistema del mundo real introducen importantes distorsiones no lineales. Estas distorsiones deben ser modeladas y reconstruidas, pero los modelos existentes no las captan todas con la suficiente precisión. Las técnicas de ML ofrecen

un enfoque para reconstruir la señal. Esta idea básica se ha descrito por primera vez en [16] y se ilustra en la Figura 1.4. En el artículo [17] publicado este mismo año 2020, se demuestra que la arquitectura hardware que se propone es capaz de igualar el rendimiento de la cancelación del la auto-interferencia de los sistemas convencionales con una complejidad computacional significativamente menor. Para la segunda aplicación nos basaremos en [18] donde los autores proponen utilizar una red neuronal para realizar una predistorsión para reducir el impacto negativo de las no linealidades de los amplificadores de potencia (PA) en la calidad de la señal transmitida. Esta tarea capta tanto la dificultad de modelar exactamente esas no linealidades, como el diseño correspondiente con una complejidad limitada dando unos resultados que indican que, una vez más, las técnicas de ML pueden al menos igualar el rendimiento de las técnicas convencionales con una complejidad global significativamente menor.

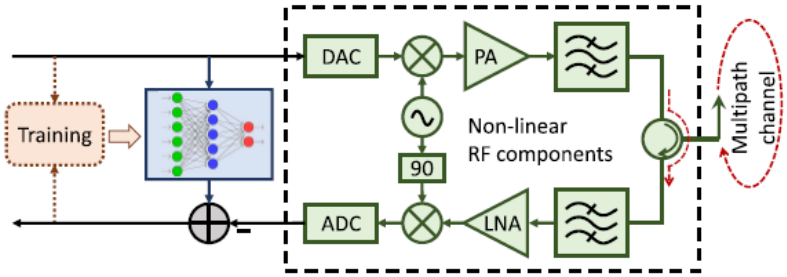


Figura 1.4: Reconstrucción por distorsión no-lineal por autointerferencia basadas en una red neural para sistemas *full-duplex* [1].

En última instancia y aunque ya se han demostrado estudios avanzados, las futuras investigaciones sobre la IA para 5G aún deben abordar más cuestiones. Primero, seguir cuestionando los métodos de aprendizaje con los convencionales. En ciertas situaciones, los métodos convencionales pueden ser un solucionador satisfactorio con una complejidad aún menor en comparación con los métodos de aprendizaje. Los trabajos existentes han demostrado que los métodos basados en el aprendizaje demostrarán sus ventajas cuando se ocupen de problemas que no pudieron ser modelados o resueltos con exactitud antes. Es decir, las investigaciones futuras deben identificar el área de aplicación de la IA para 5G/B5G. Otra cuestión a resolver es la comparación del aprendizaje basado en modelos o en datos. Combinado con el conocimiento de los expertos, el aprendizaje basado en modelos puede reducir eficazmente la complejidad de la capacitación pero, por otra parte, el aprendizaje basado en datos puede ayudar a descubrir las pautas útiles que no pudieron ser averiguadas por los expertos, para obtener un mejor rendimiento aunque la complejidad pueda ser prohibitiva. La introducción de conocimientos especializados depende de la fiabilidad de estos. El pleno aprovechamiento de los conocimientos especializados puede mejorar la eficiencia del aprendizaje y la tasa de convergencia, y reducir la complejidad para satisfacer las necesidades en tiempo real. Se espera que las investigaciones futuras señalen qué enfoque es apropiado para cada problema específico. Es importante también considerar la complejidad y ocupación en tiempo y espacio del entrenamiento necesario de los algoritmos y redes para las predicciones. Para acelerar el entrenamiento y que ocupe menos espacio de memoria, los diseñadores suelen preferir el entrenamiento *offline*. Si esto no da buen resultado se considera el entrenamiento *online*. Por consideraciones de tiempo real, se debería reducir el número de parámetros que se han de entrenar *online* y se requiere una aceleración del equipo informático. La forma de obtener eficientemente los datos de capacitación para el aprendizaje *online* es otra cuestión que debe abordarse.

Los trabajos existentes centran la mayoría de las aplicaciones de la IA en una sola capa, por ejemplo, la capa física. Aunque se puede lograr un buen rendimiento y equilibrio, el aprendizaje de la capa física está de alguna manera limitado. Algunos investigadores sostienen que no se debe limitar sino que debemos considerar la posibilidad de aplicaciones en varias capas. Por ejemplo, hoy en día los investigadores han comenzado a optimizar las redes basadas en la IA y se espera que el aprendizaje que cruza dos o tres capas obtengan más beneficios. Es de interés comentar que el aprendizaje con el software siempre será la primera opción debido a su conveniencia y adaptabilidad. Sin embargo, para las tareas de aprendizaje complicadas, la implementación de software

introducirá una gran complejidad de tiempo y espacio. Por lo tanto, el aprendizaje con hardware también ha sido considerado. Para una implementación eficiente del hardware, el algoritmo correspondiente debe ser fácil de usar, por ejemplo, robusto a la cuantificación y de estructura regular. Los conocimientos de diseño de hardware tanto en FPGA como en ASIC son esenciales para esta cuestión. Por otra parte, la IA también puede ayudar a diseñar hardware reconfigurable para el aprendizaje. Muchos estudios aportan soluciones al diseño hardware, una de las comentadas anteriormente es [17].

El objetivo del estudio que se presenta en este documento, como bien se describió en la sección anterior, es el de contextualizar la predicción de las ondas electromagnéticas usando redes neuronales apropiadas con el fin de seleccionar de forma óptima el canal o ancho de banda de un filtro paso de banda. Es decir, este trabajo se limita a proponer un ejemplo de aplicación que es demostrado mediante software. Por ello, se ha hecho una búsqueda del estado del arte que trata sobre el tema y se estudian los aspectos más importantes para aplicar este tipo de sistema radio cognitivo. Esto abrirá un camino de investigación sobre este área pudiendo incluso en un futuro hablar de su implantación *on-chip*.

1.4. Estructura de la memoria y material adicional

Este documento se divide en esta parte introductoria y dos más, una para el marco teórico y otra para el desarrollo, simulaciones y conclusiones tomadas. A su vez, el marco teórico se fragmenta en dos capítulos. El primero, el capítulo 2, trata de dar un enfoque general de la radio cognitiva dando una introducción y definición que sitúe al lector en el problema de este estudio. También incluye los conceptos de espectro electromagnético y comunicaciones necesarios para poder entender el caso de estudio y los próximos pasos. Además da una introducción a la inteligencia artificial aplicada a sistemas de radio cognitiva describiendo los problemas a los que estas técnicas de aprendizaje automático se van a enfrentar y, así, explicar razonadamente la red neuronal empleada en este estudio. El capítulo 3 se enfoca en la red elegida Long Short-Term Memory (LSTM) explicando detalladamente su arquitectura y funcionamiento. Por otro lado la parte de desarrollo, simulaciones y conclusiones también se divide en tres capítulos. El capítulo 4 trata la arquitectura de la red LSTM aplicada en Matlab® mostrando sencillas predicciones que demuestren el comportamiento de las mismas. El capítulo 5 presenta el sistema de concepto propuesto en este proyecto en el que las entradas al sistemas será las ondas electromagnéticas a predecir y un filtro analogico/RF paso de banda cambiará según la predicción hecha por la red LSTM. El capítulo final, el número 6, expone las conclusiones tomadas y las mejoras hacia futuros estudios sobre este área de la radio cognitiva.

Este trabajo incluye un código fuente escrito en MATLAB® formado por varios ficheros que se incluyen como material adicional junto a esta memoria. Se prosigue a enumerar justificando cada uno de dichos ficheros contenidos en dos carpetas digitales diferente: **capitulo4** y **capitulo5**. En primer lugar, para las pruebas realizadas en el capítulo cuatro, contenidas en la carpeta **capitulo4**, se demuestra y justifica como la red neuronal se modela tras el entrenamiento. Las pruebas incluyen los ficheros siguientes:

- **TimeSeriesForecastingUsingDeepLearningExample_vzg.mlx** Para estudiar de forma sencilla y segura el comportamiento de la red neuronal a usar, se decide usar un ejemplo incorporado en la base de datos de Matlab[3]. Este ejemplo es un buen punto de partida para el sistema que se quiere implementar en este trabajo y que se hace en el capítulo 5. Lo único que se modifica del código original es la adición de unas variables a las que se le asignan el modelo de la red neuronal en diferentes etapas y situaciones del código.

- **Study_network.m** Este fichero incorpora de forma sencilla las diferentes pruebas y gráficas

[3]Contenido en la página web: https://www.mathworks.com/help/deeplearning/ug/time-series-forecasting-using-deep-learning.html y puede ser abierto en Matlab simplemente escribiendo en la ventana de comandos `openExample('nnet/TimeSeriesForecastingUsingDeepLearningExample')`.

para verificar cómo la red es modificada a lo largo del fichero **TimeSeriesForecastingUsingDeepLearningExample_vzg.mlx**.

- **Sim1-3.m** Estos son tres ficheros que guardan la red entrenada tras tres ejecuciones completas del código. Se usarán para realizar las pruebas.

Luego, para la prueba de concepto que se desarrolla en este trabajo y que es detallada en el capitulo cinco, se incluyen los siguientes ficheros almacenados en la carpeta **capitulo5**:

- **build_input_data.m** Usa una secuencia de niveles binarios de la ocupación de la banda o canal de comunicación (aunque puede ser también multi-nivel) para crear una señal dinámica similar a la evolución temporal de las ondas electromagnéticas. La señal de salida es ideal, no está afectada por componentes aleatorias.
- **generate_data_input.m** Usa la función build_input_data.m y le añade ruido para acercar más la señal a un comportamiento real.
- **forescast.m** Es el fichero de predicción en el que en cada paso de tiempo se realiza una predicción futura del espectro electromagnético. Para ello, se debe tomar una serie de requisitos de diseño que se irán aclarando a lo largo del capítulo cinco. En la sección 5.4 se detallan los pasos más importantes de dicha función.
- **filtro.m** Es el fichero del filtro de aplicación. En esta prueba de concepto de un sistema radio cognitivo, el filtro únicamente actúa como el transmisor pero no es lo fundamental del sistema, su objetivo es simplemente demostrar el buen comportamiento del trabajo propuesto. Por ello, este consiste de un circuito Butterworth básico.
- **AlgoritmoPrediccionSistemaRC.mlx** Este es el fichero principal a ejecutar para obtener la respuesta del sistema. Incluye la generación de datos, el entrenamiento de la red, la predicción y la respuesta del filtro ante la predicción. Este fichero da la opción de contemplar otro tipo de datos pero esto aún sigue siendo de estudio y se pretenden usar en el futuro de esta línea de investigación.
- **channels.mat** Para poder realizar varias pruebas con los mismos datos, se guarda en este fichero los datos de test usados en los resultados de este trabajo. solo se debe tener en cuenta que este fichero de datos debe estar en la misma carpeta que el fichero **PrediccionSistemaRC.m** junto con las demás funciones. Este es llamado en el código para cargar los datos.
- **workspace.mat** Para agilizar las pruebas, se guarda en este fichero todos los datos generados en la creación de las señales de entrada, diseño de la red neuronal y su entrenamiento. Obtener dichos datos conlleva bastante tiempo y potencia computacional principalmente el entrenamiento de la red que, según el número de datos y potencia del ordenador donde se ejecute, puede llevar incluso más de 60 minutos (se comentará más a profundidad en el capitulo 5). Para usarlo, simplemente ejecutar este fichero y tras ello solo hace falta ejecutar la sección del bucle de predicción (referenciado así en el código con letras de color anaranjado) dentro del fichero principal **PrediccionSistemaRC.m**.

Para un uso normal de este último algoritmo, a parte de saber que todos los ficheros deben estar almacenados en la misma carpeta, se ejecuta el fichero principal **PrediccionSistemaRC.m** y este a su vez llama a los demás. El fichero principal está dividido en tres secciones diferenciadas. La primera de ellas, genera los datos a usar. La segunda crea la red neuronal y la entrena. Y la última sección tiene como fin la predicción de las series temporales en base a las señales y la red previamente definidas. Para agilizar las pruebas, se puede reutilizar los datos y la red definida en las pruebas del capítulo 5 cargando el fichero **workspace.m** y, luego, ejecutar únicamente la sección de predicción del fichero principal.

Parte II

Marco teórico

Capítulo 2

Inteligencia artificial en sistemas de Radio Cognitiva

2.1. Radio Cognitiva. ¿Qué es y a dónde quiere llegar?

La demanda de velocidad de datos cada vez más altas en las comunicaciones inalámbricas ante la limitación o la mala utilización de los recursos espectrales ha motivado la introducción de la radio cognitiva. Diversas mediciones de la utilización del espectro han mostrado sustanciales recursos no utilizados en frecuencia, tiempo y espacio. El concepto detrás de la radio cognitiva es explotar estos recursos espectrales reutilizando el espectro no utilizado de manera eficiente. El término "radio cognitiva" se atribuye al artículo de Mitola [19] publicado en 1993, aunque la idea de utilizar máquinas de aprendizaje y detección para sondear el espectro radioeléctrico se previó varias décadas antes. En esencia, la radio cognitiva se basa en la convergencia de tecnologías de comunicación y de computación que permiten ajustar de forma autónoma, y transparente para el usuario, los parámetros de transmisión y recepción de los dispositivos electrónicos, en función de la información que sensan del entorno radioeléctrico en el que se utilizan. Para ello se requiere dotar a dichos dispositivos de sistemas de comunicaciones en los que la digitalización o transformación digital de las señales que portan la información se realice lo más cerca posible de la antena tanto en el receptor como en el transmisor (ver Figura 2.1). Ello permite aumentar significativamente el grado de programabilidad y adaptabilidad de los terminales móviles a diferentes modos o estándares de comunicación y, debido a que el procesamiento de la información se hace mediante software, puede ejecutarse en un microprocesador digital. Además de un sistema de comunicación basado en software, también denominado radio definida por software o SDR (*Software-Defined-Radio*), la radio cognitiva requiere del uso de algoritmos de Inteligencia Artificial (IA), que permitan identificar de forma automática la banda óptima del espectro electromagnético en la que se pueda transmitir mejor información, maximizando así la cobertura, minimizando el efecto de las interferencias e incrementando la durabilidad y la vida útil de la batería, entre otras muchas ventajas.

En los sistemas de radiocomunicación cognitiva suelen participar sobre todo los usuarios primarios del espectro, que son los titulares de las licencias, es decir, las entidades que abonan por usar determinados anchos de banda del espectro electromagnético. Por otro lado están los usuarios secundarios que tratan de utilizar el espectro de manera oportunista cuando los usuarios primarios están inactivos. La introducción de las radios cognitivas crea inevitablemente un aumento de las interferencias y degrada la calidad del servicio del sistema primario. Este impacto debe minimizarse y para ello, los sistemas radio cognitivos deben percibir el espectro para detectar si está disponible o no y deben ser capaces de detectar las señales débiles de los usuario primarios para que así puedan ser usadas por los secundarios. Esto conlleva a que la detección del espectro es uno de los componentes más esenciales de la radio cognitiva [20]. Debido a ello, en secciones posteriores se aportan diferentes técnicas predictivas aplicadas a la detección de señales que presentan el mismo o similar comportamiento que las ondas electromagnéticas. Por lo tanto, será necesario describir

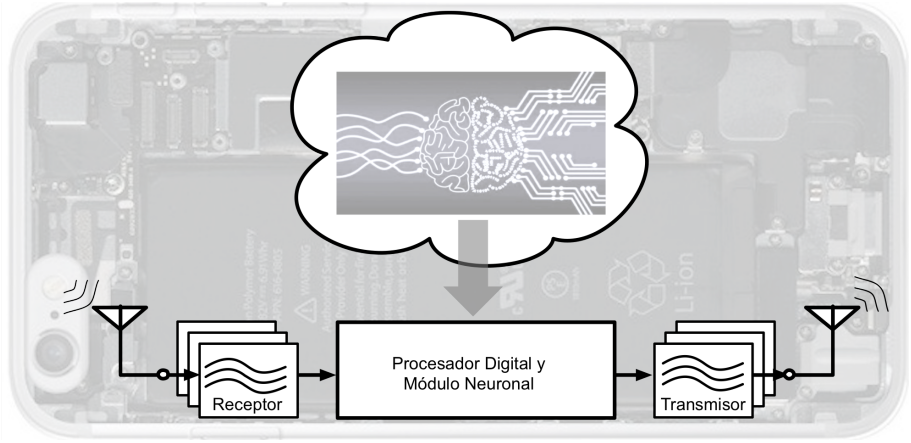


Figura 2.1: Esquema conceptual de un sistema radio cognitiva

antes las características y conceptos más relevantes de estas señales.

2.2. Evaluación y medidas del espectro de ocupación

Sobre las bases recogidas en la unión internacional de la telecomunicación o ITU (*International Telecommunication Union*) específicamente en el sector de la radiocomunicación (ITU-R) en la edición de 2016 del manual de comprobación técnica del espectro [2], se ofrece un debate detallado sobre los diferentes enfoques de las mediciones de ocupación del espectro, los posibles temas relacionados con ellos y sus soluciones empleando las recomendaciones ITU-R SM.1880 y ITU-R SM.1809. Aun así, esto es extensible a otras técnicas de multiplezación sin pérdida de generalidad.

La función del sector de radiocomunicaciones es asegurar la utilización racional, equitativa, eficaz y económica del espectro de radiofrecuencias para todos los servicios de radiocomunicaciones (incluidos los servicios por satélite), y realizar estudios sin límite de gama de frecuencias sobre la base de los cuales se adopten las recomendaciones dadas. Las funciones reglamentarias y de política del sector son desempeñadas por las conferencias mundiales y regionales de radiocomunicaciones y las asambleas con el apoyo de las comisiones de estudio.

2.2.1. Cálculo de la ocupación espectral

Las técnicas de modulación de paso de banda digital pueden clasificarse a grandes rasgos en dos categorías. La primera es la modulación de una sola portadora o de simple acceso, en la que los datos se transmiten utilizando una sola portadora de radiofrecuencia (RF). La otra es la modulación multiportadora o de múltiple acceso, en la que los datos se transmiten mediante la modulación simultánea de varias portadoras RF. Los sistemas de comunicación inalámbrica proporcionan diversos tipos de comunicación, como la voz y los datos, a varios usuarios. Es por ello que nos centraremos en las técnicas de acceso múltiple y en concreto en la Multiplexación por División de Frecuencia o en inglés *Frequency Division Multiplexing* (FDM) [21]. El método FDM consiste en dividir un espectro de frecuencias en varios canales con anchos de banda más pequeños que el del propio espectro de forma que entre ellos exista suficiente espacio para no interferirse entre ellos y perder información a consecuencia de ello. En este documento, cuando se habla de canales o bandas se hace referencia a un ancho de banda concreto asignado en el espectro de ocupación de la señal de interés.

Algunos estándares de redes inalámbricas como la IEEE 802.11s funcionan utilizando uno de los varios canales predefinidos dentro del espectro. La asignación de los ancho de banda para los distintos canales debe ser tomada de manera que se utilice eficazmente el espectro disponible. Esto

es importante para detectar las colisiones con otros sistemas y/o reasignar en un ancho de banda alternativo cuando se detecte interferencia [22]. En la Figura 2.2 se reproduce un diagrama simple de concepto de la división de la frecuencia en n canal. El ancho de banda BW de los canales se supone constante. Los límites entre canales consecuentes (frecuencias entre f_2 y f_3, f_4 y f_5...) son especificados por los estándares como por ejemplo lo hace el mencionado IEEE 802.11s.

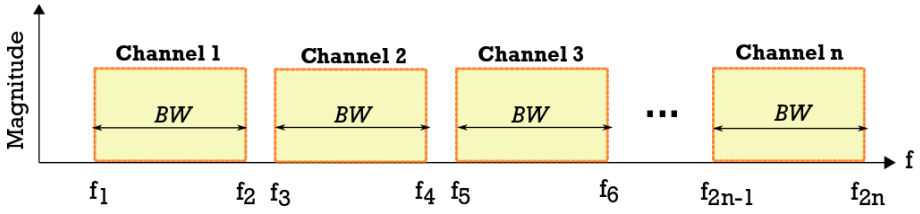


Figura 2.2: Representación de la división en frecuencia del espectro electromagnético mediante asignación por canales.

También un dato de interés es que existen bandas de frecuencias compartidas por usuarios con diferentes anchos de banda. En algunas bandas de frecuencia, los sistemas coexistentes pueden tener un ancho de banda y un espaciamiento de canal completamente diferentes. Un ejemplo es la banda de radiodifusión en ondas decimétricas, que puede ser utilizada por transmisores de televisión con un ancho de banda de canal de 8 MHz y por micrófonos inalámbricos con una separación de 25 kHz. Una medición de la ocupación con una resolución de 8 MHz puede mostrar los canales de televisión ocupados, pero no puede reconocer los micrófonos inalámbricos que ocupan solo una fracción de un canal de televisión. Si la medición se realiza sobre la base de una separación de canales de 25 kHz, la transmisión de la TV se mostraría como una serie de canales de micrófonos inalámbricos adyacentes totalmente ocupados.

Por otro lado, la ocupación momentánea espectral de un canal se entiende como señales binarias de todo o nada asumiendo que la señal está ocupada o no lo está. Esto es lo mismo que considerar que la ocupación solo será del 0 % o 100 %. Para definir el estado de la ocupación en un momento específico se define un valor umbral. Si este es sobrepasado, la banda estará ocupada y si no es así, estará vacía. Para dar significado, todas las cifras de ocupación a calcular deben ser promedios sobre un cierto período de tiempo. Este tiempo promedio se llama tiempo de integración. A continuación se enumeran diferentes aspectos y conceptos para comprender la medición y evaluación de la ocupación del espectro. Estos términos son necesarios para posteriormente referirnos a ellos con el objetivo de caracterizar las señales electromagnéticas [2]:

- **El recurso del espectro o *Spectrum resource*:** Este término describe la disponibilidad del espectro en términos de espacio (por ejemplo, ubicación, zona de servicio), tiempo y número de canales (en una banda canalizada) a los que pueden acceder todos los usuarios de un determinado territorio. En lo que respecta a una asignación de frecuencia única, el recurso del espectro puede ser en un solo canal de frecuencia. En el caso de redes auto-organizadas, como el sistema de teléfonos móviles, el recurso espectral puede consistir en todos los canales de frecuencias en una determinada banda, pero puede estar limitado en el tiempo, por ejemplo, en una franja horaria concreta. Por consiguiente, el recurso de espectro depende en gran medida del servicio de radiocomunicaciones y del problema específico que se esté examinando.

- **Valor umbral:** El umbral es un cierto nivel en la entrada del receptor que determina si un canal se considera ocupado o no. Puede ser un nivel fijo predefinido o variable. La ocupación resultante depende en gran medida del umbral por lo que es necesario establecer cuidadosamente su valor.

- **Ocupación del canal de frecuencia o FCO (*Frequency channel occupancy*):** Un canal de frecuencia está ocupado mientras el nivel o magnitud medida de la onda electromagnética esté por encima del umbral. Para un canal, el FCO es calculado de la siguiente forma:

$$FCO = \frac{T_o}{T_I} \tag{2.1}$$

donde T_o es el tiempo en el que el canal tiene un nivel medido por encima del umbral y T_I es el tiempo de integración. Asumiendo que el tiempo empleado en observar todos los canales para medir si están ocupados o no es constante, la distancia entre todas las medidas realizadas debe ser igual y entonces el FCO también puede ser definido como:

$$FCO = \frac{N_o}{N} \tag{2.2}$$

siendo N_o el número de muestras medidas en el canal con un nivel por encima del umbral y N el número total de medidas del canal durante el tiempo de integración.

- **Ocupación de la banda de frecuencia o FBO (*Frequency band occupancy*):** La ocupación de toda una banda mide cada frecuencia y calcula la ocupación media para la banda completa. El número de frecuencias medidas determinado por la resolución de la frecuencia es normalmente mayor que el número de canales utilizables en una banda. Al igual que para el FCO, si el tiempo entre medidas se considera constante el FBO es calculado como sigue:

$$FBO = \frac{N_o}{N} \tag{2.3}$$

siendo N_o el número de muestras medidas en toda la banda con un nivel por encima del umbral y N el número total de medidas durante el tiempo de integración.

