Escuela Superior Politécnica del Litoral

Facultad de Ingeniería en Electricidad y Computación

Diseño y análisis de un algoritmo de predicción del espectro basado en Machine Learning para la banda ISM 2.4GHz

INGE-2513

Proyecto Integrador

Previo la obtención del Título de:

Ingeniero en Telecomunicaciones

Presentado por:
Vera Cabello David Josué
Riofrio Muñoz Lilly Cristina

Guayaquil – Ecuador
Año: 2024

Dedicatoria

A mi papá David Vera, el ingeniero que siempre me alentó a seguir mis sueños. A mi mamá Janneth Cabello, la maestra que me enseñó el valor de la educación.

A mi novia Beberlym Goya, mi cómplice en este sueño que es parte de más sueños por cumplir.

A mis hermanos Juan e Iván, mis compañeros de juegos y confidentes.

A mi abuela Francisca Cusme, la tejedora de mis sueños más hermosos. A mi abuela Juana Franco, la jardinera que cultivó en mí la paciencia y la perseverancia.

A mis abuelos paterno y materno, a mis tías, mi familia extendida que siempre me ha arropado con su cariño.

David Josué Vera Cabello

Dedicatoria

A mi padre William Riofrio y mi madre Ivonne Muñoz, por su amor incondicional, su guía y su incansable apoyo en cada paso de mi vida. Gracias por inculcarme la importancia del esfuerzo y la perseverancia. A mi hermana Paula Riofrio, por compartir conmigo cada sueño, cada risa y cada reto.

A mis abuelos maternos, Hermes Muñoz y Lilia Pogo, cuya sabiduría y cariño han sido fundamentales en mi vida, dejándome un legado de valores que siempre atesoraré.

A mi mejor amiga Angie García, mi mejor amigo Daniel Villavicencio, y Boris Piyasagua, por estar a mi lado en cada momento, brindándome su amistad, comprensión y su fortaleza en los momentos más importantes de este viaje.

Este logro es también de ustedes.

Lilly Cristina Riofrio Muñoz

Declaración Expresa

Nosotros Vera Cabello David Josué y Riofrio Muñoz Lilly Cristina acordamos y reconocemos que:

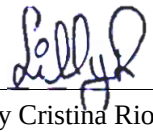
La titularidad de los derechos patrimoniales de autor (derechos de autor) del proyecto de graduación corresponderá al autor o autores, sin perjuicio de lo cual la ESPOL recibe en este acto una licencia gratuita de plazo indefinido para el uso no comercial y comercial de la obra con facultad de sublicenciar, incluyendo la autorización para su divulgación, así como para la creación y uso de obras derivadas. En el caso de usos comerciales se respetará el porcentaje de participación en beneficios que corresponda a favor del autor o autores.

La titularidad total y exclusiva sobre los derechos patrimoniales de patente de invención, modelo de utilidad, diseño industrial, secreto industrial, software o información no divulgada que corresponda o pueda corresponder respecto de cualquier investigación, desarrollo tecnológico o invención realizada por mí/nosotros durante el desarrollo del proyecto de graduación, pertenecerán de forma total, exclusiva e indivisible a la ESPOL, sin perjuicio del porcentaje que me/nos corresponda de los beneficios económicos que la ESPOL reciba por la explotación de mi/nuestra innovación, de ser el caso.

En los casos donde la Oficina de Transferencia de Resultados de Investigación (OTRI) de la ESPOL comunique los autores que existe una innovación potencialmente patentable sobre los resultados del proyecto de graduación, no se realizará publicación o divulgación alguna, sin la autorización expresa y previa de la ESPOL.

Guayaquil, 10 de septiembre del 2024.

David Josué Vera Cabello



Lilly Cristina Riofrio Muñoz

Evaluadores

Firmado electrónicamente por:
VERONICA ALEXANDRA SOTO VERA

MsC. Verónica Alexandra Soto Vera

Profesor de Materia

Firmado electrónicamente por:
WASHINGTON ADOLFO MEDINA MOREIRA

PhD. Washington Adolfo Medina Moreira

Tutor de proyecto

Resumen

Este proyecto aborda el diseño y análisis de un algoritmo de predicción de espectro basado en técnicas de Machine Learning (ML) para la banda ISM de 2.4GHz, ampliamente utilizada en aplicaciones inalámbricas como Wi-Fi, Bluetooth y dispositivos IoT. La creciente congestión de esta banda ha motivado la implementación de un algoritmo de predicción del uso del espectro para mejorar la eficiencia en entornos de alta densidad de usuarios. El trabajo se centra en el desarrollo de un modelo de redes neuronales tipo LSTM (Long Short-Term Memory) para predecir la disponibilidad del espectro. El sistema fue entrenado con datos recolectados en entornos reales y se validó con pruebas controladas. Los resultados obtenidos demuestran la eficacia del modelo con un nivel de precisión del 80%, lo que permite optimizar el uso del espectro en redes inalámbricas, reduciendo interferencias y mejorando la calidad de servicio. Finalmente, se presentan recomendaciones para mejorar la precisión del modelo y su aplicabilidad en futuros estudios sobre redes cognitivas y aplicaciones de IoT.

Palabras Clave: algoritmo de predicción, Machine Learning, LSTM, espectro

Abstract

This project focuses on the design and analysis of a spectrum prediction algorithm based on Machine Learning (ML) techniques for the 2.4GHz ISM band, widely used in wireless applications such as Wi-Fi, Bluetooth, and IoT devices. The increasing congestion in this band has driven the development of a prediction algorithm to enhance spectrum efficiency in high-density user environments. The work involves the development of a Long Short-Term Memory (LSTM) neural network model to predict spectrum availability. The system was trained with real-world data and validated through controlled tests. The results show the effectiveness of the model with an accuracy rate of 80%, optimizing spectrum usage in wireless networks, reducing interference, and improving service quality. Finally, recommendations are provided to enhance the model's accuracy and its applicability in future studies on cognitive networks and IoT applications.

***Keywords**: prediction algorithm, Machine Learning, LSTM, spectrum*

Índice General

Abreviaturas

ADAM Vínima Dinámico al Medio

AI Inteligencia Artificial

IoT Internet de las Cosas

ESPOL Escuela Superior Politécnica del Litoral

ISM Industrial, Scientific and Medical

KNN K-Nearest Neighbors

ML Machine Learning

MST Vínim de Envergadura Vínima

RC Radio Cognitiva

RL Aprendizaje por refuerzo

Índice de Tablas

Índice de Figuras

Índice de Anexos

Capítulo 1

1.1 Introducción

En el panorama actual de las comunicaciones inalámbricas, la predicción del espectro en bandas específicas se ha convertido en una herramienta fundamental para optimizar el uso de los recursos disponibles y mitigar la interferencia entre diferentes usuarios [1]. En este contexto, el Machine Learning (ML) ha emergido como una alternativa para abordar este desafío, ofreciendo soluciones eficientes y precisas para la predicción del espectro en bandas como la de 2.4 GHz [1].

La banda ISM (2.4 GHz) al ser una banda libre es utilizada para diversas aplicaciones inalámbricas, incluyendo redes Wi-Fi, Bluetooth y dispositivos IoT [2]. Sin embargo, el uso creciente de estos dispositivos ha generado un aumento significativo en la congestión del espectro, lo que puede ocasionar interferencias y degradación del rendimiento [3].

Para abordar este desafío, el análisis del comportamiento del espectro, en la banda de 2.4GHz utilizando técnicas de ML ha emergido como una herramienta prometedora para optimizar el uso de las radios cognitivas (RC), al permitir la recolección de datos de los canales de la banda ISM que se tomarán como alternativos para mejorar la eficiencia y rendimiento de las comunicaciones inalámbricas [4].

Este proyecto presenta una descripción general de la predicción del espectro en la banda de 2.4GHz utilizando ML, destacando sus beneficios, aplicaciones potenciales y desafíos.

1.2 Descripción del problema

El presente trabajo busca abordar el desarrollo de una solución innovadora para una eficiente gestión del espectro radioeléctrico en la frecuencia asignada de 2.4GHz en entornos de alta densidad de usuarios en recintos o instituciones cerradas. En este caso, la banda de 2.4 GHz, que es utilizada de manera común para redes Wi-Fi, Bluetooth y dispositivos inalámbricos, presenta limitaciones de capacidad cuando los usuarios de servicios se multiplican sobre la misma,

en ausencia de regulación de espacios, lo cual genera la aparición de cuellos de botellas, que se manifiesta con una baja de la velocidad de datos, latencia y la frecuente caída de la calidad de la red.

1.3 Justificación del problema

Se plantea la propuesta de implementación de un algoritmo de ML, el cual, a través de la observación de la serie de tiempo correspondiente, realizará la predicción de los instantes de disponibilidad de espectro en la banda de frecuencia en cuestión. Tal predicción facilitará la comunicación de la información en estos instantes sin sobresaturar el medio, lo que permitirá optimizarse. Asimismo, se proyecta alcanzar una confiabilidad en comunicación de por lo menos un 80%, lo cual otorgaría una percepción más eficiente del uso del espectro radioeléctrico en cuestión de fiabilidad comunicacional presente en la banda ISM.2.4GHz. Con ello, se tendrá una mejor conectividad, velocidad y ancho de banda mayor, permitiendo la atención de grandes masas de usuarios y escalabilidad de redes de comunicación como las SMA, TV y otras. Por lo tanto, se quiere hacer énfasis en que la aplicación de este algoritmo representa un gran paso para paliar los desafíos en conectividad presentes y futuros.

1.4 Objetivos

1.4.1 Objetivo general

Analizar la predicción del espectro de la banda ISM 2.4Ghz en ambientes internos a través de técnicas de Machine Learning para el uso eficiente de dicho rango de frecuencia.

