



# **UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA**

FACSISTEL

INGENIERÍA EN TELECOMUNICACIONES

TRABAJO DE INTEGRACIÓN CURRICULAR

**Clasificación automática en tiempo real de  
modulaciones OTA usando CNN-1D  
residual para sensado espectral**

**Jonathan Andrés Almeida Castillo**

**Dirigido por:**

Ing. Daniel Jaramillo Chamba, M.Sc.

La Libertad - 2026



**UNIVERSIDAD ESTATAL PENÍNSULA  
DE SANTA ELENA  
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES**

**TRIBUNAL DE SUSTENTACIÓN**

Ing. Ronald Rovira Jurado, Ph.D.  
**DIRECTOR DE LA CARRERA**

Ing. Daniel Jaramillo, Mgtr.  
**TUTOR**

Ing. Luis Amaya Fariño, Mgtr.  
**DOCENTE ESPECIALISTA**

Ing. Corina Gonzabay De La A, Mgtr.  
**SECRETARIA**



**UNIVERSIDAD ESTATAL PENÍNSULA  
DE SANTA ELENA  
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES**

**CERTIFICACIÓN**

Certifico que luego de haber dirigido científica y técnicamente el desarrollo y estructura final del trabajo, este cumple y se ajusta a los estándares académicos, razón por el cual apruebo en todas sus partes el presente trabajo de titulación que fue realizado en su totalidad por **Almeida Castillo Jonathan Andrés**, como requerimiento para la obtención del título de Ingeniero en telecomunicaciones.

La Libertad, a los 25 días del mes de agosto del año 2026

**TUTOR**

---

**Ing. Daniel Armando Jaramillo Chamba, Mgtr.**



**UNIVERSIDAD ESTATAL PENÍNSULA  
DE SANTA ELENA  
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES**

**DECLARACIÓN DE RESPONSABILIDAD**

**Yo, Almeida Castillo Jonathan Andrés**

**DECLARO QUE:**

El trabajo de Titulación, **Clasificación automática en tiempo real de modulaciones OTA usando CNN-1D residual para sensado espectral** previo a la obtención del título en Ingeniero en Telecomunicaciones, ha sido desarrollado respetando derechos intelectuales de terceros conforme las citas que constan en el documento, cuyas fuentes se incorporan en las referencias o bibliografías. Consecuentemente este trabajo es de mi total autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance del Trabajo de Titulación referido.

La Libertad, a los 25 días del mes de agosto del año 2026

**EL AUTOR**

---

**Almeida Castillo Jonathan Andrés**



**UNIVERSIDAD ESTATAL PENÍNSULA  
DE SANTA ELENA  
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES**

**CERTIFICACIÓN DE ANTIPLAGIO**

Certifico que después de revisar el documento final del trabajo de titulación denominado **Clasificación automática en tiempo real de modulaciones OTA usando CNN-1D residual para sensado espectral**, presentado por el estudiante, **Almeida Castillo Jonathan Andrés** fue enviado al Sistema Antiplagio, presentando un porcentaje de similitud correspondiente al 09%, por lo que se aprueba el trabajo para que continúe con el proceso de titulación.



Certificado de análisis  
Compilatio Magister+ | UPSE-ECU

Trabajo\_de\_integracion\_curricular\_Almeida  
ID : 01bdf638e0f4316a3fa9cd9896efa7d3e40b5b38



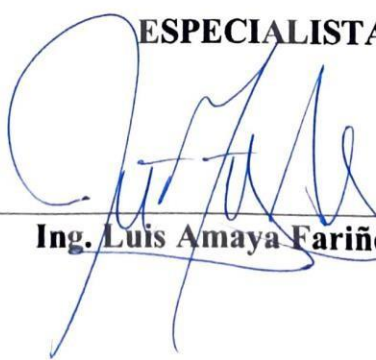
9%

Textos sospechosos

Nombre del fichero : Trabajo\_de\_integracion\_curricular\_Almeida.txt  
Tamaño del archivo original : 11,12 MB  
Número de palabras : 28.340

Depositante : AMAYA FARIÑO LUIS MIGUEL  
Fecha de depósito : 24 de agosto de 2026  
Tipo de carga : interface  
fecha de fin de análisis : 24 de agosto de 2026

**ESPECIALISTA**

  
Ing. Luis Amaya Fariño, Mgtr.



**UNIVERSIDAD ESTATAL PENÍNSULA  
DE SANTA ELENA  
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES**

**AUTORIZACIÓN**

**Yo, Almeida Castillo Jonathan Andrés**

Autorizo a la Universidad Estatal Península de Santa Elena, para que haga de este trabajo de titulación o parte de él, un documento disponible para su lectura consulta y procesos de investigación, según las normas de la Institución.

Cedo los derechos en línea patrimoniales de artículo profesional de alto nivel con fines de difusión pública, además apruebo la reproducción de este artículo académico dentro de las regulaciones de la Universidad, siempre y cuando esta reproducción no suponga una ganancia económica y se realice respetando mis derechos de autor

La libertad, a los 25 días del mes de agosto del año 2026

**EL AUTOR**

---

**Almeida Castillo Jonathan Andres**

## **AGRADECIMIENTO**

Agradezco a Dios por darme la salud y la constancia necesarias para llegar hasta este momento. Agradezco también a mi familia por su apoyo incondicional y por enseñarme que ninguna meta se alcanza sin disciplina y sacrificio. A mi tutor, Ing. Daniel Jaramillo Chamba, por su guía y sus observaciones oportunas, determinantes para dar forma y rigor a esta investigación. A los docentes de la Facultad de Sistemas y Telecomunicaciones, por los conocimientos compartidos a lo largo de la carrera. A mis compañeros, por las horas de estudio y el ánimo que nunca faltó en los momentos difíciles. A la Universidad Estatal Península de Santa Elena, por la formación y los espacios que hicieron posible este proyecto. Y de manera especial, a mi primo Javier Guamán, quien en más de una ocasión me dio trabajo para cubrir los gastos de la carrera y conseguir los componentes necesarios; sin esa ayuda oportuna, este trabajo no habría sido posible, y a mi tía Alicia González, por su generosidad y por brindarme su apoyo en un momento importante de mi formación.

*Jonathan Andrés Almeida Castillo*

## **DEDICATORIA**

Dedico este trabajo a Dios por ser mi fortaleza en cada dificultad y por permitirme culminar esta etapa. A mi madre, Giselle Castillo, por su amor incondicional, por su esfuerzo silencioso y por ser el ejemplo de perseverancia que me sostuvo durante toda la carrera; todo lo que soy se lo debo a ella. A mi tía, Sonia González, por su apoyo y por estar presente cuando más lo necesité, pues su generosidad hizo posible que continuara y llegara hasta el final de este camino. A ellas, con todo mi cariño y gratitud.

*Jonathan Andrés Almeida Castillo*

## RESUMEN

La clasificación automática de modulaciones es una etapa clave en los sistemas de radio inteligente y el monitoreo del espectro radioeléctrico, donde debe identificarse la modulación de una señal sin disponer de información previa del transmisor. La mayor parte de los trabajos publicados emplea conjuntos de datos sintéticos, que no reproducen las imperfecciones del *hardware* ni las condiciones reales del canal.

Este trabajo desarrolla un sistema de clasificación de modulaciones en tiempo real a partir de señales capturadas *Over-The-Air* mediante radio definida por *software*. Con un Pluto+ SDR como transmisor y un RTL-SDR como receptor se construyó un conjunto de datos balanceado de 55 000 ventanas I/Q distribuidas en once clases y validadas mediante un procedimiento de tres etapas. Sobre él se entrenó AMCNet, una red neuronal convolucional unidimensional con bloques residuales, invariante al desplazamiento de frecuencia de portadora, que se exportó al formato ONNX y se integró en una aplicación de escritorio en C++ de sensado espectral.

El modelo alcanzó una exactitud del 100 % en el conjunto interno de prueba y clasificó correctamente las noventa transmisiones de la validación funcional. La exactitud superó el 99 % para relaciones señal-ruido iguales o superiores a 5 dB y se degradó de forma abrupta por debajo de ese nivel. La aplicación operó con un consumo aproximado del 30 % de CPU y 25 MB de memoria en un equipo de dos núcleos sin GPU dedicada, lo que confirma la viabilidad de aplicar aprendizaje profundo a la clasificación de modulaciones reales sobre *hardware* de bajo costo.

**Palabras clave:** clasificación automática de modulaciones, radio definida por *software*, red neuronal convolucional, sensado espectral, aprendizaje profundo.

## ABSTRACT

Automatic modulation classification is a key stage in intelligent radio systems and radio spectrum monitoring, where the modulation of a signal must be identified without prior information about the transmitter. Most published studies use synthetic datasets that do not reproduce hardware imperfections or real-world channel conditions.

This work develops a real-time modulation classification system based on Over-The-Air signals captured with software defined radio. Using a Pluto+ SDR as transmitter and an RTL-SDR as receiver, a balanced dataset of 55000 I/Q windows distributed across eleven classes was built and validated through a three-stage procedure. AMcNet, a one-dimensional convolutional neural network with residual blocks and invariant to carrier frequency offset, was trained on this dataset, exported to the ONNX format and integrated into a C++ desktop application for spectrum sensing.

The model reached 100% accuracy on the internal test set and correctly classified the ninety transmissions of the functional validation. Accuracy remained above 99% for signal-to-noise ratios equal to or greater than 5 dB and degraded abruptly below that level. The application ran with an approximate consumption of 30% CPU and 25 MB of memory on a dual-core computer without a dedicated GPU, which confirms the feasibility of applying deep learning to the classification of real modulations on low-cost hardware.

**Keywords:** automatic modulation classification, software defined radio, convolutional neural network, spectrum sensing, deep learning.

## ÍNDICE GENERAL

|                                                          |    |
|----------------------------------------------------------|----|
| AGRADECIMIENTO .....                                     | 7  |
| DEDICATORIA .....                                        | 8  |
| RESUMEN .....                                            | 9  |
| ABSTRACT .....                                           | 10 |
| ÍNDICE GENERAL.....                                      | 11 |
| ÍNDICE DE TABLAS .....                                   | 16 |
| ÍNDICE DE FIGURAS .....                                  | 17 |
| ÍNDICE DE SIGLAS Y ACRÓNIMOS.....                        | 19 |
| INTRODUCCIÓN.....                                        | 23 |
| Capítulo I: Generalidades del proyecto .....             | 24 |
| 1.1 Antecedentes.....                                    | 24 |
| 1.2 Descripción del proyecto.....                        | 28 |
| 1.3 Objetivos .....                                      | 29 |
| 1.3.1 Objetivo general.....                              | 29 |
| 1.3.2 Objetivos específicos.....                         | 29 |
| 1.4 Justificación .....                                  | 30 |
| 1.5 Alcance del proyecto.....                            | 32 |
| 1.6 Metodología .....                                    | 34 |
| 1.6.1 Revisión bibliográfica .....                       | 34 |
| 1.6.2 Configuración del entorno experimental .....       | 34 |
| 1.6.3 Recolección de señales OTA .....                   | 34 |
| 1.6.4 Validación y organización del <i>dataset</i> ..... | 35 |
| 1.6.5 Entrenamiento del modelo.....                      | 35 |
| 1.6.6 Evaluación del sistema .....                       | 35 |
| 1.6.7 Herramientas y recursos .....                      | 36 |
| 1.6.8 Proceso de recolección de muestras.....            | 36 |
| 1.6.9 Etapas del entrenamiento y despliegue.....         | 37 |
| Capítulo II: Bases teóricas .....                        | 39 |
| 2.1 Marco contextual.....                                | 39 |
| 2.2 Marco conceptual .....                               | 40 |
| 2.3 Marco teórico .....                                  | 40 |
| 2.3.1 Modulación digital .....                           | 40 |

|                                                                                        |    |
|----------------------------------------------------------------------------------------|----|
| 2.3.2 Relación Señal-Ruido.....                                                        | 40 |
| 2.3.3 Constelación de señal.....                                                       | 41 |
| 2.3.4 Modulación BPSK .....                                                            | 41 |
| 2.3.5 Modulación QPSK.....                                                             | 42 |
| 2.3.6 Modulación FSK .....                                                             | 43 |
| 2.3.7 Justificación académica del uso de BPSK, QPSK y FSK .....                        | 44 |
| 2.3.8 Concepto de AMC .....                                                            | 45 |
| 2.3.9 Principios de la clasificación automática de modulación .....                    | 45 |
| 2.3.10 Clasificación automática de modulaciones digitales.....                         | 46 |
| 2.3.11 Clasificación de señales .....                                                  | 46 |
| 2.3.12 Diferencias entre clasificación de señales y clasificación de modulaciones..... | 47 |
| 2.3.13 Método basado en verosimilitud .....                                            | 48 |
| 2.3.14 Método basado en características .....                                          | 49 |
| 2.3.15 Métodos basados en <i>Machine Learning</i> .....                                | 50 |
| 2.3.16 Métodos basados en <i>Deep Learning</i> .....                                   | 51 |
| 2.3.17 Clasificación basada en <i>Deep Learning</i> .....                              | 51 |
| 2.3.18 Redes neuronales convolucionales (CNN).....                                     | 52 |
| 2.3.19 Redes LSTM e híbridas CNN-LSTM .....                                            | 53 |
| 2.3.20 Justificación del uso de CNN .....                                              | 53 |
| 2.3.21 Radio definida por <i>software</i> .....                                        | 54 |
| 2.3.22 Implementación con radio definida por <i>software</i> .....                     | 54 |
| 2.3.23 Aumento de datos.....                                                           | 55 |
| 2.3.24 Radio cognitiva.....                                                            | 56 |
| 2.3.25 Aplicaciones reales de la clasificación automática de modulación.....           | 56 |
| 2.3.26 Panorama actual y tendencias .....                                              | 57 |
| 2.3.27 Desafíos en la clasificación de modulaciones .....                              | 58 |
| Capítulo III: Desarrollo e implementación experimental.....                            | 60 |
| 3.1 Introducción al desarrollo experimental.....                                       | 60 |
| 3.2 Configuración del entorno experimental.....                                        | 60 |
| 3.2.1 <i>Hardware</i> utilizado.....                                                   | 60 |
| 3.2.2 Decisiones de diseño RF .....                                                    | 61 |
| 3.2.3 Estructura del <i>software</i> .....                                             | 62 |
| 3.3 Fase 1: recolección y validación del <i>dataset</i> .....                          | 63 |

|                                                                                            |    |
|--------------------------------------------------------------------------------------------|----|
| 3.3.1 Proceso de captura de modulaciones digitales .....                                   | 63 |
| 3.3.2 Captura de WBFM y ruido .....                                                        | 64 |
| 3.3.3 <i>Pipeline</i> de validación de ventanas .....                                      | 65 |
| 3.3.4 Efecto del desplazamiento de frecuencia de portadora (CFO) en las señales.....       | 66 |
| 3.3.5 Implicaciones del CFO en la elección de arquitectura .....                           | 67 |
| 3.4 Fase 2: arquitectura del modelo y entrenamiento .....                                  | 69 |
| 3.4.1 Arquitectura AMCNet (CNN-1D residual).....                                           | 69 |
| 3.4.2 Preprocesamiento de datos.....                                                       | 71 |
| 3.4.3 Aumento de datos ( <i>data augmentation</i> ) .....                                  | 71 |
| 3.4.4 Proceso de entrenamiento .....                                                       | 73 |
| 3.4.5 Exportación del modelo a formato ONNX.....                                           | 73 |
| 3.5 Fase 3: Aplicación del sensado espectral en tiempo real.....                           | 75 |
| 3.5.1 <i>Stack</i> tecnológico y arquitectura multihilo.....                               | 76 |
| 3.5.2 Demodulación FM y reproducción de audio en vivo .....                                | 78 |
| 3.5.3 Interfaz gráfica con Dear ImGui e ImPlot .....                                       | 80 |
| 3.5.4 Clasificación de roles, contexto regulatorio y detección de huecos espectrales ..... | 80 |
| 3.5.5 Validación DSP complementaria para predicciones WBFM .....                           | 81 |
| 3.5.6 Análisis de capacidad: Shannon y Nyquist.....                                        | 82 |
| 3.5.7 Validación con transmisor de prueba .....                                            | 84 |
| 3.5.8 Diseño del mecanismo de Open-Set Recognition (OSR) .....                             | 85 |
| 3.6 Restricciones del <i>hardware</i> y decisiones de diseño .....                         | 86 |
| 3.6.1 Estabilidad del búfer USB: fijación de la tasa de muestreo .....                     | 86 |
| 3.6.2 Gestión térmica del Pluto+ SDR durante la captura .....                              | 87 |
| 3.6.3 Diseño del control de ganancia para evitar saturación del ADC.....                   | 87 |
| 3.6.4 Sincronización TX/RX sin protocolo nativo de coordinación .....                      | 87 |
| 3.6.5 CFO entre osciladores: descarte de arquitecturas CNN-2D .....                        | 88 |
| 3.6.6 Balance del <i>dataset</i> : mecanismo de <i>fill-up</i> por clase.....              | 88 |
| 3.6.7 Criterio de selección del tamaño de ventana I/Q.....                                 | 88 |
| 3.6.8 Migración a C++ por limitaciones del GIL de Python .....                             | 89 |
| 3.7 Consideraciones finales del capítulo.....                                              | 89 |
| Capítulo IV: Resultados y validación Experimental .....                                    | 91 |
| 4.1 Introducción .....                                                                     | 91 |
| 4.2 Caracterización de las señales capturadas.....                                         | 91 |

|                                                                                 |     |
|---------------------------------------------------------------------------------|-----|
| 4.2.1 Diagramas de constelación I/Q.....                                        | 92  |
| 4.2.2 Densidad espectral de potencia.....                                       | 93  |
| 4.3 Resultados del entrenamiento del modelo CNN-1D residual .....               | 95  |
| 4.4 Evaluación del modelo mediante matriz de confusión.....                     | 96  |
| 4.5 Análisis de confianza del clasificador.....                                 | 98  |
| 4.5.1 Comportamiento del clasificador ante señales no entrenadas .....          | 100 |
| 4.6 Alcance de la robustez frente al ruido .....                                | 103 |
| 4.6.1 Condiciones de ruido consideradas durante el entrenamiento .....          | 103 |
| 4.6.2 Evaluación de la exactitud en función del SNR .....                       | 104 |
| 4.7 Validación funcional de la aplicación en C++ .....                          | 106 |
| 4.7.1 Pantalla principal del <i>dashboard</i> .....                             | 106 |
| 4.7.2 Visualización de PSD y constelación en tiempo real .....                  | 107 |
| 4.7.3 Clasificación de modulaciones transmitidas con Pluto+ SDR .....           | 108 |
| 4.7.4 Detección de señales WBFM comerciales.....                                | 112 |
| 4.7.5 Identificación de huecos espectrales .....                                | 113 |
| 4.8 Caracterización operativa y limitaciones de medición .....                  | 114 |
| 4.8.1 Plataforma de ejecución.....                                              | 114 |
| 4.8.2 Uso observado de CPU y memoria.....                                       | 115 |
| 4.8.3 Operación continua y actualización .....                                  | 117 |
| 4.8.4 Continuidad de adquisición y pérdida de muestras.....                     | 117 |
| 4.8.5 Limitaciones de la medición temporal .....                                | 118 |
| 4.9 Discusión general de resultados .....                                       | 119 |
| 4.10 Limitaciones observadas .....                                              | 120 |
| 4.10.1 Caso de estudio: falsos positivos WBFM fuera de la banda comercial ..... | 122 |
| 4.11 Trabajo futuro.....                                                        | 123 |
| 4.12 Consideraciones finales del capítulo.....                                  | 124 |
| CONCLUSIONES .....                                                              | 125 |
| RECOMENDACIONES .....                                                           | 127 |
| REFERENCIAS.....                                                                | 128 |
| ANEXO A. MANUAL DE INSTALACIÓN, CONFIGURACIÓN Y OPERACIÓN .....                 | 134 |
| A.1 Objetivo .....                                                              | 134 |
| A.2 Requisitos.....                                                             | 134 |
| A.2.1 <i>Hardware</i> .....                                                     | 134 |

|                                                  |     |
|--------------------------------------------------|-----|
| A.2.2 <i>Software</i> .....                      | 134 |
| A.3 Archivos principales.....                    | 135 |
| A.4 Compilación y ejecución mediante CLion ..... | 135 |
| A.5 Preparación del RTL-SDR.....                 | 136 |
| A.6 Configuración desde la interfaz.....         | 136 |
| A.7 Operación del <i>dashboard</i> .....         | 137 |
| A.7.1 Modo de barrido.....                       | 137 |
| A.7.2 Modo detalle .....                         | 137 |
| A.8 Ejecución del transmisor de prueba .....     | 138 |
| A.9 Solución de errores frecuentes .....         | 138 |
| A.10 Seguridad y advertencias .....              | 139 |
| A.11 Cierre del sistema .....                    | 140 |

## ÍNDICE DE TABLAS

|                                                                                                                                   |     |
|-----------------------------------------------------------------------------------------------------------------------------------|-----|
| Tabla 1. Diferencias entre clasificación de señales y clasificación de modulaciones .....                                         | 47  |
| Tabla 2. Comparación de la CNN con otros modelos de clasificación.....                                                            | 53  |
| Tabla 3. Especificaciones del <i>hardware</i> utilizado .....                                                                     | 61  |
| Tabla 4. Interpretación de los diagramas de constelación I/Q.....                                                                 | 93  |
| Tabla 5. Interpretación de la densidad espectral de potencia .....                                                                | 94  |
| Tabla 6. Análisis del umbral de confianza softmax.....                                                                            | 99  |
| Tabla 7. Falso rechazo y falsa aceptación del sistema completo frente a modulaciones no<br>entrenadas de familias distintas ..... | 102 |
| Tabla 8. Exactitud de clasificación en función del SNR del ruido añadido .....                                                    | 104 |
| Tabla 9. Resultados estabilizados de clasificación en transmisiones OTA repetidas.....                                            | 109 |
| Tabla 10. Intervalos aproximados de SNR durante las transmisiones OTA .....                                                       | 110 |
| Tabla 11. Comportamiento temporal observado durante los cambios de modulación .....                                               | 111 |
| Tabla 12. Funciones validadas en el <i>dashboard</i> C++ .....                                                                    | 113 |
| Tabla 13. Características del equipo utilizado para ejecutar la aplicación.....                                                   | 114 |
| Tabla 14. Uso de recursos observado durante la ejecución de <i>amc_ota</i> .....                                                  | 115 |
| Tabla A1. Parámetros configurables desde el <i>dashboard</i> .....                                                                | 136 |
| Tabla A2. Errores frecuentes y acciones recomendadas .....                                                                        | 138 |

## ÍNDICE DE FIGURAS

|                                                                                                             |    |
|-------------------------------------------------------------------------------------------------------------|----|
| Figura 1. Diagrama de flujo de la metodología .....                                                         | 36 |
| Figura 2. Topología del sistema experimental.....                                                           | 37 |
| Figura 3. Forma de onda de señal BPSK.....                                                                  | 42 |
| Figura 4. Forma de onda de señal QPSK .....                                                                 | 43 |
| Figura 5. Forma de onda de señal FSK .....                                                                  | 44 |
| Figura 6. Constelación de modulación BPSK y QPSK.....                                                       | 44 |
| Figura 7. Constelación de modulación FSK.....                                                               | 45 |
| Figura 8. Clasificación de señales y clasificación de modulaciones.....                                     | 48 |
| Figura 9. Algoritmos esenciales en <i>Machine Learning</i> .....                                            | 51 |
| Figura 10. Métodos de clasificación de modulaciones .....                                                   | 51 |
| Figura 11. Flujo de recepción de señales .....                                                              | 55 |
| Figura 12. Ilustración de guerra electrónica .....                                                          | 57 |
| Figura 13. Comparación con enfoques de última generación .....                                              | 58 |
| Figura 14. Flujo de datos y control entre los módulos de <i>software</i> de las tres fases del sistema..... | 63 |
| Figura 15. Proceso desde la captura pasando por la validación hasta llegar al <i>dataset</i> .....          | 66 |
| Figura 16. Efecto del <i>Carrier Frequency Offset</i> .....                                                 | 67 |
| Figura 17. Solución ante el CFO .....                                                                       | 69 |
| Figura 18. Arquitectura AMCNet: CNN-1D residual para clasificación en tiempo real.....                      | 71 |
| Figura 19. Flujo de preprocesamiento, división del <i>dataset</i> y aumento dinámico de datos I/Q.....      | 72 |
| Figura 20. Entrenamiento y exportación del modelo.....                                                      | 75 |
| Figura 21. Diagrama de arquitectura multihilo y <i>stack</i> tecnológico del sistema SDR .....              | 78 |
| Figura 22. Diagrama de bloques de la cadena de demodulación WBFM .....                                      | 79 |

|                                                                                                                       |     |
|-----------------------------------------------------------------------------------------------------------------------|-----|
| Figura 23. Flujo de análisis de capacidad y selección de modulación basado en los teoremas de Shannon y Nyquist ..... | 84  |
| Figura 24. Diagramas de constelación I/Q de las modulaciones capturadas.....                                          | 92  |
| Figura 25. Densidad espectral de potencia de las señales capturadas.....                                              | 94  |
| Figura 26. Curvas de pérdida y exactitud durante el entrenamiento y validación.....                                   | 96  |
| Figura 27. Matriz de confusión normalizada del conjunto de prueba.....                                                | 97  |
| Figura 28. Distribución de confianza entre predicciones correctas e incorrectas.....                                  | 98  |
| Figura 29. Etiqueta final asignada por la aplicación a cada tipo de señal no entrenada .....                          | 101 |
| Figura 30. Exactitud de clasificación en función del SNR del ruido añadido .....                                      | 105 |
| Figura 31. Pantalla principal de la aplicación de sensado espectral en C++ .....                                      | 107 |
| Figura 32. Visualización de PSD y constelación en tiempo real dentro del <i>dashboard</i> .....                       | 108 |
| Figura 33. Clasificación estabilizada de una modulación transmitida mediante Pluto+ SDR109                            |     |
| Figura 34. Detección de señales WBFM comerciales mediante el <i>dashboard</i> .....                                   | 113 |
| Figura 35. Identificación de huecos espectrales en la aplicación de sensado .....                                     | 113 |
| Figura 36. Consumo instantáneo de CPU y memoria del proceso amc_ota durante la ejecución .....                        | 116 |
| Figura 37. Características del procesador y utilización general del equipo durante la ejecución .....                 | 117 |