Si la resolución de frecuencia de una medición de ocupación de la banda es muy alta, el valor del FBO suele ser muy inferior a los valores del FCO de los canales de esta banda. Si ponemos un ejemplo en el que la frecuencia de partida es $F_{start} = 1$ MHz y la frecuencia final del ancho de banda es $F_{stop} = 2$ MHz mientras la resolución de frecuencia es de $\Delta F = 1$ KHz se puede obtener el número de frecuencias medidas por:

$$N_F = \frac{F_{stop} - F_{start}}{\Delta F} = 1000 \tag{2.4}$$

La resolución de frecuencia se entiende como la anchura de paso en la medición de una banda de frecuencias que debe ser menor que la separación entre canales. La separación habitual de canales es de 25 kHz, por lo que la banda medida cubre 40 canales utilizables. Si suponemos que constantemente 20 canales están ocupados y que el ancho de banda de cada transmisión es de 4 kHz, el número de muestras sobre el nivel umbral es de $20 \times 4 = 80$ resultando en una ocupación de la banda de frecuencia de $80/1000 = 0.08$ o del 8 %.

- **Ocupación del recurso de espectro o SRO (*Spectrum resource occupancy*):** La ocupación de los recursos del espectro es la relación entre el número de canales en uso y el número total de canales en toda una banda de frecuencias. Cuando solo un canal es usado, el SRO es igual al FCO.

2.2.2. Parámetros a considerar en la medición del espectro

Uno de los aspectos más críticos cuando se miden múltiples canales o bandas de frecuencia es separar las emisiones de los canales adyacentes, incluso cuando su nivel es bastante diferente. Si la anchura de banda de medición es demasiado amplia, una emisión fuerte hace que los canales vecinos aparezcan también ocupados como se muestra en la Figura 2.3(b). Ambas gráficas, (a) y (b), muestran ejemplos de la dinámica de una señal RF dividido el ancho de banda en cinco canales adyacentes. Las líneas horizontales negras representan el nivel del canal después de la evaluación. En el primer caso mostrado en la Figura 2.3(a), tanto el canal 2 como el 4 están ocupados ya que sobrepasan el valor umbral mientras que el resto no lo hacen resultando de un ajuste del ancho de banda correcto. Sin embargo, en gráfica de la Figura 2.3(b) la señal en el canal 4 es fuerte

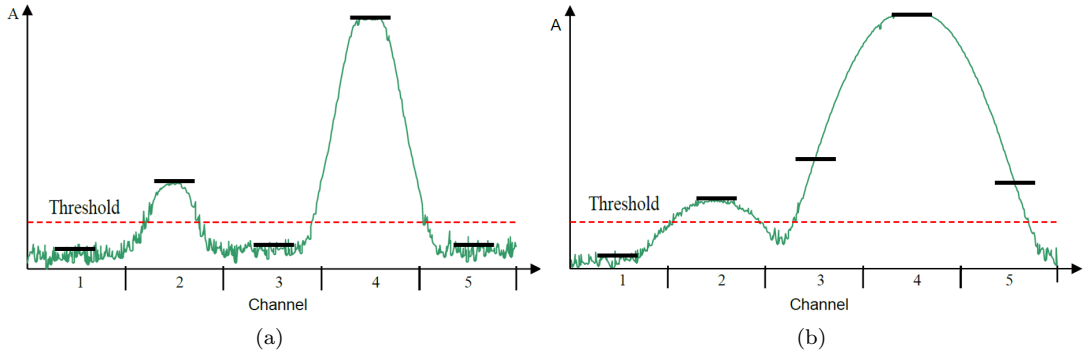


Figura 2.3: Definición del rango de frecuencias para múltiples canales adyacentes en un ancho de banda completo a estudiar. Se muestra en (a) un correcto ajuste y separación entre canales mientras que en (b) la anchura de banda de medición es demasiado amplia [2].

produciendo ocupaciones "fantasmas" o falsas en los canales 3 y 5 mientras que la ocupación del canal 2 sigue mostrándose correctamente. Es por esto que la resolución en frecuencia del equipo de medición debe ser más precisa o estrecha que la separación de los canales.

Otro parámetro crítico de un sistema de medición de la ocupación es el rango dinámico. Por un lado, debe ser lo suficientemente sensible para detectar señales débiles; por otro lado, tiene que hacer frente a señales muy fuertes de transmisores cercanos. Cuando se determina la atenuación o amplificación de RF adecuada en el sistema de medición y los lugares de medición, debe tenerse cuidado de evitar la sobrecarga del receptor durante la medición. La sobrecarga a menudo resulta en un aumento considerable del nivel de ruido. Dependiendo del ajuste del umbral, esto puede dar lugar a mediciones de la ocupación falsas. En la Figura 2.4 se observa la misma situación de ocupación que en la Figura 2.3(a), solo que el alto nivel de la emisión en el canal 4 lleva al equipo de medición a la sobrecarga. El efecto es que los cinco canales aparecen como ocupados ya que están por encima del nivel umbral. En este caso ni siquiera podría resolverse el problema elevando el umbral porque la ocupación real en el canal 2 desaparecería.

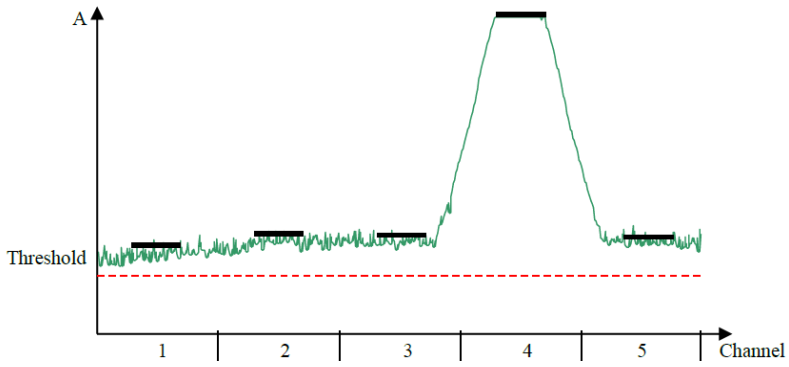


Figura 2.4: Sobrecarga del receptor durante la medición de la señal RF por un alto nivel de ocupación en el canal 4 [2].

También debe considerarse la relación señal ruido o SNR (*Signal to noise ratio*) de las señales detectables a tener en cuenta en la medición de estas. Para este fin puede suponerse la relación S/N mínima de 20 dB para las comunicaciones analógicas de banda estrecha (por ejemplo, redes privadas), de 40 dB para las comunicaciones analógicas de banda ancha (por ejemplo, radiodifusión en FM) o de 15 dB para sistemas digitales.

2.2.3. Dispositivos para la medida del espectro de ocupación

A medida que se van adecuando los sistemas a los requerimientos del sector de las comunicaciones, van apareciendo "administradores" del espectro que proporcionan un uso más ideal de él. Uno de ellos son los sistemas de radiocomunicaciones auto-organizados (o *Self-organizing radio systems*) los cuales no tienen una frecuencia única y/o fija con la que operan. En su lugar, pueden sensar la ocupación actual dentro de una cierta banda de frecuencia y seleccionar automáticamente una frecuencia que esté actualmente libre. Un ejemplo de este comportamiento es el teléfono personal DECT (*Digital Enhanced Cordless Telecomunications*), es decir, dispositivo que sigue al estándar europeo para las comunicaciones inalámbricas para alcance de entre 50 y 300 metros.

Otros de interés son los sistemas de radio con agilidad de frecuencia (o *Frequency agile radio systems*) los cuales cambian de frecuencia muy rápidamente basándose en un esquema fijo o incluso flexible pudiendo poner de ejemplo el Bluetooth. Los sistemas de medición de la ocupación estándar no suelen ser lo suficientemente rápidos para captar cada pequeña ráfaga y pueden considerar que se utiliza toda la banda de frecuencias aunque solo esté activa una estación.

Por otro lado se encuentran los sistemas digitales pulsados (o *Pulsed digital systems*) que usan métodos de acceso múltiple por división de tiempo TDMA (*Time Division Multiple Access*) o dicho de otro modo, permiten la transmisión de señales digitales mediante ráfagas ocupando un único canal de transmisión a partir de distintas fuentes. Esto supone una mejor forma de aprovechar el medio de transmisión. Incluso si un sistema de medición de la ocupación fuera lo suficientemente rápido para captar cada una de las ráfagas, sigue planteándose la cuestión de cómo se define la ocupación en este caso: ¿Debe considerarse que la frecuencia está ocupada solo porque se utiliza una franja horaria, o debe indicarse como "disponible" el tiempo que transcurre entre las ráfagas? [2].

2.3. Inteligencia artificial para sistemas de comunicación inalámbrica

La inteligencia artificial o, como la llama Poole en 1998 [3], inteligencia computacional es el estudio de un sistema que actúa de forma apropiada para ciertas circunstancias y en base a su objetivo, es flexible a los entornos cambiantes y a los diferentes objetivos, aprende de la experiencia y toma las decisiones apropiadas dadas las limitaciones perceptivas y la computación finita. El objetivo científico central de la inteligencia computacional es entender los principios que hacen posible el comportamiento inteligente, en sistemas naturales o artificiales imitando las funciones "cognitivas" del cerebro humano, como por ejemplo "observar", "aprender" y "aportar soluciones". Según Poole, para entender cómo el comportamiento inteligente podría ser algorítmico se se debe desarrollar una teoría que explique cómo ese comportamiento puede manifestarse en una máquina, y luego mostrar la viabilidad de esa teoría construyendo una implementación. Un sistema inteligente podría ser, por ejemplo, un acoplamiento de un motor de cómputo con actuadores y sensores físicos, llamado robot o podría ser un programa que actúa en un entorno puramente computacional, un *infobot*. Estos sistemas inteligentes reciben unas entradas que les proporcionan experiencia pasada de la que pueden aprender, conocimiento previo sobre el mundo, los objetivos que debe perseguir y observaciones sobre el entorno actual y sobre sí mismo. Como salida, este sistema inteligente, o agente como especifica Poole, da una acción respecto a dichas entradas. El entorno del agente puede incluir otros agentes y cada uno tiene un estado interno propio que percibe, razona y aprende sobre su entorno y sobre sí mismo. esta es una visión global de los sistemas inteligentes que varían en complejidad, desde un simple termostato hasta un equipo de robots móviles, pasando por un sistema de asesoramiento de diagnóstico cuyas percepciones y acciones son mediadas por los seres humanos.

El éxito en la construcción de un agente inteligente depende naturalmente del problema a investigar. Los sistemas AI forman parte hoy en día del avance en los campos de la economía, la medicina, la ingeniería o el transporte entre muchos otros y se ha usado en gran variedad de apli-

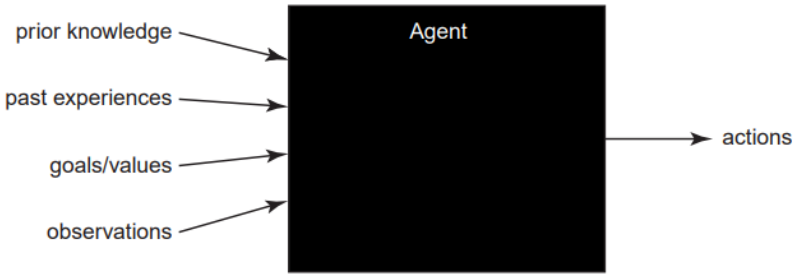


Figura 2.5: Esquema representativo de un sistema inteligente como una caja negra que recibe información del entorno y actúa frente a esta [3].

caciones como el reconocimiento de voz o control de sistemas industriales. Definir el problema a tratar es uno de los pasos más importantes antes de aplicar cualquier método de inteligencia. Para direccionar mejor la solución a aportar, es necesario distinguir dos grandes áreas de la inteligencia artificial: el aprendizaje automático o *Machine Learning* (ML) y el aprendizaje profundo o *Deep Learning* (DL). En ambos, los ingenieros utilizan herramientas de software, como por ejemplo MATLAB, para que las computadoras puedan identificar las tendencias y características de los datos aprendiendo de un ejemplo de conjunto de datos [23]. En el caso del aprendizaje automático, los datos se utilizan para construir un modelo que la computadora puede utilizar para tratar los próximos datos de entrada y, en última instancia, los datos del mundo real. Tradicionalmente, un paso importante en este flujo de trabajo es la definición manual de las características, o métricas adicionales, que ayudan a que el modelo sea más preciso. Por otro lado, el aprendizaje profundo es considerado en muchos estudios como un subconjunto del aprendizaje automático (ver Figura 2.6[1]). Se caracteriza en que los ingenieros y científicos se saltan el paso manual de crear características. En cambio, los datos se introducen en el algoritmo de aprendizaje profundo y este aprende automáticamente qué características son más útiles para determinar la salida. Además, es modelado directamente como el sistema neurológico humano.

El aprendizaje automático o ML se utiliza típicamente en aplicaciones que implican la predicción de un resultado o el descubrimiento de tendencias. En estos ejemplos, se utiliza un conjunto limitado de datos para ayudar a las máquinas a aprender patrones que puedan utilizar más tarde para hacer una determinación correcta sobre los nuevos datos de entrada. Entre los algoritmos comunes utilizados en el aprendizaje automático figuran la regresión lineal, los árboles de decisión, las máquinas de vectores de apoyo (*Support Vector Machines* o SVM), el análisis discriminatorio, las redes neuronales y los métodos de conjunto. Sin embargo el aprendizaje profundo o DL es más complejo y se suele utilizar para proyectos que implican la clasificación de imágenes, la identificación de objetos en imágenes y la mejora de imágenes y señales. En estos casos, se puede aplicar una red neuronal profunda (o *Deep Learning neural networks* (DNN)), ya que están diseñados para extraer automáticamente características de datos organizados espacial y temporalmente, como imágenes y señales. Los algoritmos comunes utilizados en el aprendizaje profundo incluyen las redes neuronales convolucionales (CNN), las redes neuronales recurrentes (RNN) y las redes Q profundas.

Los algoritmos de aprendizaje automático pueden ser más apropiados si se necesitan resultados más rápidos y que requieran menos potencia de cálculo (por ejemplo, las CPU de escritorio son suficientes para entrenar estos modelos). Los ingenieros que aplican *Machine Learning* deben esperar pasar la mayor parte de su tiempo desarrollando y evaluando características para mejorar la precisión del modelo. Por otro lado, los modelos de *Deep Learning* tomarán más tiempo para entrenar. En general, pueden tardar desde un minuto hasta unas pocas semanas en entrenarse, dependiendo de su hardware y su potencia de cálculo. Los científicos que usan el aprendizaje profundo deben esperar pasar la mayor parte de su tiempo entrenando modelos y haciendo modificaciones en la arquitectura de su red neuronal. En el caso de los modelos de DL, normalmente se requiere un hardware especializado ya que es necesario mayores requisitos de memoria y computación. Debido a que supone altos costos en GPUs, la formación de modelos de aprendizaje profundo en clusters

[1]Imagen tomada de la página web de Digital Guide IONOS: https://www.ionos.es/digitalguide/online-marketing/marketing-para-motores-de-busqueda/deep-learning-vs-machine-learning/

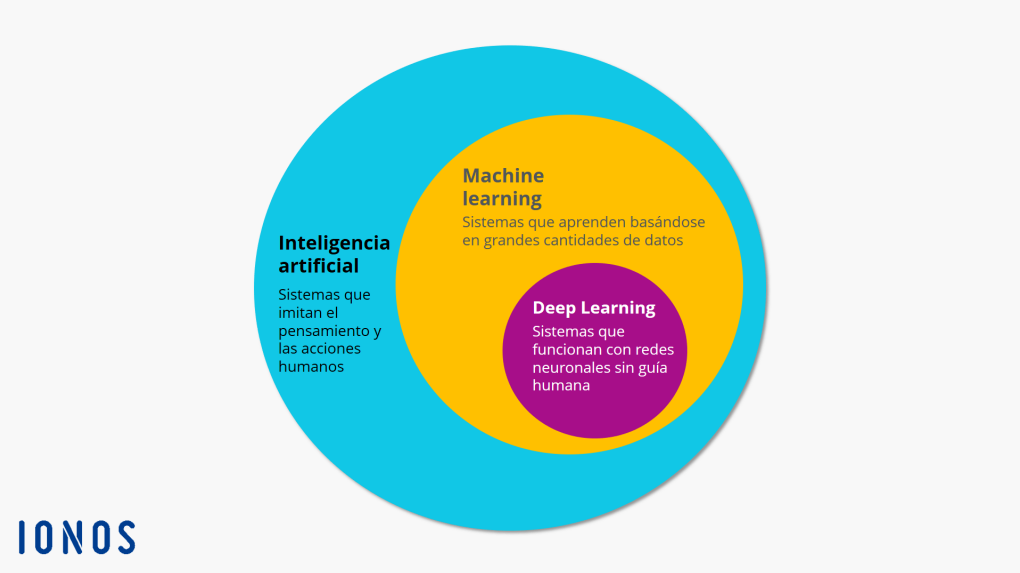


Figura 2.6: Esquema conceptual del aprendizaje automático como una subcategoría de la inteligencia artificial. A su vez, el aprendizaje profundo se considera una forma de aprendizaje automático.

o en la nube ha ganado popularidad.

La comprensión del conjunto de datos disponibles puede ayudar a elegir cuál de los dos aprendizajes debe aplicarse a una tarea determinada. Por lo general, el aprendizaje automático se utiliza cuando se dispone de datos más limitados y estructurados. La mayoría de los algoritmos de aprendizaje automático están diseñados para entrenar a los modelos con datos tabulares (organizados en filas y columnas independientes). Si los datos no son tabulares se puede seguir aplicando el aprendizaje automático pero con una cierta manipulación de los datos, es decir, los datos de los sensores deben convertirse en una representación tabular extrayendo características utilizando las métricas estadísticas comunes (media, mediana, desviación estándar, asimetría, curtosis, etc.), y luego utilizarlas con las técnicas tradicionales de aprendizaje automático. El aprendizaje profundo suele requerir una gran cantidad de datos para garantizar que la red funcione. Las redes neuronales convolucionales están diseñadas para funcionar con datos de imágenes, aunque también pueden utilizarse con datos de sensores realizando un cálculo de tiempo-frecuencia como un espectrograma en la señal. Las redes neuronales recurrentes están diseñadas para operar con datos secuenciales como señales y texto [23]. Para el caso que concierne este trabajo, y como ya se ha mencionado anteriormente, se desea predecir la evolución futura en el tiempo de las ondas electromagnéticas que atraviesan el espectro en un ancho de banda concreto. Dicho de otro modo, nuestros datos de entrada serán datos secuenciales que no están tabulados tal como las técnicas de aprendizaje automatico requieren. Por ello, este trabajo se encamina en el uso de las redes neuronales profundas usando la subcategoría *Deep Learning* de la inteligencia artificial.

2.3.1. El problema de aprendizaje para la predicción del espectro electromagnético

En general, los problemas de aprendizaje pueden ser clasificados en dos categorías: aprendizaje supervisado y no supervisado. En el aprendizaje supervisado, el conjunto de datos para el entrenamiento de la red neuronal donde esta aprende el modelo son ya conocidos, es decir, se sabe cual es la salida correcta frente a las entradas obteniendo así una relación entre las entradas y las salidas. Dicho de otro modo, teniendo una serie de valores de entrada, X, y de salida, H, el algoritmo de aprendizaje trata de encontrar el modelo o función que las relacione:

$$H = f(X) \tag{2.5}$$

La mayoría de los problemas de ML usan el aprendizaje. El objetivo es aproximarse tanto como sea posible a la relación real que existe entre los valores de X e H y así, cuando se tienen nuevos datos de entrada X (datos de validación o test), se pueden predecir las variables de salida H acorde a estos. Se denomina aprendizaje supervisado porque el proceso de aprendizaje de un algoritmo a partir del conjunto de datos de entrenamiento puede considerarse como un maestro que supervisa el proceso. Sabemos las respuestas correctas; el algoritmo hace predicciones de forma iterativa sobre los datos de entrenamiento y se corrige haciendo actualizaciones. El aprendizaje se detiene cuando el algoritmo alcanza un nivel de rendimiento aceptable o un mínimo error establecido. A su vez, un aprendizaje supervisado puede ser empleado en dos tipos de problemas: problema de clasificación o de regresión. Un problema de regresión trata de predecir un valor real como un precio o una magnitud física. Un ejemplo es predecir el precio de una casa en función de un conjunto de datos que relacionan el tamaño de las casas con su precio asignado. Esta relación es cumple una función continua (lineal o no lineal) por lo que el precio puede tomar un valor concreto dentro de un rango definido por el *dataset*. Un problema de clasificación se entiende como la predicción de una salida discreta o categoría discreta. Un ejemplo común es el de calificación de los objetos en una imagen, un objeto será algo concreto como un gato o un perro pero la salida no puede ser la media entre un gato y un perro ya que esto no existe. Por otra parte, un aprendizaje no supervisado permite aproximar la solución teniendo poca o ninguna información de cómo son las salidas de los datos de entrenamiento. Esto es similar al aprendizaje de un nuevo idioma por inmersión en el país donde se habla ese idioma. Inicialmente es muy difícil reconocer los sonidos del idioma, o separar las palabras individuales. Sin embargo, esta habilidad puede ser adquirida sin la ayuda de un profesor [24].

Teniendo en cuenta que el sistema de este proyecto obtendrá la evolución de la ocupación de la onda electromagnética durante un periodo de tiempo para definir el conjunto de datos de entrenamiento (o *dataset*), es fácil pensar que puede tratarse como un aprendizaje supervisado. Sin embargo, el problema de la predicción de ocupación puede ser visto como una colección de observaciones de una única variable recogidas de forma secuencial en el tiempo. Esto es conocido como predicción en series de tiempo o en inglés *Time Series Forecasting* [25]. Aún así, se puede afirmar que la previsión de series temporales puede enmarcarse como un problema de aprendizaje supervisado. Si se tiene un conjunto de datos de series temporales ordenados, claro está, cronológicamente, se puede usar los valores en el instante de tiempo anterior como variable de entrada X y utilizar el valor actual como variable de salida H. Si por ejemplo se tiene un *dataset* como el mostrado en el cuadro 2.1 y se reorganiza el conjunto de datos como se ha comentado, se obtendría el conjunto de datos mostrados en el cuadro 2.2. La primera fila de este último cuadro será eliminada al no poder usarla por la falta del valor de X al igual que la última fila que carecerá del valor de salida H. La predicción de series temporales se detalla más explícitamente en la siguiente sección. Además, la salida que se espera es el valor de ocupación de la banda pudiendo tomar un valor dentro del rango entrenado de valores por lo que nos enfrentamos a un problema de regresión. El análisis de regresión trata de encontrar parámetros para el modelo que minimicen el error entre los datos de salida conocidos y los datos obtenidos gracias al modelo aprendido, hasta que los resultados se consideren aceptables [26].

Pasos de Tiempo	Variable observada
1	84.5
2	93.8
3	79.2
...	...

Cuadro 2.1: Ejemplo de un *dataset* de un problema de series temporales.

X	H
?	84.5
84.5	93.8
93.8	79.2
79.2	...

Cuadro 2.2: Ejemplo de un *dataset* de un problema de series temporales convertidos para aplicar aprendizaje supervisado.

2.3.2. Predicción de series temporales

A diferencia de los problemas más simples de clasificación y regresión, los problemas de las series temporales añaden la complejidad de la dependencia temporal entre las observaciones [27]. La característica fundamental de las series temporales es que las observaciones sucesivas no son independientes entre sí, y el análisis debe llevarse a cabo teniendo en cuenta el orden temporal de las observaciones resultado ser de una dificultad mayor que otros análisis. Los métodos estadísticos basados en la independencia de las observaciones no son válidos para el análisis de series temporales porque las observaciones en un instante de tiempo dependen de los valores de la serie en el pasado. Aunque las series temporales aparecen en numerosos campos de aplicación, el análisis para las series temporales se diseñó específicamente para manejar las demandas de datos de comercio financiero. La economía representa multitud de mercados diferentes. En algunos, como el de los productos básicos, los precios proporcionados por las bolsas son los valores reales de las transacciones realizadas mientras que en el mercado de divisas o *Foreign Exchange* (FOREX) los datos suministrados no son el precio al que se llegó a un acuerdo, sino el precio al que un agente de bolsa está dispuesto a negociar. Los corredores vigilan constantemente los mercados y publican nuevos precios electrónicamente cuando los precios existentes están desfasados. En ambos casos los datos representan acontecimientos discretos que pueden ocurrir hasta 30 veces por segundo en las horas punta.