1.4.2 Objetivos específicos

1. Capturar datos de señales de 2.4GHz en un entorno real, asegurando la calidad y representatividad de estos para el estudio.

2. Limpiar los datos recolectados con el fin de eliminar ruido e inconsistencias, y utilizarlos adecuadamente en el desarrollo del modelo.
3. Diseñar un modelo de predicción adecuado e incorporar los datos adquiridos para su debida análisis y predicción.
4. Transformar los datos mediante el uso de modulación para lograr una predicción más certera.
5. Implementar el modelo de predicción en un entorno real y evaluar su rendimiento y precisión en dichas condiciones.

1.5 Marco teórico

1.5.1 Estado del arte

Se han realizado estudios sobre la utilidad de presumir conocer el comportamiento de la disponibilidad espectral; en [5] se propone un acceso al espectro basado en un modelo de cadenas de Markov modificado a tres estados, obteniéndose mejoras en el acceso a canales disponibles, especialmente en lugares donde la relación señal ruido es baja. Otro estudio propone un marco referencial para tomar decisiones sobre el acceso al espectro en comunicaciones satelitales considerando la operación de sistemas jammers cognitivos [6]. Así mismo, en [7] se realiza un estudio donde se muestran los beneficios de operar sistemas heterogéneos basados en la cooperación de los espacios en blanco de televisión (TVWS) y sistemas Wi-Fi para transmitir ráfagas de video en forma adaptativa y dinámica. Este mismo estudio recomienda el uso de los TVWS para cobertura interna de sistemas cognitivos, debido a sus características de propagación.

Trabajos recientes sobre predicción de disponibilidad espectral basados en redes neuronales [8] también utilizan la información histórica del comportamiento de los usuarios primarios, demostrando además que la predicción combinada con el sensado del espectro mejora los resultados esperados. Actualmente, los esfuerzos de predicción espectral se los realiza con

modelos Long Short-Time Memory (LSTM) puesto que controlan mejor los desvanecimientos o crecimientos exponenciales de los gradientes, y además brindan oportunidades para nuevos retos e ideas [9].

1.5.2 Banda ISM

Las bandas de frecuencia libre (en inglés, ISM bands, de Industrial, Scientific and Medical) son porciones del espectro radioeléctrico asignadas internacionalmente para el uso sin licencia de dispositivos que operan en los ámbitos industrial, científico y médico [10]. Entre estas bandas, la de 2,4 GHz sobresale por su amplia adopción en diversas aplicaciones debido a sus características ventajosas.

La banda ISM de 2,4 GHz abarca un rango de frecuencias que va desde los 2.400 MHz hasta los 2.4835 MHz, dividiéndose en 14 canales de 1 MHz cada uno [11]. Esta banda presenta características ventajosas descritas en la tabla 1.1, para la comunicación inalámbrica:

Tabla 1.1

Ventajas de la banda 2.4 GHz

Disponibilidad global extendida	La mayoría de los países cuentan con las mismas reglas para el uso de la banda de 2.4GHz, lo que facilita que los dispositivos funcionen en diferentes regiones. [12].
Bajo costo	Los componentes electrónicos necesarios para operar en la banda ISM de 2,4 GHz son relativamente económicos, lo que impulsa la adopción de esta tecnología en dispositivos de bajo costo.
Simplicidad de uso	El uso de la banda ISM de 2.4 GHz para operaciones no necesita un permiso especial, lo que facilita la configuración de redes y dispositivos inalámbricos.

1.5.2.1 Consideraciones sobre la congestión del espectro

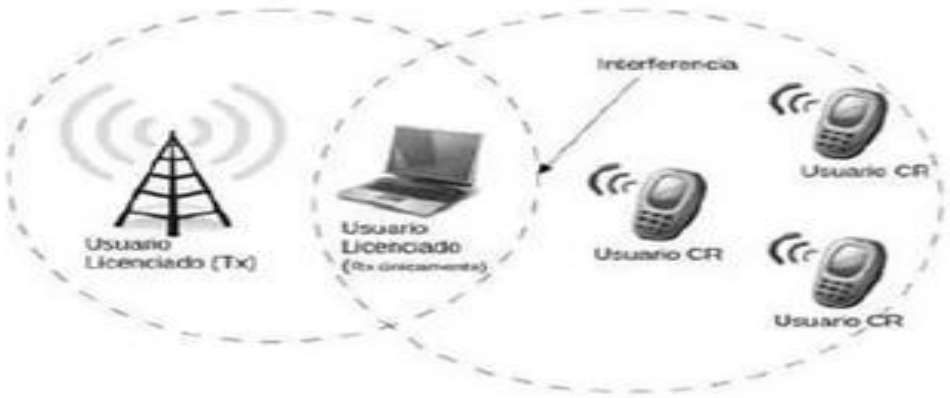
El auge de la banda ISM 2.4 GHz ha provocado un incremento en la saturación del espectro, lo cual podría derivar en interferencias y un impacto negativo en el funcionamiento de los dispositivos. Para enfrentar este desafío, se han implementado diversas estrategias de acceso al medio y protocolos de comunicación que facilitan una distribución eficiente de la banda entre los dispositivos.

1.5.3 Radio cognitivo

La radio cognitiva (RC) es una tecnología disruptiva que revoluciona el paradigma de las comunicaciones inalámbricas. A diferencia de los sistemas tradicionales, los dispositivos cognitivos aprenden y se adaptan al entorno radioeléctrico para optimizar el uso del espectro y evitar interferencias [13]. Esta capacidad se logra mediante la integración de técnicas de detección de espectro, toma de decisiones y reconfiguración.

Figura 1

Operación de RC



En la Figura 1, se muestra un ejemplo de operación de RC, lo que indica que los sistemas móviles están escuchando el medio, mientras que los equipos principales mantienen una comunicación, los sistemas de RC esperan el momento para entregar la comunicación con el transistor (Tx) al mismo tiempo que lo hace el equipo principal de receptor (Rx).

1.5.3.1 Funcionamiento

Los sistemas de RC monitorean continuamente el espectro disponible, identificando oportunidades de acceso en frecuencias sin uso o subutilizadas [14]. Basándose en esta información, el dispositivo cognitivo ajusta sus parámetros de transmisión, como potencia, frecuencia y ancho de banda, para aprovechar estos huecos espectrales de manera eficiente. Este proceso dinámico ofrece una serie de ventajas significativas, que se muestran en la Tabla 1.2, las mismas que permite una coexistencia pacífica con los usuarios primarios licenciados, sin afectar sus operaciones.

Tabla 1.2

Beneficios de RC

Mayor eficiencia del espectro	Permite un uso más eficiente del espectro radioeléctrico, un recurso escaso y valioso.
Mayor capacidad de red	Facilita la conexión de más usuarios y dispositivos a la red inalámbrica.
Reducción de la interferencia	Minimiza la interferencia entre diferentes sistemas inalámbricos, mejorando la calidad de las comunicaciones.
Flexibilidad y adaptabilidad	Permite a los dispositivos cognitivos adaptarse a entornos cambiantes y a nuevas demandas de comunicación [15].

La Tabla 1.3 muestra la amplia gama de aplicaciones que posee RC en diversos sectores, como lo son los siguientes:

Tabla 1.3

Aplicaciones de RC

Comunicaciones móviles	Redes celulares de próxima generación (5G, 6G), Internet de las Cosas (IoT).
Redes de emergencia	Comunicaciones críticas en situaciones de desastre o interrupciones del servicio.
Comunicaciones militares	Defensa, operaciones de seguridad pública.
Redes cognitivas	Redes inalámbricas autoorganizadas y optimizadas [15].

1.5.4 Acceso dinámico al medio

Dynamic Access Medium (ADAM) es una tecnología de acceso a redes inalámbricas que permite a los dispositivos utilizar de manera eficiente el espectro de radiofrecuencia disponible. La tecnología se basa en la detección del espectro para identificar oportunidades de acceso en frecuencias no utilizadas o subutilizadas, también conocidas como brechas de espectro [16]. A diferencia de los sistemas tradicionales que operan en frecuencias fijas, los dispositivos ADAM se adaptan dinámicamente al entorno radioeléctrico para optimizar el uso del espectro y evitar interferencias.

Para detectar los huecos espectrales en la banda primaria, los dispositivos ADAM emplean técnicas como el ciclado de frecuencias, el escaneo y la detección de energía [17]. Una vez identificado un hueco espectral, el dispositivo ADAM evalúa su calidad y solicita permiso para acceder a él. Si la solicitud es aprobada, el dispositivo puede transmitir sus datos en la frecuencia asignada.

1.5.5 Machine Learning y su algoritmo MST

Machine Learning (ML) o también conocido como aprendizaje autónomo, es parte de la inteligencia artificial (IA) utilizada en fotografía, arte y algunas ciencias e ingeniería. Aplicando esta tecnología se logrará mejorar el rendimiento a tareas específicas permitiendo a este sistema informático aprender grandes conjuntos de datos y de esta manera podrá identificar patrones, hacer predicciones y tomar decisiones autónomas.

Existen diferentes tipos de ML de ellos existen tres principales como son:

1.5.5.1 Aprendizaje supervisado

En este enfoque, los algoritmos se entrenan con conjuntos de datos etiquetados, donde cada ejemplo de datos tiene una etiqueta asociada que indica la categoría o valor correcto [18].

Los algoritmos aprenden a relacionar las características de los datos con las etiquetas correspondientes, lo que les permite realizar tareas como clasificación, regresión y predicción.

Algunos ejemplos de algoritmos de aprendizaje supervisado incluyen: Regresión lineal, K-Nearest Neighbors (KNN) y Árboles de decisión.

1.5.5.2 Aprendizaje no supervisado

En el aprendizaje no supervisado, los algoritmos aprenden de datos sin etiquetas. El algoritmo tiene como objetivo encontrar patrones o estructuras ocultos en los datos sin instrucciones específicas [19].