## ÍNDICE DE SIGLAS Y ACRÓNIMOS

| Sigla / Acrónimo | Significado                                               | Descripción en español                                                                         |
|------------------|-----------------------------------------------------------|------------------------------------------------------------------------------------------------|
| ADC              | Analog-to-Digital Converter                               | Convertidor analógico-digital.                                                                 |
| AGC              | Automatic Gain Control                                    | Control automático de ganancia.                                                                |
| ALRT             | Average Likelihood Ratio Test                             | Prueba de relación de verosimilitud promediada.                                                |
| AM-DSB           | Amplitude Modulation Double-Sideband                      | Modulación de amplitud de doble banda lateral.                                                 |
| AMC              | Automatic Modulation Classification                       | Clasificación automática de modulación.                                                        |
| AMC-OTA          | Automatic Modulation Classification Over-The-Air          | Clasificación automática de modulaciones en canal real (sistema desarrollado en este trabajo). |
| AMCNet           | Automatic Modulation Classification Network               | Red neuronal para clasificación automática de modulaciones.                                    |
| API              | Application Programming Interface                         | Interfaz de programación de aplicaciones.                                                      |
| ARCOTEL          | Agencia de Regulación y Control de las Telecomunicaciones | Ente regulador de las telecomunicaciones en Ecuador.                                           |
| ASK              | Amplitude Shift Keying                                    | Modulación por desplazamiento de amplitud.                                                     |
| AWGN             | Additive White Gaussian Noise                             | Ruido blanco gaussiano aditivo.                                                                |
| BB               | Baseband                                                  | Banda base.                                                                                    |
| BFSK             | Binary Frequency Shift Keying                             | Modulación FSK binaria.                                                                        |
| BPSK             | Binary Phase Shift Keying                                 | Modulación por desplazamiento de fase binaria.                                                 |
| CFO              | Carrier Frequency Offset                                  | Desplazamiento de frecuencia de portadora.                                                     |
| CNN              | Convolutional Neural Network                              | Red neuronal convolucional.                                                                    |
| CNN-1D           | One-Dimensional Convolutional Neural Network              | Red neuronal convolucional unidimensional.                                                     |
| CNN-2D           | Two-Dimensional Convolutional Neural Network              | Red neuronal convolucional bidimensional.                                                      |
| CNN-LSTM         | Convolutional Neural Network - Long Short-Term Memory     | Arquitectura híbrida de red convolucional y red de memoria a largo y corto plazo.              |
| CPU              | Central Processing Unit                                   | Unidad central de procesamiento.                                                               |
| CVNN             | Complex-Valued Neural Network                             | Red neuronal de valores complejos.                                                             |
| DC               | Direct Current                                            | Componente continua de una señal.                                                              |
| DL               | Deep Learning                                             | Aprendizaje profundo.                                                                          |

|       |                                                   |                                                                                           |
|-------|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| DNN   | Deep Neural Network                               | Red neuronal profunda.                                                                    |
| DSP   | Digital Signal Processing                         | Procesamiento digital de señales.                                                         |
| DSSS  | Direct Sequence Spread Spectrum                   | Espectro ensanchado por secuencia directa.                                                |
| ESPE  | Universidad de las Fuerzas Armadas ESPE           | Institución universitaria ecuatoriana tomada como antecedente investigativo.              |
| FAR   | False Acceptance Rate                             | Tasa de falsa aceptación: señales no entrenadas que se muestran como una modulación real. |
| FLOPs | Floating Point Operations                         | Operaciones de punto flotante.                                                            |
| FM    | Frequency Modulation                              | Modulación de frecuencia.                                                                 |
| FRR   | False Rejection Rate                              | Tasa de falso rechazo: señales conocidas que no se muestran como su modulación.           |
| FSK   | Frequency Shift Keying                            | Modulación por desplazamiento de frecuencia.                                              |
| GFSK  | Gaussian Frequency Shift Keying                   | Modulación por desplazamiento de frecuencia gaussiana.                                    |
| GIL   | Global Interpreter Lock                           | Bloqueo global del intérprete de Python.                                                  |
| GLRT  | Generalized Likelihood Ratio Test                 | Prueba de relación de verosimilitud generalizada.                                         |
| GPU   | Graphics Processing Unit                          | Unidad de procesamiento gráfico.                                                          |
| GUI   | Graphical User Interface                          | Interfaz gráfica de usuario.                                                              |
| HDF5  | Hierarchical Data Format version 5                | Formato jerárquico de datos versión 5.                                                    |
| HLRT  | Hybrid Likelihood Ratio Test                      | Prueba de relación de verosimilitud híbrida.                                              |
| HOC   | Higher Order Cumulants                            | Cumulantes de orden superior.                                                             |
| I/Q   | In-phase / Quadrature                             | Componentes en fase y cuadratura de una señal.                                            |
| IEEE  | Institute of Electrical and Electronics Engineers | Instituto de Ingeniería Eléctrica y Electrónica.                                          |
| IF    | Intermediate Frequency                            | Frecuencia intermedia.                                                                    |
| IIR   | Infinite Impulse Response                         | Filtro de respuesta infinita al impulso.                                                  |
| IMT   | International Mobile Telecommunications           | Telecomunicaciones móviles internacionales (bandas de servicios móviles).                 |
| IoT   | Internet of Things                                | Internet de las cosas.                                                                    |
| IP    | Internet Protocol                                 | Protocolo de Internet.                                                                    |
| ISI   | Inter-Symbol Interference                         | Interferencia entre símbolos.                                                             |
| k-NN  | k-Nearest Neighbors                               | Algoritmo de los k vecinos más cercanos.                                                  |
| LFM   | Linear Frequency Modulation                       | Modulación lineal de frecuencia (señal chirp de barrido).                                 |
| LSTM  | Long Short-Term Memory                            | Red de memoria a largo y corto plazo.                                                     |

|          |                                                         |                                                                                |
|----------|---------------------------------------------------------|--------------------------------------------------------------------------------|
| LWAMCNet | Lightweight Automatic Modulation Classification Network | Red neuronal ligera para clasificación automática de modulaciones.             |
| M-PSK    | M-ary Phase Shift Keying                                | Modulación por desplazamiento de fase de M niveles.                            |
| M-QAM    | M-ary Quadrature Amplitude Modulation                   | Modulación de amplitud en cuadratura de M niveles.                             |
| MATLAB   | Matrix Laboratory                                       | <i>Software</i> utilizado para cálculo numérico y procesamiento de señales.    |
| ML       | Machine Learning                                        | Aprendizaje automático.                                                        |
| MLP      | Multilayer Perceptron                                   | Perceptrón multicapa.                                                          |
| MSPS     | Mega Samples Per Second                                 | Mega muestras por segundo.                                                     |
| NN       | Neural Network                                          | Red neuronal.                                                                  |
| OFDM     | Orthogonal Frequency Division Multiplexing              | Multiplexación por división de frecuencias ortogonales (señal multiportadora). |
| ONNX     | Open Neural Network Exchange                            | Formato abierto para intercambio de modelos de redes neuronales.               |
| OOD      | Out-Of-Distribution                                     | Datos fuera de la distribución de entrenamiento.                               |
| OOK      | On-Off Keying                                           | Modulación por encendido y apagado de la portadora.                            |
| OSR      | Open-Set Recognition                                    | Reconocimiento de conjunto abierto.                                            |
| OTA      | Over-The-Air                                            | Transmisión o captura de señales por el aire.                                  |
| OTFS     | Orthogonal Time Frequency Space                         | Modulación en espacio ortogonal tiempo-frecuencia.                             |
| PAPR     | Peak-to-Average Power Ratio                             | Relación entre potencia pico y potencia promedio.                              |
| PCM      | Pulse Code Modulation                                   | Modulación por código de pulsos.                                               |
| PPM      | Parts Per Million                                       | Partes por millón (unidad para el error de frecuencia del oscilador).          |
| PSD      | Power Spectral Density                                  | Densidad espectral de potencia.                                                |
| PSK      | Phase Shift Keying                                      | Modulación por desplazamiento de fase.                                         |
| QAM      | Quadrature Amplitude Modulation                         | Modulación de amplitud en cuadratura.                                          |
| QPSK     | Quadrature Phase Shift Keying                           | Modulación por desplazamiento de fase en cuadratura.                           |
| RadioML  | Radio Machine Learning dataset                          | Conjunto de datos de referencia para clasificación de modulaciones.            |
| ReLU     | Rectified Linear Unit                                   | Unidad lineal rectificada (función de activación).                             |
| RF       | Radio Frequency                                         | Radiofrecuencia.                                                               |
| RRC      | Root Raised Cosine                                      | Filtro de coseno alzado raíz.                                                  |

|         |                                              |                                                                        |
|---------|----------------------------------------------|------------------------------------------------------------------------|
| RTL-SDR | Realtek Software Defined Radio               | Receptor de radio definida por <i>software</i> basado en chip Realtek. |
| RX      | Receiver                                     | Receptor.                                                              |
| SDR     | Software Defined Radio                       | Radio definida por <i>software</i> .                                   |
| SNR     | Signal-to-Noise Ratio                        | Relación señal-ruido.                                                  |
| SVM     | Support Vector Machine                       | Máquina de soporte vectorial.                                          |
| TX      | Transmitter                                  | Transmisor.                                                            |
| UHF     | Ultra High Frequency                         | Frecuencia ultra alta.                                                 |
| UPSE    | Universidad Estatal Península de Santa Elena | Universidad donde se desarrolla el trabajo de titulación.              |
| USB     | Universal Serial Bus                         | Bus serie universal.                                                   |
| USRP    | Universal Software Radio Peripheral          | Plataforma de radio definida por <i>software</i> .                     |
| VHF     | Very High Frequency                          | Frecuencia muy alta.                                                   |
| VRAM    | Video Random Access Memory                   | Memoria de video de acceso aleatorio.                                  |
| WBFM    | Wideband Frequency Modulation                | Modulación de frecuencia de banda ancha.                               |

## INTRODUCCIÓN

El espectro radioeléctrico es un recurso limitado que cada vez está más saturado. Esto exige sistemas de control más efectivos. En este contexto, la clasificación automática de tipos de modulaciones permite determinar qué tipo de señal llega sin saber de dónde proviene. Esta capacidad es fundamental para sistemas de recepción adaptativos, los sistemas de radio inteligente y las actividades de vigilancia del espectro.

Los métodos clásicos se basan en la máxima verosimilitud o en características que se eligen a mano. Estos métodos pierden eficacia cuando hay mucho ruido, cuando hay desvanecimiento o cuando ocurren desplazamientos de frecuencia. El aprendizaje profundo cambió esta situación. Las redes convolucionales pueden aprender por sí mismas las representaciones importantes del problema. Estas redes usan muestras sin procesar. Sin embargo, muchas investigaciones entrenan sus modelos con datos que no son reales. Estos datos no reflejan las imperfecciones del *hardware*. Por eso, los resultados obtenidos no aseguran un buen desempeño con señales reales capturadas por un receptor físico.

El presente trabajo atiende ese vacío mediante un sistema de clasificación de modulaciones en tiempo real, entrenado únicamente con señales captadas por el aire a través de radio definida por *software* e integrado en una aplicación de sensado espectral.

El documento tiene cuatro capítulos. El primer capítulo muestra los antecedentes, los objetivos y la forma de trabajo. El segundo capítulo trata el marco teórico. El tercero explica cómo se hizo el experimento. El cuarto muestra los resultados y cómo se validó el prototipo. Al final hay conclusiones, sugerencias, referencias y un anexo con el manual de operación.

# Capítulo I: Generalidades del proyecto

## 1.1 Antecedentes

La clasificación automática de modulaciones digitales, conocida como AMC, es un tema muy importante en las telecomunicaciones hoy en día. Se usa especialmente para crear receptores que no necesitan información del transmisor antes de recibir. También es útil para aplicaciones de radio cognitiva [1]. La finalidad principal de esta técnica consiste en reconocer qué tipo de modulación emplea una señal recibida. Este proceso resulta estratégico porque contribuye a un uso más eficiente del espectro, refuerza la seguridad y facilita la compatibilidad entre distintas tecnologías inalámbricas.

Al principio, los investigadores encontraron dos formas principales de estudiar AMC. Una de ellas se basaba en la máxima verosimilitud, que usaba estadísticas para encontrar la modulación más probable de una señal. Esta forma tenía una base matemática fuerte, pero necesitaba mucha capacidad de cálculo. La otra forma se centraba en características de la señal, como momentos estadísticos, varianza de amplitud o curtosis. Estos parámetros se utilizaban en clasificadores como las máquinas de soporte vectorial o los algoritmos k-NN [2]. Ambos métodos tenían problemas importantes cuando se trataba de canales con mucho ruido, señales que se desvanecían, desfases en la señal de portadora o cambios en el ancho de banda. Estas condiciones son comunes en entornos inalámbricos de la vida real y hacían que los métodos tradicionales perdieran precisión y no se pudieran aplicar de manera práctica.

La evolución de las redes neuronales y, en particular, de las técnicas de aprendizaje profundo, impulsó un cambio de paradigma en AMC. En lugar de separar la etapa de extracción de características y la de clasificación, las arquitecturas profundas permiten integrar ambas fases

en un único modelo. Así, es la propia red la que aprende representaciones jerárquicas y discriminativas de las señales a partir de datos en bruto [3]. Este cambio ha demostrado mejorar la exactitud y la robustez del proceso, especialmente en escenarios con SNR moderada o baja.

Dentro del campo del aprendizaje profundo, las redes neuronales convolucionales (CNN) han surgido como una de las herramientas más eficaces para AMC. Las CNN-1D son capaces de procesar directamente secuencias temporales I/Q, capturando patrones locales de amplitud y fase en las muestras de la señal. Las CNN-2D, por su parte, operan sobre representaciones transformadas de la señal, como espectrogramas, mapas de constelación o distribuciones tiempo-frecuencia, lo que permite aprovechar la potencia del procesamiento visual aplicado a datos de comunicaciones [4]. Investigaciones recientes reportan exactitudes superiores al 90 % en la clasificación de modulaciones comunes, siempre que el modelo sea entrenado con suficientes ejemplos representativos [5].

La literatura también tiene sus limitaciones. Cuando la relación señal-ruido es muy baja, es decir, por debajo de 0 decibelios, incluso las redes neuronales convolucionales tienen problemas para funcionar bien. Esto se debe a que las características que permiten distinguir entre las cosas se vuelven menos claras. Además, hay un riesgo de que los modelos se ajusten demasiado cuando se entrenan solo con datos generados por simulación y no se incluyen datos reales capturados [3].

Los receptores definidos por *software* son muy importantes en este contexto. Herramientas como GNU Radio, USRP, HackRF y RTL-SDR permiten capturar señales moduladas en condiciones reales. Esto incluye efectos prácticos como interferencias, problemas con la frecuencia y *jitter*. Los modelos deben aprender a manejar estos efectos prácticos [6]. Su empleo facilita la adquisición de bases de datos más fieles a la realidad y, a la vez, permite recrear escenarios de prueba específicos para evaluar la robustez de los clasificadores. La integración del aprendizaje

profundo con SDR constituye, por tanto, un avance tecnológico clave en la clasificación de modulaciones.

Investigaciones recientes han propuesto usar modelos que combinan diferentes técnicas. Por ejemplo, se han creado redes que mezclan convoluciones con mecanismos de memoria que recuerdan cosas de manera temporal. Esto permite a estos modelos reconocer patrones que ocurren a lo largo del tiempo. Además, hay variantes como las ResNet, que tienen conexiones especiales para saltar ciertas capas, lo que facilita el entrenamiento de redes muy profundas sin que se pierda mucha información durante el proceso [7]. Pese a sus ventajas, no son soluciones del todo prácticas: requieren gran capacidad de cálculo, lo que complica su aplicación en dispositivos pequeños o de bajo consumo, especialmente dentro del IoT o en sistemas embebidos.

De forma paralela, los avances en AMC con aprendizaje profundo ya se han llevado a escenarios concretos. Entre ellos destacan el ámbito de la seguridad y defensa, donde permiten detectar señales sospechosas o no autorizadas, y la radio cognitiva, que aprovecha la clasificación de la modulación de usuarios primarios para que los secundarios accedan dinámicamente al espectro sin causar interferencias [8]. En entornos civiles, la clasificación automática de modulaciones puede emplearse en la gestión de espectro compartido, sistemas satelitales y en redes vehiculares, contribuyendo a optimizar recursos y mantener la calidad de servicio.

En Ecuador se ha desarrollado un sistema de clasificación de modulaciones digitales mediante aprendizaje profundo en la Universidad de las Fuerzas Armadas ESPE, lo que evidencia la pertinencia de aplicar redes neuronales en este ámbito [9]. No obstante, dicho estudio se enfocó en un conjunto específico de modulaciones y escenarios experimentales, lo cual señala la necesidad de continuar explorando técnicas complementarias y evaluaciones con señales capturadas en condiciones reales mediante receptores SDR.

A nivel internacional, múltiples grupos han demostrado la viabilidad de clasificar modulaciones digitales mediante aprendizaje profundo en escenarios de canal retadores. En Estados Unidos, investigadores de Virginia Tech y The MITRE Corporation mostraron que redes convolucionales superan a los métodos tradicionales para el reconocimiento de modulación en presencia de ruido y desvanecimiento, lo que impulsó el uso de conjuntos como RadioML [10]. En Suiza, el Swiss AI Lab IDSIA (USI/SUPSI) exploró arquitecturas profundas con regularización y técnicas de entrenamiento robusto, reportando mejoras frente a clasificadores clásicos en condiciones de SNR baja [11].

En esta misma línea, el sensado espectral asistido por aprendizaje automático ha sido validado sobre señales reales capturadas con *hardware* de bajo costo. Saber et al. implementaron una etapa de sensado espectral orientada a dispositivos embebidos dentro de redes cognitivas, empleando señales de 433 MHz moduladas en ASK y FSK adquiridas mediante un receptor RTL-SDR. Sobre esos datos compararon redes neuronales artificiales, máquinas de soporte vectorial, árboles de decisión y k-NN, y concluyeron que las redes neuronales y las SVM alcanzaron las mejores probabilidades de detección, si bien el desempeño resultó sensible al valor de la relación señal-ruido [12]. Este antecedente resulta especialmente cercano al presente proyecto, ya que demuestra que un receptor SDR de bajo costo, combinado con clasificadores entrenados a partir de señales capturadas y no simuladas, es suficiente para determinar la ocupación del espectro. Al mismo tiempo, evidencia el límite de los clasificadores clásicos frente a niveles altos de ruido, lo que justifica el uso de arquitecturas convolucionales profundas para la clasificación de modulaciones que se propone en este trabajo.

La literatura revisada muestra cómo los métodos han cambiado con el tiempo. Antes, se utilizaban métodos tradicionales que dependían de la máxima verosimilitud y de características

manuales. Ahora, las cosas son diferentes. Las arquitecturas de aprendizaje profundo son muy importantes. Estas arquitecturas combinan dos pasos en uno: la extracción de características y la clasificación. Las redes neuronales convolucionales, también llamadas CNN, son fundamentales en este cambio. Las CNN son el elemento principal que ha hecho que las cosas cambien de esta manera. En el ámbito tecnológico, el uso de SDR permite obtener datos que reflejan condiciones reales y mejora la estabilidad de los modelos. Aun así, lograr alta precisión en condiciones difíciles de ruido y canal sigue siendo un reto. Este reto ofrece una oportunidad de investigación y respalda este trabajo. Aplicar técnicas de aprendizaje profundo a la clasificación automática de modulaciones, usando señales capturadas con SDR, es una contribución importante para el campo académico y el campo práctico de las telecomunicaciones. Este enfoque acerca el análisis a situaciones reales y fortalece el uso en el monitoreo y el aprovechamiento eficiente del espectro radioeléctrico.

## **1.2 Descripción del proyecto**

El presente proyecto plantea el desarrollo de un sistema capaz de reconocer de manera automática distintos tipos de modulación a partir de señales adquiridas con dispositivos de radio definida por *software* (SDR). Se busca obtener un clasificador práctico y verificable, destinado al análisis de señales capturadas en condiciones próximas a las de un entorno real de operación.

La metodología incluye la recopilación de señales usando SDR. Después, se organizan estas señales en un conjunto de datos con etiquetas. Luego se entrena un modelo basado en redes neuronales convolucionales. Para el sistema se usan muestras I/Q. A estas muestras se les aplican etapas de preprocesamiento, validación y aumento de datos. Todo esto busca mejorar la fiabilidad del clasificador.

El desempeño del modelo será evaluado así: se usarán métricas para clasificación y se realizarán pruebas de funcionalidad en tiempo real. También se fijará en qué medida se puede integrar el modelo en una herramienta de sensado espectral. Esto permitirá descubrir qué señales están presentes en distintas frecuencias e investigar las bandas que aún no han sido aprovechadas dentro del espectro radioeléctrico.

Un sistema que se pueda repetir y que esté bien explicado es lo que se busca. Este sistema debe mostrar pruebas reales de que el aprendizaje profundo puede usarse para clasificar modulaciones de forma automática con SDR.

## **1.3 Objetivos**

### **1.3.1 Objetivo general**

Desarrollar un sistema de clasificación automática en tiempo real de modulaciones OTA mediante una arquitectura CNN-1D residual y un receptor SDR, para apoyar tareas de sensado espectral e identificación de modulaciones en entornos inalámbricos reales.

### **1.3.2 Objetivos específicos**

- Diseñar un entorno experimental SDR orientado a la captura y validación de señales OTA, generando un *dataset* representativo para entrenar y evaluar el modelo.
- Implementar una CNN-1D residual sobre secuencias I/Q reales, con preprocesamiento y aumento de datos que refuercen su robustez ante ruido y CFO.
- Integrar el modelo entrenado en una aplicación de sensado espectral: clasificar señales capturadas por SDR y evaluar la ocupación del espectro en tiempo real.

## 1.4 Justificación

Esta tesis es importante por varias razones. Por un lado, es útil para la academia. Por otro, tiene aplicaciones prácticas. En este estudio, se analiza cómo las redes neuronales convolucionales pueden clasificar automáticamente las modulaciones. Para ello, se utilizan señales que se capturan con receptores definidos por *software*, también llamados SDR.

En el ámbito de las comunicaciones modernas (especialmente en radio cognitiva, gestión dinámica del espectro y aplicaciones militares), la identificación automática del tipo de modulación sin información previa del transmisor constituye una herramienta clave. Esta capacidad favorece la eficiencia en el uso del espectro, la interoperabilidad y la seguridad [3]. Además, estudios recientes muestran que las CNN alcanzan altos niveles de exactitud incluso en condiciones adversas: 86,1 % a  $-2$  dB SNR y 96,5 % a 0 dB SNR. Estos resultados demuestran una robustez destacable aun en entornos con ruido [3].

La literatura muestra limitaciones en las propuestas de baja complejidad, aunque estas pueden funcionar bien en tiempo real en dispositivos con pocos recursos. Esto ocurre en laboratorios universitarios o en plataformas móviles. Un ejemplo es un estudio que cambió la estructura del receptor SDR. Se usaron redes neuronales CNN-LSTM y redes residuales LSTM. El resultado fue que los parámetros se redujeron hasta seis veces. Pero la precisión no se vio afectada [13]. Este hallazgo muestra que es posible diseñar modelos ligeros y funcionales en contextos prácticos, lo cual resulta valioso en proyectos de tesis que requieren prototipos en SDR.

Gran parte de las investigaciones emplea conjuntos de datos sintéticos como RadioML. Aunque útiles para entrenar bajo condiciones controladas de SNR, modulaciones y efectos de canal, estos no siempre reflejan imperfecciones inherentes al *hardware* real, como *jitter*,

interferencias o desplazamientos de frecuencia [6]. Por ello, se vuelve necesario evaluar los clasificadores en señales captadas mediante SDR, asegurando así su generalización y robustez.

Tener contacto con señales reales supone un elemento crucial del valor adicional de esta investigación. La utilización de SDR facilita abordar fenómenos que existen en el mundo real, tales como el desvanecimiento de múltiples caminos, el drift de frecuencia o las variaciones de ganancia, aspectos que no se pueden modelar correctamente en bases de datos simuladas [13]. Esta aproximación aumenta la validez práctica del estudio frente a comparaciones exclusivamente simuladas.

Otro reto crucial es la robustez ante bajo SNR. En entornos reales las señales suelen recibirse con ruido elevado. Investigaciones recientes señalan que mejoras como bloques de dilatación, módulos *squeeze-and-excitation* o *pooling* estadístico incrementan significativamente el desempeño. Un modelo modificado, por ejemplo, alcanzó una exactitud del 98,9 % a 22 dB y cerca de 91 % a 6 dB SNR, superando propuestas previas que requerían mejores condiciones de canal [14].

Desde la perspectiva de eficiencia computacional, arquitecturas ligeras como la Red Neuronal Ligera para la Clasificación Automática de Modulaciones (LWAMCNet) han probado que es posible reducir el número de parámetros entre un 70 % y un 84 %, con lo que se acelera la inferencia sin sacrificar exactitud [6]. Esto confirma la factibilidad de implementar clasificadores en entornos con *hardware* limitado, como los disponibles en laboratorios universitarios ecuatorianos.

Las técnicas de aumento de datos (*data augmentation*) suman otro aporte. Rotación, inversión o adición de ruido gaussiano elevan la exactitud incluso con bases de datos pequeñas y

superan el rendimiento alcanzado sin estas variaciones [15]. La ventaja pesa más en proyectos académicos, donde el acceso a datos reales suele ser restringido.

En Ecuador, la investigación sobre AMC basada en redes neuronales es aún joven. Carrera, Gutiérrez y Salvador, de la ESPE [9], evalúan redes neuronales para clasificar señales moduladas digital y analógicamente a partir de un conjunto de datos artificial. Falta por realizar esa investigación en condiciones reales mediante SDR y enfocada en el sensado del espectro. Este trabajo añade esa dimensión al experimentar e introducir la investigación de AMC con pruebas experimentales, lo que refuerza el conocimiento disponible en el país.

Esta investigación responde a la necesidad de clasificar modulaciones digitales con modelos precisos, robustos y computacionalmente viables a partir de señales reales capturadas en SDR. Con ello se pretende extender el estado del arte internacional, generar soluciones prácticas para escenarios de bajo costo y contribuir al desarrollo académico de la comunidad investigadora en Ecuador.

## **1.5 Alcance del proyecto**

Este proyecto se enfoca en diseñar, implementar y validar un sistema que pueda clasificar automáticamente diferentes tipos de modulaciones que se capturan en condiciones reales mediante dispositivos de radio definidos por *software*. El objetivo es crear un sistema que pueda recolectar señales de radio del aire, entrenar un modelo basado en redes neuronales y luego integrarlo en una aplicación que pueda analizar el espectro de radio en tiempo real.