El objetivo fundamental del estudio de las series temporales es el conocimiento del comportamiento de una variable a través del tiempo para, a partir de dicho conocimiento realizar predicciones, es decir, determinar qué valor tomará la variable objeto de estudio en uno o más períodos de tiempo futuro, mediante la aplicación de un determinado modelo calculado previamente. Si se extrapola a la aplicación en economía, dado que en la mayor parte de los problemas económicos los agentes se enfrentan a una toma de decisiones bajo un contexto de incertidumbre, la predicción de una variable reviste una importancia notoria para que el agente que la realiza tome decisiones en base a una reducción de la incertidumbre y, por ende, una mejora de sus resultados.

Una serie temporal puede ser discreta o continua dependiendo de cómo las observaciones sean medidas en el tiempo. Si los datos se recogen en instantes temporales de forma continua, se debe o bien digitalizar la serie, es decir, recoger solo los valores en instantes de tiempo equiespaciados, o acumular los valores sobre intervalos de tiempo. Por otra parte, si se pueden predecir exactamente los valores, se dice que las series son *deterministas* mientras que si solo se puede determinar de modo parcial por las observaciones pasadas y no se pueden determinar exactamente, se considera que los futuros valores tienen una distribución de probabilidad que está condicionada a los valores pasados. Las series entonces son denominadas como *estocásticas*. Para series deterministas las predicciones son más o menos fiables, con un grado de error bajo. Por el contrario, si la serie es completamente aleatoria, las predicciones carecerán de validez por completo. Generalmente, no existen variables deterministas o aleatorias puras, sino que contienen ambos tipos de elementos. El estudio descriptivo de series temporales se basa en la idea de descomponer la variación de una serie en varias componentes básicas. Este enfoque no siempre resulta ser el más adecuado, pero es interesante cuando en la serie se observa cierta tendencia o cierta periodicidad. Este enfoque descriptivo consiste en encontrar componentes que correspondan a una tendencia a largo plazo, un comportamiento estacional y una parte aleatoria. Las fuentes de variación que se consideran son:

1. **La tendencia:** Puede definirse como un cambio a largo plazo que se produce en relación

al nivel medio, o el cambio a largo plazo de la media. La tendencia se identifica con un movimiento suave de la serie a largo plazo.

2. **El efecto estacional:** Muchas series temporales presentan cierta periodicidad o comportamiento similar cada cierto periodo de tiempo.
3. **Una componente aleatoria:** Existe una componente residual con comportamiento aleatorio.

De las tres componentes reseñadas, las dos primeras son componentes deterministas, mientras que la última es aleatoria. Así, se puede denotar que

$$X_t = T_t + E_t + I_t \tag{2.6}$$

donde T_t es la tendencia, E_t es la componente estacional e I_t es el ruido o parte aleatoria. En el tratamiento de series temporales que se va a abordar, únicamente considerará la información presente y pasada de la variable investigada, es decir, es univariable. Si la variable investigada es X y se dispone de los valores que toma dicha variable desde el momento inicial hasta t, el conjunto de información disponible vendrá dado por: X_1,X_2,X_3, ...,X_t.

Tradicionalmente, la previsión de series temporales ha estado dominada por métodos estocásticos lineales o paramétricos como el ARIMA (*Autoregressive Integrated Moving Average*) ya que son bien comprendidos y eficaces en muchos problemas. Pero estos métodos clásicos también tienen algunas limitaciones, tales como centrarse en los datos completos sin distinguir si existe información errónea o innecesaria, asumir una relación lineal excluyendo distribuciones más complejas, enfocarse en una dependencia temporal fija, tomar como entrada una única variable (muchos problemas del mundo real tienen múltiples variables de entrada) o realizar pronósticos de un solo paso de tiempo futuro. Es decir, si el modelo paramétrico no es el adecuado al análisis de datos que estamos realizando, puede llevar a conclusiones que queden muy alejadas de la realidad, dado que el modelo paramétrico conlleva un grado de exactitud en las afirmaciones que de él se derivan y que son adecuadas siempre y cuando se cumplan los supuestos básicos sobre los que se apoya su construcción teórica. De hecho, los modelos paramétricos presentan una estructura teórica tan rígida que no pueden adaptarse a muchos conjuntos de datos de los que hoy día se disponen, por ejemplo, para el análisis económico. La econometría no paramétrica aparece como consecuencia de intentos por solucionar problemas que existen en la econometría paramétrica llevando a los investigadores a utilizar formas funcionales flexibles para aproximarse a relaciones desconocidas entre las variables. La econometría no paramétrica no parte de supuestos sobre la distribución de probabilidad de las variables bajo estudio, sino que trata de estimar dicha distribución para encontrar la media condicional y los momentos de orden superior (por ejemplo, la varianza) de la variable de interés. Una de las desventajas de este método es la necesidad de contar con muestras muy grandes si es que se desea estimar la función de relación entre ambas variables de manera precisa. Además el tamaño de la muestra debe aumentar considerablemente conforme aumenta el número de variables involucradas en la relación. Una vez establecido el modelo, el paso siguiente consiste en estimarlo (o ajustarlo) a partir de las el número observaciones disponibles. Es decir, hay que construir un estimador de la función de regresión y un estimador de la varianza del error [28].

En numerosos estudios se aportan las redes neuronales de aprendizaje profundo como solución para aprender automáticamente el comportamiento complejo y arbitrario de las entradas a las salidas y apoyar múltiples entradas y salidas. estas son características poderosas que ofrecen una gran promesa para la predicción de series temporales, particularmente en problemas con dependencias complejas-no lineales, entradas multivariables y predicciones de más de un paso de tiempo futuro (o *multi-step*). Estas características, junto con las capacidades de las redes neuronales más modernas, pueden ser muy prometedoras.

2.3.3. Redes Neuronales para la predicción de series temporales.

Las capacidades de las redes neuronales de aprendizaje profundo o DNNs sugieren un buen ajuste para la predicción de series temporales. Técnicamente, el contexto disponible de la secuen-

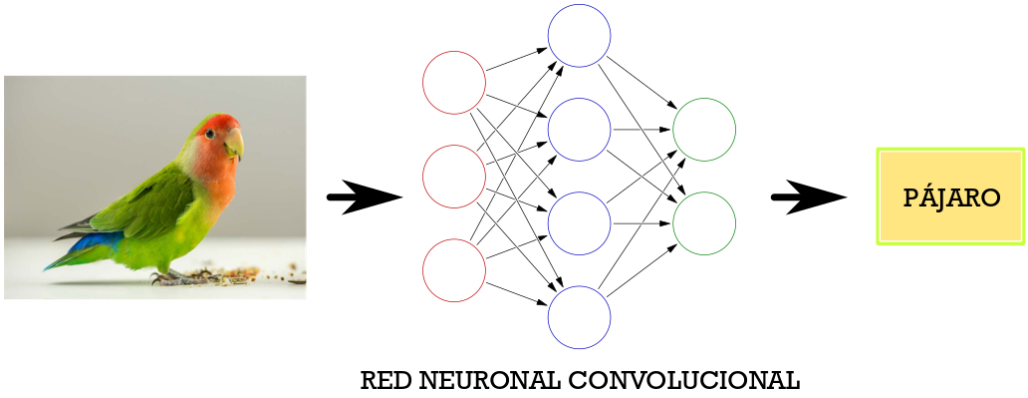


Figura 2.7: Redes neuronales convolucionales para el reconocimiento de objetos en imágenes.

cia proporcionada como entrada puede permitir que los modelos de redes neuronales aprendan directamente tanto la tendencia como la estacionalidad. Las redes neuronales más simples, como el Perceptrón Multicapa o MLP (*Multilayer Perceptron*), se aproximan a una función de mapeo de las variables de entrada a las de salida. Esta capacidad, en general, es valiosa para las series temporales por varias razones. Una de estas razones es que las redes neuronales son robustas al ruido en los datos de entrada e incluso pueden apoyar el aprendizaje y la predicción en presencia de valores perdidos o erróneos. Otro motivo es que las redes neuronales no hacen fuertes suposiciones sobre la función que relaciona las entradas con las salidas debido a que son capaces de aprender fácilmente las relaciones lineales y no lineales por lo que no se alejan de la realidad del problema. En [29] (publicado en 1996) se afirma que las redes contribuyen en gran medida en el procesamiento de series temporales y afirma que promete aplicaciones poderosas en el campo de la predicción. Más específicamente, las redes neuronales también pueden ser configuradas para soportar un número arbitrario definido pero fijo de entradas y salidas. Esto significa que las redes neuronales pueden apoyar entradas y predicciones de más de una variable.

En multitud de estudios e investigaciones [30] [31] se presentan las redes neuronales convolucionales (CNN) para problemas de clasificación como por ejemplo para reconocer el objeto de una imagen (Figura 2.7). Es más, estas redes fueron diseñadas para manejar eficientemente los datos de las imágenes pero también se ha demostrado que son eficaces en otros problemas como para la localización de objetos o el subtitulado de imágenes entre otros. Lo logran operando directamente sobre datos en bruto, como por ejemplo en los píxeles en bruto, en lugar de características específicas de los datos. El modelo aprende entonces a extraer automáticamente de los datos las características que son directamente útiles para el problema mientras aseguran un cierto grado de invariabilidad de distorsión.

La capacidad de las CNN para aprender y extraer automáticamente características de los datos de entrada en bruto puede aplicarse a los problemas de predicción de las series temporales [27]. Una secuencia de observaciones puede ser tratada como una imagen unidimensional donde un modelo de CNN puede leer y simplificar en los elementos más destacados. Esta capacidad de las CNN ha sido demostrada en tareas de clasificación de series temporales como la detección automática de actividades humanas basada en datos proporcionados en bruto por sensores de aceleración incorporados en *smartphones* [32]. Sin embargo, las redes CNN están limitadas puesto que cuando la entrada al sistema es una secuencia de datos y se requiere una secuencia de salida obtenida mediante un problema de regresión, como por ejemplo continuar con un vídeo o una canción, estas estructuras no pueden ser usadas. Esto ocurre por dos razones. Una de ellas es que la estructura de estas redes están diseñadas para que los datos de entrada y salida siempre tengan el mismo tamaño mientras que un vídeo o canción se caracterizan por ser un tipo de dato de tamaño variable. En el caso de un vídeo podemos decir que este puede contener un número concreto de imágenes o *frames* mientras que una canción puede estar compuesta por más o menos palabras. El segundo motivo, y no por ello menos importante, es que los datos secuenciales, como las imágenes en los vídeos o las palabras en las canciones, están correlacionadas queriendo decir que el dato actual tiene relación con los datos anteriores y a su vez, el dato posterior también estará condicionado

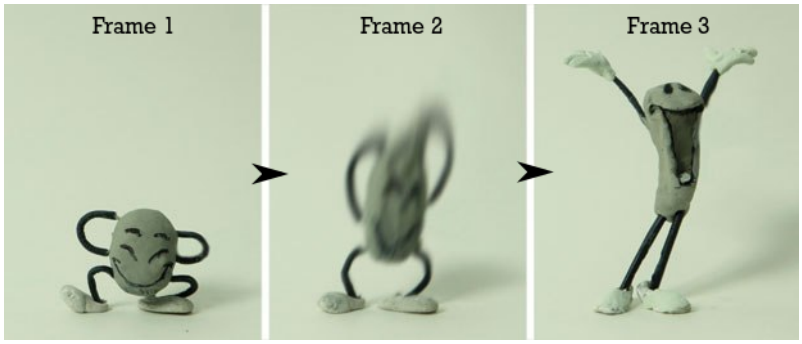


Figura 2.8: Imágenes o *frames* correlacionados en una secuencia de vídeo.

por este. En la Figura 2.8 se observa como el personaje se levanta tras haber estado agachado mediante únicamente tres imágenes. La segunda de ellas indica un movimiento ascendente ya que en la imagen anterior, el personaje estaba agachado y en la última imagen está totalmente de pie. Otro ejemplo sería la de comprender un poema, el cerebro humano concatena las palabras para entender el mensaje a transmitir pero no es posible hacerlo si se procesara cada palabra de forma individual. Así, se va a referir como secuencia a un conjunto de datos que siguen un orden especifico y que solamente tienen significado en conjunto y no de manera individual. Las redes CNN no son capaces de relacionar los datos siendo esta la gran diferencia con respecto a las redes neuronales recurrentes. Las redes neuronales recurrentes o RNN añaden el aprendizaje de la dependencia entre observaciones y además pueden procesar secuencias a la entrada y salida sin importar el tamaño de estas, a diferencia de MLP o CNN [27]. Por este motivo las RNN son más apropiadas para la predicción de señales dinámicas como el caso que presentamos en este proyecto. Pese a ello se distinguen varios usos de RNN si nos referimos a la entrada y salida. Una de ellas es conocida como *one-to-many* que se refiere a una entrada de un solo dato mientras que la salida es una secuencia, otra sería la opuesta llamada *many-to-one* pudiéndose ser, por ejemplo, la entrada una película y la salida una categoría como podría ser "Comedia". También existe *many-to-many* donde tanto la entrada como la salida son secuencias como por ejemplo el traductor de Google [33].

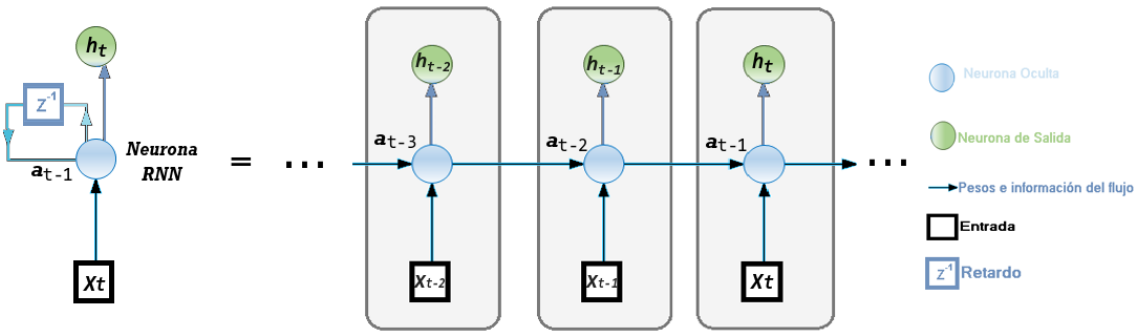


Figura 2.9: Las RNN operando en el tiempo.

Las RNN usan la recurrencia, es decir, no solo tienen como entrada la variable X sino también la activación $\mathbf{a}$ obtenida en la iteración anterior para hallar la salida H. En otras palabras, las RNN usan un tipo de memoria que permite que la información persista para generar la salida deseada que está en función de la información anterior. Aunque la arquitectura de las RNN es muy similar a las redes convencionales, estas incorporan la activación o estado oculto (en inglés *hidden state*) que es precisamente la memoria mediante la cual se comparte información entre los pasos de tiempo. El estado oculto (denotado por $\mathbf{a}$) puede almacenar información como representaciones distribuidas de alta dimensión y su dinámica no lineal puede implementar grandes cálculos para realizar tareas de modelado y predicción para secuencias complejas. Para ello se necesita una etapa de realimentación en la estructura de la red recurrente. Además, es importante decir que operan con el tiempo ya que en cada paso de tiempo o iteración, aceptan un vector de entrada y

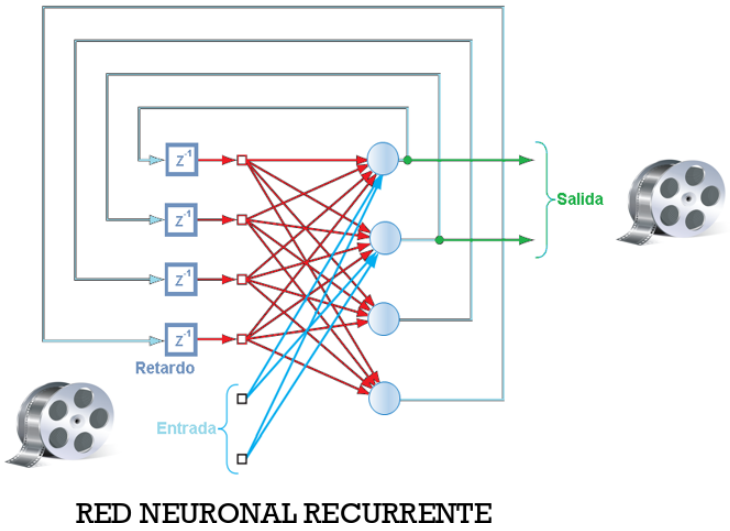


Figura 2.10: RNN *many-to-many* con capa oculta y totalmente recurrente.

actualiza su estado oculto a través de funciones de activación no lineales. La Figura 2.9 muestra el comportamiento temporal de la red desenrollado pero esto es solo una representación. Para el ejemplo de tres imágenes consecutivas dado en la Figura 2.8, cada *frame* se corresponde a un instante de tiempo.

Estas redes se caracterizan por la topología de la red o conexión entre las neuronas, el tipo de neuronas y el algoritmo de aprendizaje empleado para adaptar su función de calculo a las necesidades del problema. Empezando por la topología de la red, y sabiendo que al menos debe incorporar un circuito de realimentación, las RNN pueden incorporar capas ocultas de neuronas como la de la Figura 2.10. Las ramas se encuentran compuestas de retardo de tiempo (denominados Z^{-1}), esto resulta en una dinámica no lineal.

Por otro lado y viendo las entradas y salidas que componen las neuronas RNN (ver Figura 2.11), el funcionamiento de estas puede ser resumido por cómo la predicción o salida de dicha neurona h_t y la activación de la misma a_t son obtenidas mediante las siguientes expresiones:

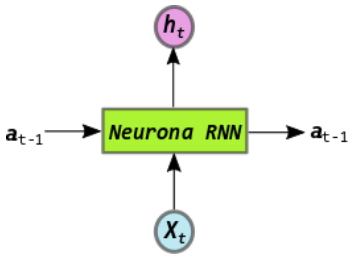


Figura 2.11: Arquitectura realimentada de las RNN con dos entradas y dos salidas.

$$a_t = f(W_a \cdot a_{t-1}) + W_x \cdot X_t + b_a) \tag{2.7}$$
$$h_t = g(W_h \cdot a_t + b_h) \tag{2.8}$$

donde los parámetros W_a, W_x, W_h, b_a y b_h son calculados durante el entrenamiento de la red neuronal y que tras este, estos valores serán fijos durante todos los pasos de tiempo. Tanto f como g indican las funciones de activación para el estado oculto a y para la salida predicha h respectivamente [30]. La estructura de las neuronas RNN estándar es muy simple ya que solo incorporan una capa que efectúa una función de activación hiperbólica tangencial (ver Figura

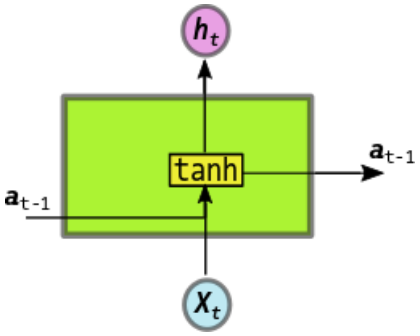


Figura 2.12: Arquitectura de las neuronas RNN estándar.

2.12). Es decir, las dos funciones f y g pueden ser sustituidas por la función *tanh* en las ecuaciones 2.7 y 2.8.

Como puede observarse, la predicción h_t depende de la activación actual a_t que a su vez depende de la activación en el instante anterior a_{t-1} y la entrada actual x_t. Esto es precisamente la memoria de la red ya que permite preservar la información entre instantes de tiempo. Si se supone, por ejemplo, tres pasos de tiempo a predecir (h_{t-2}, h_{t-1}, h_t) (como se refleja en la Figura desglosada 2.9), la salida h_t estará en función del estado oculto anterior a_{t-1} que a su vez lo estará de a_{t-2} y este de a_{t-3}. Así y debido a la anidación de las funciones de activación *tanh*, la dependencia de la salida actual h_t se va desvaneciendo con respecto a estados ocultos cada vez más alejados en el tiempo (a_{t-n} donde n es muy grande). Este es el motivo por el que las RNN estándar se usan para secuencias cortas pero no son buenas para predicciones de largas secuencias y, para el caso de interés, de series temporales con muchos pasos de tiempo futuros. El problema de la predicción de las RNN estándar para largas secuencias se conoce como problema de desvanecimiento de gradiente o *vanish gradient* y que dificulta su entrenamiento mediante el método de aprendizaje basado en descenso estocástico de gradientes y de retropropagación. Este problema es detallado por Yoshua Bengio en 1994 [34] donde razona la dificultad del entrenamiento de las redes para largas dependencias.

En numerosos estudios como [27], [30] o [35] se propone un tipo especial de RNN como la mejor solución a este problema. Estas redes reciben el nombre inglés *Long Short-Term Memory* o LSTM y fueron introducidas por Hochreiter & Schmidhuber en 1997 [36]. En el siguiente capítulo se va a profundizar en por qué estas redes resuelven el problema de *vanish gradient*, en la estructura de sus unidades o neuronas y en su funcionamiento asociándola a los problemas de series temporales que preocupan para este estudio.

Capítulo 3

LSTM para la predicción de series temporales

Todas las redes neuronales recurrentes tienen una etapa de realimentación que emulan una cadena de módulos, celdas o neuronas repetitivas como ya se comentó en el capítulo anterior. Las LSTM, al ser un tipo de red recurrente, también tienen un bloque de memoria por lo que se puede definir como una cadena de celdas pero con una estructura diferente. Esta estructura de la celda *Long Short-Term Memory* está explícitamente diseñada para evitar el problema de la dependencia a largo plazo y así mitigar los errores producidos por el desvanecimiento de gradiente [30].

Las redes neuronales LSTM pueden utilizarse para llevar a cabo diversas tareas, como la predicción, la clasificación de patrones, los tipos de reconocimiento, el análisis e incluso la generación de secuencias. Debido a la capacidad de procesar datos secuenciales de las LSTM, estas son una herramienta antigua en muchos campos diferentes como la estadística, la lingüística, la medicina, el transporte, la informática y otros. Desde 1997 se han presentado numerosos trabajos en conferencias y revistas que han mostrado un cierto reconocimiento [33]. Además, estas redes se han llevado a la implementación *hardware* en muchas investigaciones pudiendo citar algunas de ellas [37][38][39].

3.1. Arquitectura de las redes LSTM

Las celdas o neuronas LSTM presentan una entrada y salida adicional a las RNN que se conoce como estado de la celda o *Cell State*. Esta variable es la clave del funcionamiento de las redes LSTM ya que es básicamente considerada como el corazón del bloque de memoria de esta red. En la Figura 3.1 se muestra una celda LSTM y se resalta el camino del estado de la celda como una unidad lineal auto-conectada también llamada "Carrusel de Error Constante" (CEC) [33]. Al esta unidad lineal se le puede tanto añadir como eliminar datos, tal y como en un cerebro humano. El objetivo de incorporar datos es para actualizar la información que se propaga por las celdas. Por otro lado suprimir información innecesaria evita la saturación de la salida predicha ya que el CEC tiene una naturaleza lineal que podría crecer ilimitadamente si no se eliminan dichos datos. Esto hace que se propague la información o dependencia entre los pasos de tiempo más relevante solventando el problema de las largas series de tiempo.