1.5.5.3 Aprendizaje por refuerzo

El aprendizaje por refuerzo (RL) es un método de aprendizaje automático en el que un agente aprende a tomar las mejores decisiones en un entorno para obtener la mayor recompensa con el tiempo [20]. A diferencia del aprendizaje supervisado, donde el agente recibe ejemplos de

entrada y salida, en el RL, el agente explora el entorno y recibe recompensas o penalizaciones en función de sus acciones.

Dentro del amplio espectro de algoritmos de machine learning, como las redes neuronales, los más utilizados para el estudio del espectro son aquellos enfocados en series temporales, como el MST.

Una gestión eficiente del espectro radioeléctrico es fundamental para garantizar el correcto funcionamiento de las redes de comunicación inalámbrica. La capacidad de predecir el comportamiento del medio mediante algoritmos, en este caso el MST junto con el análisis de series temporales, nos proporciona información crucial sobre los patrones y tendencias del uso del espectro, mejorando así la precisión de las predicciones.

1.5.6 Aplicación del algoritmo LSTM

Para la predicción del uso del espectro radio, se implementa el algoritmo LSTM (Árbol de Envergadura Mínima) mediante la construcción de un grafo donde cada nodo representa un canal de espectro y cada arista representa la interferencia existente entre dos canales [21]. El peso de estas aristas se calcula en función de la magnitud de las interferencias.

El objetivo del algoritmo es encontrar un subconjunto de aristas que conecte todos los canales de radio sin crear ciclos, minimizando al mismo tiempo el peso total de las aristas. Esta representación identifica los canales más eficientes, minimizando las interferencias y optimizando la utilización del espectro.

La combinación del algoritmo LMST con el análisis de series temporales permite mejorar la precisión de las predicciones. Al identificar patrones y tendencias en el uso del espectro a lo largo del tiempo, se puede ajustar el peso de las aristas en el grafo MST para reflejar la demanda prevista del espectro en cada canal.

Esta metodología resulta especialmente útil para la banda ISM, específicamente la banda de 2.4 GHz, en entornos con propagación interna. La aplicación este algoritmo en estos escenarios ofrece importantes ventajas como la eficiencia, la precisión, la adaptabilidad y la escalabilidad [21].

1.5.7 Técnicas estadísticas para el procesamiento de datos

El procesamiento de datos es un aspecto fundamental en la predicción del espectro, ya que permite minimizar errores e inconsistencias en los datos para un mejor desempeño de los algoritmos de aprendizaje automático [22, 23]. En este contexto, la tabla 1.4 presentan las algunas de estas técnicas estadísticas que ayudarán al preprocesamiento de la data.

Tabla 1.4

Técnicas estadísticas de procesamientos de datos

Limpieza de datos	La presencia de errores o valores atípicos pueden afectar negativamente el rendimiento de los modelos de predicción, por lo que su eliminación es crucial para garantizar la confiabilidad de los resultados [24, 25].
Normalización	Su objetivo es transformar los datos a una escala común, lo que facilita su comparación y análisis [26, 27]. Existen diversos métodos de normalización, como el escalado min-max y el z-score, cada uno con sus propias ventajas y desventajas.
Transformación de características	Aplicación de transformaciones matemáticas a las características de los datos para mejorar su utilidad en el proceso de aprendizaje automático. Algunas transformaciones comunes incluyen la estandarización, la logaritmación y la discretización.

Las tres tecnologías anteriores son la base del procesamiento de datos en la predicción del espectro. Su aplicación adecuada puede producir modelos de predicción precisos y confiables que son esenciales para diversas aplicaciones en el campo.

Capítulo 2

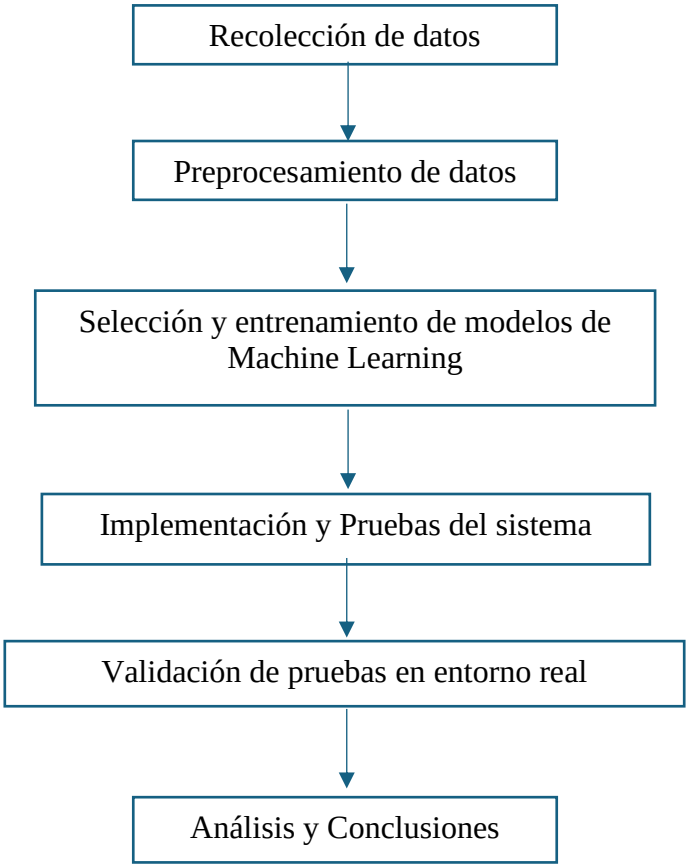
2.1 Metodología

Este trabajo se desarrolla en torno a una metodología centrada en la obtención de predicciones de espectro, lo más precisas y confiables posibles, en la banda de la ISM 2.4GHz en tiempo real, optimizando así el uso de esta tan limitado recurso y optimizando también el rendimiento de las comunicaciones inalámbricas. La Figura 2 muestra los pasos metodológicos que se siguieron para la realización de las correspondientes pruebas y validaciones de datos para el cumplimiento de este objetivo.

2.2 Diagrama de bloques del sistema

Figura 2

Diagrama de bloques del sistema



2.2.1 Recolección de datos

Se lleva a cabo un estudio detallado del espectro electromagnético dentro del establecimiento “Sweet and Coffee” (Anexo 1), utilizando un analizador espectral móvil PSA 6005 de última generación (Anexo 2). Este equipo, comparable a los sistemas de radio cognitiva, permite capturar de manera precisa y detallada la respuesta de frecuencia y potencia del espectro en diversos puntos y momentos. Las mediciones se realizaron en un entorno denso, lo que permitió evaluar el comportamiento del espectro en condiciones reales. Los datos obtenidos, almacenados en formato log, brindan información valiosa sobre la distribución de energía en diferentes bandas de frecuencia y son fundamentales para el desarrollo de modelos de ML precisos. Al realizar la recolección de datos en un escenario interno, se garantiza la representatividad de los modelos de ML, lo que a su vez mejora la capacidad de estos para predecir y analizar el comportamiento del espectro en diferentes situaciones.

2.2.2 Preprocesamiento de datos

Se implementa un proceso de preprocesamiento de datos, que incluye:

- Limpieza de datos para eliminar valores inconsistentes o erróneos.
    - Para ello, se implementa límites de varianza ±2.5, con el fin de reducir datos aberrantes y de esta manera lograr una mejor legibilidad al procesar la señal resultante.
- Eliminación del ruido con el fin de que la calidad de la señal mejore y reducir el influjo de interferencias.
- Normalización de valores que escala los datos a un rango común y facilita el entrenamiento del modelo.
- Gestión de valores atípicos con el fin de identificar y eliminar los valores excepcionales que afecten el rendimiento del modelo.

2.2.3 Selección y entrenamiento de modelos de ML

Se evalúa varios modelos de ML con diferentes arquitecturas y algoritmos de aprendizaje, incluida una comparación de dos posibles modelos (Tabla 2.1):

Tabla 2.1

Comparación de modelos ML

Característica	Redes Neuronales Recurrentes (RNN)	Redes Neuronales Convencionales (RCC)
Arquitectura	Conexiones recurrentes que permiten un flujo de información en bucle	Conexiones unidireccionales entre capas
Procesamiento de datos	Secuencial	Independiente
Memoria	Capacidad para almacenar información de entradas pasadas	Sin memoria interna
Aplicaciones	Tareas que involucran datos secuenciales, como: Traducción automática, Reconocimiento de voz, Predicción de series temporales	Tareas que no requieren memoria de entradas pasadas, como: Clasificación de imágenes, Reconocimiento de objetos, Diagnóstico médico
Ventajas	Capaces de modelar dependencias a largo plazo en datos secuenciales	Más simples de entrenar y computacionalmente eficientes
Desventajas	Sujetas a problemas de desvanecimiento y explosión del gradiente [1]	No pueden modelar dependencias a largo plazo en datos secuenciales [2]

Se utiliza un enfoque de selección de modelos basado en validación cruzada para identificar el modelo que mejor se adapta a la tarea de predicción del espectro, tal como se

especifica en la Tabla 2.1. El entrenamiento de los modelos se realiza de forma iterativa, optimizando sus hiperparámetros utilizando algoritmos de búsqueda eficientes.

2.2.4 Evaluación de modelos

Se emplea una evaluación basada en la precisión, la sensibilidad, la especificidad, en cada uno de los modelos entrenados. Se analiza su desempeño bajo las condiciones de escenario establecidas para evaluar su generalización y se compara su efectividad.