La primera parte del proyecto implica generar y transmitir diferentes tipos de señales de modulación utilizando un dispositivo de radio definida por *software* que actúa como transmisor. Luego, se capturan las señales recibidas con otro dispositivo similar que actúa como receptor. Las

muestras obtenidas se utilizan para crear un conjunto de datos etiquetados, que incluye ventanas de datos validadas y balanceadas. Este conjunto de datos incluye tanto señales de modulación digital como señales que se encuentran presentes en el espectro de radio.

El proyecto se centra en crear una red neuronal que pueda trabajar directamente con los datos tal como son. Esto significa que no es necesario cambiar los datos para que la red neuronal los entienda. De esta manera, es posible clasificar señales que no son perfectas, ya que pueden haber sido afectadas por cosas que no son ideales. Para mejorar la capacidad de la red neuronal, también se aplicarán técnicas para preparar y aumentar los datos.

El sistema se evaluará mediante métricas de clasificación y pruebas de funcionamiento en tiempo real. También se integrará en una aplicación que pueda realizar barridos de frecuencia, identificar señales activas y detectar huecos en el espectro de radio. De esta manera, la clasificación automática de modulaciones se vincula con una aplicación práctica de monitoreo y análisis del uso del espectro de radio.

Este proyecto no pretende cubrir todas las formas de modulación que existen ni crear estructuras más complicadas. No tiene como objetivo lograr la mejor optimización para dispositivos electrónicos que consumen poca energía. Aun así, utilizando un modelo que se puede exportar y una aplicación que funciona, es posible crear una base para experimentar y mejorar en el futuro la capacidad de trasladar el proyecto a diferentes lugares y hacerlo más eficiente.

Sobre el entorno de pruebas, este trabajo se limita a una validación académica y de laboratorio. No se ha implementado en redes comerciales ni en sistemas operativos críticos. El objetivo principal es demostrar que se puede usar el aprendizaje profundo para clasificar automáticamente las modulaciones en condiciones reales, creando un sistema que sea fácil de reproducir y que esté bien documentado.

## **1.6 Metodología**

El proyecto es experimental y aplicado. Se trata de crear un prototipo que clasifique de forma automática las modulaciones capturadas por el aire usando equipos de radio definida por *software*. Se usan conceptos de comunicaciones digitales, procesamiento de señales, aprendizaje profundo y detección espectral para entrenar y comprobar una red neuronal convolucional unidimensional con bloques residuales que trabaja con señales reales. El aspecto cuantitativo se enfoca en medir el rendimiento. Se usan métricas de clasificación, matrices de confusión y pruebas en tiempo real.

### **1.6.1 Revisión bibliográfica**

- Se estudiarán los métodos antiguos y nuevos para clasificar señales de forma automática.
- Se revisarán investigaciones sobre redes neuronales profundas, sistemas de radio definidos por *software*, señales en I y Q, redes neuronales convolucionales y detección espectral.
- Se establecerán criterios técnicos para elegir una estructura adecuada al procesamiento de señales obtenidas en condiciones reales.

### **1.6.2 Configuración del entorno experimental**

- Se establecerá una topología de transmisión y recepción utilizando dispositivos SDR.
- Se configurarán parámetros de operación como frecuencia central, tasa de muestreo, ganancia, atenuación y distancia entre antenas.
- Se verificará la estabilidad del enlace experimental para garantizar capturas válidas y reproducibles.

### **1.6.3 Recolección de señales OTA**

- Se transmitirán diferentes esquemas de modulación mediante el SDR transmisor.
- Las señales serán capturadas por el SDR receptor en forma de muestras I/Q.
- También se incluirán capturas de señales presentes en el espectro radioeléctrico, con el fin de ampliar el contexto experimental del sistema.

#### **1.6.4 Validación y organización del *dataset***

- Las muestras capturadas serán segmentadas en ventanas de longitud fija.
- Se aplicarán procesos de validación para descartar ventanas con saturación, baja energía o comportamiento anómalo.
- Las señales aceptadas serán etiquetadas y organizadas en un conjunto de datos balanceado para entrenamiento, validación y prueba.

#### **1.6.5 Entrenamiento del modelo**

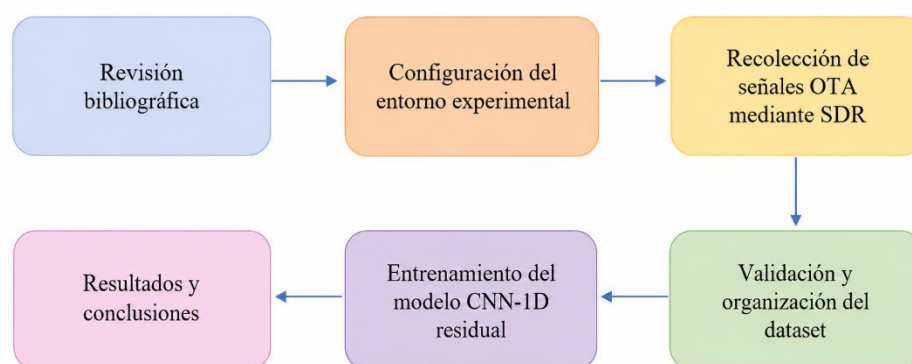
- Se implementará una arquitectura CNN-1D residual para procesar directamente las secuencias I/Q.
- Se aplicarán técnicas de preprocesamiento, como remoción de componente DC y normalización de amplitud.
- Durante el entrenamiento se emplearán estrategias de aumento de datos para mejorar la robustez frente al ruido, variaciones de amplitud y cambios de fase.
- El modelo será entrenado, validado y almacenado tomando como referencia el mejor desempeño obtenido.

#### **1.6.6 Evaluación del sistema**

- Se calcularán métricas de desempeño, incluyendo exactitud global, exactitud por clase y matriz de confusión.
- Se evaluará la capacidad del modelo para clasificar modulaciones capturadas en condiciones reales.
- Los resultados permitirán valorar la viabilidad del enfoque propuesto para aplicaciones de clasificación automática y sensado espectral.

### 1.6.7 Herramientas y recursos

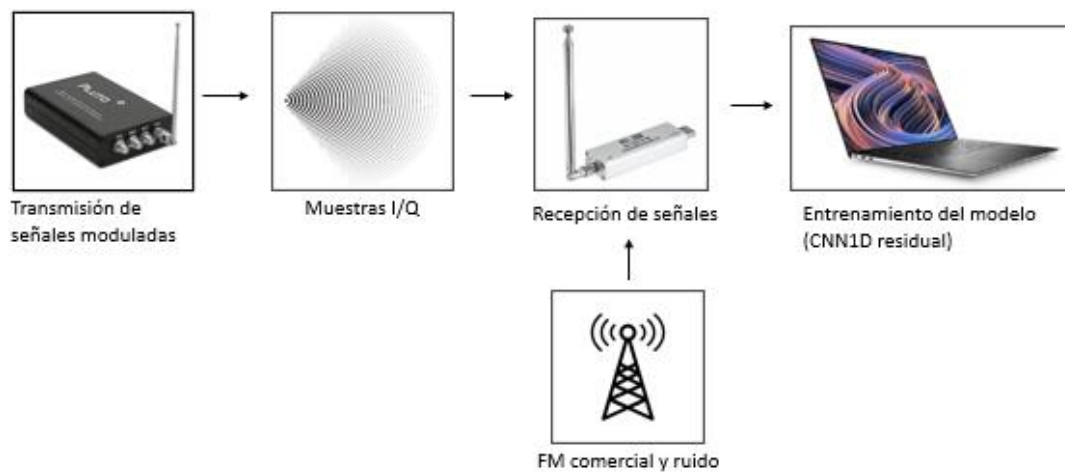
- *Software*: Python, PyTorch, librerías de control SDR, herramientas de procesamiento de señales y entorno de entrenamiento con GPU.
- *Hardware*: SDR transmisor, SDR receptor, antenas omnidireccionales y computadora para entrenamiento e inferencia.
- *Datos*: señales I/Q capturadas mediante SDR y organizadas en ventanas etiquetadas.
- *Técnicas adicionales*: normalización, remoción de offset, validación de ventanas, aumento de datos y exportación del modelo para inferencia.



**Figura 1. Diagrama de flujo de la metodología.**

### 1.6.8 Proceso de recolección de muestras

Primero, se configura el entorno experimental de transmisión y recepción mediante dispositivos SDR. Luego, el transmisor genera las señales moduladas y el receptor captura las muestras I/Q correspondientes. Posteriormente, las capturas son segmentadas en ventanas, validadas mediante criterios de calidad y etiquetadas según la modulación correspondiente. Finalmente, las muestras aceptadas se organizan en un *dataset* balanceado que será utilizado para el entrenamiento, validación y prueba del modelo CNN-1D residual.



**Figura 2. Topología del sistema experimental.**

### 1.6.9 Etapas del entrenamiento y despliegue

- Recopilar señales I/Q usando un SDR.
- Dividir las grabaciones en ventanas de tamaño fijo.
- Verificar, marcar y ajustar las muestras.
- Organizar el conjunto en partes para entrenamiento, validación y prueba.
- Crear el cargador con métodos de aumento de datos.
- Establecer y entrenar una red neuronal convolucional 1D residual.
- Usar el *early stopping* y guardar el mejor modelo.

- Analizar el desempeño con métricas de clasificación.
- Exportar el modelo listo para usar.
- Incorporar el modelo en la aplicación de detección espectral en tiempo real.

## Capítulo II: Bases teóricas

### 2.1 Marco contextual

Hoy en día, uno de los desafíos en telecomunicaciones es aprovechar mejor el espectro radioeléctrico. Para eso resulta útil reconocer qué tipo de modulación llega al receptor, incluso cuando no se tiene información previa de quién transmite. La clasificación automática de modulaciones se ha vuelto más accesible gracias al uso de equipos SDR y a técnicas modernas de aprendizaje automático, que permiten trabajar con señales reales y manejar situaciones de ruido o interferencias sin depender tanto de modelos ideales.

Las modulaciones digitales y analógicas son lo más importante en este análisis. Estas modulaciones son la base de los sistemas de comunicación inalámbricos que se usan hoy en día. Estos sistemas incluyen los enlaces satelitales, las redes móviles, la transferencia IoT y las infraestructuras de radio definidas por *software*. El proceso de estas señales consiste en cambiar la amplitud, la fase y la frecuencia para convertir la información. De esta manera, se puede ver cómo se comportan las señales cuando hay ruido, interferencia o problemas en el canal. Autoclasificarlas puede resultar beneficioso para gestionar eficazmente el espectro. Asimismo, otorga capacidades más avanzadas a los receptores y a los dispositivos cibernéticos. Tales sistemas pueden localizar señales sin conocer de antemano lo que envía el transmisor.

En Ecuador, los trabajos sobre el tema son escasos y se apoyan mayoritariamente en simulaciones; la ESPE ha evaluado redes neuronales para identificar modulaciones [9]. El paso siguiente son las capturas reales con SDR, única vía para medir el comportamiento de los clasificadores en las condiciones de propagación y ocupación espectral del país.

Esta propuesta busca impulsar investigación aplicada en la universidad. También quiere generar conocimiento útil para la gestión del espectro y la seguridad en las comunicaciones. Además, pretende ofrecer formación práctica a nuevos ingenieros en telecomunicaciones.

## **2.2 Marco conceptual**

Clasificar modulaciones con redes neuronales y SDR significa encontrar la modulación de una señal sin saber de dónde viene. El SDR captura y procesa señales digitalmente, entregando datos sobre su amplitud, fase y frecuencia. Después, la red neuronal examina esa información y se entrena para identificar los patrones de cada tipo de modulación. Estos sistemas distinguen señales con gran precisión, aun cuando el ruido o las interrupciones las alteran. En contraste con los métodos tradicionales, este enfoque ofrece más flexibilidad y no requiere habilidades técnicas muy avanzadas, lo que lo convierte en una herramienta útil para las comunicaciones modernas que buscan mejorar la disponibilidad del espectro radioeléctrico y adaptarse a diversos entornos.

## **2.3 Marco teórico**

### **2.3.1 Modulación digital**

Es el proceso mediante el cual se modifica una señal portadora (en amplitud, frecuencia o fase) en función de una secuencia de bits digitales, permitiendo la transmisión de información sobre un canal físico. Ejemplos comunes incluyen BPSK, QPSK y FSK. Por ejemplo, una revisión de los esquemas usados en comunicaciones inalámbricas menciona ASK, PSK, FSK y QAM como formas de modulación digital [16].

### **2.3.2 Relación Señal-Ruido**

Es un parámetro que mide la proporción entre la potencia de la señal útil y la potencia del ruido presente. Es un indicador crítico del desempeño de los sistemas de comunicación y afecta directamente la exactitud de la clasificación automática. Normalmente se expresa en decibelios (dB) [17].

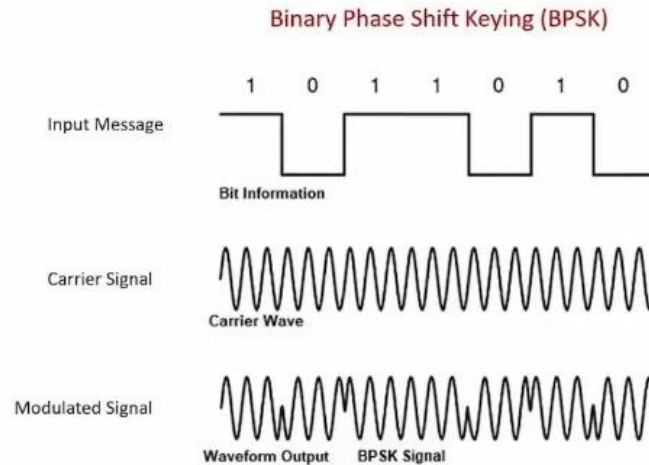
### **2.3.3 Constelación de señal**

Representación gráfica de los símbolos de una señal modulada en el plano complejo (I/Q). Facilita la visualización de la separación entre símbolos y la evaluación de la calidad de la modulación. Esta representación es típica en técnicas de modulación digital.

### **2.3.4 Modulación BPSK**

La modulación por desplazamiento de fase binaria (BPSK) transmite información binaria mediante dos estados de fase de la portadora separados por  $180^\circ$ : uno representa el bit “1” y el otro, el bit “0” [18].

- Es el esquema PSK más simple: su constelación se reduce a dos puntos antipodales.
- La separación máxima entre símbolos le otorga la menor tasa de error de bit frente al ruido y una tolerancia amplia al error de fase.
- Transmite 1 bit por símbolo, lo que limita la tasa de datos frente a esquemas de orden superior.

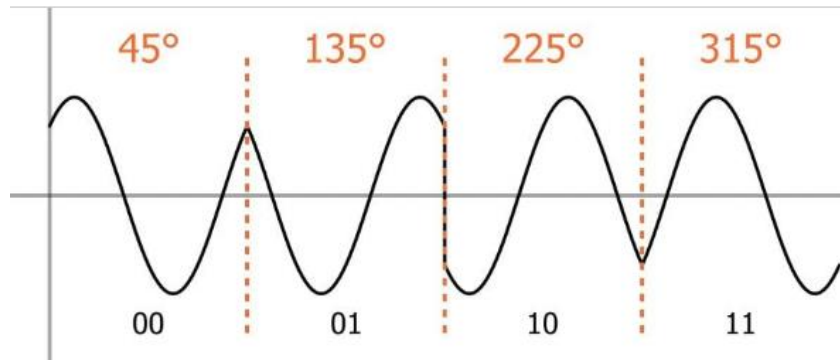


**Figura 3. Forma de onda de señal BPSK [18].**

### 2.3.5 Modulación QPSK

Es una variación de PSK que utiliza cuatro fases distintas para modular la señal portadora, de modo que con cada símbolo se transmiten dos bits [19].

- Generalmente, las cuatro fases se separan unos 90 grados entre ellas, por ejemplo, 45 grados, 135 grados, 225 grados y 315 grados. Esto ayuda a maximizar la distancia entre los estados y hace que sea más fácil distinguirlos.
- En comparación con la modulación BPSK, la QPSK puede enviar el doble de información en el mismo ancho de banda o mantener la misma velocidad de transmisión con la mitad del ancho de banda.
- La QPSK se utiliza mucho en comunicaciones por satélite, redes inalámbricas y enlaces de fibra óptica, porque encuentra un buen equilibrio entre la eficiencia en el uso del espectro y la resistencia a los errores.

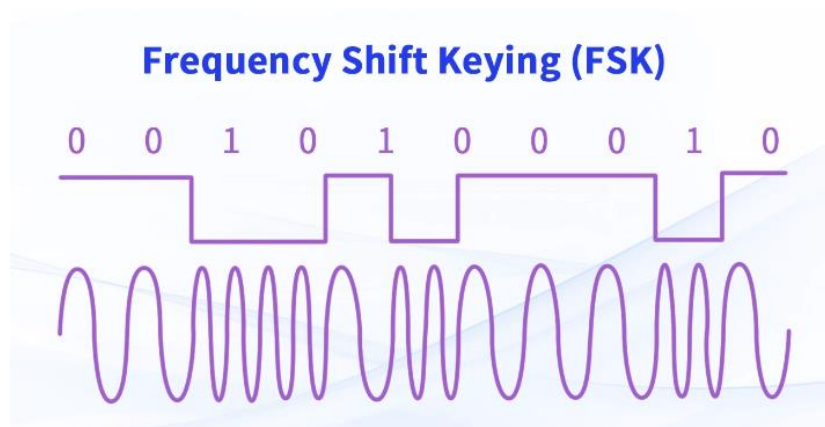


**Figura 4. Forma de onda de señal QPSK [19].**

### 2.3.6 Modulación FSK

Es una técnica de modulación digital donde la señal portadora cambia de frecuencia para representar los diferentes símbolos (por ejemplo, un bit “0” se representa con una frecuencia  $f_0$ , y un bit “1” con otra frecuencia  $f_1$ ) [20].

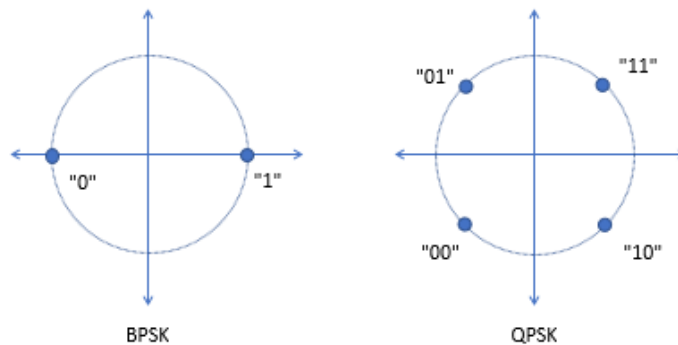
- En su forma más básica (BFSK, Binary FSK) hay dos frecuencias discretas, una para cada valor binario.
- Es relativamente inmune a variaciones de amplitud, puesto que la información está en la frecuencia.
- No es tan eficiente espectralmente como los esquemas de fase o amplitud más sofisticados, y tiende a usarse donde la robustez es clave y la tasa de datos es menor.



**Figura 5. Forma de onda de señal FSK [20].**

### **2.3.7 Justificación académica del uso de BPSK, QPSK y FSK**

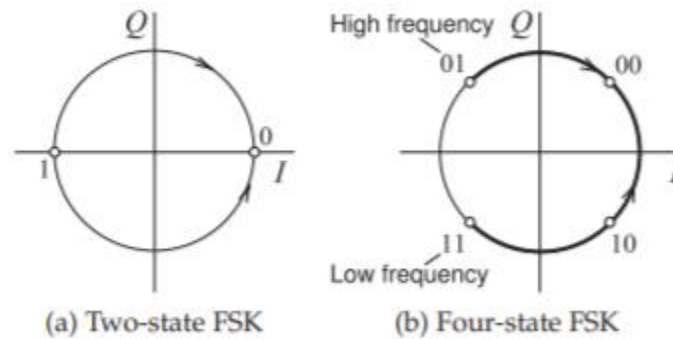
La selección de BPSK, QPSK y FSK como modulaciones de estudio se fundamenta en su relevancia dentro de los sistemas de comunicación digital y en su capacidad para representar distintas formas de codificación de la información. BPSK y QPSK, al ser modulaciones basadas en fase, permiten analizar cómo los clasificadores responden a variaciones en la cantidad de estados de fase, dispersión en constelación y sensibilidad al ruido. Su presencia constante en enlaces inalámbricos, comunicaciones satelitales y redes IoT ha sido ampliamente documentada en investigaciones recientes que emplean estas modulaciones como base para entrenar algoritmos de clasificación automática [21]. Esta característica convierte a ambas modulaciones en referentes imprescindibles para evaluar la robustez y la exactitud de modelos basados en aprendizaje automático y arquitecturas neuronales.



**Figura 6. Constelación de modulación BPSK y QPSK [21].**

Por otro lado, FSK aporta diversidad espectral al codificar la información mediante variaciones discretas de frecuencia, lo cual permite estudiar la capacidad del clasificador para identificar patrones frecuenciales y responder a canales con ruido o interferencias. La literatura señala que las modulaciones FSK continúan siendo fundamentales en sistemas de baja potencia, telemetría y comunicaciones de corto alcance, por lo que su inclusión garantiza un escenario

experimental representativo y comparable con estudios previos [22]. Asimismo, estos tres esquemas forman parte de los conjuntos de datos estándar más utilizados en AMC, como RadioML 2016.10a, que han sido empleados para validar numerosos métodos contemporáneos [14].



**Figura 7. Constelación de modulación FSK [22].**

### 2.3.8 Concepto de AMC

Técnica destinada a identificar automáticamente el tipo de modulación presente en una señal recibida, sin requerir información previa del transmisor. Representa una etapa fundamental en sistemas adaptativos, vigilancia del espectro y radio cognitiva. Esto se aborda en revisiones recientes sobre AMC que discuten métodos tradicionales y basados en aprendizaje profundo.

### 2.3.9 Principios de la clasificación automática de modulación

El proceso de AMC puede dividirse en tres etapas:

1. Adquisición y preprocesamiento de la señal, donde se captura la señal de radio a través de *hardware* SDR y se normaliza.
2. Extracción de características (manual o automática).
3. Clasificación, que consiste en asignar la señal detectada a una de las clases de modulación conocidas.

Formalmente, la señal recibida  $r(t)$  puede expresarse como:

$$r(t) = s(t; M_i, \theta_i) + n(t) \quad (1)$$

donde  $s(t; M_i, \theta_i)$  es la señal modulada bajo el esquema  $M_i \in M = \{BPSK, QPSK, FSK, \dots\}$ ;  $\theta_i$  es el vector de parámetros desconocidos del transmisor (amplitud, fase, desviación de frecuencia y temporización de símbolo); y  $n(t)$  es el ruido blanco gaussiano aditivo (AWGN) introducido por el canal.

A partir de esta formulación, el objetivo del clasificador es estimar el esquema de modulación mediante una función de decisión  $\hat{M} = f(r(t))$ , de modo que se minimice la probabilidad de error de identificación  $P_e = P(\hat{M} \neq M)$  [1].

### 2.3.10 Clasificación automática de modulaciones digitales

La clasificación automática de modulaciones identifica el esquema de modulación de una señal recibida sin información previa del transmisor. Se sitúa entre la detección y la demodulación. De ella dependen la radio cognitiva, la guerra electrónica, el monitoreo del espectro y la optimización de receptores flexibles basados en SDR.

### 2.3.11 Clasificación de señales

Por otro lado, la clasificación de señales es un proceso orientado a identificar el tipo o categoría general de una transmisión de radiofrecuencia, considerando no solo la modulación empleada sino también aspectos relacionados con su estructura, estándar de comunicación, ancho de banda, tasa de símbolos, modo de operación y características específicas del sistema emisor. Este enfoque es de utilidad en aplicaciones como monitoreo del espectro, supervisión de comunicaciones, inteligencia de señales, identificación de interferencias y sistemas de radio cognitiva [23].

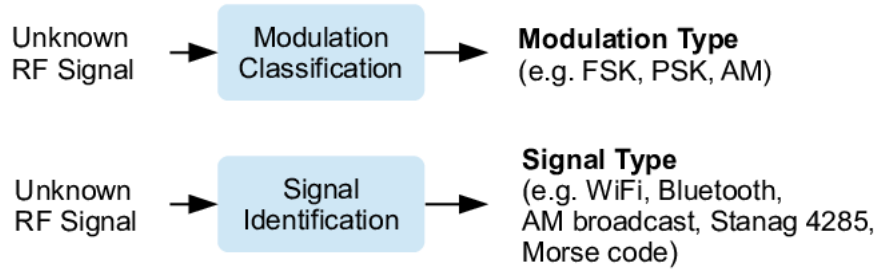
### 2.3.12 Diferencias entre clasificación de señales y clasificación de modulaciones

La clasificación de modulaciones (AMC) se enfoca en identificar exclusivamente el esquema de modulación presente en una señal (por ejemplo, BPSK, QPSK o FSK) analizando patrones relacionados con la fase, la frecuencia o la amplitud, mientras que la clasificación de señales aborda un problema más amplio que incluye el reconocimiento del estándar de transmisión, la estructura de trama, la tasa de símbolos y otras características propias del sistema emisor. Aunque ambas técnicas pueden emplear métodos similares de aprendizaje profundo, difieren en el nivel de abstracción y en el alcance del reconocimiento, siendo la clasificación de modulaciones un subproblema dentro del marco general de clasificación de señales.