En las RNN estándar, este módulo de repetición tiene una estructura muy simple de una sola capa que realiza la función tangente hiperbólica a las entradas de esta celda (ver Figura 2.12). En lugar de tener una sola capa, las LSTM incorporan cuatro capas por cada celda (o neurona artificial) interactuando de una manera especial. Una celda común LSTM está compuesta por un bloque de memoria y tres puertas. La celda LSTM recuerda las dependencias o pesos *weights* entre los elementos de la secuencia de entrada X gracias al CEC mientras que las tres puertas

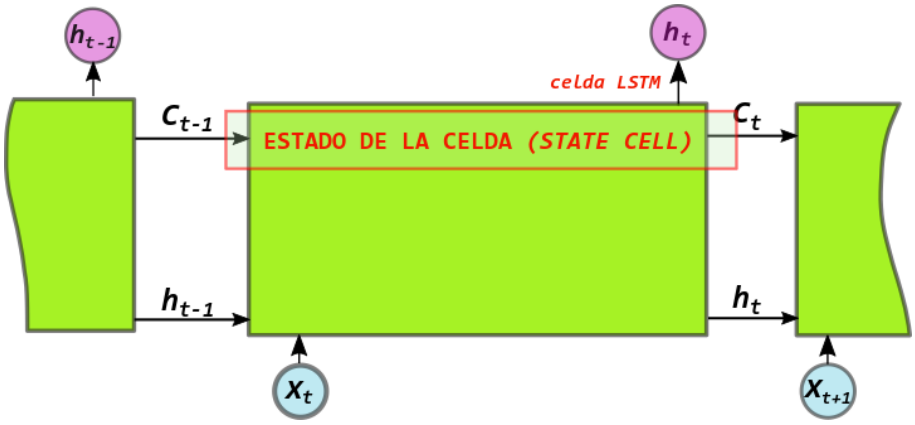


Figura 3.1: Diagrama de una cadena de celdas LSTM repetidas. El estado de la celda como solución a las largas dependencias.

regulan el flujo de información de la secuencia de entrada y salida hacia el estado de la celda C y, con ello, afecta a la predicción de la celda h. Estas puertas están formadas por una capa con función sigmoidal (es decir, la salida estará comprendida en el rango [0,1]) y una operación de multiplicación como se representa en la Figura 3.2.

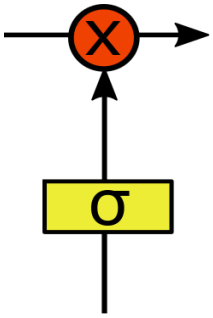


Figura 3.2: Estructuras de las tres puertas que incluye la arquitectura de una celda LSTM.

Las entradas y salidas de la celda LSTM son vectores [40]. Los datos de entrada a la unidad LSTM forman un vector concatenado de nuevos datos de entrada x_t y datos de una celda anterior h_{t-1}. Estos vectores son previamente ponderados por la matriz de pesos W y por la matriz de pesos recurrentes U respectivamente lo que regulará el funcionamiento de las puertas. La matriz de pesos W está compuesta por cuatro vectores de tamaño igual al número de neuronas o unidades ocultas de la red LSTM. Estos son los cuatro vectores de peso de las cuatro puertas que conforma la celda LSTM y ordenadas de la siguiente forma:

1. Vector de pesos de la puerta de entrada, W_i
2. Vector de pesos de la puerta de olvido, W_f
3. Vector de pesos de la celda, $W_{\tilde{c}}$
4. Vector de pesos de la puerta de salida, W_o

Es decir, la dimensión de dicha matriz está dada por $4 \cdot D \times S$ donde D significa el número de unidades ocultas y S indica el número de la secuencia de entrada x_t. Si se supone, por ejemplo, cuatro unidades ocultas y un único dato de entrada, W será de tamaño [16×1]. Por otro lado la dimensión de la matriz de pesos recurrentes es igual a $4 \cdot D \times D$ y siguiendo el ejemplo anterior sería de tamaño [16×4]. Esta también está ordenada de la forma: $U = [U_i, U_f, U_{\tilde{c}}, U_o]$. Al vector de

entrada concatenado se le añade un vector de sesgo o *bias* de tamaño [4×1] por las cuatro capas de la celda $b = [b_i, b_f, b_{\tilde{c}}, b_o]$. Tanto las matrices W y U como el vector de sesgo b son aprendidos durante el entrenamiento de la red y son invariantes. Además, sus valores se restringen dentro del rango [-1,1].

El primer paso es decidir qué información vamos a obviar o mejor dicho eliminar del estado de la celda. Esta decisión es tomada por una capa sigmoidal que es denominada como "capa de la puerta del olvido" y muestra en la Figura 3.3. Las salidas del vector concatenado y el término de sesgo pasan por la función de activación sigmoidal, propia de las puertas de la celda. La salida de la puerta de olvido queda definida por la expresión:

$$f_t = \sigma(W_f \cdot x_t + U_f \cdot h_{t-1} + b_f) \tag{3.1}$$

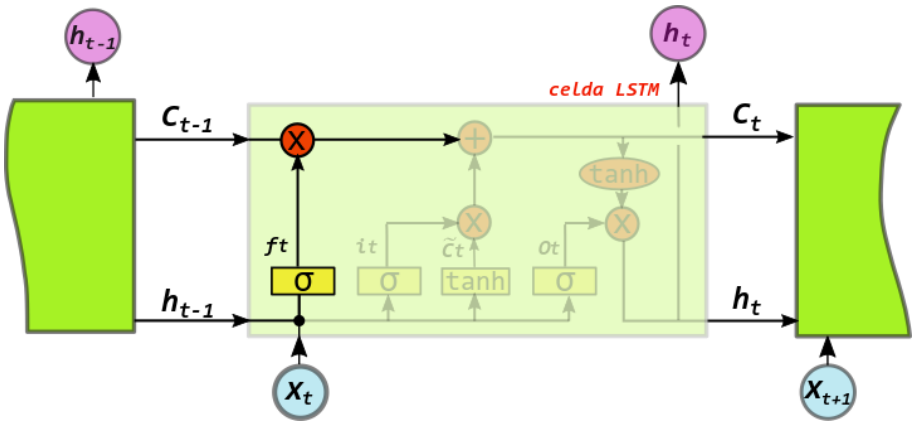


Figura 3.3: Diagrama de "capa de la puerta del olvido" de una celda LSTM.

Los valores de salida de la puerta de olvido o *forget gate*, f_t, comprendidos entre 0 y 1, multiplican mediante el producto Hadamard el estado previo de la celda C_{t_1} para decidir si se mantiene o se elimina parcial o totalmente la información. Esto puede compararse como el funcionamiento de una válvula, un 1 significa mantener completamente el valor (válvula cerrada) mientras que un 0 representa deshacerse completamente de este (válvula abierta). Para comprender mejor el objetivo de este funcionamiento se va a poner un ejemplo real. Si tenemos un sistema para predecir las palabras ocultas de un texto basándose en todas las anteriores, el estado de la celda podría incluir el género del sujeto previo. Cuando vemos un nuevo sujeto, queremos que la predicción no se base en el genero anterior por lo que es necesario que nos "olvidemos" de él.

El siguiente paso es decidir qué nueva información vamos a almacenar en el estado de la celda. Esto se divide en dos partes. En la primera, una capa sigmoide llamada "capa de la puerta de entrada" decide qué valores vamos a actualizar expresándose por la ecuación:

$$i_t = \sigma(W_i \cdot x_t + U_i \cdot h_{t-1} + b_i) \tag{3.2}$$

A continuación una capa con función de activación tangente hiperbólica, crea un vector de nuevos valores candidatos, $\tilde{C}_t$, que podrían ser añadidos al estado:

$$\tilde{C}_t = \tanh\left(W_{\tilde{c}} \cdot x_t + U_{\tilde{c}} \cdot h_{t-1} + b_{\tilde{c}}\right) \tag{3.3}$$

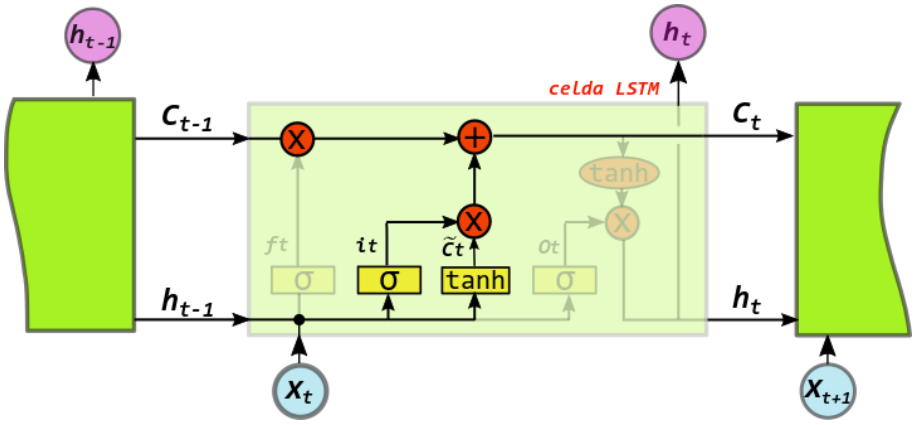


Figura 3.4: Diagrama de la "capa de la puerta de entrada" y etapa de actualización de una celda LSTM.

Estas dos se combinan mediante una operación de multiplicación resultando en un vector de actualización. Luego, se añade al estado de la celda resultando en el actual estado C_t. En el ejemplo de la predicción de palabras este paso se extrapola a añadir el género del nuevo sujeto al estado de la celda para reemplazar el antiguo que habíamos olvidado. Por tanto, siguiendo la Figura 3.4, se puede obtener cualitativamente el nuevo estado de la celda por realizar el producto Hadamard (representado por el símbolo $\odot$) entre la función de olvido y el estado anterior más la función de entrada y los nuevos valores candidatos del estado:

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{3.4}$$

Finalmente, se decide cual es la predicción actual h_t de la celda LSTM. Esta salida se basa en el estado de la celda actual, pero será una versión filtrada por la puerta de salida. Primero, ejecutamos la capa de salida sigmoidea que decide qué partes del estado de la celda se dan como salida. Esto se puede expresarse como:

$$o_t = \sigma(W_o \cdot x_t + U_o \cdot h_{t-1} + b_o) \tag{3.5}$$

Luego, el estado de la célula es multiplicado por una función *tanh* para forzar a que los valores estén comprendidos en el rango [-1, 1] y lo multiplicamos por la puerta de salida o_t, de modo que solo salgan las partes que se hayan decidido previamente. En el ejemplo del texto, como acaba de ver un sujeto nuevo, podría querer dar como salida un verbo, en caso de que eso sea lo que viene a continuación. Por ello podría dar como resultado la información de si el sujeto es singular o plural, para que sepamos en qué forma debe conjugarse un verbo si eso es lo que sigue a continuación. Teniendo en cuenta el flujo de información que se representa en la Figura 3.5, la salida de la celda se entiende por el producto Hadamard de la función de salida y el resultado de la función *tanh* del nuevo estado de la celda:

$$h_t = o_t \odot \tanh(C_t) \tag{3.6}$$

En la Figura 3.6 se muestra la arquitectura completa de una celda LSTM.

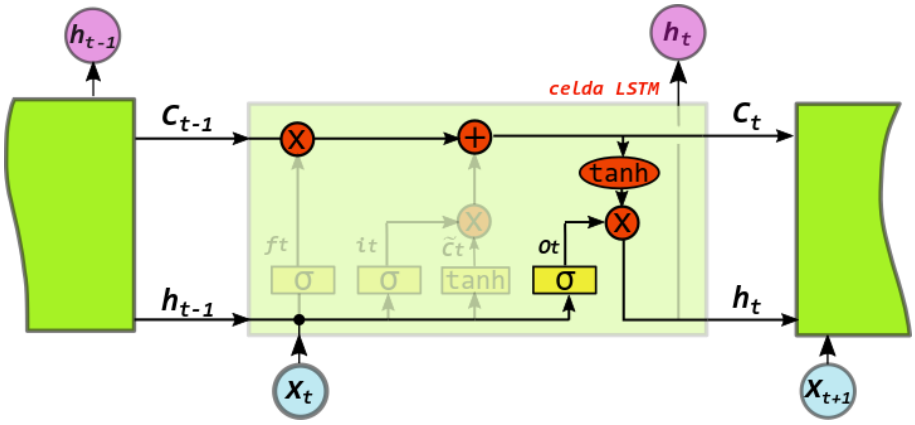


Figura 3.5: Diagrama de la puerta de salida de una celda LSTM.

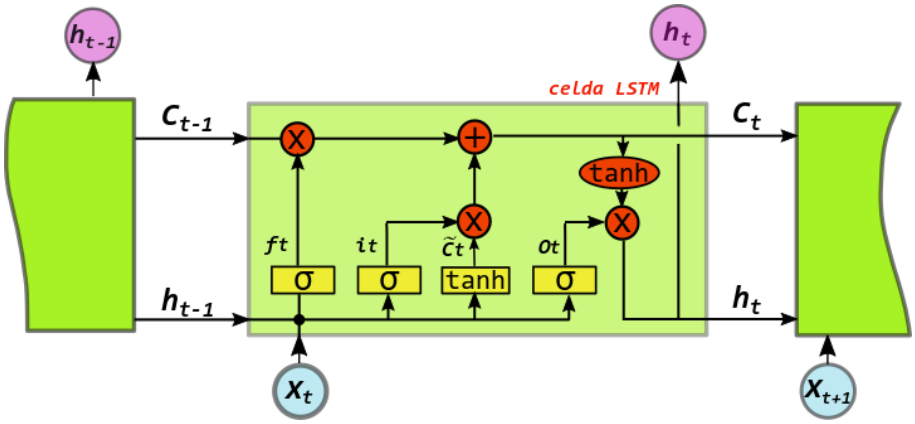


Figura 3.6: Arquitectura completa de una celda LSTM.

La arquitectura de la celda LSTM es, como se ha visto, mucho más compleja que la de las redes RNN estándar. Pero es gracias a esto que las LSTM están capacitadas para el aprendizaje de información prolongada en intervalos de tiempo resolviendo el problema del flujo de error o el desvanecimiento de gradiente. Estas redes se están usando hoy en día para este tipo de problemas de series temporales y es un tema de investigación muy innovador. Una vez conocido esto, se desarrolla a continuación las simulaciones y demostraciones del uso de estas redes para series temporales en software. Luego, se verá un ejemplo de aplicación en radio cognitiva pues las entradas al sistemas y las señales a predecir son las ondas electromagnéticas con la finalidad de modificar los parámetros de un filtro y ajustarnos al canal óptimo de comunicación.

Parte III

Desarrollo, simulaciones y conclusiones

Capítulo 4

Predicción de series temporales con redes LSTM usando MATLAB®

Este capítulo trata de explicar la arquitectura y el funcionamiento principal de las redes LSTM para series temporales con problemas de regresión usando MATLAB® como entorno de desarrollo. MATLAB® es un software altamente usado y es una buena herramienta para el análisis iterativo y procesos de diseño con un lenguaje de programación que expresa las matemáticas de matrices y arrays directamente. La versión usada es la 2020a (aunque también pueden emplearse las versiones anteriores de 2019) y es necesario disponer de *Deep Learning Toolbox*®. Además, se usa la ayuda técnica online de MathWorks® para obtener la información necesaria [1]. El código fuente que se usa para estas pruebas ha sido incluido junto con este documento. Los ficheros y el objetivo de cada uno de ellos fue especificado en la sección 1.4.

4.1. Arquitectura de la red LSTM en MATLAB®

Una red LSTM es un tipo de red neuronal recurrente que puede aprender las dependencias o pesos a largo plazo entre los pasos de tiempo de los datos de la secuencia a la entrada. La arquitectura de una red LSTM se compone de una capa de entrada, una o más capas LSTM y una capa de salida. Todas ellas están compuestas por un número concreto de unidades o neuronas. El diagrama de la Figura 4.1 ilustra mediante bloques simples la arquitectura de una red LSTM para regresión en MATLAB®. Por consiguiente, un total de cuatro capas conforman la arquitectura LSTM para este tipo de problemas. La capa de entrada o *sequence input layer* introduce secuencias de datos o series temporales a la red. El tamaño de la capa de entrada se corresponde al número de características de los datos de entrada o *dataset*. Suponiendo que el problema será univariable, es decir que únicamente se observa el comportamiento de una señal, el tamaño de la capa de entrada es de una unidad o neurona. Luego estos datos pasan a la capa LSTM la cual aprende las dependencias o pesos a largo plazo entre los pasos de tiempo de los datos de la secuencia. Estas dependencias se aprenden a través de los vectores y matrices de pesos W, pesos recurrentes U y *bias* que se especificó en el capitulo 3. La capa LSTM crea una capa de unidades o neuronas ocultas (*hidden units*). El número de unidades ocultas D se corresponde con la cantidad de datos a recordar entre los pasos de tiempo o lo que es lo mismo, al tamaño del estado oculto o *hidden state*. Por ello, es fácil pensar que es conveniente un número elevado de unidades ocultas para tener más información y predecir en base a ella. Sin embargo, si el número de unidades es demasiado grande, entonces la capa puede sobreajustarse a los datos de entrenamiento. Esto es conocido en el campo del DL como *overfitting* proveniente de la lengua inglesa. El problema de sobreajuste supone un

[1] Página web de MathWorks: https://es.mathworks.com/help/

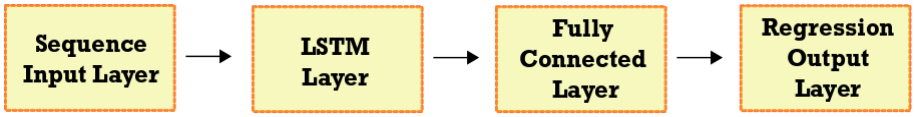


Figura 4.1: Diagrama de bloques simple de la arquitectura LSTM para un problema de series temporales por regresión.

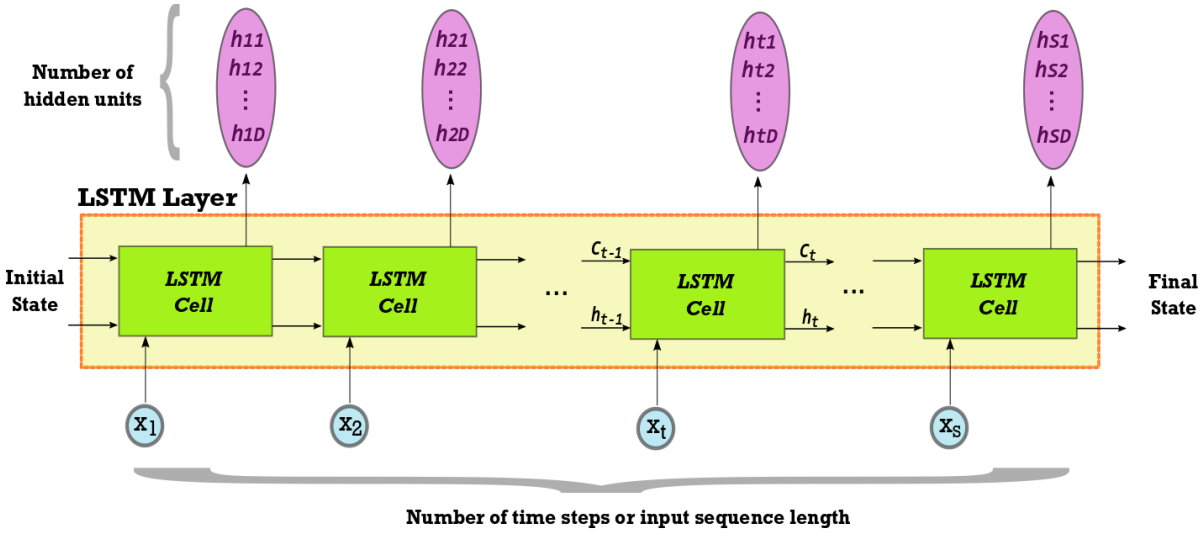


Figura 4.2: Esquema de entradas y salidas de la capa LSTM desenrollada en el tiempo.

exceso de ajuste, como su nombre indica, a los datos de entrenamiento de la red. Esto supone que la red quede entrenada para esos datos determinados y presente un mal comportamiento si la entrada varía con respecto a ellos [6]. Este es un serio problema para redes con gran número de parámetros y los investigadores tienen que lidiar con evitar este fenómeno. No obstante, el número de unidades ocultas puede variar desde docenas a unos pocos de millares. Un ejemplo de solución al sobreajuste es detallada en el artículo de Nitish Srivastava [41].

Desenrollando en el tiempo la celda LSTM como se hizo en la Figura 2.9 para una red RNN estándar, se puede representar el funcionamiento de la capa LSTM. No hay que olvidar que esto es una representación de funcionamiento temporal pero únicamente existe una celda LSTM por cada unidad o neurona oculta. Se muestra en la Figura 4.2 como cada dato de la entrada X univariable entra en la celda en cada paso de tiempo. El número total de pasos o, lo que es lo mismo, la longitud de la secuencia de entrada se representa por la letra S. A medida que se avanza en el tiempo, tanto el *hidden state* como el *cell state* se modificarán en base a las entradas y la realimentación. La salida de la celda en cada paso de tiempo es el estado oculto h de tamaño igual al número de unidades ocultas que se definan, como ya bien se aclaró antes.

Específicamente para un problema de regresión, tras la capa LSTM debe ir seguido una capa *fully connected* y a continuación de esta la capa de salida de regresión. La capa *fully connected* se encarga de multiplicar la predicción o salida de la capa LSTM por otra matriz de pesos aprendidos durante el entrenamiento. Esta matriz de pesos pondera la conexión entre la salida de la capa oculta y la neurona o neuronas de salida. La dimensión de la matriz es igual a $R \times D$ donde R se corresponde al numero de unidades de salida o respuestas de la red y D el número de unidades ocultas. En último lugar, la capa de regresión calcula el coste o pérdida del error cuadrático medio (o *Half Mean Squared Error loss*) para problemas de regresión. El objetivo es minimizar la pérdida durante el entrenamiento para aprender las matrices y vectores de pesos y obtener así el modelo de la red. Suponiendo que la entrada es una secuencia de datos al igual que en la salida, conocido esto como *sequence-to-sequence*, esta pérdida se puede expresar de la siguiente forma:

$$loss = \frac{1}{2S}\sum_{i=1}^{S}\sum_{j=1}^{R}(y_{ij} - h_{ij})^2, \tag{4.1}$$

donde y_{ij} es la salida objetivo, h_{ij} la salida proporcionada por la red y S la longitud de la secuencia de entrada. Un dato importante a remarcar es que la normalización de los datos de entrada y respuestas ayuda a estabilizar y acelerar el entrenamiento de las redes neuronales para la regresión. Por este motivo, se vuelve a comentar más adelante que esto es implementado en el código fuente.

4.2. Entrenamiento de la red LSTM en MATLAB®

El entrenamiento de una red neuronal consiste en aprender unos parámetros llamados pesos, o en inglés *weights*, que definen el modelo de la red neuronal y que ponderarán los valores de salida en función de su valor y el dato de entrada actual. Si se emplea el aprendizaje supervisado (ver sección 2.3.1), el conjunto de datos para el entrenamiento es conocido. La salida correcta es conocida frente a las entradas obteniendo así una relación entre las entradas y las salidas. Esta relación es dada por los pesos [6].

Para un mejor ajuste y para evitar que el entrenamiento se desvíe, es habitual estandarizar los datos de entrada (tanto para el entrenamiento como para el test o validación) para tener una media cero y una variación unitaria. Esto es debido a que el algoritmo supervisado genera valores de salida dentro de un intervalo fijo que es definido según la función de activación de la capa de salida. En este caso la función es sigmoidal y por tanto se comprende en el rango [0,1] como se aclaró en la sección 3.1. Estandarizar o normalizar los datos consiste en encontrar los niveles más altos y más bajos (los valores extremos) en el conjunto de datos de entrada para obtener la media. Luego se obtiene la desviación estándar del conjunto y finalmente la entrada queda normalizada según la expresión:

$$Dataset_{normalizado} = \frac{Dataset - media}{desviacion} \tag{4.2}$$

Aunque la mayoría de los algoritmos son capaces de escalar el conjunto de datos de entrada adecuadamente usando los datos reales, esto supone un aprendizaje más lento. Por consiguiente, el entrenamiento y la predicción se hace con los valores de entrada normalizados y para obtener sus valores reales simplemente hay que invertir el proceso de cálculo [42].

Tras ello, se puede decir que la línea de código de Matlab® para entrenar la red LSTM es la siguiente:

```
net = trainNetwork(sequences, Y, layers, options)
```

donde `net` es la variable estructura en la que se almacena el modelo de la red LSTM y `trainNetwork` es la función para entrenarla mediante la configuración seleccionada por el usuario a través de las entradas a dicha función[2]. Por un lado para el problema de regresión de series temporales, `sequences` contiene la entrada normalizada en forma de vector y la variable `Y` es el vector objetivo o salidas conocidas frente a la entrada `sequences`. Para la predicción de un paso futuro en el tiempo, la variable `Y` es igual a la entrada `sequences` pero desplazada una vez como se indico con el cuadro 2.2. Estas variables coinciden con las explicadas en la sección 2.3.1 correspondiendo `sequences` con X e `Y` con H. Retomando dicha nomenclatura, la señal de entrada actual conocida X_t debe dar como salida predicha el valor dado en X_{t+1} que a su vez se ha definido en el vector objetivos un instante de tiempo anterior por lo que es igual a H_t.