2.2.5 Selección del mejor modelo

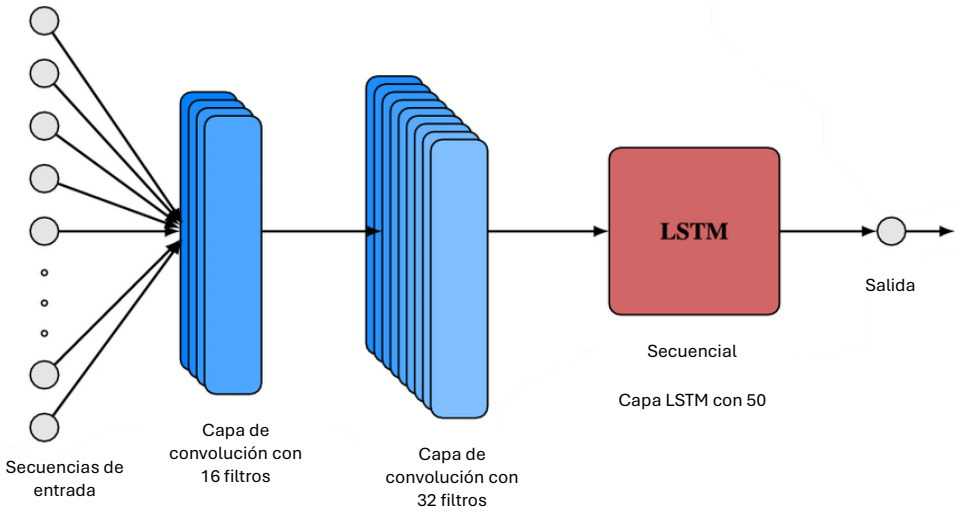
Basándose en los resultados de la evaluación, se selecciona el modelo de ML que brinda el mejor equilibrio entre precisión y generalización. El modelo seleccionado se utiliza para la implementación del sistema de predicción final del espectro.

2.2.6 Implementación del sistema

El modelo de ML seleccionado se implementa en un sistema de software, Python, que permite la predicción del espectro en tiempo real. En este caso, el modelo LSTM es la mejor opción para que el sistema sea viable de tal manera que se diseñe para ser adaptable a diferentes entornos y escenarios de operación, incluyendo redes inalámbricas estáticas y móviles. Se implementa una interfaz de usuario intuitiva para facilitar la interacción con el sistema.

Figura 3

Funcionamiento del modelo LSTM



En la Figura 3 la capa LSTM se encuentra entre dos capas convolucionales. Las capas convolucionales se utilizan para extraer características de los datos de entrada. La capa LSTM utiliza estas características para aprender dependencias a largo plazo entre los elementos de la secuencia. La salida de la capa LSTM es una secuencia de vectores, cada uno de los cuales tiene 50 unidades.

Las capas LSTM son un tipo de RNN que se utiliza para modelar secuencias de datos. Estas son capaces de aprender dependencias a largo plazo entre los elementos de una secuencia, lo que hace este modelo adecuado para tareas como el procesamiento del lenguaje natural [2].

2.2.7 Validación y pruebas

Se realizan pruebas del sistema utilizando datos nuevos no usados durante el entrenamiento del modelo. Se analizan los resultados de las pruebas para identificar posibles limitaciones del sistema y realizar ajustes necesarios para optimizar su rendimiento.

2.2.8 Análisis de resultados

Se realiza un análisis detallado de los resultados de las pruebas, incluida la precisión de la predicción, la latencia del sistema y el consumo de recursos informáticos. Se evalúa la confiabilidad del sistema y su capacidad para cumplir los objetivos establecidos, tal como se especifica posteriormente en la Figura 2. Además, se identifican oportunidades para mejorar aún más el rendimiento del sistema y ampliar su aplicabilidad.

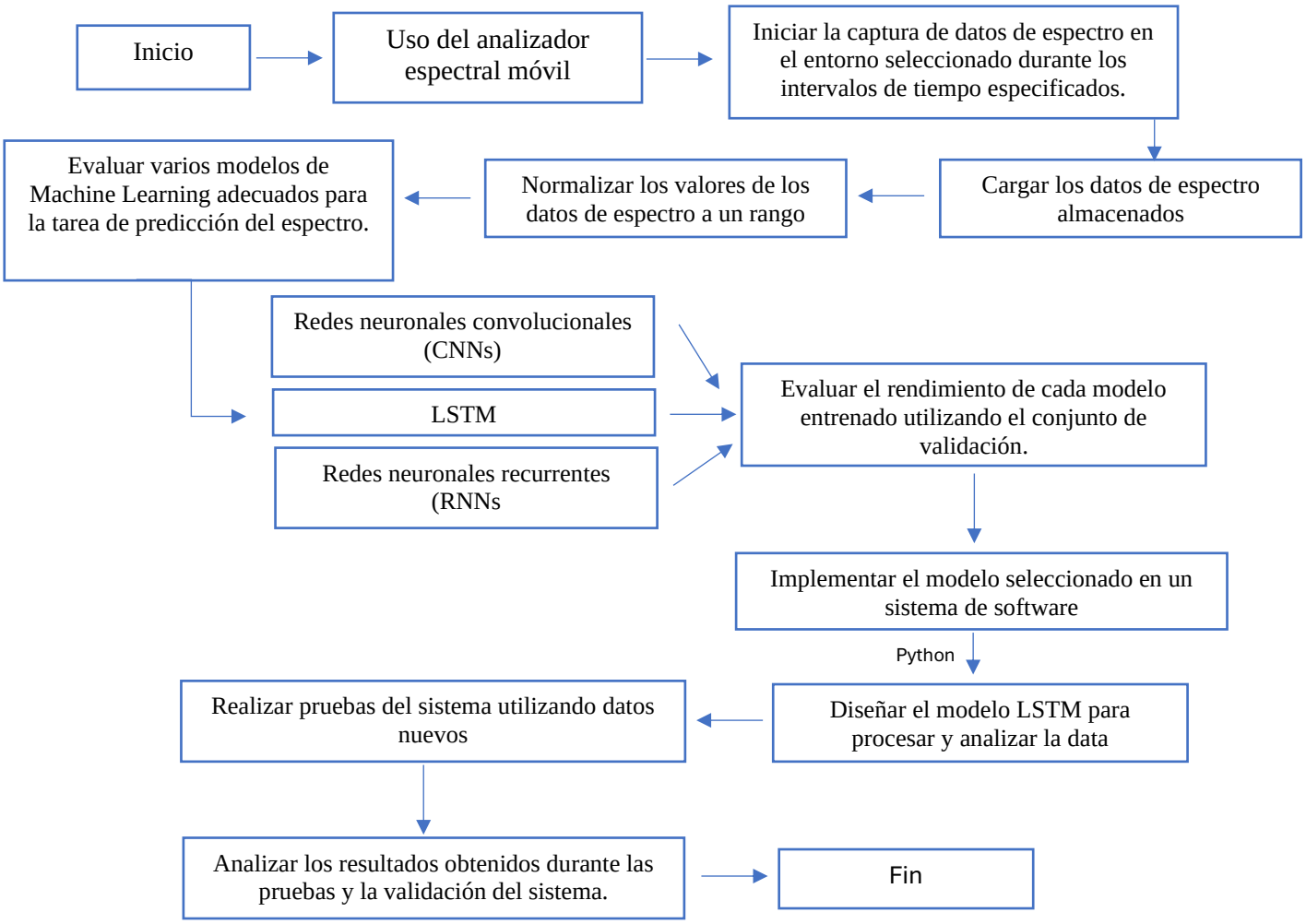
CAPÍTULO 3

3.1 Diagrama de flujo del sistema

A continuación, la Figura 4 se basa en cada proceso que se realizó para el cumplimiento del proyecto. Aquí se abarca desde la recopilación de datos mediante el analizador espectral hasta el análisis de los resultados gracias al modelo establecido.

Figura 4

Diagrama de flujo del sistema



3.2 Desarrollo y diseño del algoritmo de machine learning realizado con Python

Teniendo en cuenta la Figura 4, para lograr el desarrollo y diseño del algoritmo de LSTM antes mencionado se realizó con el lenguaje de programación Python obteniendo el análisis de canales de 2.4Ghz con el fin de alcanzar una predicción lo más certera posible.

Se empieza instalando los paquetes necesarios para el tratamiento de los datos obtenidos por medio del equipo ITT siendo este el analizador espectral móvil. Mediante la recolección de estos datos realizada, el algoritmo puede trabajar para lograr de manera autónoma la predicción deseada.

3.3 Desarrollo de solución para predicción de ML del espectro radio eléctrico.

Para la implementación inicial de técnicas de ML, se optó por un algoritmo de predicción LSTM. Se seleccionó el lenguaje de programación Python debido a su versatilidad y la amplia gama de bibliotecas disponibles. Esto permitió una implementación eficiente, facilitada por la importación de paquetes especializados para visualización, manipulación de matrices y creación de estructuras de datos, tal como se muestra en la Figura 5.