**Tabla 1**

Diferencias entre clasificación de señales y clasificación de modulaciones

| Aspecto     | Clasificación de señales              | Clasificación de modulaciones                  |
|-------------|---------------------------------------|------------------------------------------------|
| Objetivo    | Identificar el tipo completo de señal | Identificar solo la modulación                 |
| Alcance     | Amplio (estándares y protocolos)      | Reducido (señales I/Q y constelación)          |
| Complejidad | Alta                                  | Media                                          |
| Relación    | Monitoreo e inteligencia de señales   | Demodulación y SDR                             |
| Aplicación  | Incluye a AMC como subtarea           | Subtarea dentro de la clasificación de señales |



**Figura 8. Clasificación de señales y clasificación de modulaciones [23].**

### 2.3.13 Método basado en verosimilitud

El método basado en verosimilitud se apoya en los principios de detección óptima en el sentido de Bayes. En este enfoque, para cada clase de modulación  $M_i$  se calcula la función de verosimilitud de los datos recibidos  $r$ :

$$A_i(r) = p(r|M_i) \quad (2)$$

y se elige el tipo de modulación que maximiza dicha función:

$$\hat{M} = \arg \max_{M_i \in M} p(r|M_i) \quad (3)$$

[24]

En la práctica, esta probabilidad depende de parámetros desconocidos del canal y la señal (fase, amplitud, SNR, frecuencia portadora), lo cual conduce a variantes:

- ALRT (*Average Likelihood Ratio Test*)
- GLRT (*Generalized Likelihood Ratio Test*)
- HLRT (*Hybrid Likelihood Ratio Test*)

Por ejemplo, el GLRT estima primero los parámetros  $\theta_i$  mediante la técnica de máxima verosimilitud:

$$\hat{\theta}_i = \arg \max_{\theta_i} p(r|M_i, \theta_i) \quad (4)$$

y luego reemplaza dicha estimación en  $p(r|M_i, \hat{\theta}_i)$ . La modulación se clasifica según el máximo valor obtenido [2].

### **Ventajas**

- Es teóricamente óptimo (mínimo error promedio Bayesiano).
- Altamente preciso cuando los parámetros del canal son conocidos.

### **Desventajas**

- Elevada complejidad computacional.
- Dificultad en entornos con parámetros desconocidos o canal variable.
- No es práctico para radios SDR en tiempo real.

## **2.3.14 Método basado en características**

Los métodos basados en características se fundamentan en extraer descriptores estadísticos, espectrales o temporales de la señal para posteriormente alimentarlos a un clasificador (como SVM, KNN, árbol de decisión, etc.).

### **Características comunes**

- Momentos estadísticos y cumulantes de orden superior (HOCs) permiten distinguir modulaciones que difieren en su estructura de amplitud o fase.
- Características de la envolvente, fase y frecuencia instantánea.
- Características espectrales como la densidad espectral de potencia (PSD) o correlación cíclica.

Por ejemplo, los cumulantes de orden cuatro son útiles para diferenciar entre modulaciones M-PSK y M-QAM [25].

$$C_{40} = E[x^4] - 3(E[x^2])^2 \quad (5)$$

$$C_{42} = E[|x|^4] - |E[x^2]|^2 - 2(E[|x|^2])^2 \quad (6)$$

Estos valores, normalizados por la potencia media, se utilizan como vectores de entrada al clasificador:

$$f = [|C_{40}|, |C_{42}|, \dots] \quad (7)$$

### **Ventajas**

- Baja complejidad.
- Independencia del tipo de canal (si se usa normalización).

### **Desventajas**

- Requieren alto conocimiento experto para definir las características.
- Sensibles al ruido y a variaciones del canal [26].

## **2.3.15 Métodos basados en *Machine Learning***

Los métodos basados en aprendizaje automático constituyen una alternativa eficaz para la clasificación de modulaciones digitales cuando se dispone de un conjunto adecuado de características extraídas de la señal. En este enfoque, algoritmos como las Máquinas de Soporte Vectorial (SVM), Algoritmo de K vecinos más cercanos (k-NN), Bosque Aleatorio, teorema de Bayes y las redes neuronales multicapa (MLP) operan sobre descriptores obtenidos del dominio temporal, frecuencial o estadístico (por ejemplo, cumulantes, momentos, energía espectral o variaciones de fase) con el fin de distinguir entre esquemas como BPSK, QPSK y FSK [27]. Estos métodos han demostrado un desempeño estable en escenarios con distintos niveles de ruido y se utilizan con frecuencia en sistemas basados en SDR, donde la variabilidad de las condiciones del canal exige técnicas de clasificación que sean robustas y computacionalmente viables.

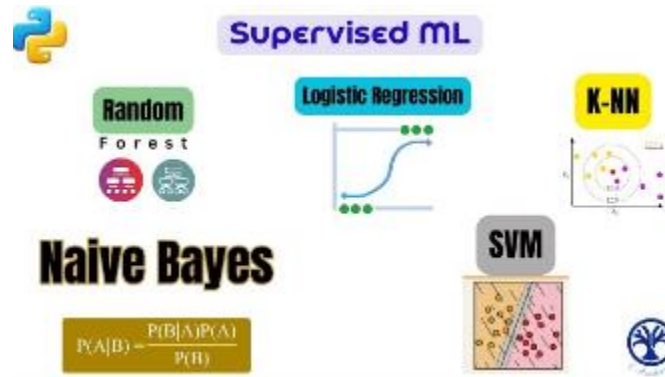


Figura 9. Algoritmos esenciales en *Machine Learning* [28].

### 2.3.16 Métodos basados en *Deep Learning*

Con la llegada de las redes neuronales profundas (DNN, CNN, LSTM, etc.), la clasificación de modulaciones ha dado un salto hacia el aprendizaje automático de características directamente desde los datos recibidos, sin necesidad de diseñarlas manualmente.

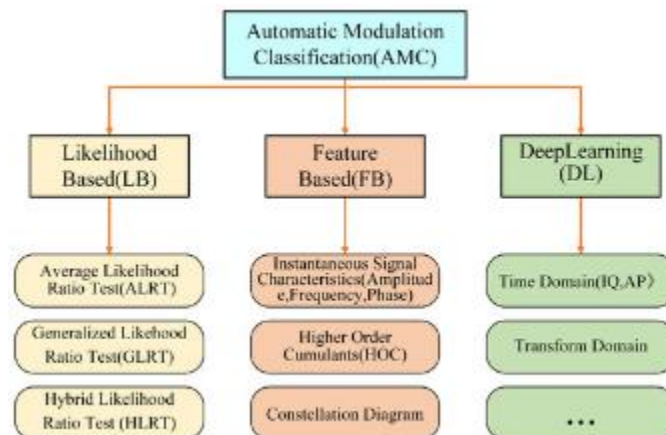


Figura 10. Métodos de clasificación de modulaciones [29].

### 2.3.17 Clasificación basada en *Deep Learning*

Es un subcampo del aprendizaje automático que utiliza redes neuronales con múltiples capas para extraer representaciones jerárquicas de los datos. En AMC, el aprendizaje profundo permite mayor exactitud y robustez frente al ruido, desvanecimiento o interferencias. Una revisión

de 2021 analiza diversos algoritmos de aprendizaje profundo para el reconocimiento de modulación [30].

### 2.3.18 Redes neuronales convolucionales (CNN)

Las redes neuronales convolucionales (CNN) son usadas en procesamiento de imágenes y señales. Reconocen patrones de amplitud, fase y frecuencia en secuencias I/Q, y los obtienen directamente de las muestras sin necesidad de realizar extracciones manuales de características previas.

La efectividad se deriva del proceso de convolución con pesos compartidos: cada capa extrae estructuras locales y las capas siguientes las combinan en representaciones jerárquicas invariantes al desplazamiento. En [3] se describe una arquitectura de CNN robusta para AMC.

El modelo básico aplica una convolución entre el filtro  $h$  y la señal  $x$ :

$$y_i = \sum_{k=1}^K h_k x_{i-k} \quad (8)$$

La salida resultante se activa mediante funciones como ReLU:

$$f(x) = \max(0, x) \quad (9)$$

Y se reduce dimensionalmente mediante *pooling*, lo cual mejora la robustez frente al ruido e interferencias.

En el contexto de AMC, las CNN pueden procesar tanto:

- Representaciones de la señal en tiempo ( $r_I r_Q$ ).
- Imágenes de espectrogramas o constelaciones complejas.

**Ejemplo relevante:**

Liu et al. (2025) propusieron un enfoque CNN de doble flujo para sistemas OTFS (Orthogonal Time Frequency Space), combinando características en los dominios temporal y retardo-Doppler. El modelo mostró una exactitud de clasificación promedio del 97,8 %, incluso con SNR bajos (−5 dB), superando métodos tradicionales [29].

### 2.3.19 Redes LSTM e híbridas CNN-LSTM

Las redes LSTM (Long Short-Term Memory) se utilizan para capturar dependencias temporales de larga duración, siendo adecuadas para secuencias de símbolos modulados. Su principal componente es la célula de memoria, que regula la información mediante compuertas (input, forget y output gates):

$$f_t = \sigma(W_f[x_t, h_{t-1}] + b_f) \quad (10)$$

$$i_t = \sigma(W_i[x_t, h_{t-1}] + b_i) \quad (11)$$

$$\bar{c}_t = \tanh(W_c[x_t, h_{t-1}] + b_c) \quad (12)$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \bar{c}_t \quad (13)$$

Estas redes son eficaces ante modulaciones FSK o QPSK, donde la evolución temporal es más marcada. No obstante, son menos eficientes en la extracción de características espaciales, por lo que a menudo se combinan con CNN [29].

Los modelos híbridos CNN-LSTM integran la potencia de ambas técnicas:

- La CNN detecta patrones espaciales o texturas en el plano I/Q.
- La LSTM modela las correlaciones temporales entre símbolos modulados.

### 2.3.20 Justificación del uso de CNN

**Tabla 2**

Comparación de la CNN con otros modelos de clasificación

| Criterio                                                  | CNN      | LSTM    | Métodos clásicos |
|-----------------------------------------------------------|----------|---------|------------------|
| Extracción automática de características                  | Sí       | Parcial | No               |
| Robustez ante ruido (SNR bajo)                            | Sí       | Sí      | No               |
| Complejidad computacional                                 | Moderada | Alta    | Variable         |
| Capacidad de generalización                               | Alta     | Media   | Baja             |
| Interpretabilidad visual (constelaciones, espectrogramas) | Sí       | No      | No               |

Por ejemplo, el enfoque de Liu et al. (2025) demostró que una CNN dual puede aprender representaciones discriminativas entre BPSK, QPSK y variantes de QAM, aprovechando convoluciones en dominios múltiples (tiempo y retardo-Doppler), algo que las LSTM no pueden explotar plenamente [31].

Además, los métodos de aprendizaje profundo eliminan la dependencia del conocimiento experto requerido por los enfoques basados en características, adaptándose automáticamente a diferentes entornos y canales.

### 2.3.21 Radio definida por *software*

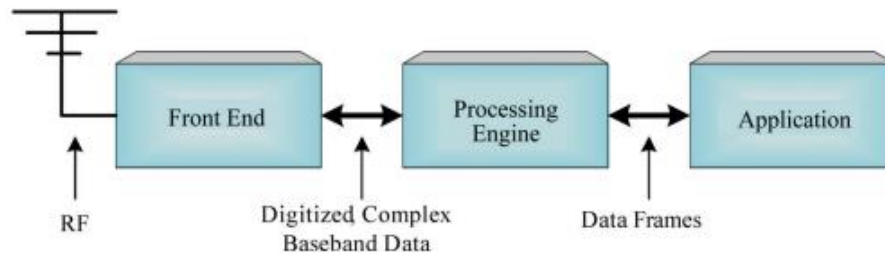
Es una plataforma de *hardware* y *software* que permite implementar funciones tradicionales de radio (modulación, demodulación, filtrado, etc.) mediante programación. Herramientas como receptores de bajo costo facilitan la experimentación y adquisición de señales reales. Por ejemplo, una revisión de 2024 lo define como un transmisor-receptor en el cual “algunos o todos” los parámetros de la capa física están definidos por *software* [32].

### 2.3.22 Implementación con radio definida por *software*

El SDR (*Software Defined Radio*) es una tecnología que implementa mediante *software* las funciones que antes se realizaban en *hardware* (mezcladores, moduladores, filtros, etc.). Esto permite capturar señales reales y probar modelos de clasificación inteligentes de manera flexible.

#### Flujo de Captura

- *Hardware*: el SDR (como USRP, RTL-SDR o HackRF) digitaliza la señal de RF.
- Preprocesamiento: se sincroniza, filtra y normaliza la señal.
- Extracción o input directo: los datos I/Q se exportan a un entorno de entrenamiento (Python, MATLAB, PyTorch).
- Inferencia: el modelo neuronal clasifica la modulación.



**Figura 11. Flujo de recepción de señales [33].**

Matemáticamente, la señal digital muestreada puede expresarse como:

$$r[n] = Ae^{j(2\pi f_c n T_s + \phi)} + w[n] \quad (14)$$

donde  $A$  es la amplitud,  $f_c$  la frecuencia de portadora,  $T_s$  el periodo de muestreo y  $w[n]$  el ruido. Este vector  $r = [r_1, r_2, \dots, r_N]$  constituye la entrada del modelo neuronal.

#### 2.3.23 Aumento de datos

La técnica que se utiliza para aumentar la variedad de los datos de entrenamiento se basa en aplicar transformaciones como ruido, rotación o inversión. Esto ayuda a mejorar la capacidad del modelo para generalizar. Un ejemplo de esto se puede ver en AMC y DL, donde se han utilizado técnicas de aumento de datos para mejorar la clasificación cuando hay mucho ruido y una relación señal-ruido baja.

#### **2.3.24 Radio cognitiva**

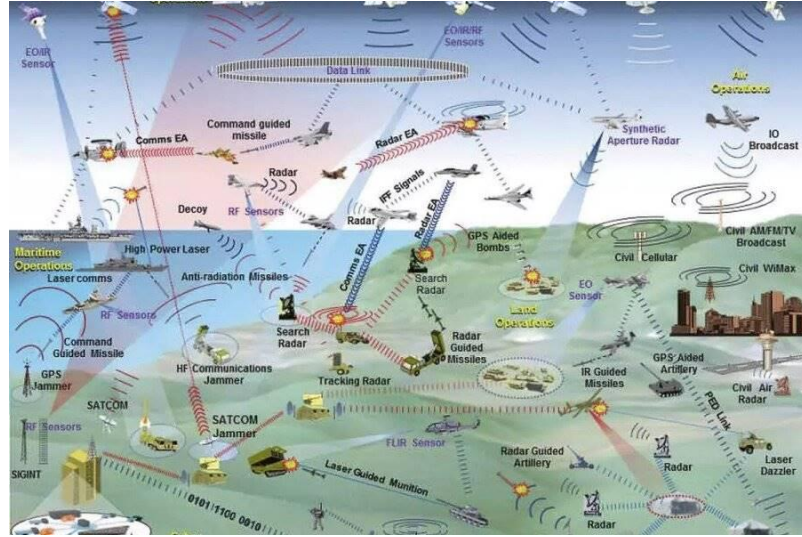
Es un paradigma tecnológico en el que los sistemas de comunicación ajustan dinámicamente su operación (frecuencia, potencia, tipo de modulación) según las condiciones del entorno, basándose en la detección y clasificación del espectro disponible. El concepto de AMC y SDR se integra frecuentemente en revisiones de radio cognitiva/spectrum sensing [3].

#### **2.3.25 Aplicaciones reales de la clasificación automática de modulación**

La clasificación automática de modulación tiene aplicaciones directas en campos donde es necesario identificar el tipo de señal presente en un entorno radioeléctrico sin información previa del transmisor. Entre las aplicaciones civiles más relevantes se encuentran la gestión del espectro, el monitoreo de interferencias y el sensado espectral en radio cognitiva. En estos escenarios, AMC facilita la detección de transmisiones irregulares, optimiza el uso del espectro y permite a los dispositivos seleccionar dinámicamente formas de onda adecuadas, tal como ha sido documentado en investigaciones recientes sobre sistemas inteligentes de comunicación [34].

En ámbitos de defensa e inteligencia electrónica, AMC se utiliza para identificar señales hostiles o desconocidas, realizar vigilancia del espectro y apoyar operaciones de guerra electrónica. Su capacidad para reconocer modulaciones incluso en condiciones adversas contribuye al análisis de amenazas y a la ejecución de contramedidas apropiadas. Estudios

recientes destacan que AMC cumple un rol crítico en aplicaciones aeroespaciales, sistemas de seguridad y plataformas tácticas donde la detección temprana de señales es esencial para la toma de decisiones [35].



**Figura 12. Ilustración de guerra electrónica [36].**

### 2.3.26 Panorama actual y tendencias

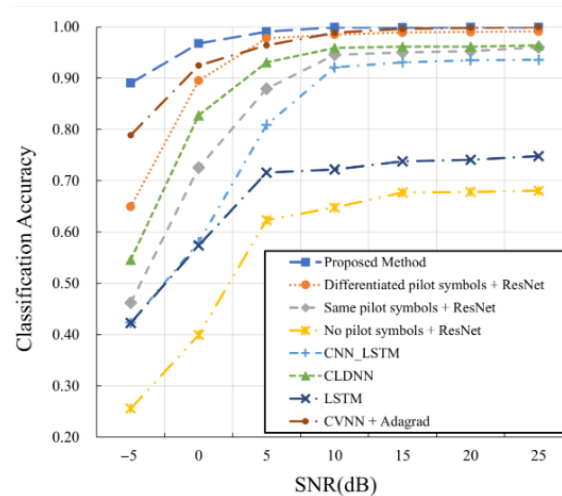
Las tendencias más recientes combinan fusiones multidominio (tiempo/frecuencia, retardo-Doppler, espacio) y estructuras diferenciadas de pilotos para mejorar la separabilidad entre modulaciones, como se muestra en el trabajo de Liu et al. (2025). Este enfoque propone pilotos específicos por tipo de modulación [28], expresados como:

$$x_{ad}[k, l] = \begin{cases} P, & (k, l) \in F_p \\ 0, & (k, l) \in G_p \\ D, & \text{caso contrario} \end{cases} \quad (15)$$

donde  $F_p$  y  $G_p$  son regiones de piloto y de guarda, respectivamente, y  $D$  contiene los símbolos de datos.

Esto permite incrementar la discriminabilidad de características antes del proceso de aprendizaje y ha mejorado la clasificación en más de un 4 % de exactitud promedio (de 93,7 % a 97,8 %).

Asimismo, se está explorando el uso de redes complejas (CVNN) que procesan directamente los valores complejos  $I + iQ$  sin dividirlos, respetando la estructura de fase; no obstante, requieren mayor coste computacional.



**Figura 13. Comparación con enfoques de última generación [29].**

### 2.3.27 Desafíos en la clasificación de modulaciones

A pesar de los avances, varios desafíos permanecen:

- Variabilidad del canal (desvanecimiento Rayleigh, Doppler).
- Baja relación señal-ruido (SNR).
- Modulaciones similares (QPSK vs. 8-PSK).
- Sobrecoste computacional en dispositivos embebidos.

Para mitigarlos, se han explorado técnicas como:

- Normalización de amplitud y fase.

- Aumento de datos (*Data Augmentation*).
- Uso de arquitecturas ligeras (CNN-1D compactas).
- Optimización con algoritmos adaptativos (Adam, RMSprop).

## Capítulo III: Desarrollo e implementación experimental

### 3.1 Introducción al desarrollo experimental

El presente capítulo describe el diseño e implementación del sistema de AMC-OTA (*Automatic Modulation Classification Over-The-Air* — Clasificación Automática de Modulaciones en Canal Real), articulado en tres fases: recolección y validación del *dataset*, entrenamiento del modelo de red neuronal y despliegue para inferencia en tiempo real. En cada fase se describen las decisiones adoptadas durante el diseño, los argumentos que las respaldan y las limitaciones de *hardware* que influyeron en la elección arquitectónica.

El diseño utiliza solo señales reales que se capturan en el aire. No se usan datos creados artificialmente. Esto hace que las pruebas sean más fáciles de realizar. El modelo tiene que manejar las condiciones reales del enlace. Estas condiciones incluyen desplazamiento de frecuencia entre osciladores, ruido térmico en el *front-end* y propagación.

### 3.2 Configuración del entorno experimental

#### 3.2.1 Hardware utilizado

El entorno experimental utiliza dos dispositivos SDR. Estos dispositivos tienen funciones diferentes. Uno actúa como transmisor. Se eligió un Pluto+ SDR como transmisor. Este dispositivo está conectado por medio de Ethernet. La dirección IP es 192.168.2.1. Se usa la librería PyADI-IIO de Analog Devices para controlarlo. El dispositivo crea las señales en banda base para cada tipo de modulación. Luego envía esas señales a una frecuencia central de 400 MHz. La señal se puede atenuar. La atenuación se puede configurar en  $-10$  dB.

El receptor fue un dispositivo llamado RTL-SDR genérico con un tuner FC0013. Este dispositivo se conectó por USB y se utilizó con una biblioteca llamada pyrtlsdr. Las capturas de señales se hicieron a una velocidad de 2,0 millones de muestras por segundo. La ganancia solicitada fue de 30 decibelios para las modulaciones digitales y de 20 decibelios para la modulación de frecuencia de banda ancha. El controlador del tuner ajustó el valor a la ganancia más cercana que soportaba.

Los dispositivos se equiparon con antenas telescópicas que pueden captar señales de cualquier dirección. Estas antenas estaban separadas por una distancia de aproximadamente 4 a 5 metros en un laboratorio. Esta configuración permitió que las señales se vieran afectadas por efectos reales como la atenuación, las reflexiones en varios caminos y el ruido térmico del *hardware*.

**Tabla 3**

Especificaciones del *hardware* utilizado

| Componente      | Dispositivo      | Parámetros Clave                                     |
|-----------------|------------------|------------------------------------------------------|
| Transmisor (TX) | Pluto+ SDR       | 400 MHz, 2 MSPS, -10 dB de atenuación, búfer cíclico |
| Receptor (RX)   | RTL-SDR (FC0013) | 400 MHz / 88-108 MHz, 2 MSPS, ganancia 20-30 dB      |
| Antenas         | Telescópicas     | Omnidireccionales, ~4-5 m separación en interiores   |
| Entrenamiento   | Google Colab     | GPU NVIDIA L4 (24 GB VRAM)                           |

### 3.2.2 Decisiones de diseño RF

**Tasa de muestreo (2,0 MS/s):** Funciona de manera estable en el tuner FC0013, sin problemas de desbordamiento del búfer USB, según las pruebas realizadas en GNU Radio. Ofrece

un sobremuestreo de  $10\times$  para WBFM. El ancho de banda de WBFM es de aproximadamente 200 kHz. Ofrece 20 muestras por símbolo a 100 ksym/s en las modulaciones digitales.

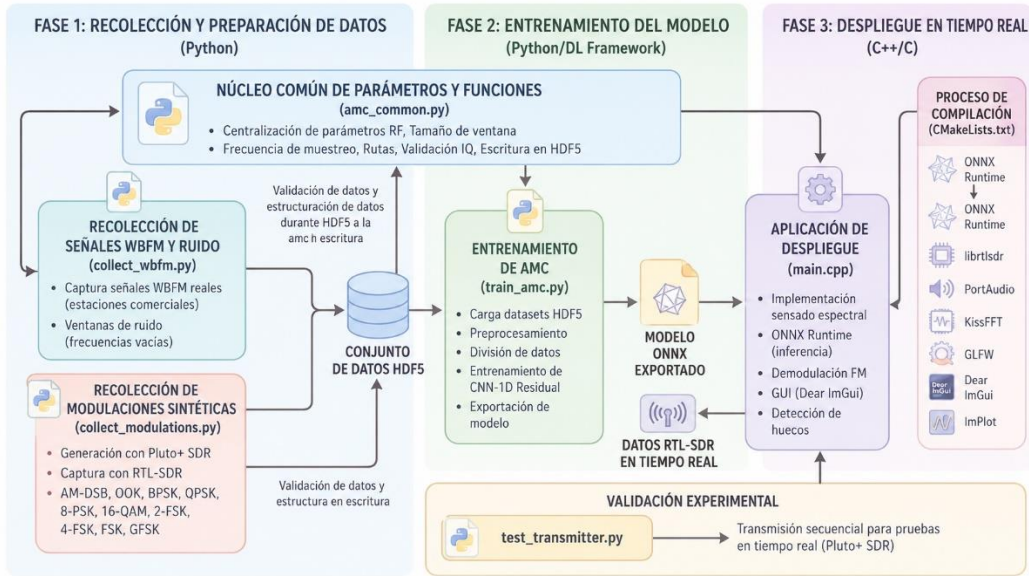
**Tamaño de ventana (1 024 muestras):** Equilibra la capacidad de distinguir las señales y la eficiencia de procesamiento. Una ventana con 128 o 256 muestras cubre poca transición entre símbolos. Para una ventana con 2 048 o más muestras, el coste aumenta, pero no se ve un beneficio visible. La ventana elegida contiene alrededor de 51 símbolos según la configuración indicada. El tiempo de inferencia no se ha registrado; por lo tanto, se pierde la oportunidad de vincular a esta opción un tiempo de inferencia experimentado.

**Ganancia fija (manual):** Se descartó el AGC (Automatic Gain Control) porque la ganancia variable en el tiempo cambia la forma de la señal dentro de una misma ventana y empeora las características que dependen de la amplitud. La ganancia fija mantiene la estadística dentro de cada captura.

### 3.2.3 Estructura del *software*

| Archivo                | Fase del sistema             | Función dentro del diseño                                                                                                                                 |
|------------------------|------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| amc_common.py          | Configuración general        | Centraliza parámetros RF, tamaño de ventana, frecuencia de muestreo, rutas, validación I/Q y escritura en HDF5.                                           |
| collect_wbfm.py        | Fase 1: recolección de datos | Captura señales WBFM reales desde estaciones FM comerciales y también recolecta ventanas de ruido en frecuencias vacías.                                  |
| collect_modulations.py | Fase 1: recolección de datos | El Pluto+ SDR genera las modulaciones AM-DSB, OOK, BPSK, QPSK, 8-PSK, 16-QAM, 2-FSK, 4-FSK y GFSK; el RTL-SDR las captura.                                |
| train_amc.py           | Fase 2: entrenamiento        | El script carga los HDF5, divide el conjunto, preprocesa, entrena la CNN-1D residual y exporta el modelo.                                                 |
| main.cpp               | Fase 3: despliegue           | La aplicación de sensado espectral en tiempo real integra RTL-SDR, ONNX Runtime, demodulación FM, detección de huecos y GUI.                              |
| CMakeLists.txt         | Fase 3: compilación          | Define el proceso de compilación de la aplicación C++ y enlaza dependencias como ONNX Runtime, librtlsdr, PortAudio, KISS FFT, GLFW, Dear ImGui e ImPlot. |

|                     |                         |                                                                                                           |
|---------------------|-------------------------|-----------------------------------------------------------------------------------------------------------|
| test_transmitter.py | Validación experimental | Transmite secuencialmente las modulaciones con el Pluto+ SDR para probar la clasificación en tiempo real. |
|---------------------|-------------------------|-----------------------------------------------------------------------------------------------------------|



**Figura 14. Flujo de datos y control entre los módulos de *software* de las tres fases del sistema.**

Diagrama de bloques que conecta los scripts de cada fase con sus artefactos (HDF5, *checkpoint*, ONNX).