[2]Página web de la función de entrenamiento: https://es.mathworks.com/help/deeplearning/ref/trainnetwork.html

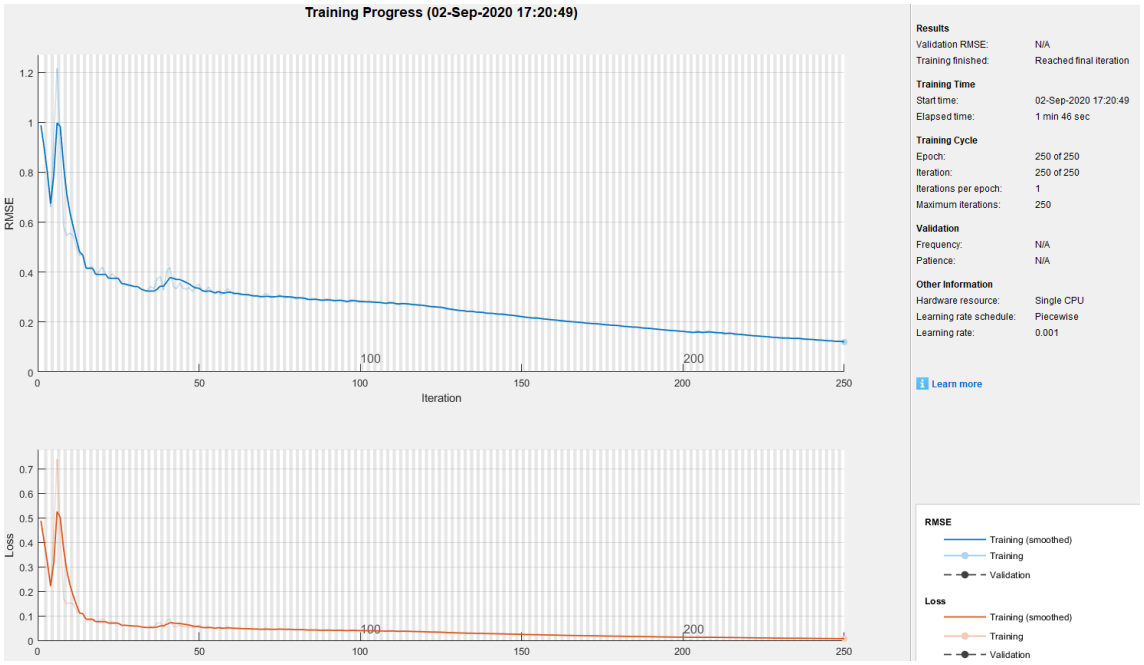


Figura 4.3: Ventana emergente que aparece durante el entrenamiento de la red neuronal donde se aprecia la disminución del error RMSE y la pérdida *loss*.

La variable `layers` es la que previamente se debe fijar con la arquitectura de la red neuronal a entrenar. En ella se indica que tipo de red se usa, cuantas capas y unidades incorpora y el tipo de problema que se presenta, de clasificación o regresión (para ver en detalle como se implementa en Matlab® es necesario consultar el código fuente suministrado junto con este documento). Por último la variable `options` se define usando otra función conocida de Matlab® y es `trainingOptions`. esta determina el tipo de optimizador del aprendizaje que en este caso se usa el conocido método de Adam el cual por si mismo tiene un estudio complejo matemático que puede ser consultado en la referencia [43]. La entrada `options` también asigna otros valores importantes en el entrenamiento de una red como lo es la velocidad de aprendizaje, crucial para que el sistema converja en una solución óptima logrando alcanzar el mínimo error. Aunque no se entre en mucho detalle debido a que se ha seleccionado un valor estándar de esta, la elección de la velocidad de aprendizaje puede suponer un sobreajuste o, por lo contrario, un mal ajuste a los datos de entrenamiento por lo que es un factor de gran envergadura.

En la Figura 4.3 se muestra la ventana emergente que aparece durante el entrenamiento de la red neuronal. En ella se representa gráficamente la evolución para cada iteración del error cuadrático medio (RMSE) obtenido y la pérdida. Ambas, disminuyen considerablemente y aún más lo hace la pérdida ya que llega a valores muy cercanos al cero (ver ecuación 4.1 para comprender el cálculo de esta). Esto hace que al final del entrenamiento la red neuronal quede ajustada en función de la dinámica de la señal de entrada.

4.3. Implementación de la red LSTM enfocada en el caso de estudio

Con lo comentado hasta ahora, ya podemos diseñar nuestra red neuronal LSTM. En primer lugar, solo tendrá una unidad de entrada y otra de salida ya que la señal es univariable y la salida solo presenta una respuesta, el valor predicho un paso de tiempo futuro en base a la entrada proporcionada. En la Figura 4.4 se ve claramente la arquitectura final. El número de unidades ocultas es lo que nos queda por determinar y esto se hará en base a los resultados obtenidos en este capítulo.

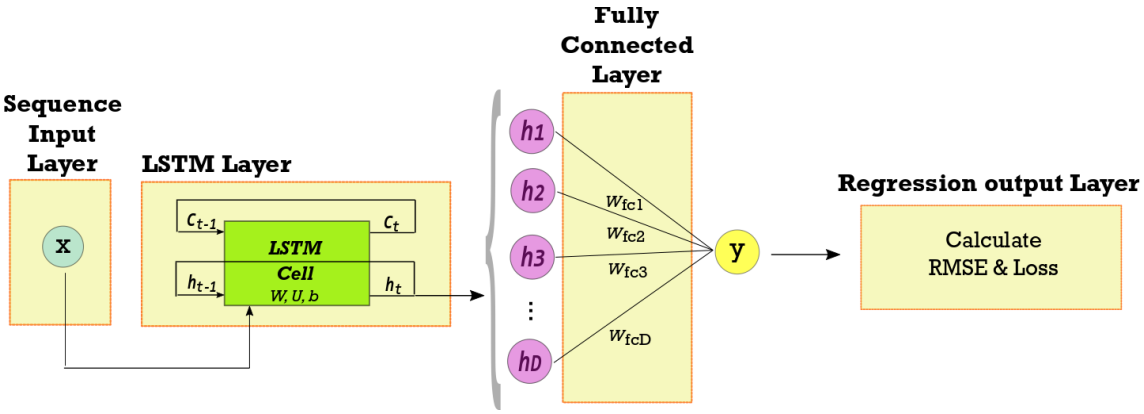
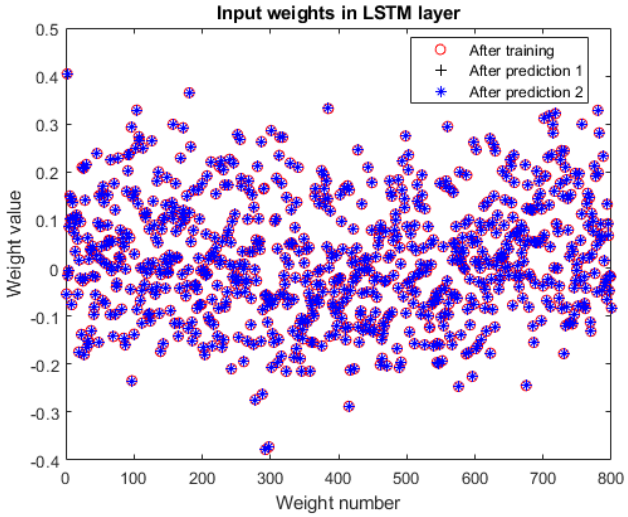


Figura 4.4: Estructura de la red neuronal implementada en Matlab® indicando una sola unidad o neurona de entrada y salida.

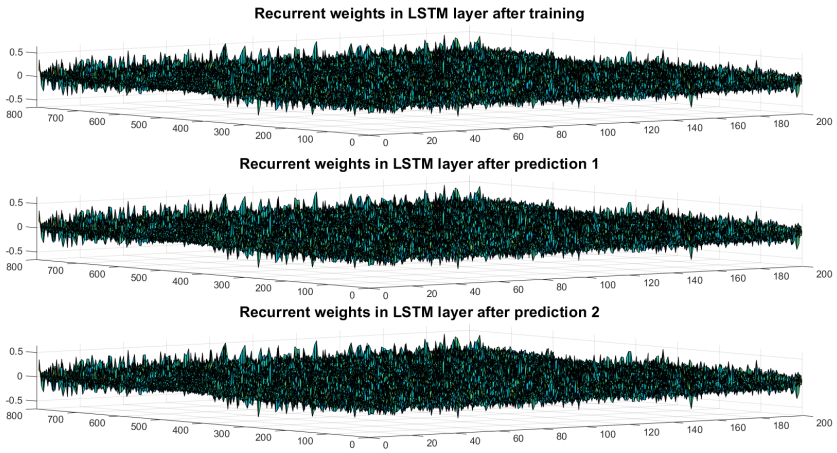
Un ejemplo de Matlab® que se dirige con cierta exactitud hacia la linea de investigación de este trabajo es nominado "*Time Series Forecasting Using Deep Learning*". El código con su explicación detallada se encuentra en la página web: https://www.mathworks.com/help/deeplearning/ug/time-series-forecasting-using-deep-learning.html y puede ser abierto en Matlab simplemente escribiendo en la ventana de comandos `openExample('nnet/TimeSeriesForecastingUsingDeepLearningExample')`. El ejemplo entrena a una red LSTM para predecir el número de casos de viruela dado el número de casos en los meses anteriores. La arquitectura que usa es la mencionada en este capítulo. El número de unidades ocultas para la capa LSTM es de 200 por lo que D es igual a 200 neuronas.

Este ejemplo demuestra que para pronosticar los valores de los futuros pasos de tiempo de una secuencia, se puede entrenar una red LSTM de regresión de secuencia a secuencia, donde las respuestas coinciden con la secuencia de entrenamiento desplazada un paso de tiempo. Es decir, en cada paso de tiempo de la secuencia de entrada, la red LSTM aprende a predecir el valor del siguiente paso de tiempo. Esto sigue el procedimiento de predicción de series temporales detallado al final de la sección 2.3.1. Primero carga una base de datos, define y entrena la red. Luego, Se utiliza la red de dos formas diferentes: prediciendo cada nuevo dato a partir de la predicción anterior lo cual acumula el error de cada paso, o bien prediciendo cada nuevo dato a partir del valor real anterior conocido evitando que se vaya acumulando el error. La primera predicción, pronostica los valores de múltiples pasos de tiempo en el futuro de uno en uno utilizando la predicción anterior como entrada de la siguiente. La segunda trata de predecir de la misma forma que la anterior pero actualizando el estado de la red con los valores observados reales conocidos en lugar del valor predicho anterior. Para hacer predicciones sobre una nueva secuencia, reinicia el estado de la red utilizando la función `resetState` y la actualiza con los datos hasta ahora conocidos (incluyendo los de entrenamiento). Al restablecer el estado de la red se evita que las predicciones anteriores afecten a las predicciones sobre los nuevos datos. En los resultados mostrados en la página web del ejemplo, el primer tipo de predicción daba como resultado un valor RMSE igual a 248.5531 mientras que para el segundo tipo, con los mismos datos, se obtiene un RMSE de 158.0959. Una mejor predicción es obtenida al resetear y actualizar la red pero para ello hace falta dos cosas: una es medir la señal de entrada en cada momento para tener la información hasta el momento actual y la otra es que este proceso de reseteo conlleva mucho más tiempo de predicción. En el ejemplo de aplicación que se presenta en este trabajo (detallado en el próximo capítulo), se realiza una combinación de ambas predicciones y se adapta a las necesidades del sistema.

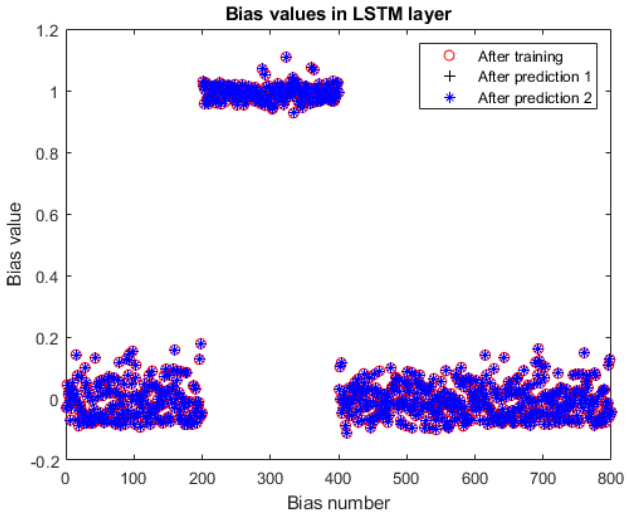
Tras almacenar el estado de la red justo después del entrenamiento y una vez hecha la predicción, se han podido representar gráficamente la evolución de los parámetros entrenables, los pesos y bias, para así analizar el comportamiento de la red. Con la ayuda de las Figuras 4.5 se observa que los pesos de entrada W, los pesos recurrentes U y los bias o sesgos b toman un valor tras el entrenamiento que no variarán tras las predicciones, reseteos o actualizaciones de la red neuronal. Los valores de los parámetros coinciden tras el entrenamiento y después de los dos tipos de predicciones realizadas. Observase que el eje horizontal recorre completamente los vectores de



(a)



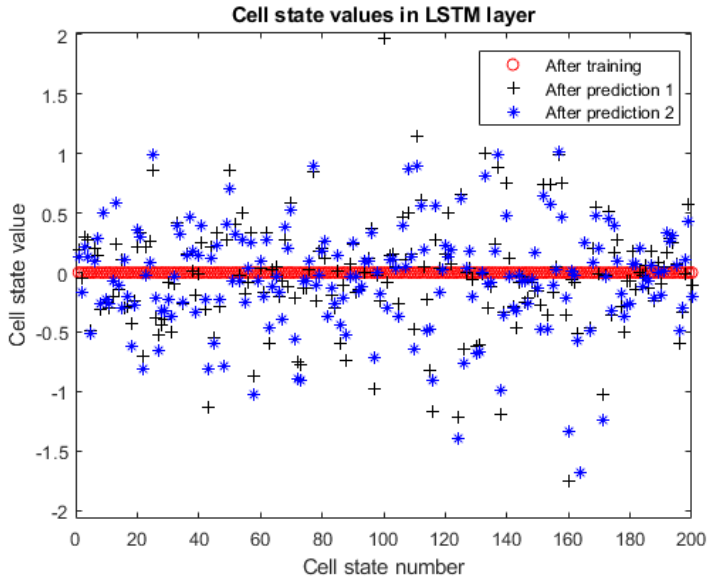
(b)



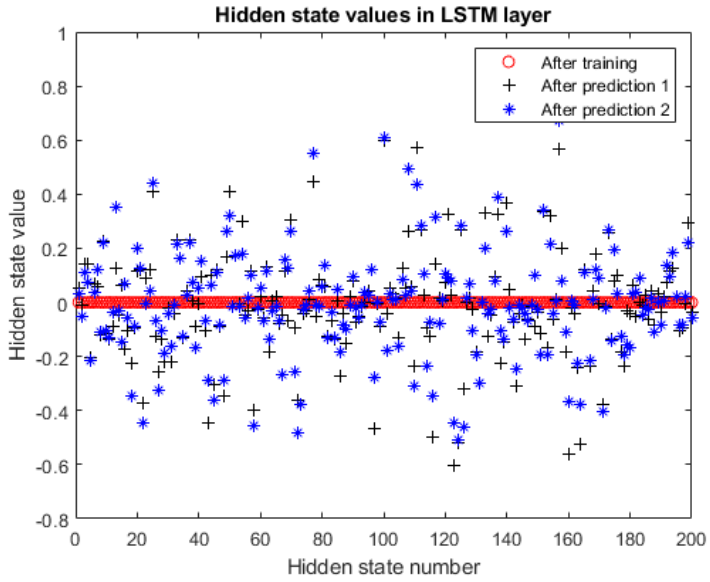
(c)

Figura 4.5: Gráficas de los pesos y sesgos aprendidos de la red LSTM. Se comprueba que estos son valores fijos por lo que no cambian con las predicciones.

dicho parámetros. Debido a que en el ejemplo se usan 200 unidades ocultas, el tamaño de los pesos de entrada y del vector bias es de 1×800 siendo 800 el producto de las 200 unidades por las 4 puertas de la celda LSTM. Por el contrario, la matriz de los pesos recurrentes es de dimensión 200×800 donde la el tamaño de 200 también se refiere al número de neuronas ocultas en la capa LSTM. Los valores de los pesos están dentro del rango [-1,1] mientras que el sesgo toma valores entre [0,1]. De la Figura 4.5(c) podemos observar que el sesgo ronda normalmente el valor 0 excepto para los valores situados entre las muestras 200 y 400. Estas se refieres a los bias de la puerta *forget* significando que mantiene los datos recordados. Todo esto fue explicado teóricamente en la sección 3.1.



(a)



(b)

Figura 4.6: Gráficas del estado de la celda y estado oculto para cada unidad oculta en dos predicciones diferentes con la misma red LSTM entrenada.

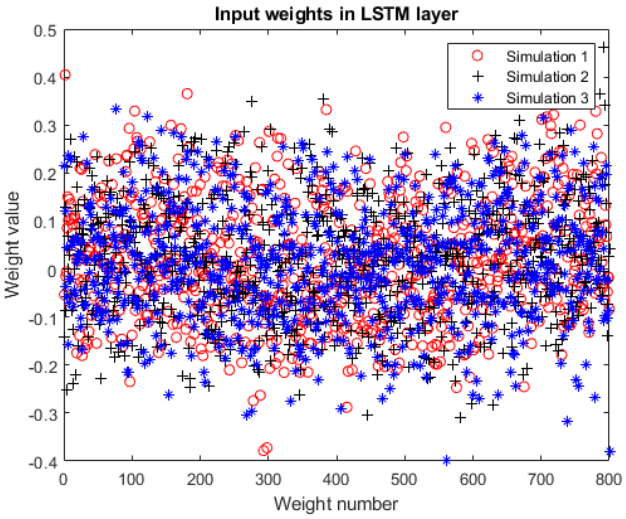
Se acaba de comprobar que los pesos no varía tras el entrenamiento, sin embargo, durante la predicción el estado de la celda y el estado oculto o salida de la celda LSTM son las que variarán

Simulación	Predicción 1 (sin actualizar la red)	Predicción 2 (actualiza la red)
Página web	248.5531	158.0959
Sim 1	203.0765	129.0444
Sim 2	273.6893	186.3402
Sim 3	290.3078	152.5403

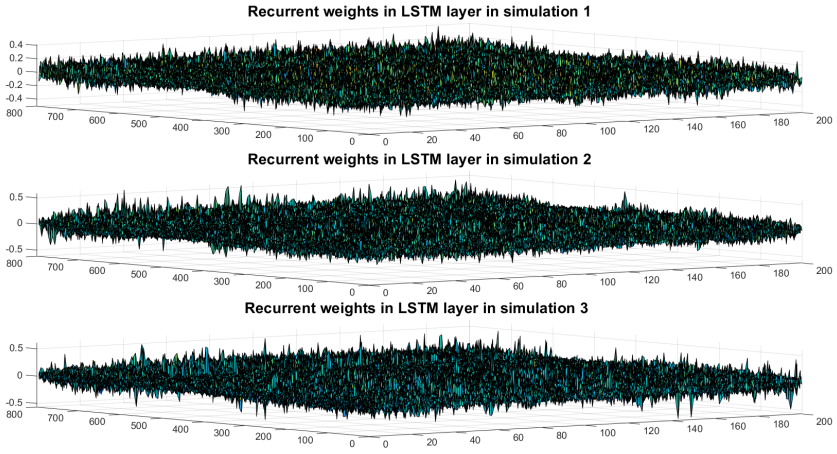
Cuadro 4.1: Valores RMSE de las predicciones sin y con actualización de la red neuronal para distintas ejecuciones del mismo ejemplo de Matlab®

provocando predicciones ajustadas a las nuevas entradas a la red neuronal. Se ve reflejado en las gráficas de las Figuras 4.6. Además para la misma red entrenada, los dos tipos de predicciones provocan diferentes valores en ambos estados. Nótese que tras el entrenamiento, sus valores son cero (círculos rojos) puesto que durante este proceso la salida es conocida y comparada por lo que se toma como una predicción perfecta para obtener los valores de los parámetros entrenables.

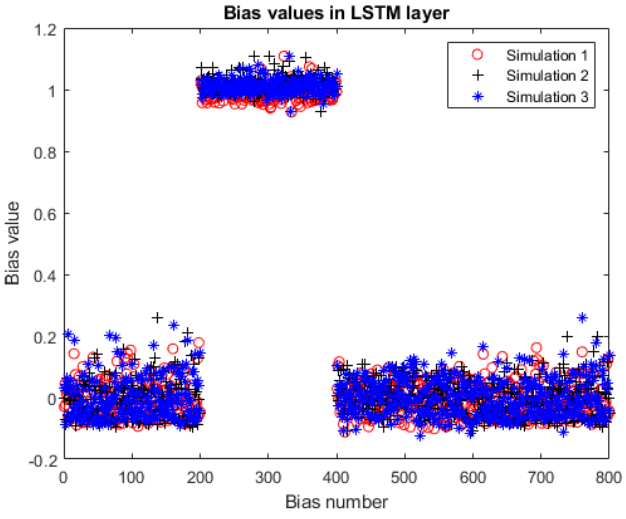
Por último, es de interés saber que para los mismos datos de entrada a la red, esta puede ser entrenada de diferentes formas, es decir, aprende unos pesos y sesgos diferentes pero que deben dar igualmente un buen comportamiento. Para comprobar esto, se ha almacenado en los ficheros *sim1*, *sim2* y *sim3* la red entrenada para 3 ejecuciones distintas del mismo ejemplo "*Time Series Forecasting Using Deep Learning*" incluido en el material adicional junto con esta memoria. En las gráficas de la Figura 4.7 se aprecia que estos parámetros tienen valores similares pero no iguales. Aunque para estas tres simulaciones el comportamiento de la red es muy parecido, esto se puede cuantificar con los valores de error obtenidos en las predicciones. En el cuadro 4.1 se recogen los valores RMSE de estas tres ejecuciones del código junto con los valores que aparecen en la página web de este mismo ejemplo y que se especificaron antes. Esta conclusión se ha tenido en cuenta en la realización de la prueba de concepto que se muestra en el próximo capítulo puesto que no siempre se tienen los mismos resultados aunque se debe conseguir que sean fiables.



(a)



(b)



(c)

Figura 4.7: Gráficas de los pesos y sesgos aprendidos de la red LSTM. Se comprueba que estos son valores fijos por lo que no cambian con las predicciones.

Capítulo 5

Sistema de selección de banda para un Filtro *Bandpass* mediante redes neuronales LSTM

Este capítulo trata de presentar un sistema de selección de banda o canal para el ajuste de un filtro *Bandpass* y así lograr un comportamiento óptimo de este. Es necesario aclarar que se pretende demostrar el concepto de radio cognitiva para una simple aplicación la cual podría extrapolarse a un sinfín de problemas. El objetivo es demostrar la predicción de una señal dinámica de entrada que cambia con el tiempo y que puede tratarse como una predicción de series temporales. Para ello se usan las redes LSTM, lo que ha sido razonado en el marco teórico de este documento. Con ello, se predice el comportamiento o tendencia futura de la señal y, en base a esta información se decidirá que ancho de banda o canal asignar cambiando los parámetros del biquad del filtro *Bandpass*.