Figura 5

Importación de Librerías

```
import numpy as np
import matplotlib.pyplot as plt
import scipy.signal as sgn
import pandas as pd
pd.set_option("display.float_format", lambda x: "%.4f" % x)
import seaborn as sns
sns.set_context("paper", font_scale=1.3)
sns.set_style("white")
import warnings
warnings.filterwarnings("ignore")
from time import time
import matplotlib.ticker as tkr
from numpy.random import randn
from numpy.random import seed
from scipy import stats
from scipy.stats import pearsonr
from statsmodels.tsa.stattools import adfuller
from sklearn import preprocessing
from statsmodels.tsa.stattools import pacf
%matplotlib inline
```

```
pip install match
```

```
Requirement already satisfied: match in c:\users\david\anaconda3\lib\site-packages (0.3.2)
Requirement already satisfied: nltk>=3.4 in c:\users\david\anaconda3\lib\site-packages (from match) (3.8.1)
Requirement already satisfied: regex>=2018.7.11 in c:\users\david\anaconda3\lib\site-packages (from match) (2023.10.3)
Requirement already satisfied: click in c:\users\david\anaconda3\lib\site-packages (from nltk>=3.4->match) (8.1.7)
Requirement already satisfied: joblib in c:\users\david\anaconda3\lib\site-packages (from nltk>=3.4->match) (1.4.2)
Requirement already satisfied: tqdm in c:\users\david\anaconda3\lib\site-packages (from nltk>=3.4->match) (4.66.4)
Requirement already satisfied: colorama in c:\users\david\anaconda3\lib\site-packages (from click->nltk>=3.4->match) (0.4.6)
Note: you may need to restart the kernel to use updated packages.
```

```
pip install keras
```

```
Requirement already satisfied: keras in c:\users\david\anaconda3\lib\site-packages (3.4.1)
Requirement already satisfied: absl-py in c:\users\david\anaconda3\lib\site-packages (from keras) (2.1.0)
Requirement already satisfied: numpy in c:\users\david\anaconda3\lib\site-packages (from keras) (1.26.4)
Requirement already satisfied: rich in c:\users\david\anaconda3\lib\site-packages (from keras) (13.3.5)
Requirement already satisfied: namex in c:\users\david\anaconda3\lib\site-packages (from keras) (0.0.8)
Requirement already satisfied: h5py in c:\users\david\anaconda3\lib\site-packages (from keras) (3.11.0)
Requirement already satisfied: optree in c:\users\david\anaconda3\lib\site-packages (from keras) (0.11.0)
Requirement already satisfied: ml-dtypes in c:\users\david\anaconda3\lib\site-packages (from keras) (0.3.2)
Requirement already satisfied: packaging in c:\users\david\anaconda3\lib\site-packages (from keras) (23.2)
Requirement already satisfied: typing-extensions>=4.0.0 in c:\users\david\anaconda3\lib\site-packages (from optree->keras) (4.11.0)
Requirement already satisfied: markdown-it-py<3.0.0,>=2.2.0 in c:\users\david\anaconda3\lib\site-packages (from rich->keras) (2.2.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in c:\users\david\anaconda3\lib\site-packages (from rich->keras) (2.15.1)
Requirement already satisfied: mdurl~=0.1 in c:\users\david\anaconda3\lib\site-packages (from markdown-it-py<3.0.0,>=2.2.0->rich->keras) (0.1.0)
Note: you may need to restart the kernel to use updated packages.
```

```
import match
```

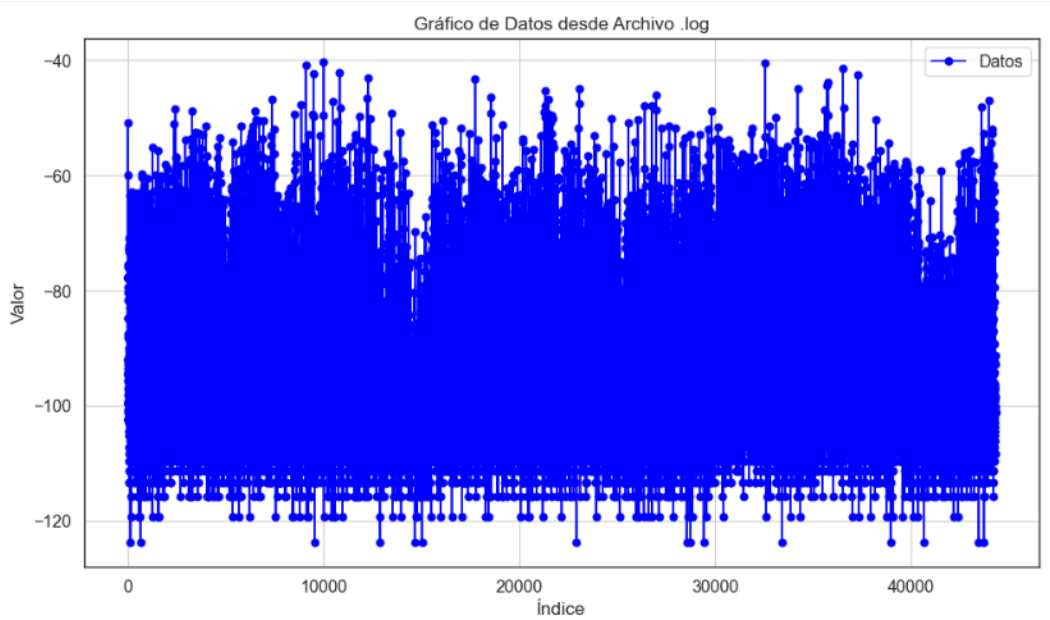
Los datos recopilados en formato .log durante un estudio realizado en el establecimiento “Sweet and Coffee”, contienen información sobre la intensidad de las señales en diferentes bandas

de frecuencia. El objetivo principal de este análisis es desarrollar un modelo de predicción que permita identificar interferencias y congestiones en el espectro radioeléctrico, contribuyendo así a una gestión más eficiente del mismo.

A continuación, se visualiza el gráfico de todos los datos en bruto (Figura 6) para lograr una idea visual de los datos que se están trabajando.

Figura 6

Visualización de en bruto



Con el objetivo de obtener una señal limpia y confiable, se llevó a cabo un proceso de preprocesamiento de los datos (Anexo 3). Este incluyó la eliminación de ruido, la identificación y tratamiento de datos atípicos, así como la mitigación de efectos adversos como el *fast fading* y el esparcimiento multicamino, los cuales pueden introducir distorsiones en las mediciones.

Una vez depurada la señal, se procederá a su transformación mediante el empleo de medias móviles y la técnica de limitación por media-varianza de ±2.5 tanto superior como inferiormente (Anexo 4). Estas técnicas permitirán suavizar la serie temporal y eliminar los valores atípicos, facilitando así el análisis posterior de los datos vectoriales.

Se dividió el conjunto de datos en dos subconjuntos: un conjunto de entrenamiento (80%) y un conjunto de prueba (20%) (Anexo 5). El modelo de aprendizaje automático se entrenó exhaustivamente utilizando el conjunto de entrenamiento.

Luego, como muestra el Anexo 6, se realiza una división en los índices del tiempo en múltiplos de 7.2 segundos y múltiples de 0.01 segundos. Una vez logrado esto, se separa en 80% y 20% los datos divididos anteriormente para el entrenamiento de ML, con el fin de continuar entrenando al ML y prediga el 20% faltante.

Una vez obtenido el dataframe se realiza la modulación, en este caso una modulación por fase o conocido como MP, para los datos limpios (Anexo 7). Luego, se realiza un cambio de fase de modulación para una posible facilidad de tratamiento de los datos.

Ya que se obtuvo una definición de función en la cual como entradas se tiene los datos originales limpios, el entrenamiento del 80% que posteriormente se dividió y predijo con la modulación, ahora mediante el algoritmo LSTM diseñado (Anexo 8) se genera la posible predicción real.

Posteriormente, se llevó a cabo el proceso de demodulación de la señal (Anexo 9) para extraer la información relevante. Los datos demodulados fueron normalizados y representados en un vector unidimensional. A continuación, este vector se utilizó como entrada para entrenar y evaluar un modelo de red neuronal recurrente LSTM, con el objetivo de realizar las predicciones deseadas.

Con los datos preprocesados y normalizados, se procede a aplicar el algoritmo LSTM. El modelo, entrenado previamente con una porción de los datos, será utilizado ahora para realizar predicciones sobre el conjunto completo de datos. De esta manera, se evaluará la eficiencia y precisión del modelo en la tarea de predicción.

A continuación, se realiza un entrenamiento previo de los datos únicamente demodulados (Anexo 10) para el algoritmo neuronal LSTM, separándolos en un 80% de entrenamiento y un 20% en validación de entrenamiento o predicción.

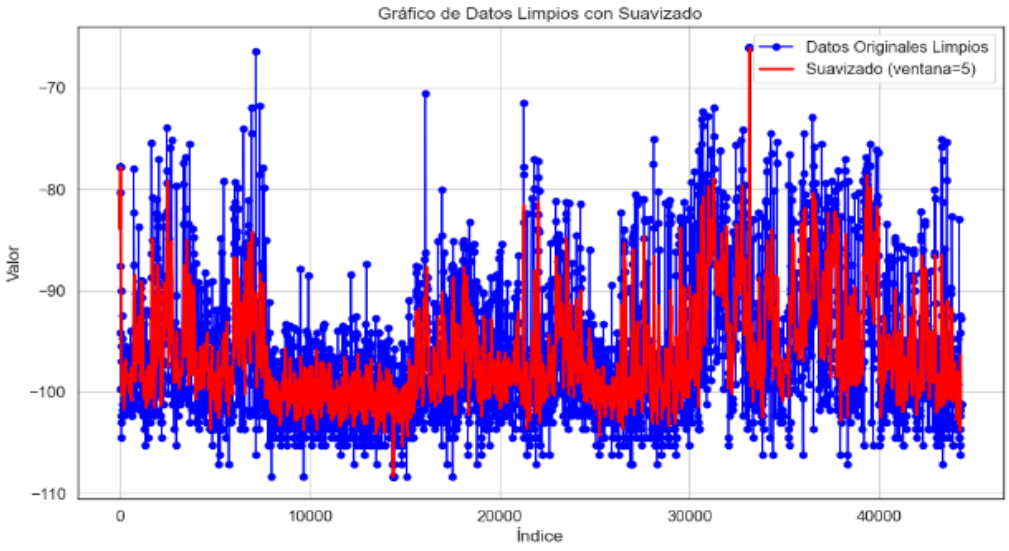
Finalmente, se redimensionan los datos, es decir, se los lleva a un vector unidimensional. Luego se define el modelo tanto con la secuencia temporal como con el historial para obtener una predicción según el modelo implementado en este proyecto (Anexo 11).