### 3.3 Fase 1: recolección y validación del *dataset*

#### 3.3.1 Proceso de captura de modulaciones digitales

La recolección de las nueve formas de modulación digital y analógica (AM-DSB, OOK, BPSK, QPSK, 8-PSK, 16-QAM, 2-FSK, 4-FSK y GFSK) se lleva a cabo utilizando el script `collect_modulations.py`. El Pluto+ SDR produce cada forma de onda de banda base empleando la clase `ModulationGenerator`, la cual soporta los algoritmos correspondientes: filtros RRC (Root Raised Cosine) para las señales de modulación PSK y QAM, modulación de frecuencia para las variantes FSK y filtrado gaussiano para GFSK.

La captura sincroniza TX/RX. Se evalúa primero la potencia recibida antes de recoger los segmentos válidos. Se añaden periodos de reposo de enfriamiento cada tres transmisiones moduladas. Estos descansos ayudan a mantener la temperatura estable durante sesiones prolongadas. La configuración de diseño aspiraba a recopilar 5 000 segmentos válidos por clase. Esto proporciona 55 000 registros. Los registros se distribuyen en 11 clases. Hay un mecanismo de llenado que se encarga de completar los espacios que quedan sin registro. Cuando concluye la fase principal de recogida, evalúa las clases que aún no han alcanzado el número mínimo necesario y realiza nuevas vueltas hasta obtener el valor requerido. Los datos se almacenan en dos archivos en formato HDF5, que es un formato de almacenamiento de datos jerárquico. Estos archivos se llaman `wbfm_captures.h5` y `digital_captures.h5`. Se usa compresión GZIP para reducir el tamaño de los archivos. Se eligió HDF5 en lugar de NPY porque permite incluir metadatos como tasa de muestreo, ganancia, frecuencia y marca de tiempo. Además, la compresión ayuda a reducir el espacio de almacenamiento en casi un 40 %.

### **3.3.2 Captura de WBFM y ruido**

Se tomaron muestras del espectro de FM, que va de 88 MHz a 108 MHz, utilizando un RTL-SDR. Se capturaron 27 estaciones de radio. El archivo de script `collect_wbfm.py` incluye una función de ganancia adaptativa por estación. Por defecto tiene una ganancia base de 20 dB y se disminuye automáticamente si se detectan interferencias en las muestras capturadas. El sistema pasa por cada estación varias veces para asegurar un total de 5 000 muestras válidas.

Las muestras de la clase “NOISE” vienen de bandas vacías en VHF y UHF. Allí, el RTL-SDR solo registra el ruido de fondo del receptor y del ambiente electromagnético. El criterio de aceptación es el contrario al aplicado a las señales moduladas. La ventana se admite si no se satura

y su relación señal-ruido espectral queda por debajo del umbral de detección. El modelo aprende así la estadística del canal vacío y no atribuye la ausencia de señal a una modulación conocida.

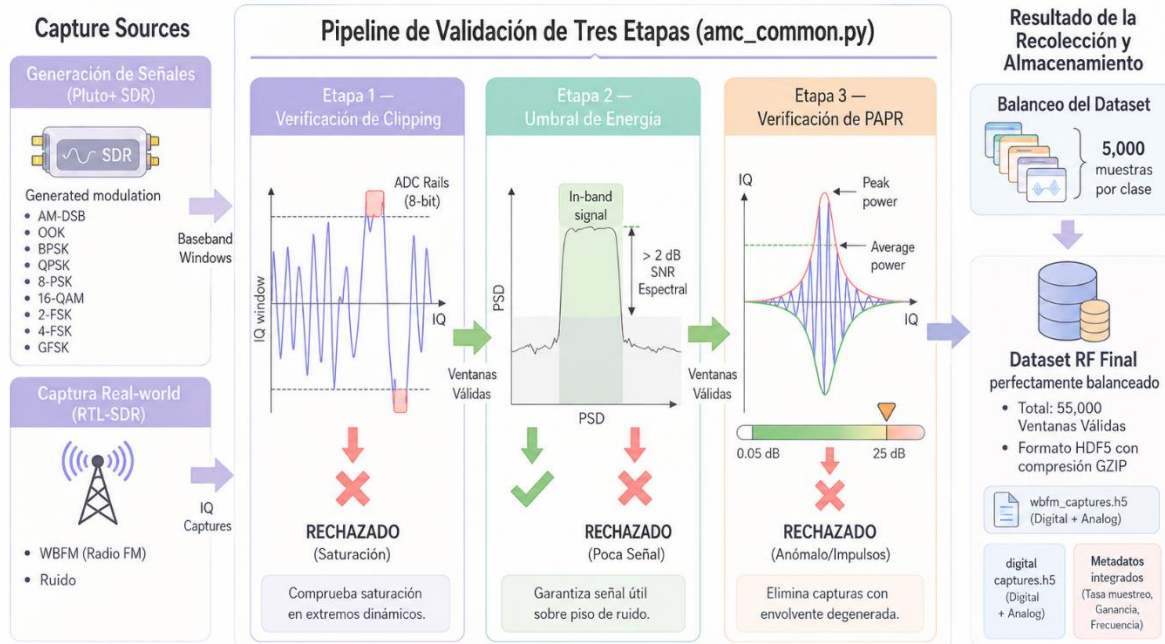
### 3.3.3 Pipeline de validación de ventanas

Para asegurar la calidad de los datos de entrenamiento, se implementó un proceso de validación con tres etapas. Esta configuración fue incorporada en el archivo ``amc_common.py``. Solo aquellas ventanas que superan las tres fases son consideradas para el conjunto de datos final. El sistema realiza esta validación automáticamente durante la recopilación de datos. De esta manera, no hay necesidad de verificar los datos manualmente después.

**Etapas 1 — Verificación de *clipping*:** Los valores I y Q no deben superar los límites del ADC de 8 bits del RTL-SDR. Se eliminan las muestras dentro de la ventana si la señal muestra valores extremos, como estar saturada y por tanto distorsionada.

**Etapas 2 — Umbral de energía:** Se calcula la relación señal-ruido espectral (SNR). Esta relación se obtiene dividiendo la energía dentro de banda entre la energía fuera de banda. Se aceptan las ventanas donde el SNR espectral es mayor que 2 dB. Este valor asegura que la señal esté por encima del ruido.

**Etapas 3 — Verificación de PAPR:** Se mide la relación potencia de pico a potencia promedio (PAPR) de cada ventana. Si los valores están fuera del rango de 0,05 a 25 dB, se consideran capturas anómalas. Estas pueden ser impulsos no deseados o señales con una forma de onda irregular. Estas ventanas se eliminan para no afectar la calidad del *dataset*.



**Figura 15. Proceso desde la captura pasando por la validación hasta llegar al *dataset*.**

### 3.3.4 Efecto del desplazamiento de frecuencia de portadora (CFO) en las señales

Una consecuencia directa del uso de osciladores físicos independientes en el transmisor y el receptor es la deformación de los diagramas de constelación respecto al ideal teórico. En un sistema con oscilador compartido, las constelaciones de BPSK, QPSK o 16-QAM presentan puntos nítidos y separados. En las capturas reales, en cambio, se observan patrones circulares en forma de anillo, fenómeno que tiene consecuencias determinantes en la elección de la arquitectura del clasificador.

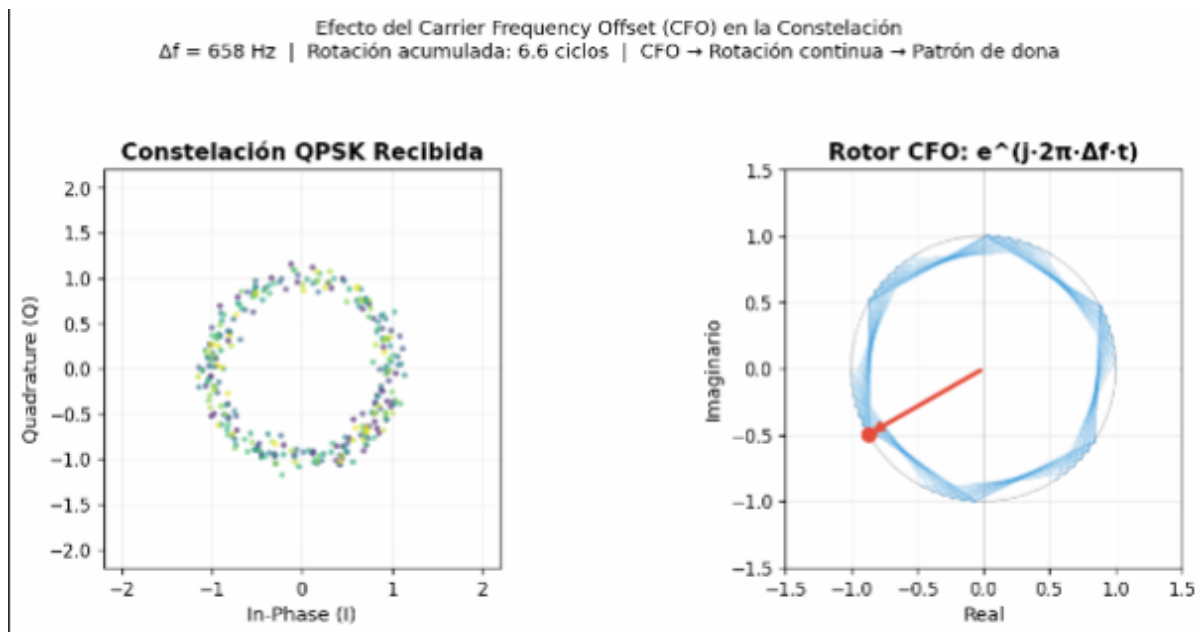
El desplazamiento viene del Offset de Frecuencia de Portadora, también llamado CFO. El transmisor y el receptor usan osciladores separados para generar sus señales de portadora. Cada oscilador tiene una tolerancia determinada. El Pluto+ SDR (TX) y el RTL-SDR con tuner FC0013 (RX) están ajustados a 400 MHz en teoría. Sin embargo, cada oscilador se aleja de su valor nominal en un rango que va de unidades a decenas de partes por millón, es decir, ppm. En 400 MHz, 10 ppm

equivalen a 4 kHz. Esta diferencia se transfiere a la portadora y genera un residuo  $\Delta f$  entre la señal que se envía y el oscilador local del receptor.

Matemáticamente, si la señal transmitida es  $s(t)$  y existe un offset de frecuencia  $\Delta f$ , la señal recibida en banda base se expresa como:

$$r(t) = s(t) \cdot e^{j \cdot 2\pi \cdot \Delta f \cdot t} + n(t) \quad (16)$$

El término exponencial  $e^{j \cdot 2\pi \cdot \Delta f \cdot t}$  actúa como un rotor complejo que gira continuamente la fase de la señal recibida. Cada punto de la constelación original traza un arco circular a medida que la fase acumulada avanza con el tiempo. Si la ventana de observación es suficientemente larga respecto al período de rotación  $1/\Delta f$ , los puntos de la constelación barren circunferencias completas, produciendo los anillos característicos observados en la Figura 16.



**Figura 16. Efecto del *Carrier Frequency Offset*.**

### 3.3.5 Implicaciones del CFO en la elección de arquitectura

La presencia del CFO tuvo un impacto directo y fundamental en la elección de la arquitectura de clasificación. Los enfoques basados en una CNN-2D que usan imágenes de constelación o las representaciones bidimensionales del espacio I/Q también fueron rechazados, ya que el cambio continuo de la fase de rotación desfigura la estructura espacial fija necesaria para trabajar con estos métodos de manera efectiva. Una CNN-2D entrenada con constelaciones claras y nítidas (donde los puntos son exactos y separados) no sería útil si enfrentase un patrón circular como el que se ve en la vida real. La ubicación angular de cada muestra en la constelación es arbitraria y difícil de prever.

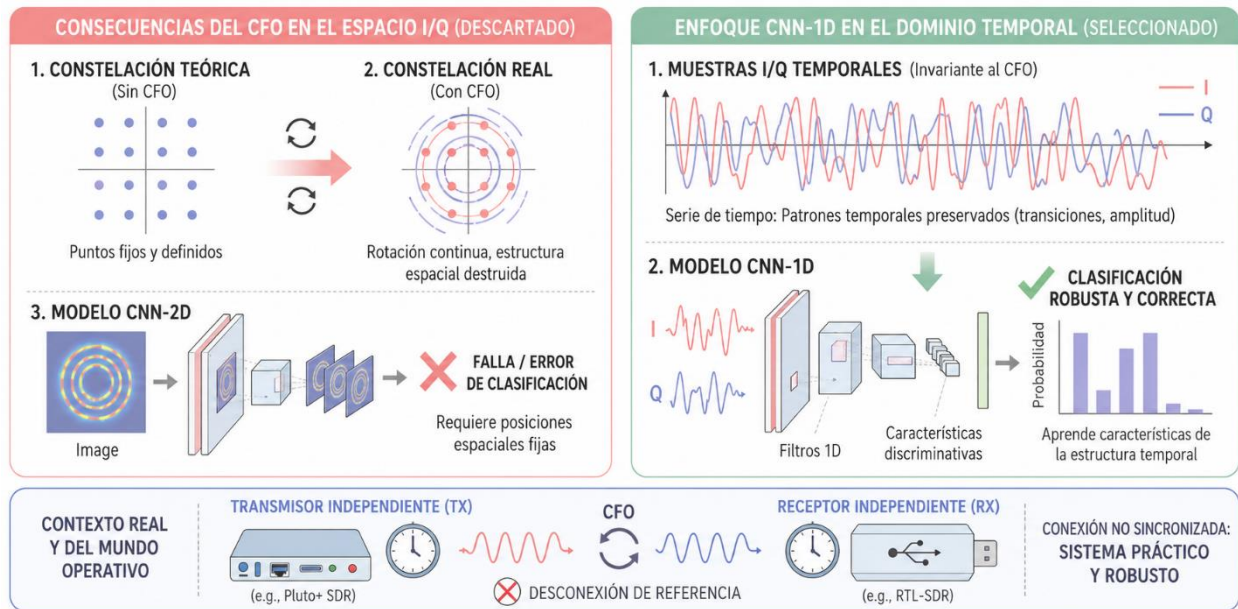
Por otro lado, la arquitectura CNN-1D elegida opera directamente sobre la secuencia temporal de muestras I/Q. Trata las componentes *In-Phase* y *Quadrature* como dos canales de una señal de tiempo. La representación temporal conserva características que la rotación de fase no afecta. Estas características incluyen cambios de amplitud, la forma de la señal y la periodicidad de los símbolos. La red no necesita que los puntos de la constelación estén en sus posiciones teóricas. Trabaja con la estructura temporal de la secuencia I/Q.

El escenario en el que se trabaja apoya esta decisión. En situaciones como monitorear el espectro, detectar señales no autorizadas o vigilar el espectro electromagnético, el receptor y el transmisor no comparten un dispositivo que genere señales de frecuencia fija. Un clasificador que funcione bien debe ser capaz de soportar cambios en la frecuencia. La red neuronal convolucional que trabaja con muestras de señales de radio en el dominio del tiempo, entrenada con datos que incluyen cambios en la frecuencia, puede manejar esto de manera efectiva.

Es importante mencionar que si se hubiera utilizado un enlace con un oscilador compartido, por ejemplo, un Pluto+ SDR para la transmisión y la recepción, o un par de dispositivos HackRF con sincronización externa, las constelaciones aparecerían como puntos claros y fijos. Esto es

similar a lo que se ve en los libros de texto. Sin embargo, esta configuración no es realista porque el transmisor y el receptor suelen ser dispositivos independientes.

Al decidir trabajar con osciladores separados, como un Pluto+ para la transmisión y un RTL-SDR para la recepción, se presenta un desafío. Esto se hace a propósito para simular situaciones del mundo real. El resultado es un conjunto de datos y un modelo más sólidos y representativos de la realidad.



**Figura 17. Solución ante el CFO.**

### 3.4 Fase 2: arquitectura del modelo y entrenamiento

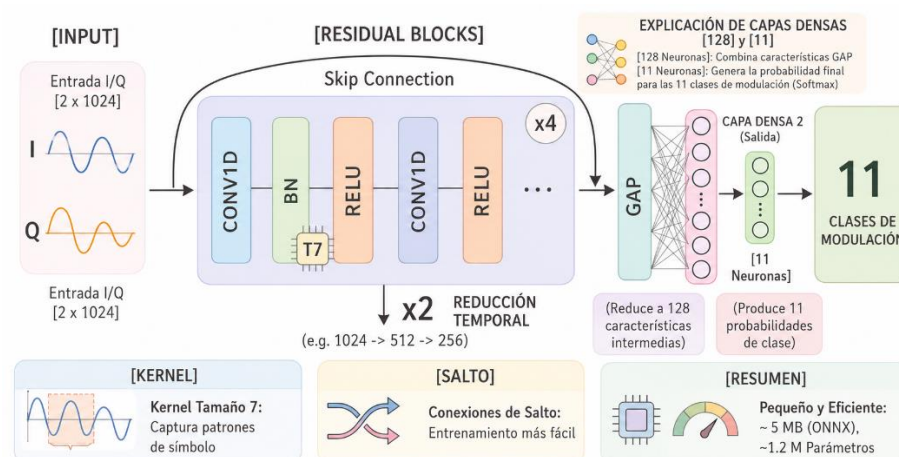
#### 3.4.1 Arquitectura AMCNet (CNN-1D residual)

La arquitectura AMCNet es una red neuronal diseñada para procesar señales que varían con el tiempo. Esta red tiene una capa inicial y luego cuatro bloques que se repiten. Cada bloque tiene dos capas de convolución con *kernels* de tamaño 7, lo que ayuda a entender patrones en la señal. También tiene una forma de normalizar los datos y una función de activación llamada ReLU.

La entrada a la red tiene un tamaño determinado:  $[batch, 2, 1024]$ . Estos dos canales sirven para las partes en fase y cuadratura de la señal. La red consta de una capa convolucional inicial y cuatro bloques residuales. Cada bloque residual tiene dos convoluciones en una dimensión con *kernels* de tamaño 7, normalización por lotes y funciones de activación ReLU, con una conexión que suma la entrada directamente a la salida del bloque.

Los bloques 2 y 4 reducen la longitud temporal de la señal de 1 024 a 512 y después a 256. Después de los bloques residuales, una capa de *Global Average Pooling* procesa la señal en el eje temporal y crea un vector con 256 características. Luego, dos capas densas utilizan este vector para producir los logits de las 11 clases. La función softmax se usa para calcular las probabilidades de estas clases durante la inferencia.

El modelo consta de aproximadamente 1,2 millones de parámetros y puede ser exportado al formato ONNX con un tamaño de alrededor de 5 MB. Los *kernels* con tamaño 7 reconocen patrones en la señal que abarcan varias muestras por símbolo. Si cada símbolo tiene una secuencia de 20 muestras, el *kernel* del primer bloque cubre aproximadamente el tercio del periodo de símbolo. La utilización de conexiones residuales mejora el flujo de gradientes durante el entrenamiento y permite que la red aprenda cómo obtener mejor rendimiento.



**Figura 18. Arquitectura AMCNet: CNN-1D residual para clasificación en tiempo real.**

### **3.4.2 Preprocesamiento de datos**

Antes del entrenamiento, cada ventana I/Q atraviesa dos etapas de preprocesamiento: se centra restando su media compleja, lo que suprime la componente de continua. A continuación, se normaliza a potencia unitaria dividiendo por la raíz cuadrada de la potencia media; la escala de amplitud queda homogénea entre ventanas y se preservan la estructura temporal y las relaciones de fase.

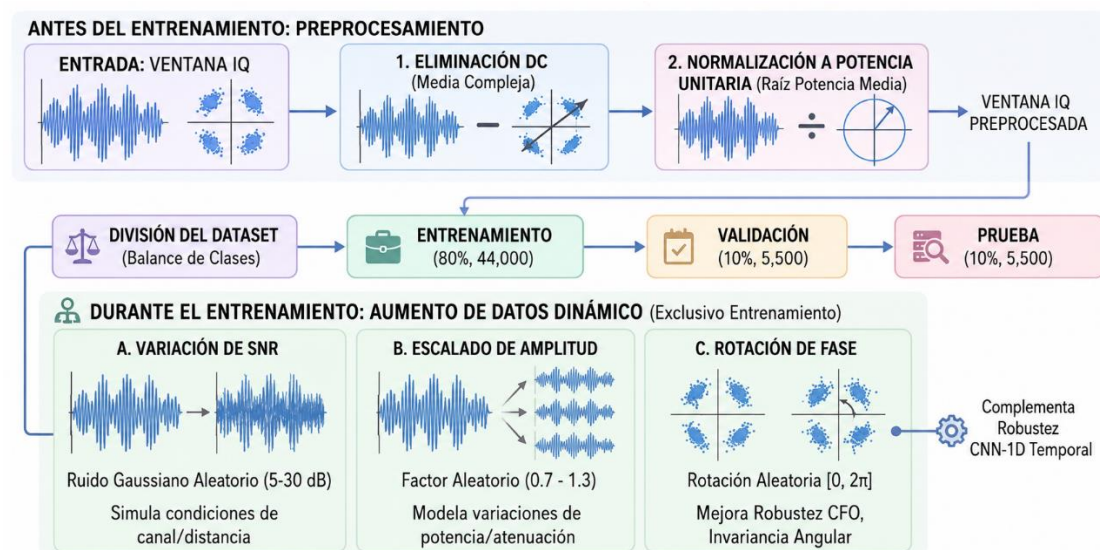
El conjunto se reparte en 80 % de entrenamiento (44 000 ventanas), 10 % de validación (5 500) y 10 % de prueba (5 500), con las clases balanceadas en cada partición.

### **3.4.3 Aumento de datos (*data augmentation*)**

Durante el entrenamiento se aplican tres técnicas de aumento de datos dinámico, exclusivamente sobre el conjunto de entrenamiento, para mejorar la robustez del modelo frente a variaciones del canal:

- Variación de la relación señal-ruido: se agrega ruido blanco gaussiano con una relación señal-ruido entre 5 y 30 decibelios. Esto simula diferentes condiciones del canal y distancias de propagación.
- Aumento de volumen: la señal se multiplica por un número al azar entre 0,7 y 1,3. Esto simula cambios en la potencia que se recibe debido a la pérdida en el canal.
- Rotación de fase: se le da una rotación de fase aleatoria entre 0 y  $2\pi$ . Esto ayuda al modelo a ser más resistente al desfase de frecuencia que se ve en las señales reales.

La rotación de fase al azar es parte de una técnica de aumento de datos. Esta rotación tiene que ver con el efecto de desfase de frecuencia que se menciona. Al entrenar con rotaciones de fase que están distribuidas de manera uniforme entre  $0$  y  $2\pi$ , se mejora la capacidad del modelo para clasificar señales reales. Esto ocurre sin importar la posición angular de la constelación. Esto ayuda a que el modelo sea más resistente. Esta resistencia ya está presente por la forma en que funciona la arquitectura al trabajar en el dominio temporal.



**Figura 19. Flujo de preprocesamiento, división del *dataset* y aumento dinámico de datos I/Q.**

Un diagrama que muestra el ciclo de preparación de las señales antes y durante el entrenamiento. El proceso empieza con el preprocesamiento de las ventanas I/Q. Se elimina la componente DC. Se normaliza a potencia unitaria para tener una amplitud estándar. La figura muestra cómo se reparte el conjunto en entrenamiento, que es el 80 %, en validación, que es el 10 %, y en prueba, que también es el 10 %. Luego se aplican tres técnicas de aumento dinámico. Estas técnicas solo se usan durante el entrenamiento. Las técnicas son variación de SNR, escalado de amplitud y rotación de fase. Estas técnicas ayudan al modelo a ser más robusto frente a las distorsiones del canal.

#### 3.4.4 Proceso de entrenamiento

El entrenamiento se ejecutó en Google Colab utilizando una GPU NVIDIA L4 con 24 GB de VRAM. Se empleó el optimizador AdamW con tasa de aprendizaje inicial de  $1 \times 10^{-3}$  y *weight decay* de  $1 \times 10^{-4}$ . La función de pérdida utilizada fue *Cross-Entropy Loss*, adecuada para problemas de clasificación multiclase.

Se implementó un mecanismo de *early stopping* que detiene el entrenamiento si la pérdida de validación no mejora durante 15 épocas consecutivas, previniendo el sobreajuste. El modelo con la menor pérdida de validación se guarda como *checkpoint* (best\_model.pth) y se utiliza para la evaluación final y la exportación.

#### 3.4.5 Exportación del modelo a formato ONNX

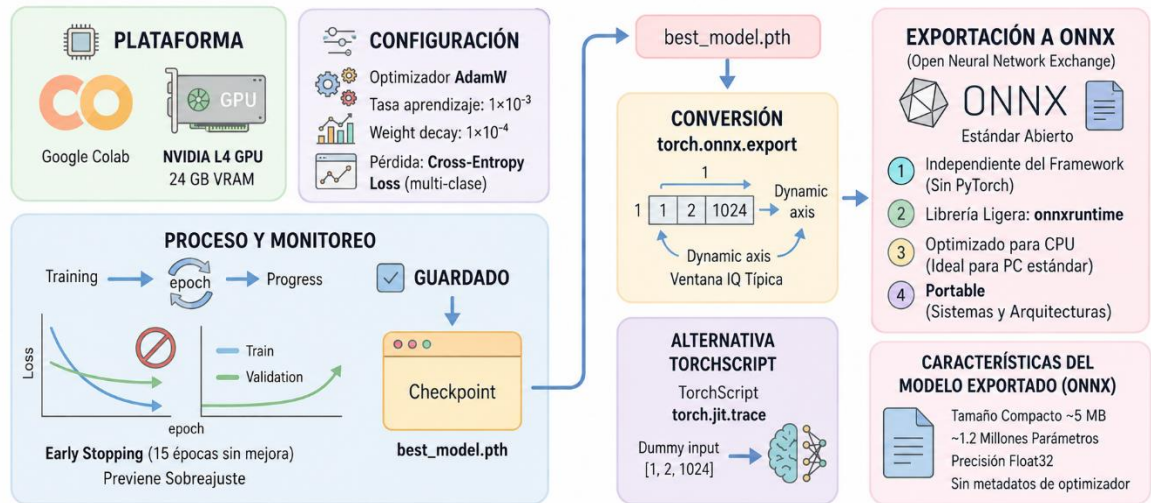
Cuando termina el entrenamiento, se exporta el modelo que funciona mejor en la validación, llamado best\_model.pth, a un formato llamado ONNX. Esto se hace con una herramienta de PyTorch llamada torch.onnx.export. Para hacer esta exportación, se utiliza un tensor de entrada de prueba que tiene dimensiones de 1, 2 y 1 024, que es similar a una ventana I/Q normal. Se especifican ejes dinámicos en la dimensión de *batch* para permitir que el modelo funcione con diferentes tamaños de lote.