El caso de estudio para este trabajo se representa en el diagrama de bloques conceptual de la Figura 5.1. Se muestra un receptor de conversión directa basado en radio cognitiva. El receptor consiste en un frontal RF (o *RF front-end*) analógico programable, compuesto por un filtro de radiofrecuencia y un amplificador de bajo ruido (o *Low-Noise Amplifier*(LNA)), seguido de un convertidor analógico-digital de paso de banda (o *Band-Pass Analog-to-Digital Converter*(BP-ADC)), que digitaliza la señal entrante de la antena de manera que la mayor parte del procesamiento de la señal se lleva a cabo por el procesador de señales digitales (o *Digital Signal Processor*(DSP)). Todos estos bloques pueden ser asistidos y controlados por un motor de red LSTM para optimizar la calidad de servicio (QoS) en el receptor. Para ello, el bloque LSTM interactúa con el frontal RF analógico/RF, el ADC y el DSP para percibir la información sobre el nivel de las interferencias fuera de banda y el ruido base, con el fin de predecir el nivel de ocupación en una banda y/o canal determinado. Sobre la base de esta información, el DSP adaptará la métrica de actuación de los diferentes bloques en el frontal RF analógico a fin de seleccionar el canal óptimo para procesar las diferentes señales entrantes en la antena. Con el fin de simplificar y sin pérdida de generalidad, este trabajo se centrará en la forma en que el motor de red LSTM propuesto puede utilizarse para predecir la mejor banda/canal para la señal que se va a transmitir modificando los parámetros del filtro de radiofrecuencia. En la Figura 5.1 se resaltan con cajas grises los dos bloques que engloban el ejemplo de aplicación que se presenta en este capítulo. Los demás bloques completarían este ejemplo y podría hacerse a partir del estudio que se presenta pero, aún así, es suficiente para demostrar el funcionamiento de las redes LSTM para la predicción de la señal de ocupación por lo que solo engloba los bloques resaltados.

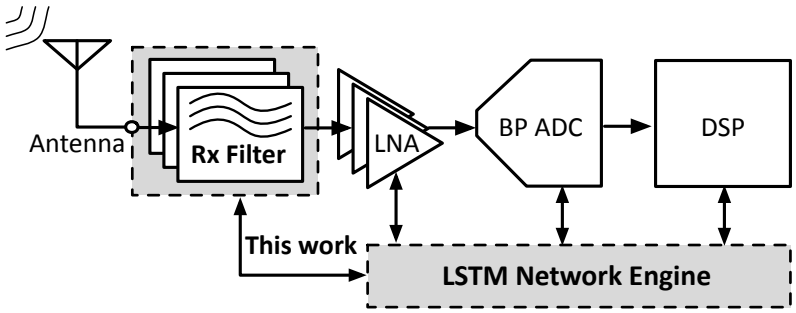


Figura 5.1: Diagrama de bloques del receptor basado en radio cognitiva usado como caso de estudio.

5.1. Planteamiento del sistema de la selección de banda para un Filtro *Bandpass* gobernado por la predicción dada por una red neuronal LSTM

En particular, se considera que el receptor es parte de un dispositivo IoT genérico con un transceptor inalámbrico que puede operar en diferentes bandas de frecuencia. La Figura 5.2(a) ilustra el espectro de frecuencias con n canales diferentes, como se explicó previamente con la Figura 2.2, pero esta vez se centran en las frecuencias $f_1, f_2, \dots fn$ considerando todos los anchos de banda iguales. Para cada canal i, suponemos que podemos medir la ocupación de la banda en tiempo real, obteniendo las señales representadas en la Figura 5.2(b) para los canales $1, 2, \dots n$. Considerando estas señales de ocupación Oc_i como la entrada a nuestro sistema, representamos el motor de red LSTM propuesto como el cuadro de puntos de la Figura 5.2(c), con n redes LSTM en paralelo, cada una de las cuales predice la evolución futura de una señal de ocupación Oc_i^{pre} , y un bloque de decisión que utiliza estas predicciones para decidir qué canal estará menos ocupado a continuación. Basándose en esta decisión, este bloque modificará un filtro BP para ajustar su función de transferencia y así seleccionar la banda de frecuencias apropiada.

Las redes LSTM representadas en la Figura 5.2(c) son univariables - tienen una variable (o neurona) para las capas de entrada y de salida - mientras que tienen una capa oculta con D unidades LSTM. Además, se implementa una red LSTM de múltiples pasos para pronosticar más de un paso de tiempo futuro, como se definió para los problemas de modelado predictivo de regresión. El objetivo de cada red LSTM (una para cada canal) es predecir un cierto número de muestras futuras M de su señal de entrada Oc_i. Para ello, la red tiene que ser entrenada con un conjunto de datos que representen señales reales de ocupación de la banda, permitiendo a la red aprender sus características y dinámica. Después del entrenamiento, cada red puede empezar a predecir la ocupación en tiempo real. Cada vez que la nueva red recibe una muestra de tiempo, calcula un número programable de muestras futuras M que pueden ser procesadas por el bloque de decisión. El número de muestras predichas M es un compromiso: un valor muy pequeño dará una predicción más precisa pero limitará la capacidad del sistema para anticipar la evolución futura del espectro de frecuencias, mientras que un valor muy grande mejorará la anticipación pero el error de la señal predicha aumentará, produciendo decisiones erróneas.

El bloque de decisión recibe todas las señales de ocupación predichas para cada canal y decide cuál debe seleccionarse con cierta anticipación. Dicha anticipación está determinada por el número asignado de pasos futuros a predecir M. Este valor M coincide con el tamaño de la secuencia de entrada que se denominaba con la letra S en el capítulo anterior, más concretamente en la Figura 4.2. Consideremos el caso de estudio de la Figura 5.1. La frecuencia de muesca/resonancia del filtro BP puede seleccionarse adecuadamente según los datos proporcionados por el motor de la red LSTM, de modo que la frecuencia de resonancia pueda desplazarse al canal más conveniente en términos de número de interferencias, ruido base y calidad de la información que se transmite. Esto puede ser modelado de la siguiente manera: en cada momento t_0, este bloque tiene que comparar la evolución futura de cada señal Oc_i^{pre} $(t_0; t_0 + 1; t_0 + 2; \dots; t_0 + M)$. A partir de la comparación de estas n predicciones, el bloque de decisión dirá qué canal alcanzará el valor de ocupación más bajo

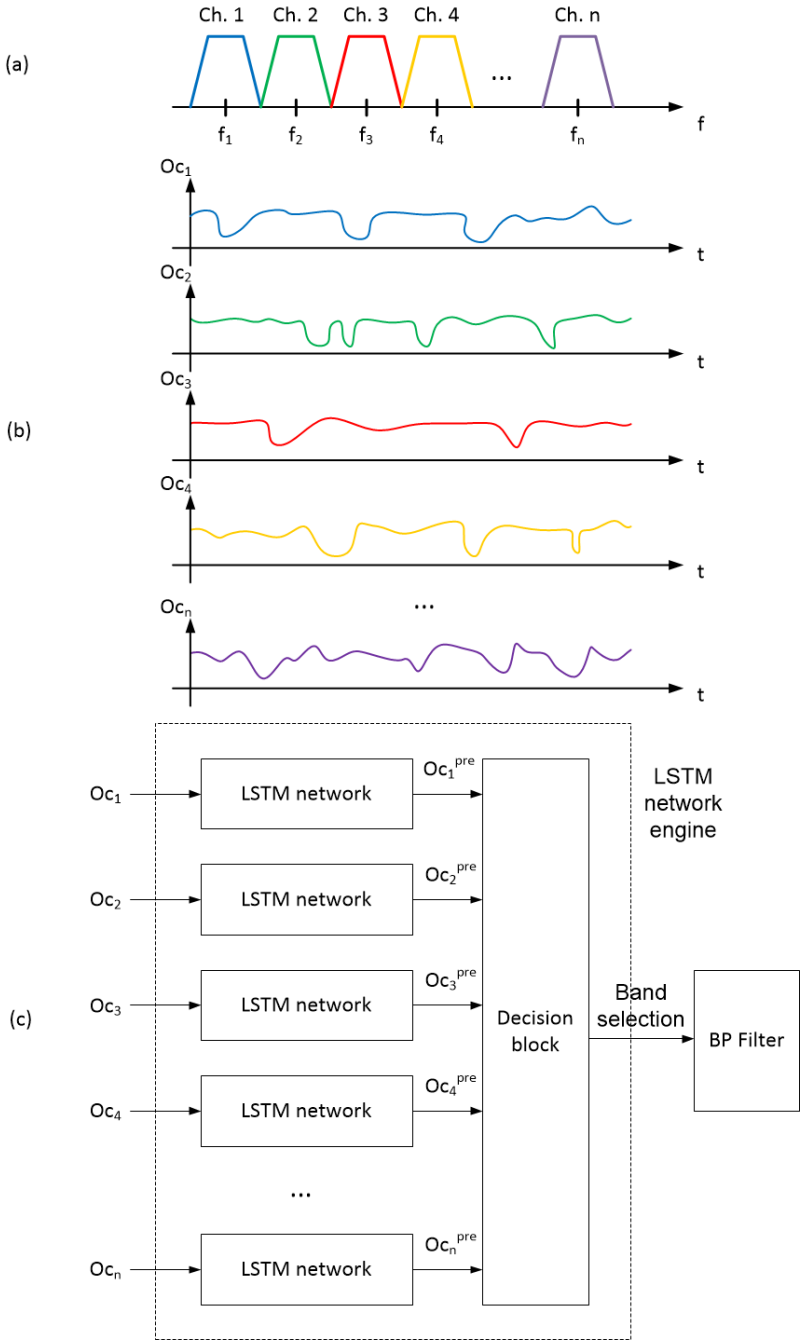


Figura 5.2: (a) Ilustración del espectro de frecuencia con n canales. (b) Representación de la señal de ocupación en el tiempo para los distintos canales. (c) Diagrama de bloques del sistema propuesto.

para este intervalo. Es decir, qué canal estará más libre para poder ser ocupado próximamente evitando interferencias. Si el valor más bajo corresponde a un canal distinto del seleccionado actualmente t_0, el bloque de decisión tiene que elegir los parámetros del filtro ajustando su banda de paso a la frecuencia apropiada del canal más libre. Para ello se definen previamente las frecuencias en las que se centran cada uno de los canales $(f_1, f_2, \ldots, f_n)$ suponiendo que tienen el mismo ancho de banda. En el caso en que el canal más libre sea el actual, no se produce cambio en el filtro. De esta manera, el sistema puede cambiar dinámicamente el canal de comunicación, mejorando el rendimiento del dispositivo IoT. En secciones posteriores se detalla mejor cómo se diseña el bloque de decisión para que su funcionamiento sea óptimo.

El sistema propuesto a modo de caso de estudio consta de cuatro canales y ha sido implementado

en MATLAB® siendo un caso particular del propuesto en la Figura 5.2(c) con cuatro redes LSTM (una para cada canal). A continuación se detallan los pasos y consideraciones que se han tenido en cuenta para esta implementación. También se especificará el flujo de predicción en MATLAB® y los resultados obtenidos aportando las conclusiones tomadas.

5.2. Señal de entrada. Datos de entrenamiento y de test

Una medida especial de la ocupación del espectro es presentada en la Figura 5.3 para la banda ISM (*Industrial, Scientific and Medical*) de 2.4 GHz. Se utiliza principalmente para los estándares de área local inalámbrica o WLAN (*Wireless Local Area Network*) denominado técnicamente por "IEEE 802.11b/g/n", Bluetooth, Zigbee y DECT (en América del Norte) sin necesidad de una licencia individual. Dado que el uso de internet aumenta rápidamente, a menudo hay varios puntos de acceso WLAN y estaciones móviles en el mismo canal siento este el problema del que se ha estado hablando en el trabajo y que motiva su realización. Existe, por tanto, una superposición de canales, y los canales vecinos presentan potencial de interferencia. En la Figura se muestra la potencia en el tiempo de un solo canal de WLAN para un solo usuario. Como se comenta en [2], para algunos fines es útil obtener cifras de ocupación para un único usuario específico en una frecuencia, por ejemplo, para identificar las fuentes de interferencia o para recomendar cambios de canal a fin de utilizar la banda disponible de la manera más eficiente. Esto es justamente lo que se pretende con el selector de bandas que se propone.

En la sección anterior en la que se planteaba de manera general el sistema propuesto, se introdujo la señal de entrada como la señal de ocupación de un solo canal en el tiempo con la Figura 5.2(b). Con esta entrada se pretende simular el comportamiento de la señal mostrada para la banda ISM. En la Figura 5.3 se observa como la señal está ocupada cuando la magnitud aumenta de los -65 decibelios (o dB) aproximadamente y está desocupada cuando la señal está comprendida entre -90 y -75 dB. Este rango de valores que toma la señal cuando está libre se debe al ruido base. Además, se aprecia que la señal también puede aumentar con diferentes valores de magnitud entre los -65 dB hasta los -55 dB. En esta prueba de concepto se va a suponer una ocupación binaria, es decir, solo tendrá dos niveles de ocupación: ocupado o libre.

Todas las conclusiones tomadas para definir la señal de entrada se han hecho a base de aplicar los conceptos y observar los resultados basándonos en la predicción gráfica y el valor del error cuadrático medio (RMSE) obtenido (en la sección 5.5 se muestran los resultados finales). Para definir en Matlab la señal, se genera una señal aleatoria de ceros y unos que representan los dos niveles de ocupación del canal, ocupado y desocupado respectivamente. Estos ceros y unos, a los que los llamamos como elementos, representan intervalos de tiempo de tamaño fijo en los cuales asumimos que las señales de entrada permanecen en un estado constante. Los intervalos de tiempo son definidos por un número de muestras que definen el tiempo o tamaño del mismo. La señal de entrada se define con esta dinámica a lo largo del tiempo. La señal de entrada se divide en un conjunto de datos de entrenamiento y unos datos de test. En primer lugar los datos de entrenamiento son usados para, como es obvio, entrenar la red neuronal y que así esta aprenda la dinámica de la señal de entrada. Como se sabe, un buen entrenamiento es clave en *Deep Learning* [20] ya que condiciona la respuesta de la red neuronal y, en este caso, la predicción y futura decisión del canal a elegir. Es conocido que para un buen entrenamiento, el conjunto de datos para ello debe contemplar o definir en la mejor medida de lo posible el comportamiento de la señal por lo que se partió de un total de 10.000 muestras. Para ello se definió 100 elementos (entre ceros y unos) que se representaban cada uno de ellos con 100 muestras. Tras varias pruebas, se concluyó una mejora significativa tras doblar esta cantidad y que a partir de este número de muestras los resultados no experimentaban gran mejora. Es decir, finalmente se han tomado 20.000 muestras para el conjunto de entrenamiento lo que tampoco supone una diferencia notoria en computación de cálculo. Un conjunto mayor podría incluso traducirse a un sobreajuste de la red. Se toma una secuencia de 200 elementos definidos con 100 muestras por elemento. Para definir el número de muestras por elemento se ha considerado que cuanto menor sea su valor, menos realista es y puede ser más difícil aprender la dinámica de esta. En cambio si se usa muchas más muestras por elemento, se verá reflejado en un mayor gasto computacional al tratar con más datos que, a la vez, no aportan

mayor información. Un porcentaje de las muestras para cada elemento deben ser usadas para la transición de estado en el caso en que se produzca. Es decir, si el elemento actual es un cero y el siguiente a este es un uno, la señal tiene una transición de subida y de bajada para el caso opuesto. Estas transiciones o cambios de estado no pueden ser abruptos ya que no sería real y además la predicción no sería tan buena frente a estos cambios. Varias iteraciones del sistema han sido necesarias para deducir que un porcentaje alto de muestras para las transiciones resultan en un mejor comportamiento. Finalmente se obtuvo los mejores resultados (mínimos valores de RMSE) para un porcentaje del 70 % o, lo que es lo mismo para 70 muestras para representar cada cambio de estado. Para el conjunto de datos test o validación no se debe tomar de importancia el número de muestras totales ya que únicamente se deduce en más tiempo de ejecución del sistema. Sin embargo, los elementos tienen que definirse con el mismo número de muestras, que en este caso es de 100, y el mismo porcentaje de ellas para definir las transiciones. Por otro lado, para aproximar la señal a una real como la que se mostró en la Figura 5.3 es necesario añadir un cierto ruido base. En concreto se añade un ruido de media cero y varianza 0,05 a la señal. Después se suavizan estas componentes aleatorias mediante un filtrado paso de baja que aplicará una función de transferencia racional definida por un coeficiente numerador y denominador que en este caso se han definido con los valores 0.05 y 1, respectivamente. Para ello se usa la función *filter* de Matlab. Atenuar las componentes aleatorias del ruido promueve una mejor predicción mientras que la señal es más realista. Este proceso de filtrado es fácilmente usado en una implementación real en hardware lo que no implica añadir mucha complejidad y su uso es bastante notorio.

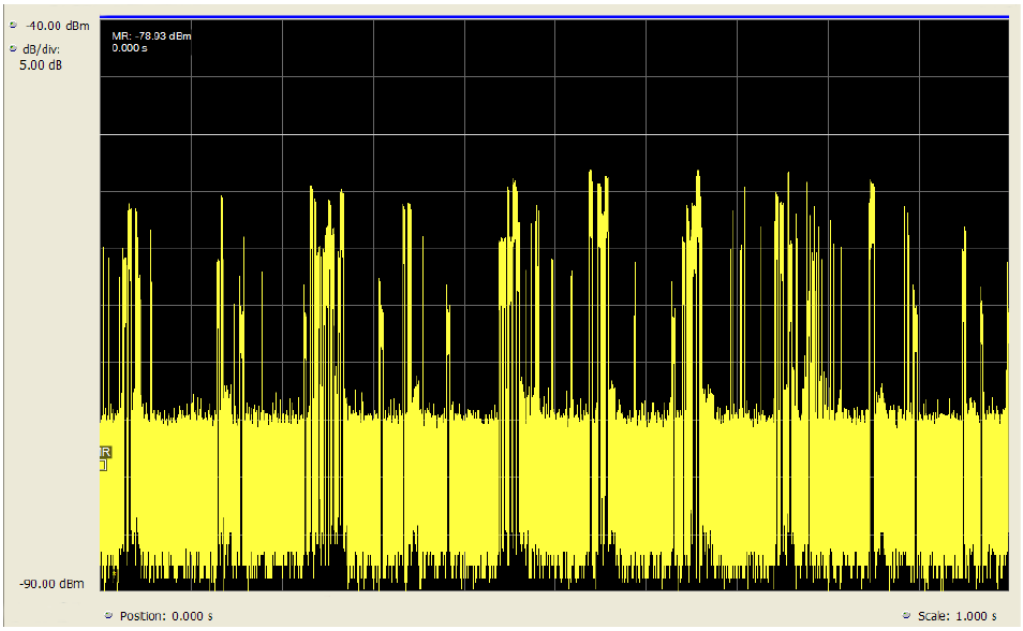


Figura 5.3: Evolución en el tiempo de la potencia para una señal de un canal especifico que sigue el estándar WLAN (frecuencia de 2.412 GHz con un ancho de banda de 5 MHz) [2].

Para agilizar el sistema y sabiendo que la dinámica de la señal es igual para todos los anchos de banda o canales, se ha usado un único conjunto de datos de entrenamiento para entrenar una red. Es decir, solo debe entrenarse una red y, siguiendo el diagrama de la Figura 5.2(b), luego se replica tantas veces como canales hayan que para este ejemplo se usan cuatro. Las cuatro redes en un principio son idénticas ya que los parámetros entrenables o pesos y bias serán iguales. Sin embargo, luego recibirán los datos de test definidos para cada canal por lo que se irán actualizando y aportarán una predicción en base a dichos datos test. En la Figura 5.4 se muestran los datos de entrada usados. En azul claro se representa los datos de entrenamiento y a partir de esto, se observa con otros colores los datos de válidación para los cuatro canales. Un total de 2.000 muestras para cada conjunto de datos de validación se ha definido. Se debe remarcar que aunque las secuencias de elementos se han generado aleatoriamente, luego se han guardado dichas secuencias para analizar estos resultados concretos ya que son excelentes para comentar las diferentes situaciones que pueden

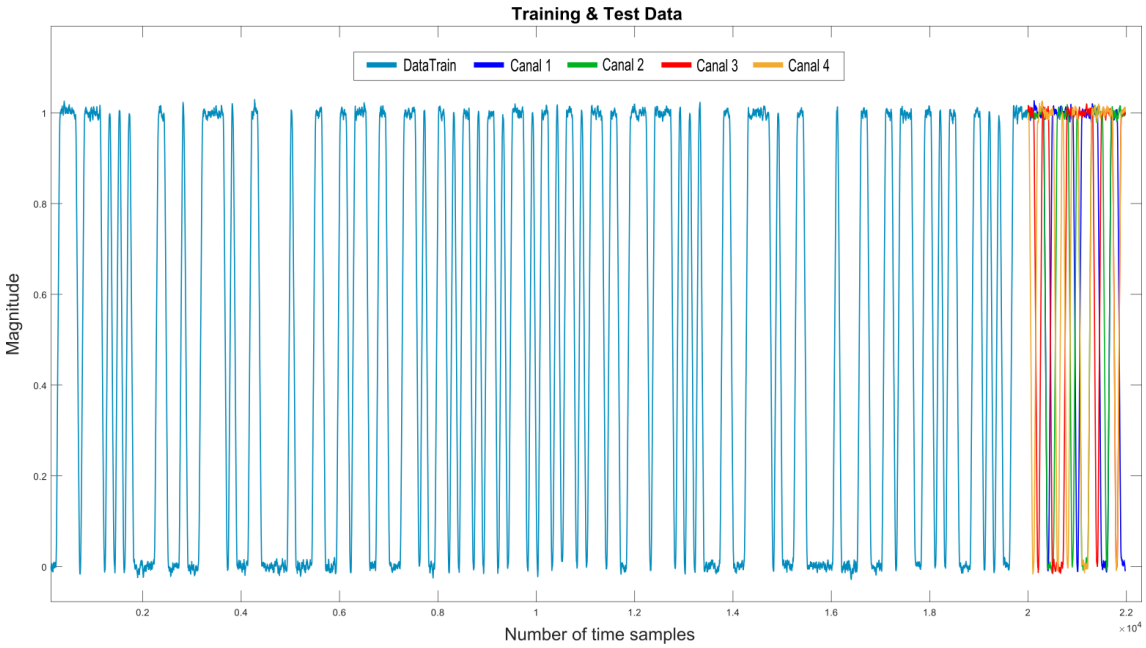


Figura 5.4: *Dataset* para el sistema de selección de bandas. En azul claro se representan los datos para el entrenamiento de las redes LSTM (DataTrain). Las demás señal, representan los datos de validación o test para cada uno de los cuatro canales usados.

darse. Esto se hará en la sección 5.5.

5.3. Entrenamiento de la red LSTM

Siguiendo la sección 4.2 del capítulo anterior, para el entrenamiento de la red LSTM solo se debe tener en cuenta las muestras de entrenamiento almacenadas en una variable que llamaremos X_{train}. Esta guarda todos los datos excepto el último por lo que su dimensión final es de 1×19999. Se crea una nueva variable Y_{train} que será el vector objetivos almacenando desde el segundo dato de entrenamiento (igual a $X_{train}(2)$) hasta el último de ellos eliminado para anteriormente para el vector de entradaX_{train}. Previamente estos datos fueron normalizados.

5.4. Bucle de predicción de la banda ocupacional para los distintos canales de comunicación

El bucle para la predicción de series temporales es el núcleo del sistema que se propone. El objetivo es, que a partir de los datos de entrada definidos y la red LSTM seleccionada y entrenada, obtener una predicción futura de la tendencia de las señales de test. En total serán cuatro señales diferentes, una para cada canal, que irán aumentando o disminuyendo su nivel de ocupación. Tras realizar la predicción se genera la selección del canal y se actúa a consecuencia modificando el filtro de paso de banda que permitirá el paso de las señales de comunicación en el ancho de banda correspondiente al canal libre seleccionado.