3.3.1 Gráficos de resultados obtenidos

Tal como se observa en la Figura 7, representa los datos originales con una gran cantidad de fluctuaciones aleatorias (línea azul), lo que significa una presencia considerable de ruido. Mientras que, la línea roja representa los datos depurados para el proceso de modulación de la señal, la misma que se centró en un suavizado con una ventana tamaño 5, donde se reduce significativamente el ruido presente en los datos originales y se visualiza de mejor manera la tendencia de los datos.

Figura 7

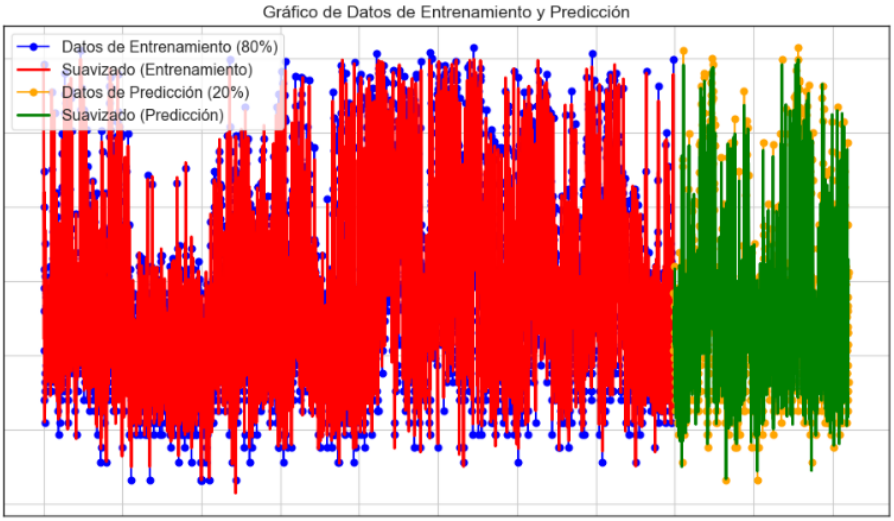
Visualización de datos originales y datos limpios



El suavizado de los datos de entrenamiento (línea roja) indica un buen ajuste del modelo a los datos históricos, es decir que captura las características principales de los datos (Figura 8). Los datos de prueba a la predicción (línea verde) sugiere que el modelo generaliza bien a nuevos datos.

Figura 8

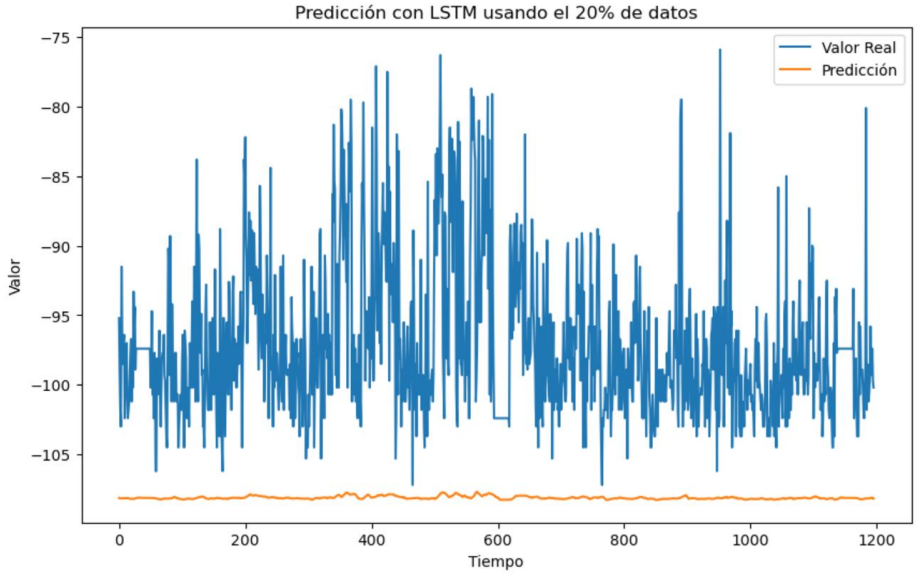
División de nuevos datos en entrenamiento y prueba



En continuidad con la Figura 9, se observan los nuevos datos obtenidos mediante el modelo LSTM, es decir la predicción a base del 20% de datos de prueba es bastante incierta, no existe el seguimiento correcto al patrón de la señal. La parte superior (línea azul) muestra los datos de

Figura 9

Predicción a base del 20% de data

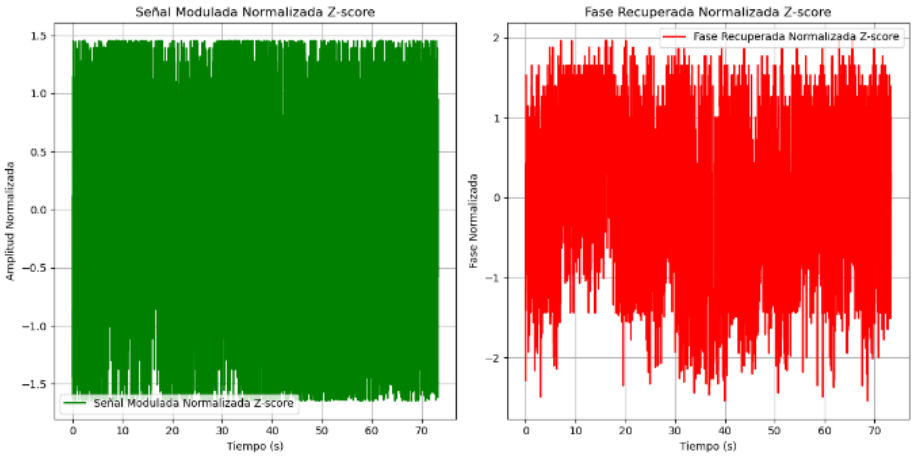


entrenamiento que se le brindó al modelo, mientras que la parte inferior (línea amarilla) son los datos que el modelo predijo y que son lejanos a los datos de entrenamiento previo que se brindó.

Dichos datos se logran visualizar en la Figura 10, donde se muestra la señal modulada y demodulada tanto en vector 1D mostrando los puntos máximos y mínimos que se alcanzaron.

Figura 10

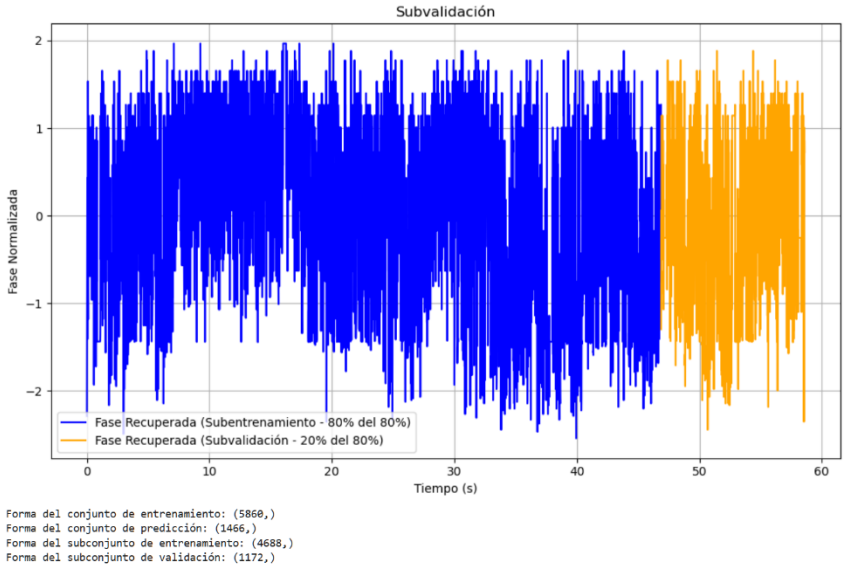
Visualización de la demodulación de la señal



Una vez los datos son demodulados, se normalizan y prueban en el entrenamiento (línea azul) y validación de la predicción (línea naranja) como se observa en la Figura 11. De esta manera, se conoce el correcto funcionamiento del modelo LSTM.

Figura 11

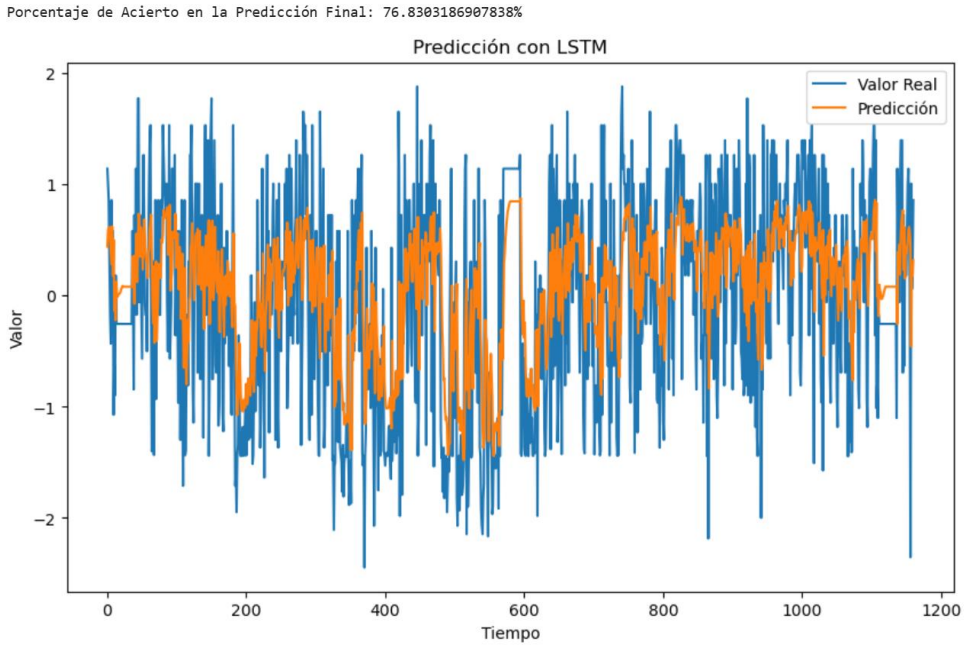
Validación de la predicción



Con la fase normalizada y los datos modulados en fase se realiza la predicción con ayuda del modelo desarrollado LSTM como se aprecia en la Figura 12 La gráfica corresponde a la predicción real de los datos que se entrenó previamente mediante ML, dando un resultado bastante fiable sobre el modelo de LSTM, con un valor del 76.83% de precisión sobre los datos cargados a entrenar y predecir.