Se eligió el formato ONNX por tres razones importantes. En primer lugar, es un estándar abierto que no depende de ningún *framework* en particular, lo que significa que no es necesario instalar PyTorch en la computadora donde se va a utilizar el modelo. Solo se necesita una herramienta llamada onnxruntime, que es mucho más ligera. En segundo lugar, onnxruntime puede optimizar el modelo y ejecutarlo de manera vectorizada en la CPU, lo que es necesario para que funcione en computadoras sin una GPU dedicada. En tercer lugar, el formato ONNX es portable,

lo que significa que se puede utilizar en diferentes arquitecturas y sistemas operativos, lo que facilita la migración a plataformas embebidas.

Además de exportar el modelo en formato ONNX, también se exporta una versión en formato TorchScript, utilizando una herramienta llamada `torch.jit.trace`. Esta versión se utiliza como respaldo en caso de que haya problemas con `onnxruntime` y permite ejecutar el modelo desde Python con PyTorch sin necesidad de tener todo el entorno de entrenamiento.

El modelo exportado ocupa alrededor de 5 MB, distribuidos en dos archivos: el grafo computacional (`amc_model.onnx`, que ocupa alrededor de 51 KB) y los pesos en un archivo de datos externo (`amc_model.onnx.data`, que ocupa alrededor de 4,9 MB). Esto se debe a que el modelo tiene alrededor de 1,2 millones de parámetros en `float32`. Ambos archivos se incluyen juntos, ya que `onnxruntime` busca el archivo de pesos de manera relativa al archivo del modelo. El conjunto es más ligero que el *checkpoint* de entrenamiento (que ocupa aproximadamente 14,8 MB), que además almacena el estado del optimizador para permitir la reanudación del entrenamiento. El tamaño reducido del modelo exportado permite reducir los tiempos de inicio del sistema de inferencia y abre la posibilidad de cuantizarlo posteriormente a `INT8` para implementarlo en *hardware* embebido en los trabajos futuros.



**Figura 20. Entrenamiento y exportación del modelo.**

El trabajo empieza con la configuración básica en Google Colab con una GPU NVIDIA L4 y termina con la exportación final. Se usó *early stopping* para evitar el *overfitting*. El mejor modelo, llamado `best_model.pth`, se cambió a formato ONNX usando un tensor de ejemplo de una ventana I/Q. La dimensión del *batch* se definió como dinámica. El modelo final pesa unos 5 MB y contiene la estructura y los pesos. También funciona en la CPU con `onnxruntime`, sin depender de PyTorch. Además, se creó una versión en TorchScript que se guardó como *backup*.

### 3.5 Fase 3: Aplicación del sensado espectral en tiempo real

El modelo ONNX de la Fase 2 se incluyó en una aplicación de escritorio desarrollada en C++17. Esta aplicación realiza un escaneo espectral en tiempo real. La aplicación recorre el espectro usando el RTL-SDR. Luego clasifica la modulación de cada canal. Después demodula y reproduce el audio de las estaciones FM que encuentra. Todos los resultados se muestran en una interfaz gráfica. En esa interfaz hay gráficos de PSD, constelaciones y estadísticas sobre la ocupación.

La elección de C++ como lenguaje de implementación responde a problemas de rendimiento detectados en una versión previa basada en Python (onnxruntime + matplotlib), cuyas limitaciones se documentan en detalle en la sección 3.6.8. La versión actual en C++ ejecuta el ciclo completo de captura, clasificación, demodulación y renderizado a tasas de cuadros sostenidas y sin cortes audibles en el audio FM.

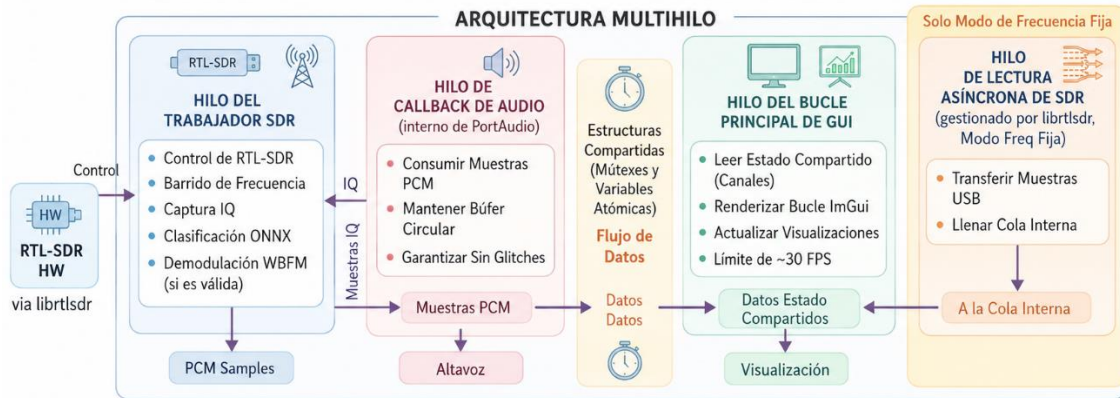
Antes de cada inferencia, la aplicación prepara la ventana I/Q de manera consistente con la forma en que se recopiló el conjunto de datos. La aplicación utiliza nuevamente el detector de energía del validador. Se usa el 12 % central del espectro frente a los bordes del 15 %. El umbral es de 2 dB. Las ventanas que están por debajo del umbral se marcan como “NOISE” sin usar la red neuronal. Cuando la energía principal está desplazada del centro de sintonía, se realiza un desplazamiento digital para llevar la señal a banda base. Esto ocurre antes del detector y antes de la CNN. Esto coincide con el conjunto de entrenamiento. Las señales en el conjunto de entrenamiento están centradas en DC. Este desplazamiento está acotado a  $\pm 150$  kHz (y nunca más de medio paso de barrido) de modo que corrige únicamente offsets residuales de sintonía sin capturar energía perteneciente a canales adyacentes. Por último, la PSD mostrada en la interfaz se estima mediante promediado de periodogramas (método de Welch, hasta 200 segmentos de 256 muestras), lo que reduce la varianza de la estimación espectral frente a un periodograma único.

### **3.5.1 Stack tecnológico y arquitectura multihilo**

La aplicación se crea con un conjunto de bibliotecas que funcionan bien y son eficientes. Estas bibliotecas pueden usarse en Linux y Windows gracias a MSYS2/MinGW. La aplicación controla el *hardware* con la ayuda de librtlsdr, que es una biblioteca en C para receptores RTL2832U. Para hacer las inferencias, el sistema usa la API de ONNX Runtime en C++. Todo este proceso se ejecuta en la CPU. También se hace un análisis espectral con KISS FFT, que es

una versión de la FFT más ligera y fácil de usar. El sistema usa PortAudio para reproducir el audio, ya que PortAudio es una biblioteca que funciona en muchas plataformas y tiene una baja latencia. La interfaz gráfica se ha hecho con Dear ImGui e ImPlot, y utiliza el soporte de OpenGL mediante GLFW. Para organizar todo, se usa CMake como gestor de compilación y con CMake se asignan todas las dependencias necesarias. Estas dependencias se descargan automáticamente utilizando FetchContent, una herramienta para descargar contenido de forma sencilla.

La arquitectura interna se organiza en hilos concurrentes que se comunican mediante estructuras compartidas protegidas por *mutex* y variables atómicas. El primer hilo (*worker* SDR) gestiona el control del RTL-SDR, ejecuta el barrido de frecuencias, captura las muestras I/Q y ejecuta tanto la clasificación con el modelo ONNX como la demodulación FM cuando la señal sintonizada se clasifica y valida como WBFM. PortAudio usa un segundo hilo para el audio. Este hilo extrae las muestras PCM del demodulador FM usando un búfer circular. También hay un tercer hilo que se encarga de dibujar la interfaz con ImGui. El bucle de ImGui corre a un máximo de 30 cuadros por segundo. Este hilo lee el estado de los canales para actualizar las visualizaciones en pantalla. Al separar la grabación, el audio y la interfaz, cada proceso puede trabajar a su propio ritmo sin molestar a los demás. Además, cuando se usa el modo de análisis de frecuencia constante, se activa un cuarto hilo de lectura asíncrono. Este cuarto hilo usa librtlsdr para pasar las muestras del bus USB a unas colas internas. Lo hace a la velocidad que permite el *hardware*. Esta separación entre la llegada de las muestras y su procesamiento asegura que el audio FM no se interrumpa, incluso si hay latencia en la clasificación o en el renderizado.



**Figura 21. Diagrama de arquitectura multihilo y *stack* tecnológico del sistema SDR.**

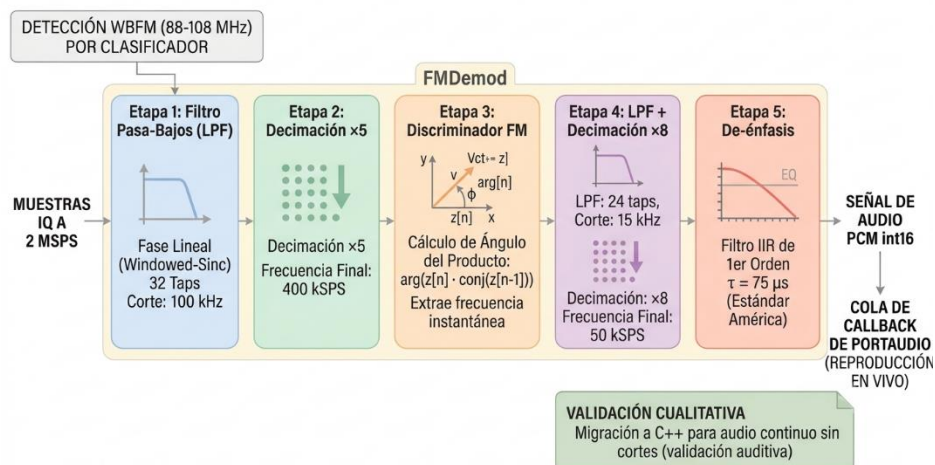
Diagrama estructurado de la arquitectura de *software* organizada en hilos concurrentes independientes, diseñada para garantizar un procesamiento en tiempo real de baja latencia y sin bloqueos en la interfaz.

### 3.5.2 Demodulación FM y reproducción de audio en vivo

Cuando el clasificador detecta WBFM en la frecuencia observada (y, fuera de la banda comercial, la predicción supera además la validación DSP descrita en la sección 3.5.5), se activa la cadena de demodulación implementada en la clase FMDemod. El proceso comprende cinco etapas. Primero, las muestras I/Q a 2 MSPS atraviesan un filtro pasa-bajos de fase lineal (*windowed-sinc*, 32 *taps*) con frecuencia de corte de 120 kHz para eliminar canales adyacentes y reducir el ancho de banda. Segundo, se deciman las señales a un factor de 5, llegando a 400 kSPS. A continuación, se aplica el discriminador FM tradicional calculando el ángulo del producto  $z[n] \cdot \text{conj}(z[n-1])$ , que se encarga de extraer la información angular instantánea proporcional a la frecuencia modulante. Posteriormente, la señal resultante pasa por un filtro pasa-bajos FIR de 24 *taps* y un corte en 15 kHz. Este ancho de banda coincide con el del audio FM comercial. Se deciman las señales de nuevo a 8 para llegar a una tasa final de 50 kS/s. Después se realiza el de-

énfasis con una constante  $\tau$  de 75  $\mu$ s. Este es el estándar en América del Norte y se aplica como un filtro IIR de orden uno. El audio se convierte en PCM tipo 16 bits y es enviado a la cola del *callback* de PortAudio. Los filtros y el discriminador mantuvieron su estado mientras los bloques se procesaban, de manera que el procesamiento por bloques equivale a realizar un filtrado continuo. La escala se estableció en función de la máxima desviación de FM, que oscila entre  $\pm 75$  kHz. No se ajusta por bloque, lo que evita cambios de volumen.

La demodulación en vivo cierra el sistema por el lado práctico: el operador confirma de oído que una estación clasificada como WBFM corresponde a una emisora comercial. Es una verificación funcional de la cadena de recepción, complementaria a las métricas y no sustitutiva. Esta característica fue precisamente uno de los principales motivos para migrar de Python a C++, ya que la versión Python no era capaz de mantener un flujo de audio continuo sin cortes, como se detalla en la sección 3.6.8.



**Figura 22. Diagrama de bloques de la cadena de demodulación WBFM.**

Este diagrama detalla el *pipeline* DSP de cinco etapas de la clase FMDemod para procesar muestras I/Q a 2 MSPS. El flujo comprende un filtro pasa-bajos inicial (120 kHz), una decimación por 5 (400 kSPS), la demodulación mediante el discriminador FM, un segundo filtro de audio (15

kHz) con decimación por 8 (50 kSPS), y un filtro de  $\tau=75 \mu\text{s}$ . El proceso concluye con la conversión a PCM int16 y su envío a PortAudio, permitiendo validar auditivamente y sin cortes la clasificación WBFM gracias a la eficiencia de la migración a C++.

### 3.5.3 Interfaz gráfica con Dear ImGui e ImPlot

La interfaz gráfica se construye con Dear ImGui, una biblioteca de modo inmediato (immediate-mode GUI) ampliamente utilizada en aplicaciones de bajo nivel y herramientas de desarrollo. A diferencia de los *frameworks* tradicionales basados en eventos y árboles de *widgets* persistentes, ImGui reconstruye toda la interfaz en cada cuadro de renderizado, lo que la hace especialmente adecuada para aplicaciones que muestran datos en constante actualización como *dashboards* de instrumentación SDR. ImPlot, una extensión de ImGui especializada en gráficas de alto rendimiento, se utiliza para renderizar la PSD, la constelación I/Q y los histogramas de confianza.

El renderizado se realiza directamente sobre OpenGL a través de los *backends* `imgui_impl_glfw` e `imgui_impl_opengl3`, lo que permite alcanzar tasas de cuadro elevadas con un consumo de recursos reducido. La ventana se gestiona con GLFW, que provee también el contexto OpenGL multiplataforma, eliminando así los cuellos de botella identificados en la versión previa en Python.

### 3.5.4 Clasificación de roles, contexto regulatorio y detección de huecos espectrales

La aplicación guarda el estado de cada canal usando la estructura `ChannelState`. Esta estructura guarda las últimas mediciones de clase, confianza, SNR y tiempo de actividad. Para evitar que cambios pequeños y rápidos afecten el resultado, la clase que se muestra no se basa en una sola ventana de datos. En lugar de eso, se usa un voto mayoritario entre varias ventanas

recientes. Esto ayuda a que el estado en el *dashboard* no cambie de forma inestable debido a variaciones breves en el canal.

Cada frecuencia que se escanea tiene un rol operativo. La asignación se basa en dos criterios: si la frecuencia está en una banda regulada y si el detector detecta actividad. PRIMARIO indica una señal activa en una banda regulada. PRI-LIBRE es cuando la banda regulada no tiene actividad detectada. ACTIVO se usa para señales que están fuera de las bandas primarias. HUECO es un canal sin actividad y candidato para reutilización. GUARDA es para frecuencias cerca del borde de una banda regulada.

El marco regulatorio utiliza una tabla local de atribuciones de ARCOTEL. Esta tabla incluye radiodifusión AM y FM, televisión VHF/UHF/TDT y bandas IMT de telefonía móvil. La capa no cambia la salida de la red. Simplemente, sitúa la frecuencia observada dentro del uso atribuido del espectro.

Por lo tanto, la clasificación de modulación sigue dependiendo del modelo AMC y de las validaciones DSP. El contexto ARCOTEL se utiliza para etiquetar el servicio esperado. De esta manera, se evitan recomendaciones de reutilización en zonas reguladas o de guarda.

Para detectar huecos, se tienen en cuenta dos cosas: la clase estimada y la energía medida en el canal. Si una señal es desconocida, no se considera un hueco automáticamente, aunque tenga una relación señal-ruido muy alta. Esto se debe a que una alta relación señal-ruido indica que el espectro está ocupado, incluso si el sistema no puede identificar la modulación. Esta decisión es importante porque evita recomendar una frecuencia como reutilizable cuando en realidad hay actividad real que no está incluida en las clases que se han entrenado.

### **3.5.5 Validación DSP complementaria para predicciones WBFM**

Se agregó una etapa de validación a la red neuronal para verificar las predicciones de WBFM que se realizan fuera del rango de frecuencia FM comercial. Esta etapa no reemplaza a la red neuronal convolucional ni crea una nueva categoría. Lo que hace es comprobar si la señal tiene sentido desde el punto de vista físico cuando el modelo dice que es WBFM en un rango donde se necesita más evidencia para estar seguro.

La validación se basa en un discriminador FM que estima la variación instantánea de fase entre muestras consecutivas:

$$y[n] = \text{angle}(x[n] \cdot \text{conj}(x[n - 1])) \quad (17)$$

En FM, el discriminador genera una señal que es proporcional al modulante, por lo que su energía se encuentra concentrada en frecuencias bajas. El sistema analiza esta concentración y la compara con otras métricas, como SNR, ancho de banda y PAPR.

La validación solo ocurre fuera del rango comercial de FM, que va desde 88 hasta 108 MHz. Dentro de este rango, se mantiene la predicción del clasificador neuronal para no descartar estaciones reales en situaciones de recepción cambiantes. Fuera de este rango, si una predicción WBFM no cumple los requisitos físicos mínimos, el resultado se marca como “SEÑAL DESCONOCIDA”.

### **3.5.6 Análisis de capacidad: Shannon y Nyquist**

Para complementar la detección de huecos espectrales, la aplicación incorpora un módulo de análisis de capacidad basado en los teoremas de Shannon y Nyquist. Este módulo permite estimar la viabilidad teórica de cada hueco como canal de comunicación secundario, utilizando el ancho de banda asociado al paso de barrido y el SNR espectral medido durante la observación del canal.

La capacidad de Shannon establece la tasa máxima teórica de transmisión sobre un canal con ancho de banda  $B$  y relación señal-ruido SNR, expresada como:

$$C = B \cdot \log_2(1 + SNR_{lin}) \quad (18)$$

donde  $SNR_{lin}$  corresponde al valor de SNR en escala lineal. En la aplicación, este cálculo se realiza a partir del SNR observado durante el sensado espectral. El resultado es entonces una estimación válida para las condiciones de canal medidas, no una predicción de desempeño para enlaces futuros.

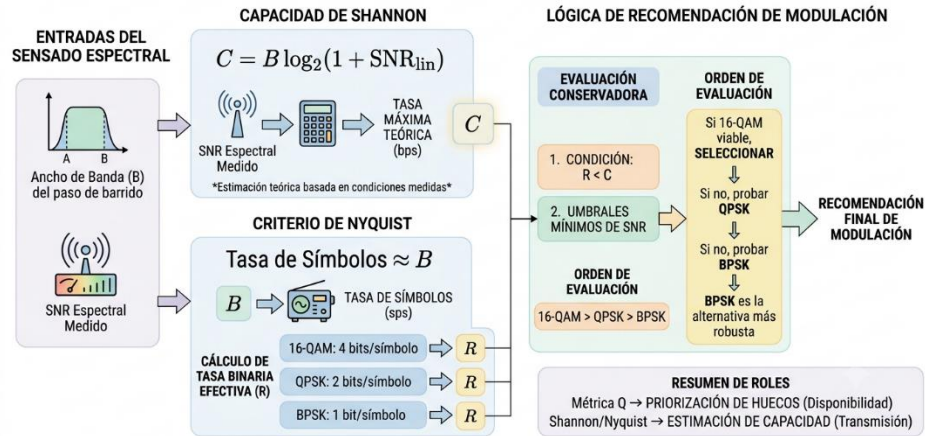
El criterio de Nyquist acota la tasa máxima de símbolos que un canal limitado en banda admite sin interferencia intersimbólica (ISI). La implementación la aproxima como:

$$R_{s\_max} = B / 2 \quad (19)$$

La tasa de símbolos determina la tasa binaria de cada esquema candidato según sus bits por símbolo: 1 en BPSK, 2 en QPSK y 4 en 16-QAM.

El sistema examina los esquemas desde el menos complejo hasta el más complejo: BPSK, QPSK y 16-QAM. Elige el primero que se ajuste a la capacidad de Shannon estimada y cuya relación señal-ruido solicitada sea menor o igual a lo estipulado. En caso de que ningún esquema satisfaga estas reglas, se elige BPSK por defecto.

De esta manera, los criterios de disponibilidad permiten priorizar los huecos espectrales, mientras que Shannon y Nyquist permiten estimar la modulación más adecuada bajo las condiciones observadas. Esta separación evita que factores como el tiempo de inactividad o la distancia a bandas reguladas determinen directamente la modulación, ya que dichos factores describen disponibilidad espectral, no capacidad de transmisión.



**Figura 23. Flujo de análisis de capacidad y selección de modulación basado en los teoremas de Shannon y Nyquist.**

El diagrama muestra cómo se elige el tipo de modulación en los espacios que se detectan. Con la información sobre el ancho de banda y la relación señal-ruido que se obtiene al analizar la situación, el sistema calcula la capacidad del canal según la fórmula de Shannon. Luego, con el criterio de Nyquist, se determina la velocidad a la que se pueden enviar los símbolos. La cantidad de bits que se pueden enviar en cada símbolo en cada esquema determina la tasa de bits. La decisión se toma probando los esquemas, desde el BPSK hasta el 16-QAM, y se elige el primero que cumpla con los límites y los umbrales de calidad establecidos.

### 3.5.7 Validación con transmisor de prueba

El sistema se validó con un prototipo de pruebas basado en el script test\_transmitter.py, que hace que el Pluto+ SDR recorra las nueve modulaciones de forma secuencial. Durante cada transmisión, que dura un tiempo configurable (de 20 segundos por defecto), hay pausas de silencio de 10 segundos entre ellas. Esta variación permite verificar al mismo tiempo la capacidad de clasificación en tiempo real y la detección dinámica de huecos espectrales. Además, el uso del equipo comercial de radio FM permite verificar la cadena desde la demodulación hasta la

reproducción de audio. Los resultados cuantitativos de estas pruebas se presentan en el Capítulo IV.

### **3.5.8 Diseño del mecanismo de Open-Set Recognition (OSR)**

Un sistema de clasificación de modulaciones que se usa en situaciones del mundo real a menudo se enfrenta a señales que no pertenecen a ninguna de las clases con las que fue entrenado. Esto ocurre cuando se sintonizan modulaciones que no están en el conjunto de entrenamiento, señales que están codificadas, formas de onda que combinan diferentes tipos o interferencias que tienen una estructura definida. Un clasificador cerrado reparte toda la probabilidad entre las once clases que conoce. Cuando recibe una entrada que no está dentro de su distribución, le asigna necesariamente una de esas clases, a menudo con mucha confianza. El sistema AMC-OTA incluye un mecanismo para reconocer señales que no están en el conjunto de clases conocidas. Este mecanismo se llama reconocimiento de conjunto abierto.

El modelo fue entrenado como un clasificador de conjunto cerrado que tiene once clases. Por lo tanto, cuando recibe una señal que no estuvo en el entrenamiento, la red neuronal predecirá una de las clases conocidas. Para evitar ese comportamiento se usó un criterio de desviación basado en el valor máximo de la distribución softmax. Cuando el valor de confianza sea menor a 0,65, la interpretación cambiará a la categoría “SEÑAL DESCONOCIDA”. Este mecanismo funciona después de la inferencia y no afecta los parámetros ni la estructura de la CNN.

El valor 0,65 se tomó durante el desarrollo como un umbral operativo heurístico. Con once clases, una salida uniforme tendría una probabilidad de cerca de 0,091 por clase. Pero esta comparación solo muestra que 0,65 es mayor que el nivel de azar y no prueba que sea el umbral óptimo. En el conjunto cerrado de prueba, la confianza mínima de las predicciones correctas fue alrededor de 0,769. Por tanto, el umbral no rechazó las predicciones correctas observadas en ese

conjunto. No se hizo una optimización independiente con una colección representativa de señales desconocidas. Por eso, el valor debe considerarse un parámetro operativo inicial y no una frontera estadísticamente óptima.

Cabe destacar que el enfoque por umbral sobre softmax es la forma más simple de OSR y presenta limitaciones conocidas en la literatura. Métodos más sofisticados como OpenMax, los enfoques basados en energía (energy-based OOD detection) o las técnicas de calibración por temperatura podrían ofrecer mayor robustez ante señales fuera de distribución, especialmente cuando estas comparten características temporales con alguna clase conocida. La incorporación de estos métodos avanzados se identifica como una línea relevante de trabajo futuro. Para los objetivos de esta tesis, el enfoque por umbral simple resultó suficiente para diferenciar transmisiones activas de pausas de silencio durante las pruebas en tiempo real.

### **3.6 Restricciones del *hardware* y decisiones de diseño**

El diseño del sistema AMC-OTA estuvo condicionado por restricciones propias del *hardware* de bajo coste empleado. A continuación se documentan las decisiones de diseño que estas restricciones motivaron en cada fase del proyecto, junto con la justificación técnica de cada elección. Comprender estas decisiones es esencial para contextualizar los parámetros de configuración descritos en las secciones anteriores.

#### **3.6.1 Estabilidad del búfer USB: fijación de la tasa de muestreo**

El tuner FC0013 del RTL-SDR genera desbordamientos del búfer USB a tasas de muestreo de 2,4 MSPS o superiores, lo que corrompe las capturas. Esta restricción de *hardware* determinó la fijación de la tasa de muestreo en 2,0 MSPS y el límite de bloque de lectura en 128 K muestras.

La elección se justifica además por el sobremuestreo suficiente que esta tasa ofrece para las modulaciones empleadas, tal como se detalla en la sección 3.2.2.

### **3.6.2 Gestión térmica del Pluto+ SDR durante la captura**

El Pluto+ SDR carece de sistema de refrigeración activo, lo que provoca degradación en la calidad de las señales generadas durante sesiones prolongadas. Para garantizar la integridad del *dataset*, se incorporaron ciclos de enfriamiento térmico de 40 segundos cada tres modulaciones durante la recolección, y pausas de silencio entre transmisiones durante las pruebas de inferencia. Esta restricción térmica es inherente al trabajo con *hardware* real de bajo coste y no se manifiesta en entornos de simulación.