Varias restricciones de diseño se han considerado para la implantación del sistema. Una de ellas es establecer un umbral del 50 % de ocupación ya que, como se dijo en la sección 2.2.1, la señal se va a considerar binaria. Dicho de otro modo, se considerará una transición de cambio de nivel de la ocupación cuando se predice que la señal supera el 50 % de los dos niveles asignados, cero como un canal libre y uno para un canal ocupado completamente. La señal pasa de estar libre

a tener una transición de subida si se predice en la secuencia de salida un nivel mayor a 0.5. De forma opuesta, cuando la señal está ocupada y se predice que el nivel estará bajo el nivel de 0.5 próximamente, entonces se pronostica un cambio de estado de bajada. Mediante el desarrollo y pruebas se ha ido seleccionando el valor más adecuado para el número de pasos de predicción M. También a lo largo de la evaluación del selector de bandas, otras consideraciones se han tenido en cuenta y serán comentadas más adelante. A continuación se muestra un cuadro resumen de los pasos seguidos durante el algoritmo de predicción:

Algoritmo de predicción de la banda ocupacional

1. Asignar un número de pasos futuros M a predecir.
2. Actualizar la red con la función *predictAndUpdateState* con los datos de entrenamiento y obtener el primer valor predicho gracias a usar el último valor de entrenamiento antes inutilizado.
3. Normalizar los datos de test para cada canal.
4. Crear una variable *nett* que contiene todas las redes del sistema (un total de cuatro para este ejemplo).
5. Se inician las variables necesarias para la predicción.
6. Se inicia el bucle de predicción. Este se compone de dos bucles anidados; el primero de ellos recorre las muestras o datos de validación y para cada una de estas, se realizan para cada canal los siguiente pasos:
   a) Tomar los datos de test a partir del canal y la muestra actual más los M a predecir.
   b) En la primera iteración, predecir en base a la primera predicción realizada en el paso 2. Además, iniciar una variable estructura que contiene toda la información necesaria para la predicción. Para el resto de iteraciones se usa la función **forecast** creada para que, a través de las nuevas predicciones se evalúen los cambios de transición o cambios de estado. Esta función se detalla más adelante.
   c) Cada 10 iteraciones, representar el valor RMSE y visualizar gráficamente la secuencia de datos de salida predicha comparándola con los datos reales para cada canal.
   d) Si anteriormente en el paso 6(b) se ha detectado cambio de estado (la señal pasa de ocupada a desocupada o viceversa), se obtiene la información necesaria que sería el momento (o muestra) actual, momento en el que se sobrepasa el valor umbral definido y si la transición es de subida o bajada.
   e) Si el canal seleccionado va a ser ocupado y otro canal está o va a ser desocupado, realizar la modificación de los parámetros del biquad del filtro para cambiar al ancho de banda asociado al nuevo canal.
7. Por último se visualiza gráficamente.

Una vez que se han aclarado los pasos más generales del algoritmo de predicción, es de importancia entender cómo se realiza la predicción con la red LSTM. Esto se detalla a continuación enumerando los pasos en la función **forecast** que ha sido creada para este sistema:

Función de predicción forecast:

1. Realizar la predicción con M muestras usando la función *predictAndUpdateState*.
2. Desnormalizar la predicción para poder representarla gráficamente *a posteriori*.

3. Resetear la red LSTM usando la función *resetState* para que no le afecte las actualizaciones hechas en base a las predicciones anteriores ya que estos valores no son los reales y llevan un cierto error.
4. Actualizar de nuevo la red con los valores conocidos, es decir, los de entrenamiento y las muestras del conjunto de test que ya sean conocidas.
5. Implementar un filtro para restringir la detección de cambio de estado. Hasta que no se alcance un valor mínimo (2 %) o máximo (98 %) de la señal no se habilita la búsqueda de un nuevo cambio de estado. Esto evita posibles resultados falsos provocados por el ruido.
6. Si se detecta que la señal sobrepasa el valor umbral, modificar las variables que indicarán en el bucle general de predicción un cambio de estado.
7. Actualizar la información de predicción.

Comparando el algoritmo de predicción junto con la función **forecast** se afirma que la predicción que se establece es una combinación de los dos tipos que se presentan en el ejemplo de la sección 4.3. Por un lado, en la función **forecast** se pronostica la salida con M pasos futuros en base a la predicción hecha en el paso de tiempo anterior. Es decir, se predice la ocupación en el momento $t+1$ usando como dato conocido la predicción de su valor en el momento t y así continuamente hasta completar la secuencia. El motivo es que se suponen conocidos los datos reales hasta el momento actual t pero a partir de él los datos son desconocidos (son datos futuros que aún no han ocurrido). Esta es realmente la esencia del problema. Pero, por el otro lado, tras obtener esta predicción de M pasos futuros, la red se resetea y se actualiza con los valores conocidos hasta el momento actual que son los datos de entrenamiento y los datos de test que hayan sido transcurridos ya. El motivo es ir actualizando la red con los datos que sí son conocidos para ir corrigiendo los errores que introducen las predicciones. Así, para la siguiente iteración, la red no está influenciada por este error y puede dar una nueva predicción lo más ajustada posible a la realidad.

Una vez explicado el bucle de predicción se puede mostrar el flujo del código final del sistema para una visualización más clara de los pasos a seguir para la predicción de series temporales que se presentan en este capítulo. El diagrama es el de la Figura 5.5. En él, pueden verse las decisiones que se toman en el bucle de predicción (no hay que olvidar que estos son los procesos y subprocesos más importantes pero que dentro del código son necesarios muchos más).

5.5. Resultados obtenidos

El número de neuronas ocultas es relevante para las redes LSTM. A mayor número de neuronas, mejor predicción. En el ejemplo que se usaba en la sección 4.3, el número de unidades o celdas LSTM eran 200. Partiendo de este ejemplo similar al que ahora presentamos, se probó a aumentar este número de unidades. Esto se veía reflejado en un mayor coste computacional pero también en una mejor predicción. En primer lugar se aumentó a solo unas pocas unidades más, exactamente a 2^8 o 256 neuronas. Esto no suponía un cambio notorio en la velocidad de cálculo pero los valores RMSE indicaban una mejoría de centésimas. Luego se probó con 500 unidades, pero esto ralentizaba las simulaciones en un factor de 10 como mínimo por lo que se descartó. Finalmente, todas las pruebas han sido realizadas con 256 unidades ocultas para la capa LSTM y ha proporcionado buenos resultados.

También mediante prueba y error se ha obtenido el valor de M (los pasos a predecir). Es fácil pensar que cuanto mayor sea, antes se puede anticipar al cambio de canal por lo que esto es mejor. El valor máximo óptimo en el que no se han obtenido transiciones erróneas ha sido de $M = 60$.

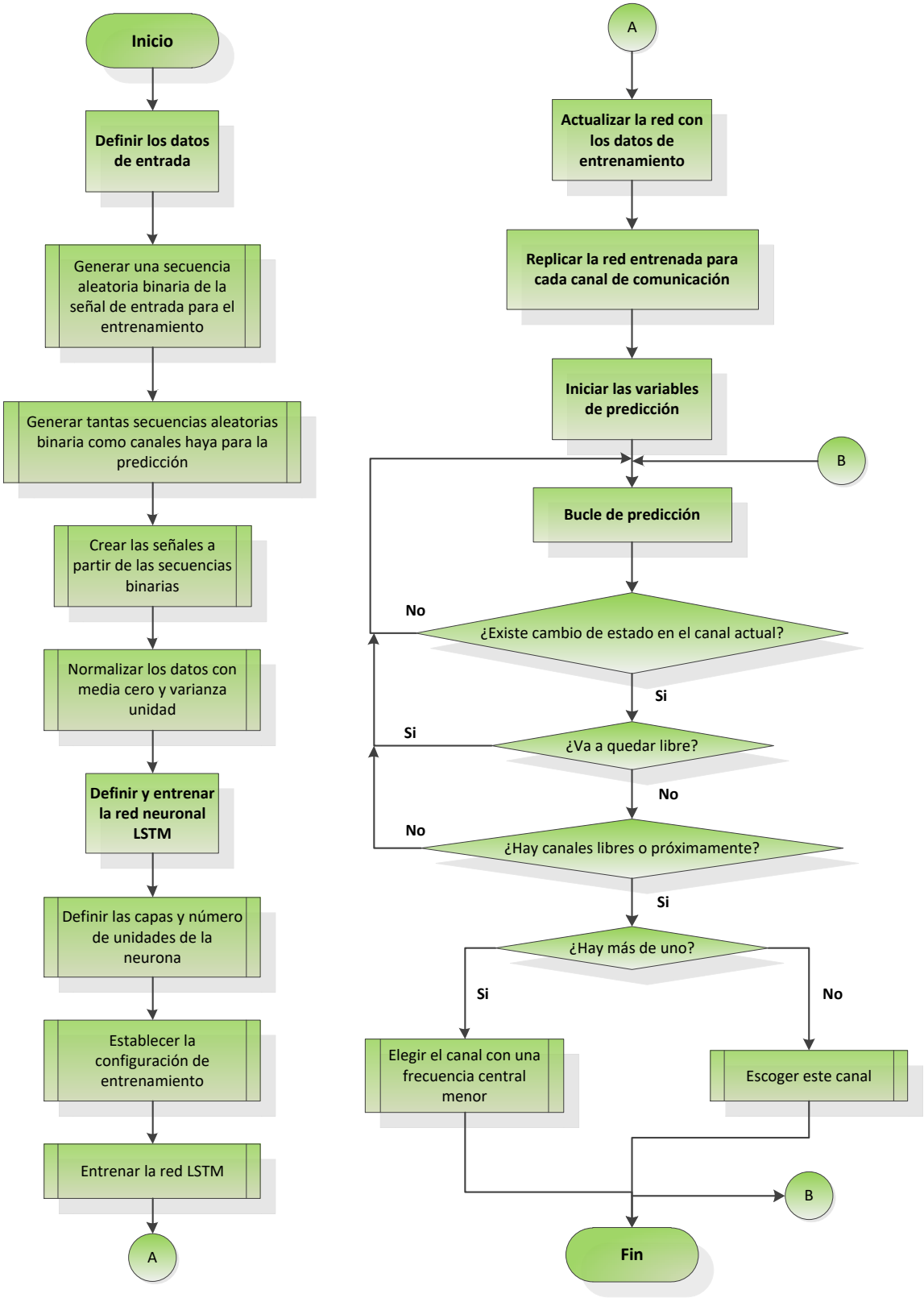


Figura 5.5: Diagrama de flujo de la implantación software del selector de bandas mediante la predicción de series temporales usando redes neuronales LSTM.

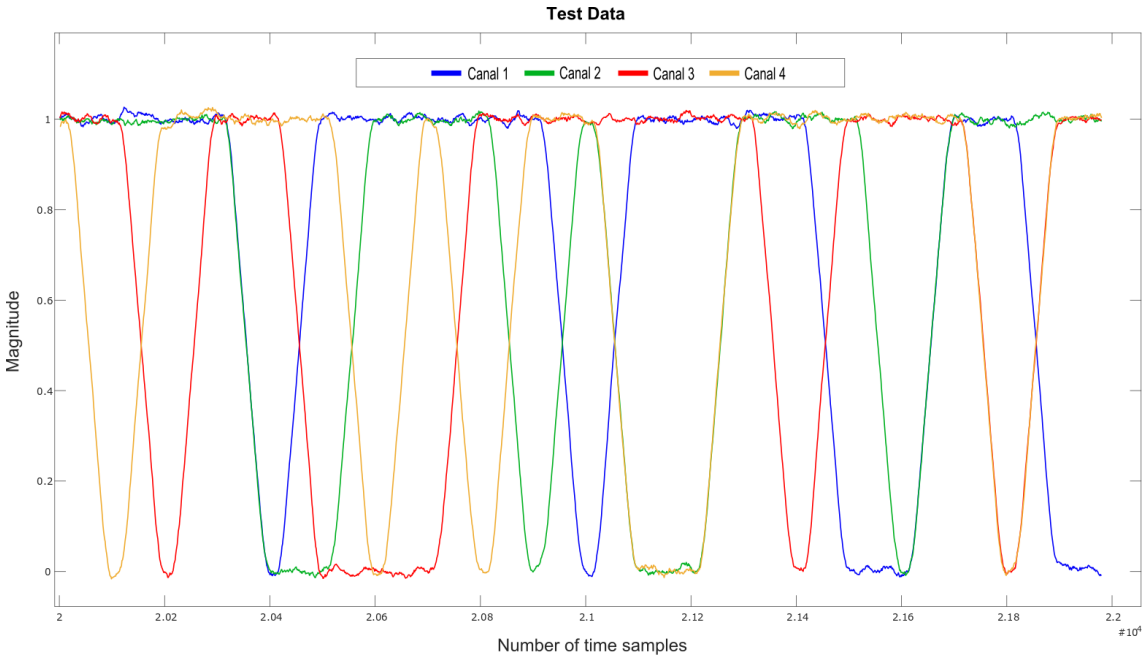


Figura 5.6: Conjunto de datos test o de validación usados para la predicción del selector de banda ocupacional.

La Figura 5.6 muestra el conjunto de datos de test que se ha usado para evaluar el sistema propuesto y obtener los resultados finales (contenido en el fichero **channels.mat**, ver sección 1.4). Estas señales ya fueron representadas en la Figura 5.4 pero se ha considerado necesario ampliarlas para visualizar gráficamente y de forma clara el ejemplo de test. En la Figura 5.7(a)-(d) se muestra un fragmento con las primeras 1.000 muestras de las cuatro señales de test para cada canal. Mientras, en la Figura 5.7(e) se observa el comportamiento de la selección de banda a lo largo de dicho fragmento generado por el bloque de decisión (ver diagrama 5.2(c)). Cada paso de esta señal indica qué canal debe ser seleccionado. Esto es finalmente lo que interesa obtener ya que gracias a ello se modificará la selección de banda mediante el filtro paso de banda. Si observamos detenidamente las Figuras 5.7(a)-(d), podemos observar diferentes situaciones que provocan cambio de canal con t_i donde i enumera dichas situaciones. Todos estos cambios pueden ser explicados si se sigue el flujo que se refleja en el diagrama de la Figura 5.5. En primer lugar es importante decir que se considera como canal seleccionado por defecto al canal número uno como se indica en la gráfica de selección 5.7(e). Luego, en el primer instante de cambio, t_1, el sistema predice que mientras que el canal actualmente seleccionado (canal 1) está ocupado, la ocupación del canal 4 disminuirá por lo que selecciona la banda de frecuencias correspondiente. En t_2, las predicciones dicen que la ocupación de la banda seleccionada 4 aumentará, mientras que disminuirá para el canal 3. Por lo tanto, se selecciona el canal 3. Puede observarse que entre t_2 y t_3 el canal 3 aumenta y esto es predicho por el sistema pero, al no disminuir ningún otro canal durante este tiempo, el selector no realiza ningún cambio de canal para no incrementar un consumo innecesario. En t_3, cuando el sistema predice que dos canales diferentes estarán menos ocupados (1 y 2), se selecciona el primero siguiendo una lista de prioridades en la que se prefieren las bandas de menor frecuencia. Esto es simplemente una decisión de diseño. En t_4, se espera que el canal 1 seleccionado se ocupe de nuevo, por lo que el bloque de decisión compara todas las predicciones, seleccionando el canal 2. Durante los instantes t_5 y t_6, el canal 4 tiene una caída de la ocupación pero debido a que el canal 3 no incrementa su nivel de ocupación, no se realiza el cambio de canal. Este algoritmo funciona continuamente.

En resumen, el sistema predice la tendencia futura de ocupación o desocupación de los cuatro canales. Se ha decidido por diseño, que solo cuando el canal seleccionado va a ser nuevamente ocupado, el selector buscará un canal libre secuencialmente. En el cuadro 5.1 se detalla la información más relevante de los cambios de estado que se producen en estas primeras 1000 muestras de test.

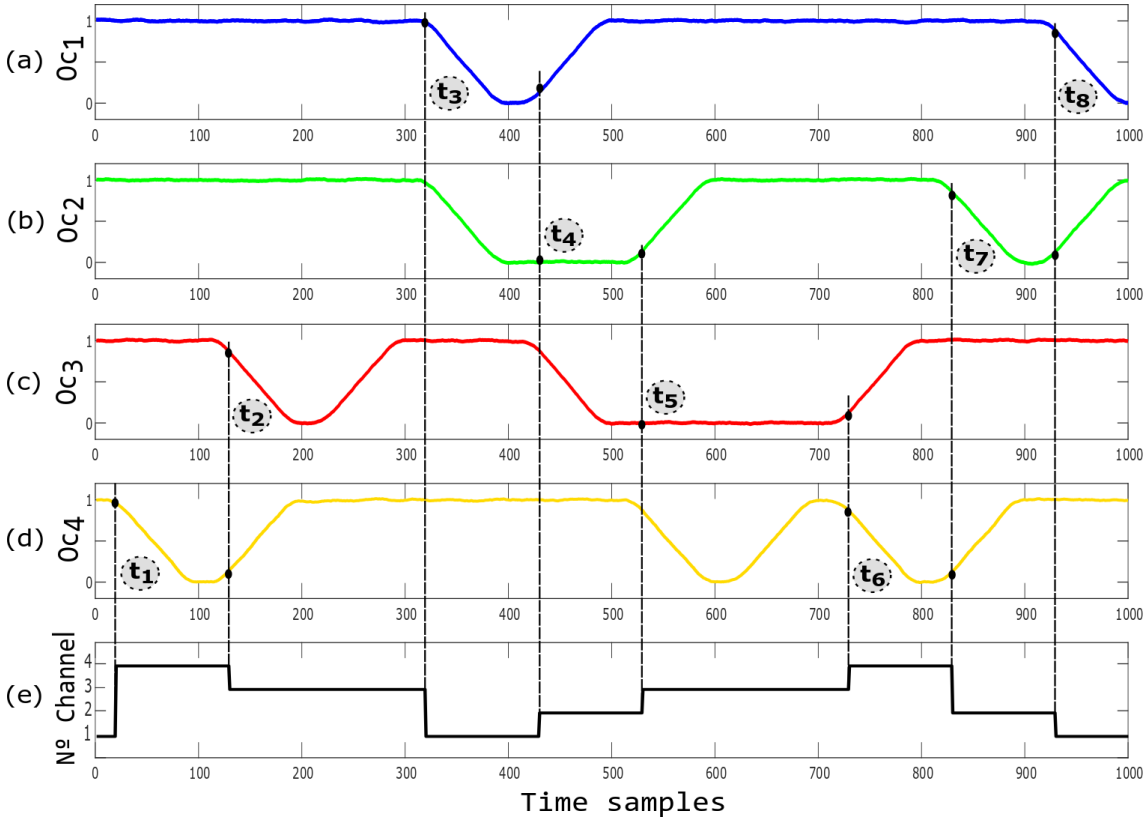


Figura 5.7: Resultados de simulación obtenidos para los cuatro canales donde las señales (a)-(d) representan las medidas de la ocupación de canal en el tiempo y (e) muestra la banda seleccionada.

Canal	**Transición**	**Muestra actual**	**Muestra de transición**	**Pasos anticipados N_p**
4	Bajada	22	55	33
4	Subida	98	157	59
3	Bajada	121	156	35
3	Subida	198	256	58
1	Bajada	322	357	35
2	Bajada	322	356	34
1	Subida	402	456	54
3	Bajada	423	456	33
2	Subida	503	556	53
4	Bajada	523	556	33
4	Subida	600	656	56
3	Subida	699	756	57
4	Bajada	720	756	30
4	Subida	800	856	56
2	Bajada	821	857	36
2	Subida	901	956	55
1	Bajada	922	955	33

Cuadro 5.1: Información de la predicción para la ocupación de los canales usando los primeros 1000 muestras o pasos de tiempo. Muestra el tipo de transición, la subida supone la ocupación del canal y la bajada la desocupación de este. También, la muestra actual en la que se predice el cambio y la muestra o paso de tiempo en la que realmente se sobrepasa el umbral que marca esta transición. La resta de las dos anteriores refleja el número de pasos con los que se esta anticipando la predicción.

La última columna del cuadro 5.1 indica el número de pasos con los que se adelanta la predicción a que ocurra el cambio de estado. A este número de pasos se le llama con la nomenclatura N_p. Dicho de otro modo, N_p representa la anticipación obtenida (en número de muestras de tiempo) para adaptar el transceptor a la mejor banda. Si se observa con detenimiento, este número de pasos es mayor cuando la transición es de subida, es decir, cuando la señal va a estar ocupada con una media de 56 muestras (ver cuadro 5.2). Sin embargo, se se estima de media 34 pasos para la transición de bajada. La transición que indica que el canal va a quedar libre es crucial para realizar el cambio de canal por parte del selector así que, para que este valor sea fiable, se dice que el sistema predice con aproximadamente $N_p = 34$ pasos de anticipación.

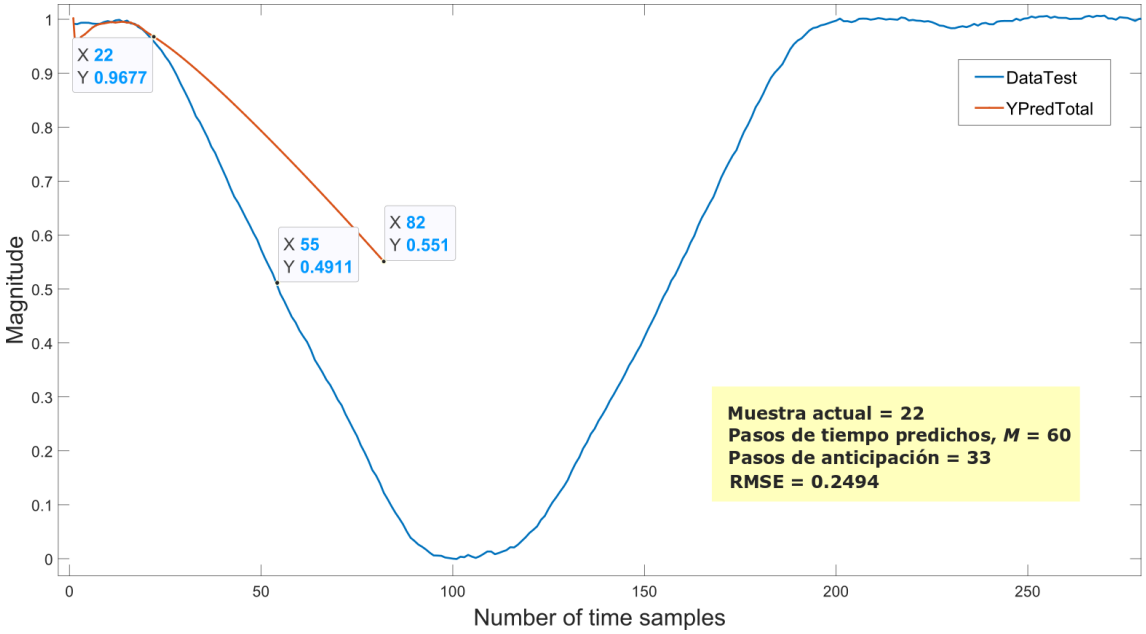
Media de pasos para la subida	**Media de pasos para la bajada**
56	34

Cuadro 5.2: Media de pasos de tiempo con los que se predice la transición de estado de la ocupación de los canales. La subida supone la próxima ocupación del canal y la bajada la desocupación de este.