Figura 12

Predicción final

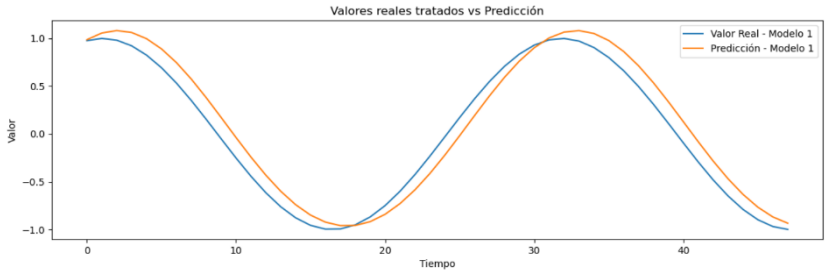


Finalmente, se presenta la comparación entre cada caso donde se utilizó el modelo LTSM (Figura 13). En la parte superior (parte a) se observa la diferencia de predicción que se obtuvo con la data demodulada y normalizada, donde el patrón de seguimiento (línea naranja) es bastante bueno en base a los datos originales (línea azul). Mientras que, en la parte inferior (parte b), se muestra la misma diferencia tomando los datos originales sin ningún tipo de procesamiento. Aquí el patrón no sigue correctamente a la señal original, por lo tanto, la predicción no es buena en ese momento.

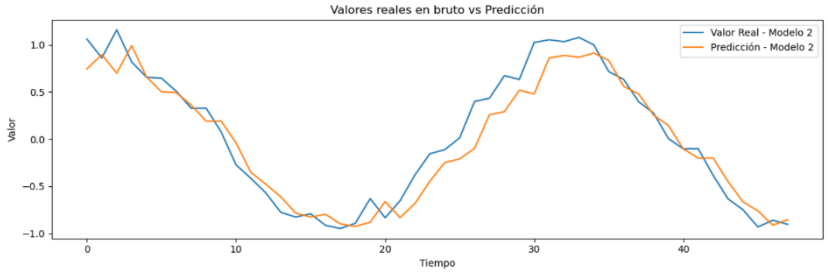
Figura 13

Comparación final

a. *Comportamiento de predicción con datos tratados*



b. *Comportamiento de predicción con datos antes del preprocesamiento*



Capítulo 4

4.1 Conclusiones y recomendaciones

4.1.1 Conclusiones

- En base a la data no tratada, el modelo LSTM entrenado con el 80% de los datos y validando dicho entrenamiento con el 20% faltante no logra capturar la dinámica de la serie temporal representada en los datos reales. Tal como se visualizó en la Figura 9 existe un desajuste, lo que significa que el modelo no puede aprender los patrones subyacentes en los datos debido a los cambios bruscos que experimenta la señal al trabajarse con potencias.
- Este trabajo, en base a la data preprocesada, demuestra la eficacia de la predicción del espectro utilizando el modelo LSTM en la banda ISM de 2.4GHz, proporcionando una poderosa herramienta para optimizar la gestión del espectro con un 80% de éxito en entornos inalámbricos. Además, sienta las bases para futuras investigaciones que puedan mejorar aún más la precisión y aplicabilidad de estos modelos en diferentes escenarios.
- La aplicación de un umbral de 2.5 veces la varianza para la detección y eliminación de ausencias demostró ser una estrategia efectiva para limpiar y transformar los datos. Al eliminar los valores anormales, se mejoró la calidad de los datos y, por lo tanto, la habilidad del modelo para detectar las tendencias subyacentes en el espectro.
- En el contexto de esta investigación, LSTM han demostrado ser una herramienta valiosa en el campo de las comunicaciones inalámbricas. Gracias a su capacidad para manejar secuencias temporales y detectar patrones complejos, se ha logrado obtener predicciones fiables, dejando atrás los resultados de modelos tradicionales.

4.1.2 Recomendaciones

- Para mejorar las predicciones, se recomienda examinar el tamaño del conjunto de datos de entrenamiento, la estructura del modelo LSTM y posiblemente aumentar su complejidad o el tiempo de entrenamiento.
- Para una mayor investigación en el área, se recomienda profundizar en este tipo de aplicaciones de IoT para agricultura de precisión basadas en la predicción espectral. Esto con el fin de optimizar la comunicación entre dispositivos, mejorando sistemas como el riego inteligente y el monitoreo de cultivos. Esta investigación en particular contribuiría significativamente a la eficiencia en el uso de recursos y la productividad de la agricultura.

Referencias

[1] S. Haykin, «Neural networks and learning machines. Pearson Prentice Hall,» 2009.

[2] IEEE Std 802.11a-1999, «Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 2: Enhancements for Higher Throughput. IEEE.».

[3] D. &. H. M.-H. Miorandi, «Cognitive radio for spectrum management in wireless networks: Recent advances and future directions. IEEE Access, 4,,» 2016, pp. 5484-5509.

[4] I. F. S. W. S. Y. &. C. E. Akyildiz, «Wireless sensor networks: A survey. IEEE Communications Surveys & Tutorials,» 2007, pp. 220-253.

[5] Y. C. a. H. J. Lin, «"A HMM-Based Spectrum Access Method Used for HF Communication," 2015 8th International Symposium on Computational Intelligence and Design (ISCID), Hangzhou,» 2015, pp. 426-431.

[6] Y. S. a. Y. E. Sagduyu, «"Spectrum learning and access for cognitive satellite communications under jamming," 2016 IEEE Conference on Communications and Network Security (CNS),,» Philadelphia, PA, 2016, pp. 472-479.

[7] L. B. e. al, «"Dynamic Adaptive Video Streaming on Heterogeneous TVWS and Wi-Fi Networks,",» de *IEEE/ACM Transactions on Networking*, 2017, pp. 3253-3266.

[8] S. P. S. N. S. a. M. G. S. DasMahapatra, «"Performance Analysis of Prediction Based Spectrum Sensing for Cognitive Radio Networks,",» de *2nd International Conference on Intelligent Communication and Computational Techniques (ICCT)*, Jaipur, India, 2019, pp. 271-274.

[9] F. d. B. V. a. F. M.-T. J. Salas, «Deep Learning: Current State,» de *IEEE Latin America Transactions, vol. 17*, 2019, pp. 1925-1945.

[10] FCC, «CFR 47 Part 15 Subpart C - Unlicensed Devices," Federal Communications Commission,» 3 Mayo 2024. [En línea]. Available: https://www.ecfr.gov/current/title-47/chapter-I/subchapter-A/part-15/subpart-E. [Último acceso: 2024].

[11] ETSI EN 300 328, «"Technical requirements for radio equipment operating in the 2,4 GHz ISM band and intended for industrial, scientific and medical (ISM) purposes,",» European Telecommunications Standards Institute, 2020.

[12] IEEE Std 802.15.1-2020, «Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs),» 2020. [En línea]. Available: https://standards.ieee.org/ieee/802.11/7028/.

[13] I. B. Y. &. C. A. Goodfellow, «Deep learning. MIT press.,» 2016.

[14] G. &. G. L. Csurka, «Supervised learning vs. unsupervised learning. In Proceedings of the 27th International Conference on Machine Learning,» 2010, pp. 246-253.

[15] R. S. &. B. A. G. Sutton, «Reinforcement learning: An introduction. MIT press.,» 2018.

[16] J. &. G. G. Corominas, «Spectral resource allocation using minimum spanning trees. In Proceedings of the 2014 IEEE International Conference on Communications (ICC),» 2014, pp. 1-6.

[17] W. &. L. Y. Zhang, «Data preprocessing for machine learning: A survey.,» 2019, pp. 142862-142876.

[18] X. K. V. Q. J. R. G. J. Y. Q. M. H. .. &. L. Y. Wu, «TOWARDS A PRACTICAL CLASSIFICATION SYSTEM FOR DATA MINING. ACM SIGKDD Explorations,» 2006, pp. 41-56.

[19] Y. &. W. X. He, «Data Preprocessing for Big Data Analysis. In 2017 IEEE 13th International Conference on Automation, Robotics and Vision (ICARV),» 2017, pp. 1013-1016.

[20] N. V. &. B. D. A. Chawla, «Remove Outliers Using One-Class SVM. In Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,» 2004, pp. 409-417.

[21] P. &. W. G. Min, «Normalization and Data Preprocessing for Classification. In 2011 11th International Conference on Machine Learning and Applications (ICMLA),» 2011, pp. 202-205.

[22] M. &. A. L. Re, «Normalization Methods for Feature Scaling in Machine Learning. Artificial Intelligence Research,» 2013, pp. 113-122.

[23] J. Mitola, «The cognitive radio: Software radios that compute. IEEE Journal on Selected Areas in Communications,» 2000, pp. 353-364.

[24] S. Haykin, «Cognitive radio: Brain-inspired wireless communications. IEEE Journal on Selected Areas in Communications,» 2005, pp. 29-40.

[25] I. F. L. W. Y. C. M. C. &. Z. H. Akyildiz, «Wireless cognitive networks: A survey. IEEE Communications Magazine,» 2009, pp. 86-95.