### **3.6.3 Diseño del control de ganancia para evitar saturación del ADC**

El ADC del RTL-SDR es de solo 8 bits, lo que limita su rango dinámico a 48 dB teóricos. Una ganancia excesiva satura el convertidor en presencia de estaciones FM con señal fuerte, introduciendo distorsión irreparable. Para mitigar esta restricción se diseñó un sistema de ganancia adaptativa por estación: comienza con 20 dB y la reduce automáticamente cuando detecta valores I/Q cercanos a los límites del ADC. Esta calibración por estación es esencial para obtener muestras WBFM de calidad sin distorsión por saturación.

### **3.6.4 Sincronización TX/RX sin protocolo nativo de coordinación**

El Pluto+ SDR y el RTL-SDR proceden de fabricantes distintos, se controlan por interfaces separadas (Ethernet y USB) y no comparten mecanismo de sincronización. La emisión se detecta entonces por umbral de energía: el receptor vigila la potencia de las muestras e inicia la recolección cuando el nivel resulta compatible con una transmisión activa. La situación reproduce la de un sistema de monitoreo del espectro, donde el receptor no coordina con los transmisores.

### 3.6.5 CFO entre osciladores: descarte de arquitecturas CNN-2D

Como se detalló en la sección 3.3.4, el CFO entre osciladores independientes produce constelaciones circulares en lugar de los puntos discretos que requieren las arquitecturas CNN-2D (Red Neuronal Convolutiva bidimensional) basadas en imágenes de constelación. Esta restricción física condicionó la selección de la arquitectura CNN-1D (Red Neuronal Convolutiva unidimensional), que opera directamente sobre secuencias temporales I/Q y es inherentemente invariante al CFO, sin necesitar un paso previo de estimación y compensación.

### 3.6.6 Balance del *dataset*: mecanismo de *fill-up* por clase

La variabilidad en la tasa de rechazo del *pipeline* de validación (especialmente en modulaciones de alto PAPR como 16-QAM y en capturas WBFM de estaciones débiles) hace que ciertas clases acumulen muestras válidas más lentamente. Para garantizar el balance perfecto del *dataset* sin sesgos de clase, se diseñó el mecanismo de *fill-up*: al finalizar la recolección principal, el sistema identifica automáticamente las clases con déficit y ejecuta rondas adicionales dirigidas hasta completar las 5 000 ventanas requeridas por clase.

### 3.6.7 Criterio de selección del tamaño de ventana I/Q

El tamaño de la ventana es importante porque decide cuánta estructura temporal se puede ver en cada predicción. Si la ventana es demasiado corta, tendrá pocos cambios de símbolo. Por otro lado, ventanas más largas necesitan más memoria y cuestan más en procesamiento.

Se eligió una ventana de 1 024 muestras I/Q. Con una tasa de muestreo de 2 millones de muestras por segundo y una tasa de símbolos de 100 000 símbolos por segundo, esta configuración tiene alrededor de 51 símbolos por ventana. Esto es suficiente para clasificar.

Para saber si funciona, se probó el modelo dentro de una aplicación en C++. Sin embargo, no se agregaron herramientas para medir solo el tiempo que toma hacer predicciones ni para calcular su media y desviación. Por lo tanto, no hay un resultado específico sobre cuánto tiempo toma el modelo en sí.

### **3.6.8 Migración a C++ por limitaciones del GIL de Python**

Una primera versión del sistema de inferencia se creó con Python. Se utilizó `onnxruntime` para el modelo ONNX y `matplotlib` para hacer gráficas. Esto ayudó a comprobar que todo funcionara bien con el *hardware*. Sin embargo, hubo dos problemas: las gráficas se actualizaban muy despacio y había cortes en el audio FM demodulado.

El problema principal es el GIL de CPython. Esto significa que solo un hilo puede ejecutar código a la vez. Entonces, cuando se necesita procesar el audio, esto compite con la creación de gráficas y la captura de datos. Todo intenta usar el mismo recurso al mismo tiempo. El resultado es que el audio no se entrega de manera suave a la tarjeta de sonido. Esto causa problemas con la calidad del sonido.

Esta restricción motivó la decisión de reescribir el sistema en C++17 como aplicación nativa de escritorio, descrita en la sección 3.5. Los hilos nativos de C++ se ejecutan en paralelo real sin GIL; el estado se comparte directamente en memoria entre hilos y el renderizado OpenGL de Dear ImGui accede directamente a la GPU (Graphics Processing Unit — Unidad de Procesamiento Gráfico). El resultado es una aplicación con audio FM continuo y gráficas actualizadas de forma fluida (limitadas por diseño a ~30 cuadros por segundo para moderar el consumo de CPU).

## **3.7 Consideraciones finales del capítulo**

El capítulo muestra que trabajar con señales reales en el aire impone limitaciones que influyen en las decisiones de diseño. La principal es la diferencia de frecuencia entre osciladores independientes. Esto hace que no se pueda utilizar una red neuronal convolucional 2D sobre imágenes de constelación, ya que el giro continuo de fase las distorsiona. Por lo tanto, se optó por una red neuronal convolucional 1D en el dominio temporal, cuya tolerancia a la diferencia de frecuencia se logra mediante el aumento de datos.

El *hardware* de bajo costo también tuvo un impacto en el diseño. La saturación del convertidor analógico-digital de 8 bits, los desbordamientos del búfer USB y la deriva térmica determinaron la ganancia manual, la tasa de 2,0 millones de muestras por segundo, el validador de tres etapas y el almacenamiento en formato HDF5. El resultado es un sistema compacto que tiene alrededor de 1,2 millones de parámetros y un modelo ONNX de aproximadamente 5 megabytes. Este sistema puede clasificar modulaciones e identificar huecos espectrales en la banda observada.

El proceso consta de tres fases: recolección, entrenamiento y despliegue. Estas fases están interconectadas. La ventana de 1 024 muestras, la ganancia fija y el aumento de datos influyen en la robustez del modelo. La arquitectura multihilo en C++, la inferencia con ONNX Runtime y el rechazo por conjunto abierto garantizan que la robustez del modelo se mantenga durante la operación continua. El Capítulo IV presenta los resultados numéricos del entrenamiento, la evaluación y la inferencia.

## Capítulo IV: Resultados y validación Experimental

### 4.1 Introducción

En este capítulo se presentan los resultados obtenidos durante la validación experimental del sistema de clasificación automática de modulaciones OTA en tiempo real. Los resultados se organizan en función de las principales etapas desarrolladas en el proyecto: caracterización visual de las señales capturadas, análisis espectral de las modulaciones, entrenamiento del modelo CNN-1D residual, evaluación mediante matriz de confusión, análisis de confianza del clasificador y verificación funcional de la aplicación de sensado espectral implementada en C++.

El sistema se validó con señales I/Q capturadas por SDR: ocho modulaciones digitales y AM-DSB generadas con el Pluto+, WBFM de emisoras comerciales y una clase de ruido, once en total. Las muestras se procesaron y organizaron para entrenar y evaluar el modelo. El modelo exportado se integró luego en una aplicación de escritorio que infiere en tiempo real sobre las muestras del RTL-SDR.

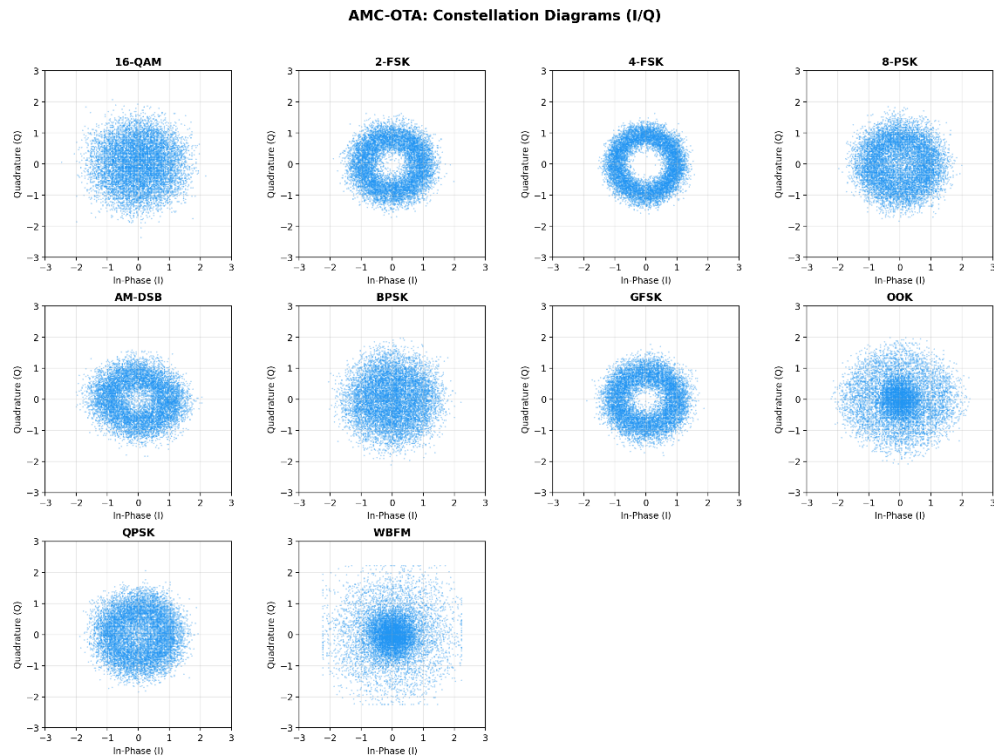
### 4.2 Caracterización de las señales capturadas

Antes de evaluar el desempeño del modelo, se analizaron visualmente las señales capturadas con el fin de verificar que el *dataset* contuviera patrones diferenciables entre clases. Para ello, se generaron diagramas de constelación I/Q y gráficas de densidad espectral de potencia para cada tipo de señal.

Las clases consideradas en esta etapa fueron: AM-DSB, BPSK, QPSK, 8-PSK, 16-QAM, OOK, 2-FSK, 4-FSK, GFSK y WBFM. La clase “NOISE” se evaluó principalmente en la etapa de clasificación, ya que no representa una modulación específica, sino ausencia de señal útil.

### 4.2.1 Diagramas de constelación I/Q

Los diagramas de constelación muestran la distribución de las componentes en fase I y cuadratura Q de las señales capturadas. A diferencia de constelaciones ideales generadas en simulación, las capturas OTA presentan dispersión, rotación de fase y deformaciones propias del canal y del *hardware* SDR. Estas variaciones son esperadas debido al ruido, al *Carrier Frequency Offset*, a la falta de sincronización perfecta entre transmisor y receptor, y a las condiciones reales de propagación.



**Figura 24. Diagramas de constelación I/Q de las modulaciones capturadas.**

En las modulaciones PSK y QAM, no aparecen puntos definidos. Se ve una nube de muestras cerca del centro. Esto pasa porque las señales se tomaron como secuencias I/Q sin sincronizar la señal ni recuperar la portadora. En las modulaciones FSK y GFSK, las constelaciones parecen círculos o anillos. Eso ocurre porque la información no sigue posiciones

fijas de fase, sino que cambia de frecuencia. Por otro lado, la distribución de WBFM es más grande y menos ordenada. Eso se debe a que el sistema es analógico y tiene un ancho de banda grande.

**Tabla 4**

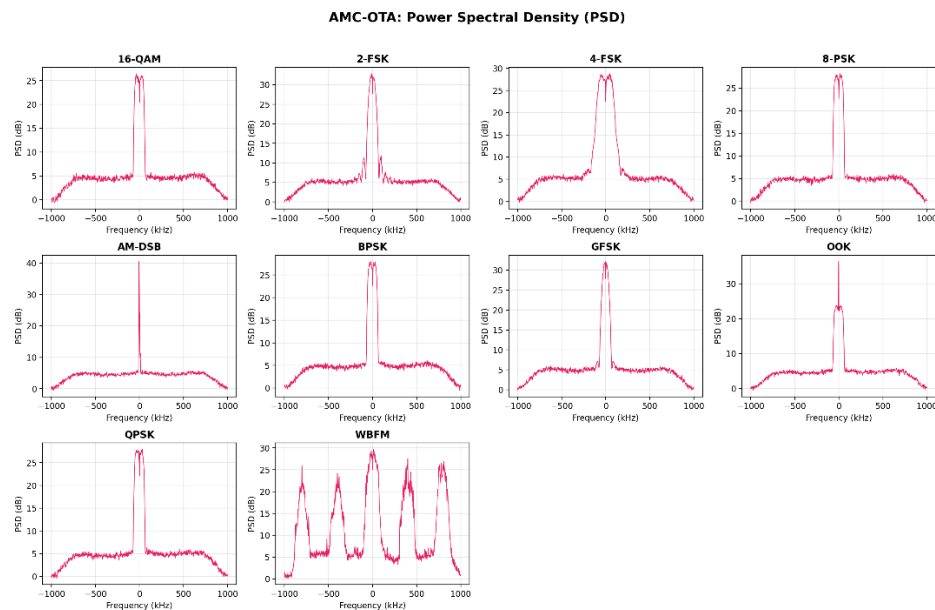
Interpretación de los diagramas de constelación I/Q

| Clase  | Comportamiento observado                                      | Interpretación                                                                                     |
|--------|---------------------------------------------------------------|----------------------------------------------------------------------------------------------------|
| AM-DSB | Distribución con variaciones de amplitud alrededor del origen | La información se refleja principalmente en cambios de amplitud.                                   |
| BPSK   | Nube de puntos con dispersión sobre el plano I/Q              | La modulación por fase se ve afectada por ruido y desfase de portadora.                            |
| QPSK   | Distribución circular dispersa                                | Los cuatro estados de fase no aparecen ideales por tratarse de una captura OTA sin sincronización. |
| 8-PSK  | Nube más densa y distribuida angularmente                     | Mayor cantidad de fases, por lo que la dispersión angular es más evidente.                         |
| 16-QAM | Distribución amplia en amplitud y fase                        | La modulación combina variaciones de amplitud y fase.                                              |
| OOK    | Nube con concentración central y dispersión de amplitud       | Se relaciona con estados de encendido y apagado de la portadora.                                   |
| 2-FSK  | Estructura anular                                             | La información depende de cambios entre dos frecuencias.                                           |
| 4-FSK  | Estructura anular con mayor variación                         | Representa múltiples estados frecuenciales.                                                        |
| GFSK   | Anillo suavizado y compacto                                   | El filtrado gaussiano suaviza las transiciones de frecuencia.                                      |
| WBFM   | Distribución amplia e irregular                               | Señal analógica real con mayor variabilidad espectral y temporal.                                  |

Estos resultados visuales confirman que las señales capturadas no corresponden a condiciones ideales de simulación, sino a un entorno experimental real. Esto justifica el uso de una CNN-1D capaz de aprender patrones directamente desde las secuencias I/Q, sin depender de constelaciones perfectamente sincronizadas.

#### 4.2.2 Densidad espectral de potencia

La densidad espectral de potencia permite observar cómo se distribuye la energía de cada señal en el dominio de la frecuencia. En las modulaciones digitales transmitidas mediante el Pluto+ SDR se observa una concentración de energía alrededor de la frecuencia central, con diferentes formas espectrales según el tipo de modulación.



**Figura 25. Densidad espectral de potencia de las señales capturadas.**

Las modulaciones PSK y QAM tienen un ancho de banda definido porque se filtra la forma de los pulsos. Por otro lado, las modulaciones FSK y GFSK tienen componentes que dependen de los cambios de frecuencia que se utilizan para representar los símbolos.

La señal AM-DSB tiene un pico claro en el medio, que es la portadora, y dos bandas laterales que se deben a la señal moduladora. La WBFM, por su parte, se extiende sobre un ancho de banda mayor. Tiene muchos componentes en su espectro y esto se debe a que es una emisión de FM como las que se escuchan en la radio, que lleva audio y otros contenidos dentro de su ancho de banda.

## Tabla 5

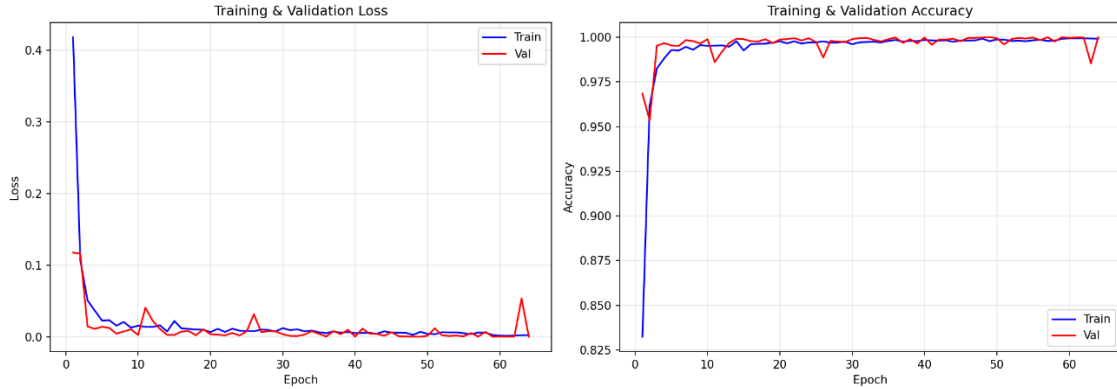
Interpretación de la densidad espectral de potencia

| Clase  | Comportamiento espectral observado                     | Interpretación                                                          |
|--------|--------------------------------------------------------|-------------------------------------------------------------------------|
| AM-DSB | Pico central pronunciado                               | Presencia de portadora y bandas laterales de amplitud.                  |
| BPSK   | Energía concentrada alrededor de la frecuencia central | Modulación de fase binaria con ancho de banda limitado por el filtrado. |
| QPSK   | Espectro compacto similar a BPSK                       | Mayor eficiencia espectral al transmitir dos bits por símbolo.          |
| 8-PSK  | Energía central con forma compacta                     | Modulación de fase de mayor orden.                                      |
| 16-QAM | Espectro concentrado con variaciones de amplitud       | Combinación de amplitud y fase.                                         |
| OOK    | Pico central y energía distribuida alrededor           | Cambios de amplitud por encendido y apagado de la portadora.            |
| 2-FSK  | Componentes cercanas a la frecuencia central           | Desplazamientos entre dos frecuencias.                                  |
| 4-FSK  | Mayor extensión respecto a 2-FSK                       | Uso de cuatro estados frecuenciales.                                    |
| GFSK   | Espectro más suavizado                                 | El filtrado gaussiano reduce transiciones abruptas.                     |
| WBFM   | Espectro amplio con varios picos                       | Señal FM comercial real con mayor ocupación de ancho de banda.          |

El análisis espectral permitió comprobar que cada clase posee rasgos frecuenciales diferenciables. Estos patrones complementan la información temporal de las muestras I/Q y explican por qué el modelo puede aprender a separar adecuadamente las clases durante el entrenamiento.

### 4.3 Resultados del entrenamiento del modelo CNN-1D residual

El modelo que se utilizó se llama CNN-1D residual. Este modelo se entrenó con ventanas I/Q que ya habían sido procesadas. Cada ventana tenía 1 024 muestras. Luego, cada ventana se convirtió en un tensor con dos canales. Uno de los canales era para la componente I y el otro para la componente Q. Durante el entrenamiento, se hicieron algunas cosas para que el modelo fuera mejor. Se cambiaron las amplitudes, se desplazaron las fases y se agregó ruido. Todo esto se hizo para que el modelo pueda funcionar bien incluso cuando las condiciones no son ideales.



**Figura 26. Curvas de pérdida y exactitud durante el entrenamiento y validación.**

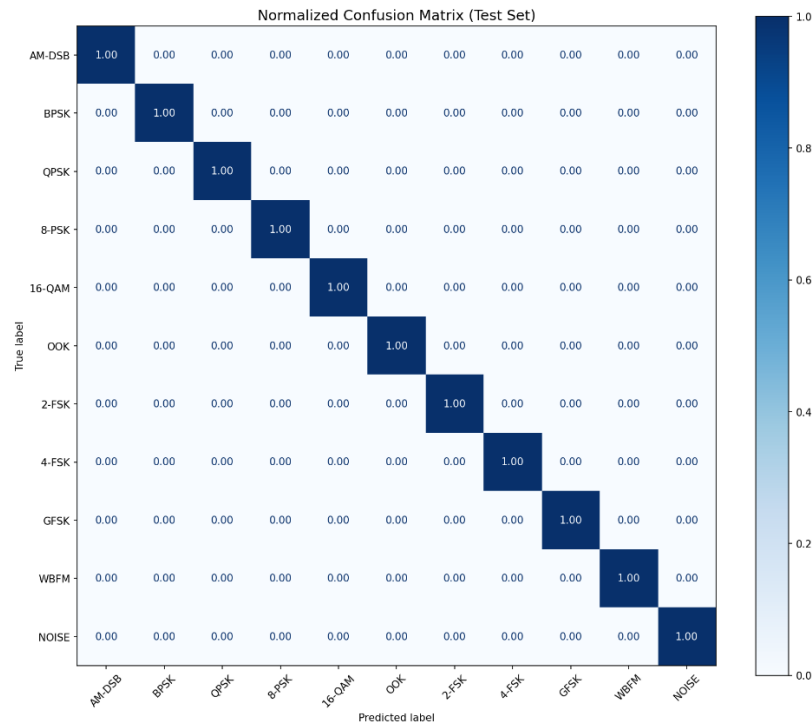
En las curvas de entrenamiento se observa una reducción rápida de la función de pérdida durante las primeras épocas. La exactitud de entrenamiento y la de validación aumentan de forma acelerada y se mantienen cercanas a 1,00 durante la mayor parte del proceso. Esto indica que la arquitectura CNN-1D residual logró aprender patrones discriminativos entre las clases de modulación consideradas.

No hay muchas diferencias en la curva de validación, pero la curva general del modelo no cambia por ello. La pérdida se mantiene en valores bajos, mientras que la precisión es constante. Esto indica que el modelo funciona correctamente y no hay mucha diferencia entre la fase de entrenamiento y la de validación, teniendo en cuenta los datos utilizados.

Los resultados indican que el modelo se entrenó correctamente. Sin embargo, como el desempeño es muy bueno, es importante considerar el entorno en el que se realizó el experimento. El modelo se evaluó con datos del mismo conjunto y preprocesamiento, por lo que es posible que los resultados cambien en situaciones diferentes. Sería útil realizar más pruebas en diferentes escenarios para obtener una visión más completa.

#### **4.4 Evaluación del modelo mediante matriz de confusión**

Para evaluar el desempeño del clasificador en el conjunto de prueba, se generó una matriz de confusión normalizada. Esta matriz permite observar la relación entre las clases reales y las clases predichas por el modelo. Los valores ubicados en la diagonal principal representan clasificaciones correctas, mientras que los valores fuera de la diagonal representan errores o confusiones entre clases.



**Figura 27. Matriz de confusión normalizada del conjunto de prueba.**

La matriz de confusión obtenida presenta valores de 1,00 en la diagonal principal para todas las clases evaluadas. Esto indica que, dentro del conjunto de prueba utilizado, el modelo clasificó correctamente las muestras de cada clase sin presentar confusiones visibles entre modulaciones. Las clases AM-DSB, BPSK, QPSK, 8-PSK, 16-QAM, OOK, 2-FSK, 4-FSK, GFSK, WBFM y NOISE se identificaron correctamente.

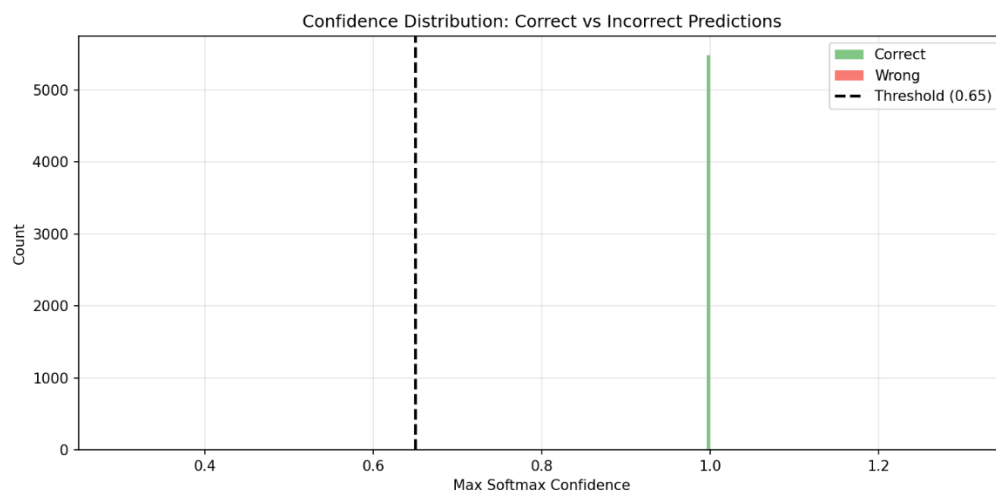
El resultado indica que la red CNN-1D residual capta las diferencias entre clases en el conjunto de pruebas. Esto implica que los datos son suficientemente representativos como para

entrenar el modelo de clasificación, y que la configuración utilizada es útil para el tratamiento de secuencias I/Q reales.

El resultado que se obtuvo es válido en un laboratorio donde se controlan las condiciones. Sin embargo, si las antenas están muy lejos, si hay interferencias de fuera, si la relación señal-ruido varía mucho o si se usan dispositivos SDR diferentes, el desempeño puede cambiar. Por eso, el sistema tiene un mecanismo que reconoce patrones de manera abierta y utiliza un umbral de confianza.

#### 4.5 Análisis de confianza del clasificador

Además de la clase predicha, el modelo genera una distribución de probabilidades mediante la función softmax. La confianza máxima permite describir el grado de seguridad asignado por el modelo a una predicción. En la implementación se adoptó heurísticamente un umbral operativo de 0,65, de modo que las predicciones con confianza inferior sean presentadas como señales desconocidas.



**Figura 28. Distribución de confianza entre predicciones correctas e incorrectas.**

La distribución de confianza muestra que las predicciones correctas del conjunto de prueba se concentran en valores elevados. Las predicciones correctas tienen una confianza mínima de 0,769. El umbral operativo se fijó en 0,65, lo que significa que ninguna de las predicciones correctas sería rechazada. El valor de 0,65 no se eligió al azar. Está por debajo de la confianza mínima de las predicciones correctas, con un margen de 0,12. Esto permite absorber la variabilidad de la confianza en predicciones legítimas y descartar a la vez las salidas con confianza inferior. Este resultado permite comprobar que el valor seleccionado es compatible con las clases conocidas del conjunto de prueba. Sin embargo, no significa que sea el valor óptimo para rechazar señales desconocidas. Una señal externa al entrenamiento puede producir una confianza softmax elevada para alguna clase conocida.