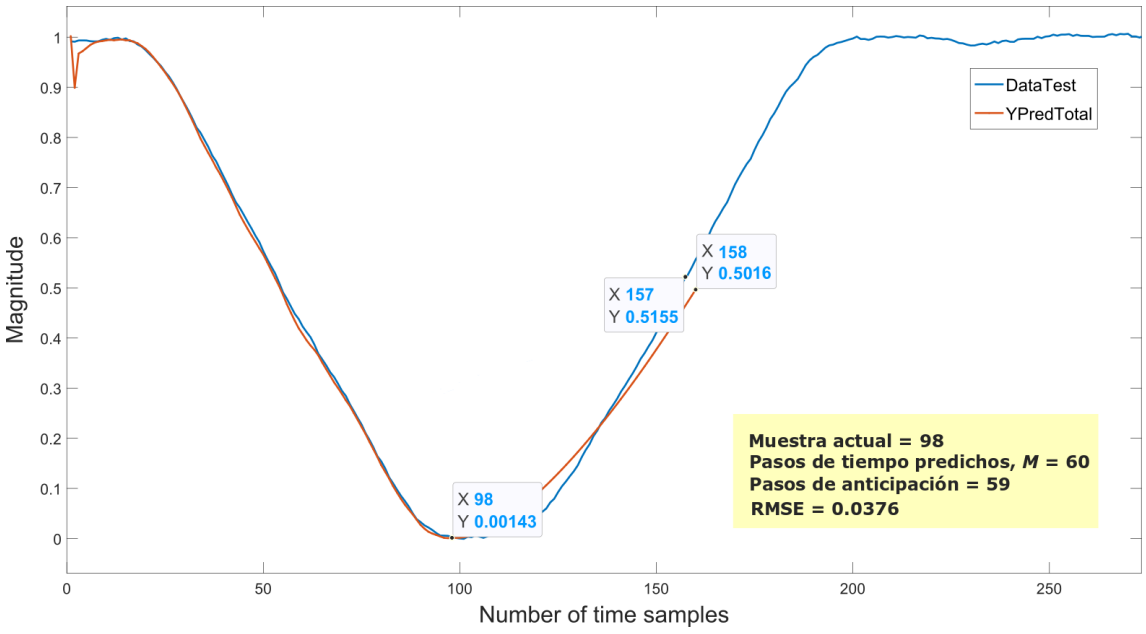
Mediante los valores de error cuadrático medio (RMSE) se obtiene la información de cuánto de buena o mala es la predicción. En los instantes en que no se produce una transición este valor es mayor y se debe a que las componentes aleatorias del ruido son más notorias cuando la señal está en un nivel "fijo". Durante las transiciones, la predicción se ajusta mejor a la señal por lo que se obtienen valores más pequeños de RMSE. Para las 1000 muestras se ha obtenido valores de RMSE dentro del rango [0.03, 0.28] o lo que es lo mismo entre el 3 % y el 28 %. En la Figura 5.8(a) se representa la predicción de una próxima transición de bajada del canal 4 ya que sobrepasa el nivel de umbral establecido del 50 % de ocupación (valor de magnitud del 0.5). El valor RMSE es de casi un 25 % y la predicción se anticipa 33 pasos de tiempo antes de que realmente, en el futuro, se sobrepase el valor umbral. Luego, en la Figura 5.8(b) se muestra este mismo canal cuando se detecta que próximamente va a ser ocupado. Esta vez, el error de la predicción es menor, del 3 % aproximadamente, por lo que esta se ajusta mejor a los M pasos de tiempo futuros de ocupación. Además, se predice con 59 pasos de antelación. Si se observan las primeras muestras predichas en la Figura 5.8(b) se ve que estas sufren de mayor error al cambiar de la señal definida para el entrenamiento a la señal de test por lo que se han tenido que considerar los primeros cinco pasos de tiempo como margen de estabilización. Esta secuencia de salida predicha (de tamaño $M = 60$) tiene un error apreciable pero, al actualizar la red con el valor conocido actual para la próxima predicción, este error no se acumula. Con los cursores y mediante el valor de RMSE se puede afirmar que esta predicción concreta se aproxima bastante bien a la señal real.

En último lugar, cada vez que el bloque de decisión selecciona una determinada banda, modifica los parámetros del filtro BP, cambiando sus frecuencias mínimas y máximas. La Figura 5.9 representa las cuatro funciones de transferencia para una anchura de banda total de 1 GHz (de 2 a 3 GHz). Parte de esta banda se ha dividido en los cuatro canales, cada uno de un ancho de banda igual a 200 MHz empezando el canal 1 con la frecuencia de 2,2 GHz y acabando el canal 4 con los 3 GHz. Este ancho de banda es únicamente un ejemplo para el filtro propuesto pero, a nivel conceptual, podría ser de otro valor.

El tiempo que conlleva la ejecución de este programa depende en gran medida del equipo u ordenador que se esté usando. Para este trabajo se ha usado un ordenador de gama media con 4 GB de memoria RAM y un procesador Intel(R) Core(TM) i7-6500U CPU @ 2.5 GHz, siendo un poca potencia para este tipo de simulaciones. Aún así, de forma clara podemos decir que el entrenamiento de la red neuronal para los datos usados supone un gran tiempo de ejecución de entorno a los 50 minutos. Otros tiempos de ejecución se han obtenido haciendo uso de las funciones `timeit` y `tic-toc` de Matlab y se presentan en el cuadro 5.3. Para comparar se ha usado (por poco tiempo ya que no disponíamos de él) un ordenador más potente con 16 GB de RAM y solo le ha llevado 4 minutos aproximadamente para el entrenamiento completo. Sin embargo, para el bucle de predicción, los tiempos medios son del mismo rango que los recogidos en el cuadro 5.3.



(a)



(b)

Figura 5.8: Gráfica de datos de test para el canal 4 y la predicción realizada en la transición de (a) bajada y (b) subida.

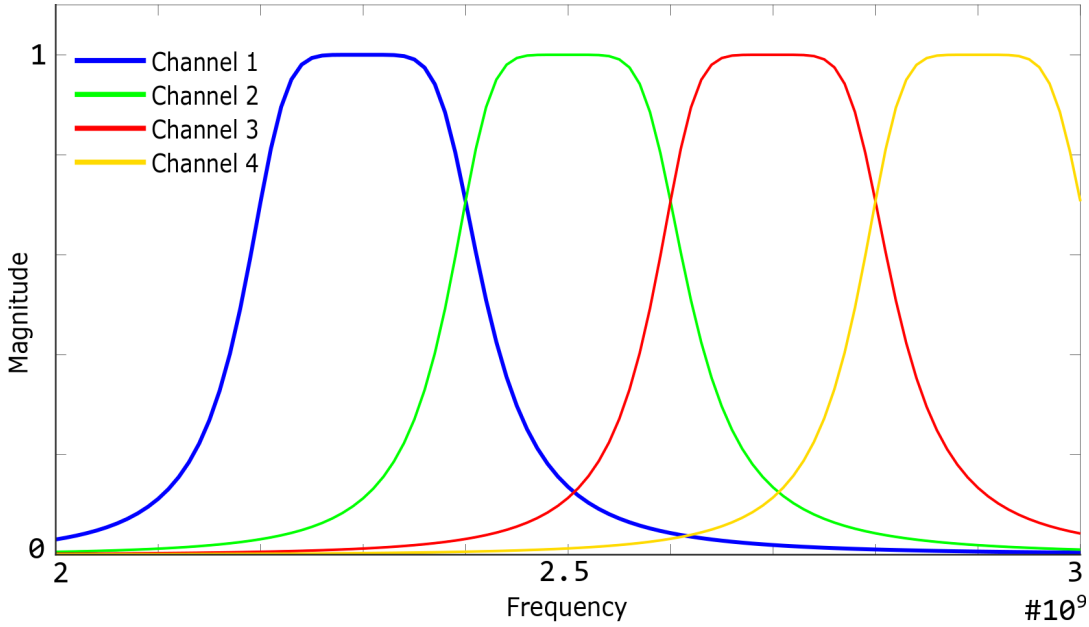


Figura 5.9: Conjunto de las cuatro funciones de transferencia asociadas al filtro BP cuando cada canal es seleccionado.

Proceso	**Tiempo medio de ejecución (s)**
Generar los datos de entrada	1.0396
Definir y entrenar la red	50 min (aprox.)
`forecast`	1.2317
Predecir los 4 canales (primera iteración)	0.7551
Predecir los 4 canales (sin cambio de canal)	5.6064
Predecir los 4 canales (con cambio de canal)	10.2547
Predecir 10 muestras	57.5122

Cuadro 5.3: Tiempos de ejecución para diferentes procesos del selector de banda implementado en Matlab®.

Capítulo 6

Conclusiones y Mejoras

6.1. Conclusiones

Se propone una estrategia novedosa para seleccionar la banda de frecuencia de menor ocupación en un sistema de radio cognitiva. Para ello, se ha implementado cuatro redes LSTM para predecir la evolución futura en el tiempo de la ocupación en cada canal definido. Estas redes proporcionan una secuencia de salida con una media de $N_p = 34$ pasos de tiempo futuros con los que se decide que canal es el más óptimo a ser seleccionado por un filtro paso de banda. Es decir, una vez que se detecta que próximamente el canal actualmente seleccionado va a ser ocupado, el bloque diseñado para seleccionar el canal busca una banda diferente que esté o vaya a estar desocupada. Cuando un cambio de canal es necesario, se modifican los parámetros de un biquad para seleccionar la frecuencia mínima y máxima del nuevo canal a seleccionar (todos los anchos de banda de los cuatro canales son iguales). El filtro cambia su función de transferencia para cada cambio dando un comportamiento óptimo a un sistema de comunicaciones que pretende funcionar con las mínimas interferencias posibles. Esta estrategia ha sido implementada en Matlab y validada usando datos sintéticos para dar una prueba de concepto sobre un determinado sistema radio cognitivo.

Una de las conclusiones más importantes a las que se ha llegado con este trabajo es la envergadura de los datos de entrada y la estructura del sistema que incluye la arquitectura de la red neuronal seleccionada para un buen resultado del sistema. Finalmente, se ha llegado a un compromiso entre definir la señal de entrada de manera realista conociendo cómo cambian las ondas electromagnéticas y su simplificación para obtener una buena predicción. La simplificación se ha basado en filtrar el ruido previamente añadido a la señal y además, tratarla como una señal binaria y no multi-nivel. También es de significado el número de datos de entrenamiento para la red neuronal seleccionada ya que si se define una señal de entrada demasiado filtrada y/o con muchos datos de entrenamiento, la red neuronal puede quedar sobreajustada. Todo ello es fundamental en el área de la inteligencia artificial. Este trabajo abre las puertas a una línea de investigación más ambiciosa que pretende aprender las ventajas de aplicar la inteligencia artificial, tan empleada hoy en día, en uno de los campos de estudio de más interés y en constante desarrollo como son las comunicaciones que tienen gran impacto en nuestra sociedad.

6.2. Mejoras y trabajos futuros

El sistema desarrollado a lo largo de este trabajo no es más que una prueba de concepto que sirve como punto de partida de una línea de investigación mucho más ambiciosa. El objetivo general de esta línea es explorar distintas estrategias para aplicar redes neuronales a sistemas de comunicaciones, principalmente bajo el paradigma de la radio cognitiva. Dentro de este objetivo general, se plantean una serie de ideas, algunas más concretas para desarrollar a corto plazo, y otras

más abiertas que se irán concretando a medio-largo plazo en el marco de esta línea de investigación. A continuación, a partir del sistema de selección dinámica de canal descrito en este documento, se enumeran una serie de mejoras:

- Por una parte, analizar más a fondo el comportamiento de las redes LSTM para optimizar su rendimiento alcanzando un equilibro entre maximizar la capacidad de anticiparse a las transiciones en las señales de entrada (N_p, número de muestras antes de que se produzca realmente una transición) y minimizar los errores de predicción (predecir transiciones que no se producen o no predecir transiciones que sí se producen).
- Mejorar el bloque de decisión para que sea capaz de predecir qué canal tendrá un menor nivel de ocupación en un escenario de señales de ocupación multi-nivel. Con una señal multi-nivel nos queremos referir a que se define la señal de entrada con diferentes niveles de ocupación y no como con una naturaleza binaria como se define en la sección 2.2.1 gracias a la información proporcionada por las bases recogidas en la unión internacional de la telecomunicación (ITU) [2]. Con ello, podría seleccionarse el canal menos ocupado aunque su valor no sea del 0 %. En este trabajo solo se contemplaba un cambio de canal cuando se obtiene una predicción de ocupación total (100 %) próxima del actual canal seleccionado mientras que se predecía que otro iba a estar totalmente desocupado.
- Si el objetivo de la medición es detectar el mayor número posible de emisiones, independientemente de su nivel como se comenta en el punto anterior, es preferible un umbral dinámico que se adapte al nivel de ruido en cada instante. Este ajuste al ruido es crítico y existen varios métodos para hacerlo:
  - *Medición directa del nivel de ruido en una frecuencia no utilizada:* La medición del ruido debe realizarse con los mismos ajustes (tiempo de medición y ancho de banda) que los utilizados para la medición de la ocupación real. La realización más simple de este método es medir el nivel de ruido una vez y utilizar el resultado para la medición de la ocupación total. Si no puede suponerse que el nivel de ruido permanezca constante con el tiempo, es aconsejable incluir un canal (o una frecuencia) adicional a los canales usados únicamente para este fin. Esto asegura que el nivel de ruido se mide cada vez que se vuelve a medir la ocupación real. El umbral final de la medición de la ocupación debe ser superior al nivel de ruido medido con un margen de al menos 3 a 5 dB. De lo contrario, los picos a corto plazo en el nivel de ruido darán lugar a una ocupación "fantasma" o falsa.
  - *Medición directa del nivel de ruido en las franjas de tiempo libre:* El nivel de ruido también puede medirse directamente durante los momentos en que un canal no está ocupado. Se prefiere este método al descrito anteriormente, porque la medición del ruido se realiza en el canal real que se va a predecir para determinar su ocupación. La ventaja, especialmente en las mediciones de ocupación en muchos canales o en toda una banda de frecuencias, es que este método puede tener en cuenta los niveles de ruido dependientes de la frecuencia y el tiempo. Por ejemplo, una emisión fuerte en un canal puede producir un aumento del nivel de ruido en los canales vecinos debido al ruido de fase del transmisor. Al igual que en la medición del nivel de ruido en las frecuencias no utilizadas, los ajustes clave (tiempo, ancho de banda) tienen que ser iguales a los ajustes utilizados para la medición de la ocupación real, y el umbral tiene que estar al menos 3 a 5 dB por encima del nivel de ruido medido. Por las mismas razones mencionadas anteriormente, este método es más preciso si la medición del ruido se realiza cada vez que se vuelve a medir la ocupación real.
  - *Umbral calculado:* Si no se dispone de frecuencias adecuadas no utilizadas ni de tiempos en los que se sabe que un canal no está utilizado, el umbral también puede calcularse a partir de los niveles medidos a lo largo la observación de los canales. Este método solo funciona en las mediciones de ocupación de la banda de frecuencias o en las mediciones de ocupación de múltiples canales con anchos de banda iguales.
- Se ha comprobado durante el desarrollo de este trabajo que los datos de entrada tienen gran peso, sino es lo que más, sobre la predicción de series temporales. Como se ha visto, para

obtener buenos resultados en este tipo de estudios se necesita un gran *dataset* que supone un alto requerimiento de computación. Para este trabajo se ha usado un ordenador simple para realizar las simulaciones por lo que se ha empleado mucho tiempo en ellas ya que este proceso puede ser muy lento por la gran cantidad de operaciones que conlleva el uso de tantos datos o muestras. De forma estimada, unos 90 minutos son requeridos para entrenar y predecir 2000 datos de test con el PC empleado. Se considera para estudios futuros el uso de un ordenador con más potencia computacional y, si es necesario, el uso de *clusters* para realizar simulaciones en varios núcleos en paralelo. Esto facilitaría mucho las simulaciones e impulsaría la investigación.

Además de estas evoluciones directas sobre el sistema desarrollado, se plantean una serie de alternativas para afrontar el mismo problema de forma diferente en el futuro de la línea de investigación. Algunas de ellas son:

- En un futuro se pretende estudiar la posibilidad de sustituir las múltiples redes LSTM que operan en paralelo por una única red multi-variable que procese todas las señales de ocupación de canal de forma conjunta, utilizando posibles correlaciones entre ellas para mejorar la predicción. Esto podría reducir el número de unidades (o neuronas) y usar una sola red con la que se aproveche más el potencial de su arquitectura.
- De gran interés es la aplicación de este modelo a un sistema de comunicaciones más complejo que el filtro paso de banda, que incluya una mayor configurabilidad. Un ejemplo sería usar todos los bloques que se muestran en el diagrama de la Figura 5.1. Otras muchas aplicaciones pueden ser perseguidas ya que, como se ha dicho a lo largo del documento, la predicción de series temporales con redes neuronales puede ser muy versátil y de gran utilidad. Sin embargo, la línea de investigación que se quiere recorrer está directamente relacionada con los sistemas de comunicación y, por tanto, la radio cognitiva.
- Una vez concluidos todos estos estudios, se desea plantear una implementación en hardware del modelo que proporcione mejores prestaciones. Para ello, se haría una primera versión del bloque de predicción y decisión en FPGA, con la finalidad de comprobar la viabilidad de la propuesta y ajustar el diseño. Finalmente, se implementaría en VLSI un prototipo completo de un pequeño sistema de comunicaciones que pueda ser configurado de forma dinámica a partir de un sistema neuronal de predicción de la ocupación de los canales.
- Más allá del caso de estudio utilizado en este trabajo, se estudiarán distintas perspectivas de aplicación de Inteligencia Artificial al diseño de circuitos en general y a los circuitos de comunicaciones en particular. Un campo de estudio muy desarrollado es el diseño automático de circuitos analógicos basado en distintos algoritmos de optimización. Una propuesta interesante es utilizar redes neuronales para realizar la función del algoritmo de optimización. Otra propuesta prometedora es utilizar redes neuronales para configurar un sistema de modulación en un sistema de Radio Cognitiva, optimizando los diagramas de constelación en función de las características sensadas del entorno.

Bibliografía

[1] Chuan Zhang, Yeong-Luh Ueng, Christoph Studer, and Andreas Burg. Artificial intelligence for 5g and beyond 5g: Implementations, algorithms, and optimizations. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 10(2):149–163, 2020.

[2] SM Series. Spectrum occupancy measurements and evaluation. 2018.

[3] David Poole, Alan Mackworth, and Randy Goebel. Computational intelligence. 1998.

[4] ITUR SG05. Draft new report itu-r m.[imt-2020. tech perf req]-minimum requirements related to technical performance for imt-2020 radio interface (s). *ITU-R SG05 Contribution*, 40, 2017.

[5] Marco Giordani, Michele Polese, Marco Mezzavilla, Sundeep Rangan, and Michele Zorzi. Toward 6g networks: Use cases and technologies. *IEEE Communications Magazine*, 58(3):55–61, 2020.

[6] Alessio Zappone, Marco Di Renzo, and Mérouane Debbah. Wireless networks design in the era of deep learning: Model-based, ai-based, or both? *IEEE Transactions on Communications*, 67(10):7331–7376, 2019.

[7] Chunxiao Jiang, Haijun Zhang, Yong Ren, Zhu Han, Kwang-Cheng Chen, and Lajos Hanzo. Machine learning paradigms for next-generation wireless networks. *IEEE Wireless Communications*, 24(2):98–105, 2016.

[8] S. Wu. Key technology enablers of innovations in the ai and 5g era. In *2019 IEEE International Electron Devices Meeting (IEDM)*, pages 36.3.1–36.3.4, 2019.

[9] R. Li, Z. Zhao, X. Zhou, G. Ding, Y. Chen, Z. Wang, and H. Zhang. Intelligent 5g: When cellular networks meet artificial intelligence. *IEEE Wireless Communications*, 24(5):175–183, 2017.

[10] D. Gesbert, D. GÃ¼ndÃ¼z, P. de Kerret, C. R. Murthy, M. van der Schaar, and N. D. Sidiropoulos. Guest editorial special issue on machine learning in wireless communication - part i. *IEEE Journal on Selected Areas in Communications*, 37(10):2181–2183, 2019.

[11] Xiaosi Tan, Weihong Xu, Kai Sun, Yunhao Xu, Yair Be'ery, Xiaohu You, and Chuan Zhang. Improving massive mimo message passing detectors with deep neural network. *IEEE Transactions on Vehicular Technology*, 69(2):1267–1280, 2019.

[12] Junkai Liu, Jienan Chen, Siyu Luo, Shuai Li, and Shengli Fu. Deep learning driven non-orthogonal precoding for millimeter wave communications. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 2020.

[13] Tobias Gruber, Sebastian Cammerer, Jakob Hoydis, and Stephan ten Brink. On deep learning-based channel decoding. In *2017 51st Annual Conference on Information Sciences and Systems (CISS)*, pages 1–6. IEEE, 2017.

[14] Xianbin Wang, Huazi Zhang, Rong Li, Lingchen Huang, Shengchen Dai, Yourui Huangfu, and Jun Wang. Learning to flip successive cancellation decoding of polar codes with lstm networks. In *2019 IEEE 30th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pages 1–5. IEEE, 2019.

[15] Hengtao He, Chao-Kai Wen, Shi Jin, and Geoffrey Ye Li. Deep learning-based channel estimation for beamspace mmwave massive mimo systems. *IEEE Wireless Communications Letters*, 7(5):852–855, 2018.

[16] Alexios Balatsoukas-Stimming. Non-linear digital self-interference cancellation for in-band full-duplex radios using neural networks. In *2018 IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pages 1–5. IEEE, 2018.

[17] Yann Kurzo, Andreas Toftegaard Kristensen, Andreas Burg, and Alexios Balatsoukas-Stimming. Hardware implementation of neural self-interference cancellation. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 2020.

[18] Chance Tarver, Alexios Balatsoukas-Stimming, and Joseph R Cavallaro. Design and implementation of a neural network based predistorter for enhanced mobile broadband. In *2019 IEEE International Workshop on Signal Processing Systems (SiPS)*, pages 296–301. IEEE, 2019.

[19] J. Mitola. Software radios: Survey, critical evaluation and future directions. *IEEE Aerospace and Electronic Systems Magazine*, 8(4):25–36, 1993.

[20] E. Axell, G. Leus, E. G. Larsson, and H. V. Poor. Spectrum sensing for cognitive radio : State-of-the-art and recent advances. *IEEE Signal Processing Magazine*, 29(3):101–116, 2012.

[21] Ye Geoffrey Li and Gordon L Stuber. *Orthogonal frequency division multiplexing for wireless communications*. Springer Science & Business Media, 2006.

[22] Sanjiv Nanda, Shravan K Surineni, and Jay Rodney Walton. Multiple frequency band operation in wireless networks, 19 jul 2011. US Patent 7,983,298.

[23] Embedded Computing Design. Seth Deland. Mathworks. When to use Machine Learning or Deep Learning? `https://www.embedded-computing.com/guest-blogs/when-to-use-machine-learning-or-deep-learning`, 15 de Octubre de 2019.

[24] Andrew Ng. University of Standford. Lecture notes from course tittled "machine learning", September 2019.

[25] J Marín. Series temporales. *Universidad Carlos III de Madrid. Obtenido de http://halweb. uc3m. es/esp/Personal/personas/jmmarin/esp/EDescrip/tema7. pdf*, 2017.

[26] James L McClelland, David E Rumelhart, PDP Research Group, et al. Parallel distributed processing. *Explorations in the Microstructure of Cognition*, 2:216–271, 1986.

[27] Jason Brownlee. *Deep learning for time series forecasting: Predict the future with MLPs, CNNs and LSTMs in Python*. Machine Learning Mastery, 2018.

[28] Francisco Parra. Estadística y machine learning con r. Enero 2019.

[29] Georg Dorffner. Neural networks for time series processing. In *Neural network world*. Citeseer, 1996.

[30] Chaoyun Zhang, Paul Patras, and Hamed Haddadi. Deep learning in mobile and wireless networking: A survey. *IEEE Communications Surveys & Tutorials*, 21(3):2224–2287, 2019.

[31] Anastasia Borovykh, Sander Bohte, and Cornelis W Oosterlee. Conditional time series forecasting with convolutional neural networks. *arXiv preprint arXiv:1703.04691*, 2017.

[32] Jian Bo Yang, Minh Nhut Nguyen, Phyo Phyo San, Xiao Li Li, and Shonali Krishnaswamy. Deep convolutional neural networks on multichannel time series for human activity recognition. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15. AAAI Press, 2015.

[33] Kamilya Smagulova and Alex Pappachen James. A survey on lstm memristive neural network architectures and applications. *The European Physical Journal Special Topics*, 228(10):2313–2324, 2019.

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

[35] Wentao Zhu, Cuiling Lan, Junliang Xing, Wenjun Zeng, Yanghao Li, Li Shen, and Xiaohui Xie. Co-occurrence feature learning for skeleton based action recognition using regularized deep lstm networks. *arXiv preprint arXiv:1603.07772*, 2016.

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

[37] Zhou Zhao, Ashok Srivastava, Lu Peng, and Qing Chen. Long short-term memory network design for analog computing. *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 15(1):1–27, 2019.

[38] Olga Krestinskaya, Khaled Nabil Salama, and Alex Pappachen James. Learning in memristive neural network architectures using analog backpropagation circuits. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 66(2):719–732, 2018.

[39] Jun Liu, Tong Zhang, Guangjie Han, and Yu Gou. Td-lstm: temporal dependence-based lstm networks for marine temperature prediction. *Sensors*, 18(11):3797, 2018.

[40] Kamilya Smagulova, Kazybek Adam, Olga Krestinskaya, and Alex Pappachen James. Design of cmos-memristor circuits for lstm architecture. In *2018 IEEE international conference on electron devices and solid state circuits (EDSSC)*, pages 1–2. IEEE, 2018.

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

[42] Andrew Nicola Edmonds. Time series prediction using supervised learning and tools from chaos theory. 1996.

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