[26] FCC, «Spectrum Policy Statement Regarding Unlicensed Use of the 5 GHz Band.,» 2003. [En línea]. Available: https://www.fcc.gov/document/fcc-states-spectrum-management-principles-transmitters-receivers-0.

[27] ETSI, «Technical Report 102 481: Unlicensed Use of the 5 GHz Frequency Band; Harmonized Technical Requirements for Equipment Employing Access in Licensed and Unlicensed Bands (ELAF),» 2005. [En línea]. Available: https://www.etsi.org/.

Anexos

Anexo 1

Toma de mediciones

Anexo 2

Analizador espectral móvil

Anexo 3

Preprocesamiento de datos

```
# Calcular la media móvil para suavizar los datos
window_size = 5
df['Smooth'] = df['Value'].rolling(window=window_size, min_periods=1).mean()

# Calcular la diferencia absoluta entre los datos originales y suavizados
df['Diff'] = np.abs(df['Value'] - df['Smooth'])

# Definir un umbral para identificar datos con ruido (puedes ajustar este valor según tus datos)
noise_threshold = 1.0

# Identificar índices de datos con ruido
noisy_data_indices = df[df['Diff'] > noise_threshold].index

# Verificar si hay datos con ruido
if len(noisy_data_indices) > 0:
    print(f"Se han identificado {len(noisy_data_indices)} puntos con ruido.")

    # Opcional: Mostrar los datos con ruido en el gráfico
    plt.figure(figsize=(10, 6))
    plt.plot(df.index, df['Value'], marker='o', linestyle='-', color='b', label='Datos Originales')
    plt.plot(df.index, df['Smooth'], linestyle='-', color='r', linewidth=2, label=f'Suavizado (ventana={window_size})')
    plt.scatter(df.loc[noisy_data_indices].index, df.loc[noisy_data_indices]['Value'], color='g', label='Datos con Ruido')
    plt.xlabel('Índice')
    plt.ylabel('Valor')
    plt.title('Gráfico de Datos con Identificación de Ruido')
    plt.legend()
    plt.grid(True)
    plt.tight_layout()
    plt.show()

    # Limpiar los datos eliminando puntos identificados como ruido
    df_clean = df.drop(index=noisy_data_indices)

    # Continuar trabajando con los datos limpios, por ejemplo, aplicar suavizado nuevamente
    df_clean['Smooth'] = df_clean['Value'].rolling(window=window_size, min_periods=1).mean()
```

Anexo 4

Empleo de técnica de limitación por media-varianza

```
# Calcular la media y la desviación estándar de los datos sin ruido
mean_cleaned = df_cleaned['Smooth'].mean()
std_cleaned = df_cleaned['Smooth'].std()

# Calcular los límites superior e inferior (media +/- 2.5 * desviación estándar)
upper_limit = mean_cleaned + 2.5 * std_cleaned
lower_limit = mean_cleaned - 2.5 * std_cleaned

# Identificar los datos que se encuentran dentro de los límites
within_limits_indices = df_cleaned[(df_cleaned['Smooth'] >= lower_limit) & (df_cleaned['Smooth'] <= upper_limit)].index
```

Anexo 5

División de datos en entrenamiento y prueba

```
# Dividir los datos dentro de los límites en 80% para entrenamiento y 20% para predicción
train_size = int(0.8 * len(within_limits_indices))
train_indices = within_limits_indices[:train_size]
test_indices = within_limits_indices[train_size:]

df_train = df_cleaned.loc[train_indices]
df_test = df_cleaned.loc[test_indices]
```

Anexo 6

Modulación de datos

```
# Crear DataFrame final solo con datos dentro de los límites
df_final = df_cleaned.loc[within_limits_indices].reset_index(drop=True)

# Asumir que cada índice representa un intervalo de 0.01 segundos
interval = 0.01  # en segundos
df_final.index = df_final.index * interval

# Modulación (ejemplo de modulación de fase)
frecuencia_portadora = 244.000 * 1e6  # MHz a Hz
fase_portadora = 0.285  # Fase de la portadora para modulación de fase

# Generar la señal portadora
tiempo = np.arange(len(df_final)) * interval
portadora = np.sin(2 * np.pi * frecuencia_portadora * tiempo + fase_portadora)

# Aplicar modulación de fase
senal_modulada = np.sin(2 * np.pi * frecuencia_portadora * tiempo + fase_portadora * df_final['Value'])
```

Anexo 7

División de índice de tiempo

```
# Asumir que cada índice representa un intervalo de 0.01 segundos
interval = 0.01  # en segundos
df_cleaned.index = df_cleaned.index * interval
df_train.index = df_train.index * interval
df_test.index = df_test.index * interval
df_subtrain.index = df_subtrain.index * interval
df_subtest.index = df_subtest.index * interval

# Ajustar el índice para que represente múltiplos de 0.01 segundos
df_cleaned.index = df_cleaned.index / 0.01
df_train.index = df_train.index / 0.01
df_test.index = df_test.index / 0.01
df_subtrain.index = df_subtrain.index / 0.01
df_subtest.index = df_subtest.index / 0.01
```

Anexo 8

Diseño de algoritmo LSTM

```
# Normalización de los datos
scaler = MinMaxScaler(feature_range=(0, 1))
subtrain_values = scaler.fit_transform(subtrain_values.reshape(-1, 1))
subtest_values = scaler.transform(subtest_values.reshape(-1, 1))

# Convertir datos en formato adecuado para LSTM
def create_dataset(data, time_step=1):
    X, y = [], []
    for i in range(len(data)-time_step-1):
        a = data[i:(i+time_step), 0]
        X.append(a)
        y.append(data[i + time_step, 0])
    return np.array(X), np.array(y)

time_step = 10
X_train, y_train = create_dataset(subtrain_values, time_step)
X_test, y_test = create_dataset(subtest_values, time_step)

# Reshape input to be [samples, time steps, features] which is required for LSTM
X_train = X_train.reshape(X_train.shape[0], X_train.shape[1], 1)
X_test = X_test.reshape(X_test.shape[0], X_test.shape[1], 1)

# Crear el modelo LSTM
model = Sequential()
model.add(LSTM(50, return_sequences=True, input_shape=(time_step, 1)))
model.add(LSTM(50, return_sequences=False))
model.add(Dense(25))
model.add(Dense(1))

# Compilar el modelo
model.compile(optimizer='adam', loss='mean_squared_error')

# Entrenar el modelo
model.fit(X_train, y_train, batch_size=1, epochs=1)

# Predecir
train_predict = model.predict(X_train)
test_predict = model.predict(X_test)

# Invertir la normalización
train_predict = scaler.inverse_transform(train_predict)
test_predict = scaler.inverse_transform(test_predict)
y_train = scaler.inverse_transform([y_train])
y_test = scaler.inverse_transform([y_test])
```

Anexo 9

Demodulación de la señal

```
# Demodulación en fase
# Recuperar la fase de la señal modulada
fase_modulada = np.arctan2(senal_modulada, np.cos(2 * np.pi * frecuencia_portadora * tiempo))
# Extraer la fase del mensaje (desmodulación)
fase_recuperada = (fase_modulada - fase_portadora * df_final['Value']) / fase_portadora

# Convertir la fase recuperada a un vector 1D
fase_recuperada_1D = np.array(fase_recuperada)

# Normalización Min-Max
scaler_minmax = MinMaxScaler()
senal_modulada_normalizada_minmax = scaler_minmax.fit_transform(senal_modulada_1D.reshape(-1, 1)).flatten()
fase_recuperada_normalizada_minmax = scaler_minmax.fit_transform(fase_recuperada_1D.reshape(-1, 1)).flatten()

# Normalización Z-score
scaler_zscore = StandardScaler()
senal_modulada_normalizada_zscore = scaler_zscore.fit_transform(senal_modulada_1D.reshape(-1, 1)).flatten()
fase_recuperada_normalizada_zscore = scaler_zscore.fit_transform(fase_recuperada_1D.reshape(-1, 1)).flatten()

# Graficar la señal modulada normalizada
plt.figure(figsize=(12, 12))
```

Anexo 10

Modelo LSTM con datos demodulados

```
# Dividir la señal en conjuntos de entrenamiento y predicción
fase_recuperada_train = fase_recuperada_normalizada[:n_train]
fase_recuperada_test = fase_recuperada_normalizada[n_train:]

# Calcular el tamaño del subconjunto de entrenamiento
n_subtrain = int(len(fase_recuperada_train) * subtrain_ratio)  # Tamaño del subentrenamiento
n_subtest = len(fase_recuperada_train) - n_subtrain  # Tamaño del subconjunto de validación

# Dividir el conjunto de entrenamiento en subentrenamiento y subvalidación
fase_recuperada_subtrain = fase_recuperada_train[:n_subtrain]
fase_recuperada_subtest = fase_recuperada_train[n_subtrain:]
```

Anexo 11

Realización de la predicción

```
# Preparar los datos para el modelo LSTM
time_step = 10
X_train, y_train = create_dataset(fase_recuperada_train, time_step)
X_test, y_test = create_dataset(fase_recuperada_test, time_step)

# Redimensionar para LSTM [samples, time_steps, features]
X_train = X_train.reshape((X_train.shape[0], X_train.shape[1], 1))
X_test = X_test.reshape((X_test.shape[0], X_test.shape[1], 1))

# Definir el modelo LSTM
model = Sequential([
    LSTM(50, activation='relu', input_shape=(time_step, 1)),
    Dense(1)
])
model.compile(optimizer='adam', loss='mse')

# Entrenar el modelo
history = model.fit(X_train, y_train, epochs=10, batch_size=32, validation_data=(X_test, y_test))

# Realizar una predicción
predictions = model.predict(X_test)
```