**Tabla 6**

Análisis del umbral de confianza softmax

| Parámetro                | Valor utilizado            | Función dentro del sistema                                                |
|--------------------------|----------------------------|---------------------------------------------------------------------------|
| Confianza máxima softmax | Valor entre 0 y 1          | Indica la seguridad del modelo en la clase predicha.                      |
| Umbral de reconocimiento | 0,65                       | Define el límite mínimo para aceptar una clase como conocida.             |
| Predicción $\geq 0,65$   | Clase aceptada             | La señal se clasifica como una de las modulaciones conocidas.             |
| Predicción $< 0,65$      | Señal desconocida          | La señal se etiqueta como desconocida para evitar falsas clasificaciones. |
| Distribución observada   | Confianzas cercanas a 1,00 | El modelo muestra alta seguridad en el conjunto de prueba.                |

El mecanismo implementado constituye una primera aproximación al reconocimiento de conjunto abierto. Su finalidad es evitar que predicciones con confianza moderada o baja sean presentadas como identificaciones válidas. En la versión evaluada no se realizó una optimización

exhaustiva del umbral mediante múltiples señales conocidas y desconocidas; por esta razón, no se calcularon tasas de falsa aceptación y falso rechazo para diferentes valores.

El conjunto evaluado contiene principalmente clases conocidas durante el entrenamiento. Por ello, una confianza elevada en dicho conjunto no garantiza el rechazo correcto de señales fuera de distribución. El caso de falsos positivos WBFM presentado posteriormente confirma que el máximo softmax puede ser elevado ante señales externas. La evaluación sistemática de diferentes umbrales mediante tasas de falsa aceptación y falso rechazo se plantea como trabajo futuro.

#### **4.5.1 Comportamiento del clasificador ante señales no entrenadas**

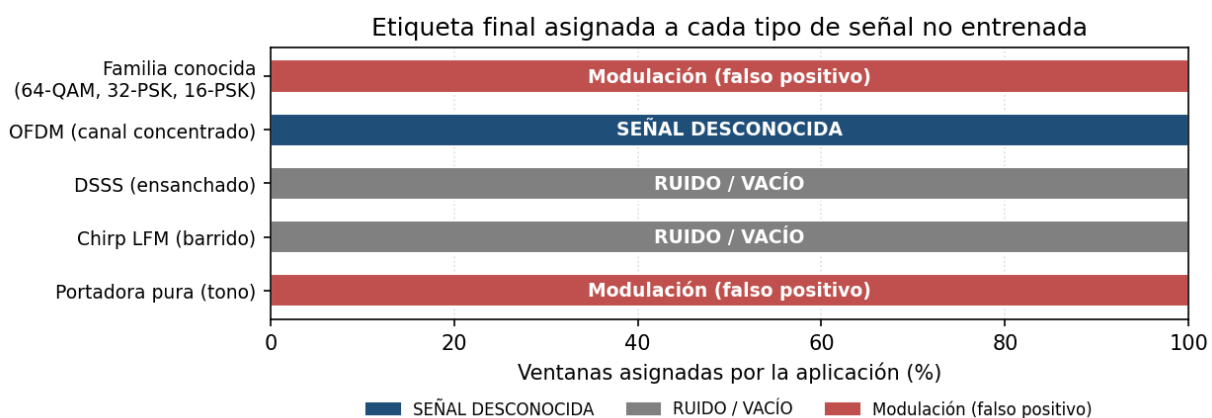
Se evaluaron dos tipos de entradas ajenas a las once clases entrenadas. El primero reúne modulaciones de familias presentes en el entrenamiento, pero de orden no observado: 64-QAM, 32-PSK y 16-PSK. El segundo, señales de estructura espectral distinta de todas las clases entrenadas: OFDM (multiportadora), chirp LFM (barrido de frecuencia), espectro ensanchado (DSSS) y portadora sin modular. En ambos casos, la aplicación utilizó toda su lógica de decisión. Esto incluyó la compuerta de ocupación, la clase de ruido y el umbral de confianza, y no solo se basó en el modelo.

Las modulaciones de familias conocidas son las más difíciles de manejar. Estas modulaciones comparten la estructura de constelación de las clases entrenadas. Por lo tanto, el clasificador las asigna con mucha confianza a la vecina más cercana. Por ejemplo, el 64-QAM se asigna al 16-QAM, y el 32-PSK y el 16-PSK se asignan al 8-PSK.

La red no tiene una clase para esos órdenes. Por eso, el máximo softmax solo no puede detectar la asignación errónea dentro de una misma familia. Esto muestra las limitaciones del sistema.

Las señales de estructura espectral distinta se comportan de manera opuesta. La OFDM, el chirp y el DSSS no se asignan a una modulación concreta, sino que producen un espectro plano y sin portadora dominante que el modelo asocia a la clase de ruido. Cabe precisar que esta clasificación como ruido no obedece a una baja relación señal-ruido: estas señales se identifican como ruido incluso generadas de forma prácticamente limpia (40 dB), porque su firma espectral es intrínsecamente plana, a diferencia de la degradación por ruido aditivo que afecta a las modulaciones conocidas descrita en la sección 4.6.

La aplicación utiliza una lógica para determinar la etiqueta final. Esta etiqueta se basa en la salida y en la ocupación espectral, como se muestra en la Figura 29. Si hay una señal desconocida que ocupa una banda concentrada con suficiente energía, como una transmisión OFDM en un canal específico, se activa la comprobación de ocupación. En este caso, hay señal donde el modelo indicó que había ruido. Por lo tanto, la ventana se reetiqueta como “SEÑAL DESCONOCIDA”. Por otro lado, las señales ensanchadas, como DSSS, o las señales de barrido de banda ancha, como el chirp, presentan un espectro plano. No superan el umbral de ocupación y se reportan como “RUIDO/VACÍO”. La portadora sin modular es otro caso. Se trata de un componente tonal concentrado que el modelo asocia con 4-FSK.



**Figura 29. Etiqueta final asignada por la aplicación a cada tipo de señal no entrenada.**

Para cuantificar el desempeño del sistema ante estas señales se barrió el umbral de confianza con la cadena de decisión completa (modelo, clase de ruido, compuerta de ocupación y umbral) sobre OFDM, chirp y DSSS. La Tabla 7 reporta la tasa de falso rechazo (FRR), fracción de señales conocidas que dejan de mostrarse como su modulación, y la de falsa aceptación (FAR), fracción de señales no entrenadas que se muestran como una modulación real. En el umbral operativo de 0,65 la falsa aceptación es nula (0,00 %) y el falso rechazo es prácticamente nulo (0,06 %), y ambas se mantienen estables en todo el rango. Es decir, ninguna de estas modulaciones no entrenadas se presenta como una clasificación válida y casi ninguna señal legítima se descarta.

**Tabla 7**

Falso rechazo y falsa aceptación del sistema completo frente a modulaciones no entrenadas de familias distintas

| Umbral de confianza | FRR (%) | FAR (%) |
|---------------------|---------|---------|
| 0,50                | 0,06    | 0,00    |
| 0,60                | 0,06    | 0,00    |
| 0,65                | 0,06    | 0,00    |
| 0,70                | 0,06    | 0,00    |
| 0,80                | 0,10    | 0,00    |
| 0,90                | 0,10    | 0,00    |
| 0,95                | 0,16    | 0,00    |

Nota: se han probado 5 000 ventanas reales. Las modulaciones que no se han entrenado son OFDM, chirp LFM y DSSS. La tasa de falso rechazo es la fracción de señales correctas que no se muestran como su modulación real. La tasa de falsa aceptación es la fracción de señales no entrenadas que se muestran como una modulación real. Se ha aplicado la cadena de decisión completa de la aplicación.

El sistema es robusto con señales desconocidas. Esto es una propiedad del sistema completo y no del umbral de confianza. La clase de ruido y la compuerta de ocupación ayudan a

resolver transmisiones no entrenadas que ocupan un canal concreto. Esto es común en un escenario de sensado. Las señales legítimas no se ven afectadas.

Además, existen tres problemas. El primero es la asignación incorrecta de modulaciones de la misma familia. El segundo es la consideración de las señales de espectro ancho como señales vacías. El tercer problema está asociado a la presencia de componentes simples del tono. Estos inconvenientes, además de la naturaleza sintética de las señales de prueba, sugieren que en el futuro será necesario añadir métodos para la identificación de conjuntos abiertos. Algunos de estos métodos son OpenMax o puntuaciones basadas en energía. Se debe realizar una comprobación de los métodos antes mencionados con capturas reales de modulaciones no entrenadas.

## **4.6 Alcance de la robustez frente al ruido**

### **4.6.1 Condiciones de ruido consideradas durante el entrenamiento**

Durante el entrenamiento se aplicó aumento de datos mediante la adición aleatoria de ruido blanco gaussiano con valores de SNR comprendidos entre 5 y 30 dB. Este procedimiento tuvo como finalidad exponer al modelo a diferentes niveles de degradación y evitar que dependiera exclusivamente de capturas con condiciones favorables. Es necesario precisar que el intervalo de 5 a 30 dB corresponde al rango de relación señal-ruido empleado por el aumento de datos sintético durante el entrenamiento, y no a un umbral de descarte aplicado a las capturas. El conjunto de datos se creó con señales reales que se capturaron por aire. Estas señales debían cumplir con ciertos requisitos de calidad. Tenían que tener una relación entre la señal y el ruido de al menos 2 decibelios. Esto significa que las señales ya contenían las condiciones del canal y los niveles de ruido del enlace. Luego, se agregaron perturbaciones gaussianas a estos datos dentro de un rango específico. La aplicación también calcula y muestra una estimación de la relación señal-ruido

durante su funcionamiento. Sin embargo, ver esta métrica no es suficiente para evaluar la exactitud en función de la relación señal-ruido.

#### 4.6.2 Evaluación de la exactitud en función del SNR

Con posterioridad a la observación planteada, se realizó una evaluación controlada de la exactitud en función del SNR con el fin de determinar el nivel de ruido en el cual el clasificador deja de ser confiable. El método utilizó los mismos conjuntos de pruebas utilizados durante el entrenamiento, obtenidos a través de la división estratificada 80/10/10 (5 500 muestras no vistas por el modelo). Para cada ventana, se introdujo ruido blanco gaussiano de diferentes niveles de ruido: -5, 0, 5, 10, 15, 20, 25 y 30 dB. Se emplearon las mismas formulaciones que en el incremento de datos durante el entrenamiento. El conjunto resultante se evaluó con el modelo ONNX. También se añadió una condición de referencia sin ruido añadido.

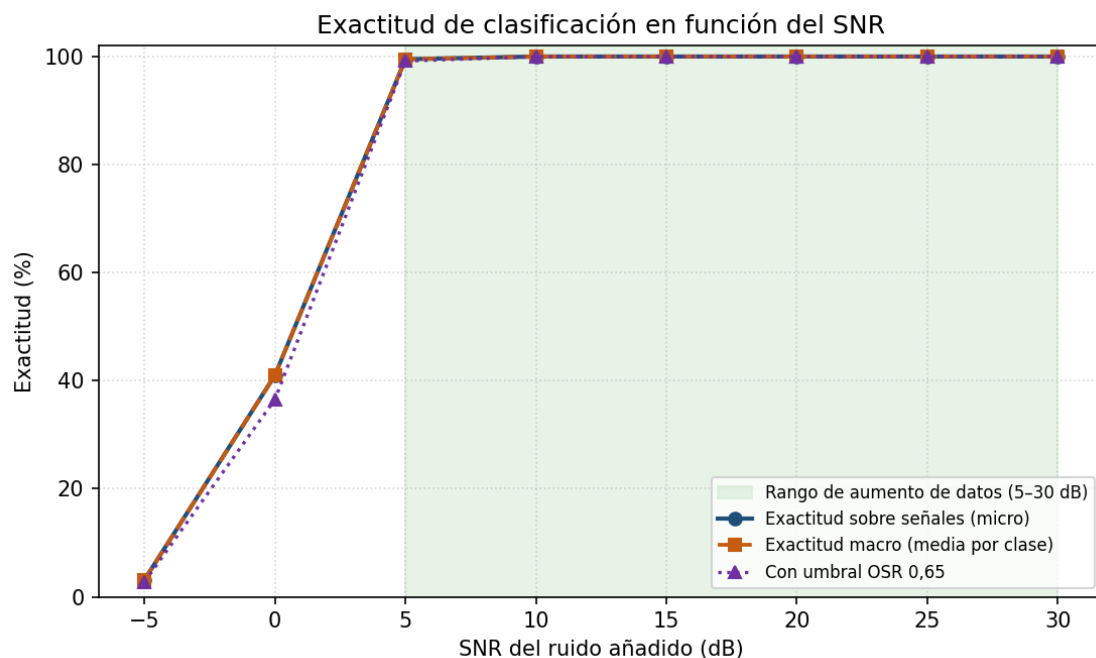
La columna con umbral aplica el criterio de conjunto abierto (OSR) de 0,65: las predicciones de confianza inferior se contabilizan como rechazadas. La confianza media es el máximo softmax promediado sobre las clases de señal.

**Tabla 8**

Exactitud de clasificación en función del SNR del ruido añadido

| SNR del ruido añadido (dB) | Exactitud sobre señales (%) | Exactitud con umbral 0,65 (%) | Confianza media |
|----------------------------|-----------------------------|-------------------------------|-----------------|
| -5                         | 3,16                        | 2,78                          | 0,930           |
| 0                          | 40,98                       | 36,50                         | 0,901           |
| 5                          | 99,46                       | 99,12                         | 0,992           |
| 10                         | 99,96                       | 99,94                         | 0,999           |
| 15                         | 99,96                       | 99,94                         | 1,000           |
| 20                         | 99,96                       | 99,96                         | 1,000           |
| 25                         | 99,96                       | 99,96                         | 1,000           |
| 30                         | 99,96                       | 99,96                         | 1,000           |
| Sin ruido añadido          | 99,98                       | 99,98                         | 1,000           |

Nota. La exactitud se determinó sobre las clases de señal, descontando la clase de ruido. El campo con umbral emplea el criterio de reconocimiento de conjunto abierto que utiliza un umbral de 0,65; las predicciones con confianza por debajo del umbral se clasifican como rechazadas. La confianza media representa el mayor valor softmax promedio entre las clases de señal.



**Figura 30. Exactitud de clasificación en función del SNR del ruido añadido.**

El deterioro de las señales no se realiza de forma gradual. Si el SNR es mayor o igual a 5 dB, la precisión de las señales se mantendrá por encima del 99 %, lo cual supone un alto nivel de confianza. Pero si el nivel de ruido de la señal se reduce a cero dB, la precisión cae hasta el 40,98 %, y al reducirlo hasta los -5 dB, la precisión se reduce al 3,16 %. Esto demuestra que la línea de corte práctica para la fiabilidad ocurre alrededor de los 5 dB.

Al analizar los resultados por clase, se observa que la caída en la exactitud no es uniforme. A 0 decibelios, las señales BPSK y OOK mantienen una exactitud superior al 97 %, mientras que las señales 8-PSK, 16-QAM y las variantes FSK tienen una exactitud inferior al 15 %. A -5

decibelios, la exactitud de las modulaciones digitales es prácticamente nula, y solo se puede identificar la clase de ruido de forma consistente.

Estos resultados son coherentes con el límite inferior de 5 decibelios establecido en el aumento de datos. Los dos niveles inferiores del barrido caen fuera del rango visto en el entrenamiento. Por debajo de 5 decibelios, la información discriminativa queda dominada por el ruido, lo que significa que ampliar el aumento hacia esa zona no aportaría información valiosa.

A menos de 5 decibelios, el modelo muestra una confianza media del 93 % pese a clasificar mal las señales. El umbral global de 0,65 elimina solo el 6,38 % de las señales. Esto respalda lo señalado en el apartado 4.5: que una mayor confianza del softmax no implica siempre un acierto con señales muy dañadas, lo que fortalece la importancia de implementar estrategias de descarte más sólidas, cuyo estudio será un trabajo futuro.

## **4.7 Validación funcional de la aplicación en C++**

El modelo entrenado e implantado se integró en una aplicación en C++ para el monitoreo espectral en tiempo real. La aplicación toma las lecturas del RTL-SDR y las transforma en ventanas I/Q de 1 024 muestras. A continuación, calcula la información espectral y realiza la inferencia usando ONNX Runtime. Por último, muestra los resultados a través de una interfaz gráfica.

La interfaz es útil porque muestra varios datos importantes. Por ejemplo, muestra la clase detectada y la confianza del modelo. También muestra la relación señal-ruido estimada, el estado del canal y la ocupación del espectro. Además, la aplicación incorpora algunas funciones adicionales. Estas funciones son demodulación FM, visualización de la densidad espectral de potencia, visualización de constelación y detección de huecos espectrales.

### **4.7.1 Pantalla principal del *dashboard***

La pantalla principal permite monitorear el estado general del sistema durante el barrido espectral. En esta vista se observa la frecuencia analizada, la clase detectada por el modelo, el nivel de confianza, el SNR estimado y el rol asignado al canal. Además, el *dashboard* incorpora una referencia de contexto regulatorio basada en bandas asociadas a servicios de radiodifusión y televisión, lo que permite distinguir si una frecuencia se encuentra dentro de una banda primaria, libre, activa, de guarda o candidata a hueco espectral.

La información regulatoria funciona como apoyo visual para interpretar el espectro y no interviene en la clasificación: toda predicción del modelo procede de la inferencia sobre la ventana I/Q procesada. El contexto de ARCOTEL permite contrastar esa predicción con el servicio atribuido a la banda observada y detectar discrepancias durante el análisis.

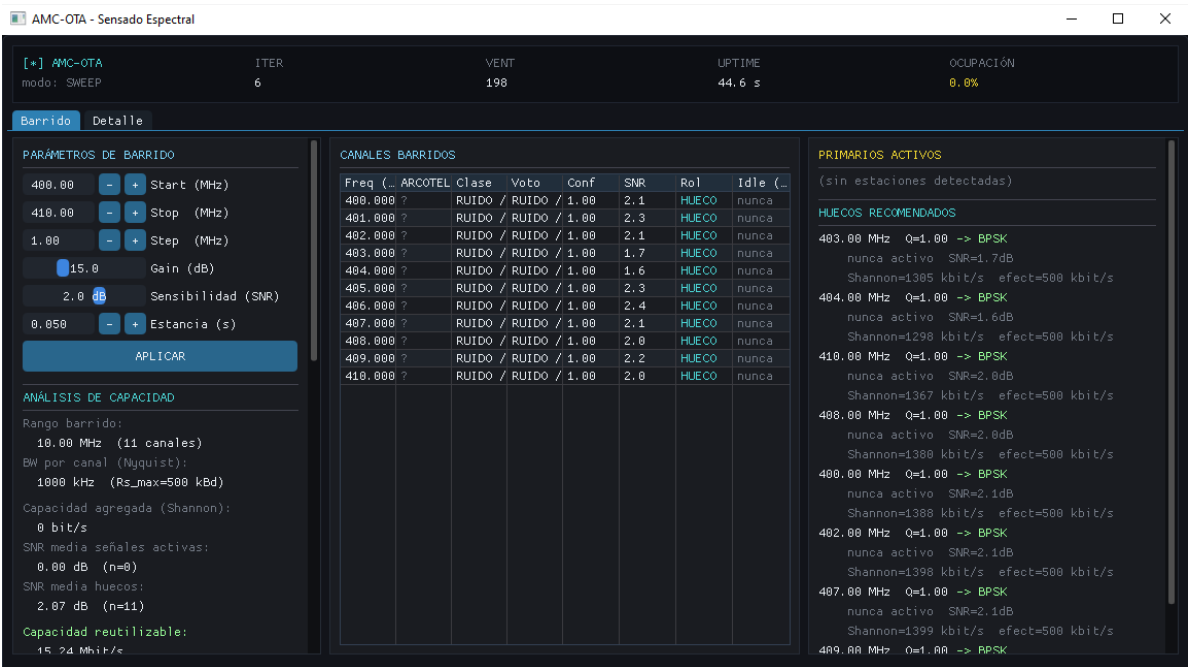


Figura 31. Pantalla principal de la aplicación de sensado espectral en C++.

#### 4.7.2 Visualización de PSD y constelación en tiempo real

La visualización de PSD permite observar la distribución de potencia de la señal recibida en la frecuencia seleccionada. De forma complementaria, la constelación I/Q permite analizar la

dispersión de las muestras complejas. Estas herramientas sirven como apoyo visual para verificar que la clasificación realizada por el modelo guarda relación con el comportamiento de la señal recibida.



**Figura 32. Visualización de PSD y constelación en tiempo real dentro del *dashboard*.**

#### 4.7.3 Clasificación de modulaciones transmitidas con Pluto+ SDR

Para evaluar la clasificación en tiempo real se ejecutaron diez rondas completas mediante el script `test_transmitter.py`. Cada ronda incluyó una transmisión de cada una de las nueve modulaciones generadas por el Pluto+ SDR: AM-DSB, OOK, 16-QAM, BPSK, QPSK, 8-PSK, 2-FSK, 4-FSK y GFSK. En total se realizaron 90 transmisiones, correspondientes a diez repeticiones por modulación.

| CANALES BARRIDOS |               |               |      |      |        |          | PRIMARIO |
|------------------|---------------|---------------|------|------|--------|----------|----------|
| Freq (MHz)       | Clase         | Voto          | Conf | SNR  | RoI    | Idle (s) | (sin es) |
| 480.000          | RUIDO / VACÍO | RUIDO / VACÍO | 1.00 | 1.9  | HUECO  | nunca    | HUECOS   |
| 481.000          | RUIDO / VACÍO | RUIDO / VACÍO | 1.00 | 1.8  | HUECO  | nunca    |          |
| 482.000          | RUIDO / VACÍO | RUIDO / VACÍO | 1.00 | 2.3  | HUECO  | nunca    | 480.00   |
| 483.000          | RUIDO / VACÍO | RUIDO / VACÍO | 1.00 | 2.5  | HUECO  | nunca    | nun      |
| 484.000          | RUIDO / VACÍO | RUIDO / VACÍO | 1.00 | 0.8  | HUECO  | nunca    | Sha      |
| 485.000          | RUIDO / VACÍO | RUIDO / VACÍO | 1.00 | 2.1  | HUECO  | nunca    | 481.00   |
| 486.000          | RUIDO / VACÍO | RUIDO / VACÍO | 1.00 | 2.0  | HUECO  | nunca    | nun      |
| 487.000          | RUIDO / VACÍO | RUIDO / VACÍO | 1.00 | -8.0 | HUECO  | nunca    | Sha      |
| 488.000          | QPSK          | QPSK          | 1.00 | 18.6 | ACTIVO | 3.3      | 484.00   |
| 489.000          | RUIDO / VACÍO | RUIDO / VACÍO | 1.00 | -4.0 | HUECO  | nunca    | nun      |
| 410.000          | RUIDO / VACÍO | RUIDO / VACÍO | 1.00 | 2.2  | HUECO  | nunca    | Sha      |
|                  |               |               |      |      |        |          | 486.00   |
|                  |               |               |      |      |        |          | nun      |
|                  |               |               |      |      |        |          | Sha      |

**Figura 33. Clasificación estabilizada de una modulación transmitida mediante Pluto+ SDR.**

Cada transmisión tuvo una duración de 20 segundos y estuvo seguida de una pausa de 10 segundos. Además, el script generó una nueva forma de onda para cada repetición. Por consiguiente, las rondas utilizaron diferentes secuencias de símbolos, aunque se mantuvieron constantes el *hardware*, la frecuencia de operación y las condiciones generales del enlace.

Durante algunos cambios de modulación se observó una predicción incorrecta con baja confianza en los primeros segundos de la transmisión. Este comportamiento fue transitorio, no se presentó en todos los cambios y desapareció dentro de un periodo máximo aproximado de cinco segundos. Después de este intervalo, la clase mostrada por el voto mayoritario coincidió con la modulación transmitida y permaneció estable durante el resto del ensayo.

Para evitar que la transición entre señales afectara la evaluación, se consideró como resultado de cada transmisión la clase estabilizada después de los primeros cinco segundos. Una coincidencia entre la clase estabilizada y la modulación transmitida se contabilizó como acierto. Una clase diferente, la etiqueta “SEÑAL DESCONOCIDA” o la etiqueta “RUIDO/VACÍO” después del periodo de estabilización se habría contabilizado como error.

**Tabla 9**

### Resultados estabilizados de clasificación en transmisiones OTA repetidas

| <b>Modulación</b>   | <b>Número de transmisiones</b> | <b>Aciertos</b> | <b>Errores</b> | <b>Exactitud (%)</b> | <b>Confianza media calculada (%)</b> |
|---------------------|--------------------------------|-----------------|----------------|----------------------|--------------------------------------|
| AM-DSB              | 10                             | 10              | 0              | 100                  | 100,00                               |
| OOK                 | 10                             | 10              | 0              | 100                  | 100,00                               |
| 16-QAM              | 10                             | 10              | 0              | 100                  | 99,88                                |
| BPSK                | 10                             | 10              | 0              | 100                  | 100,00                               |
| QPSK                | 10                             | 10              | 0              | 100                  | 99,99                                |
| 8-PSK               | 10                             | 10              | 0              | 100                  | 99,92                                |
| 2-FSK               | 10                             | 10              | 0              | 100                  | 100,00                               |
| 4-FSK               | 10                             | 10              | 0              | 100                  | 100,00                               |
| GFSK                | 10                             | 10              | 0              | 100                  | 99,99                                |
| <b>Total/global</b> | <b>90</b>                      | <b>90</b>       | <b>0</b>       | <b>100</b>           | <b>99,98</b>                         |

Nota. La confianza media es el máximo softmax promediado sobre las ventanas de prueba de cada modulación (aproximadamente 500 por clase). Los valores se mantienen próximos a la unidad en todas las modulaciones evaluadas.

La Tabla 9 recoge las transmisiones clasificadas correctamente una vez superado el periodo inicial de estabilización: 90 aciertos y ningún error, esto es, una exactitud del 100 % en las condiciones evaluadas. La exactitud obtenida debe interpretarse dentro de las condiciones favorables de recepción del laboratorio y no debe generalizarse a otros dispositivos o escenarios sin evaluaciones adicionales.

Durante las diez rondas, la relación señal-ruido que mostraba la aplicación cambió un poco. Estuvo entre 17 y 24 decibelios. El valor más bajo se vio cuando se transmitió en AM-DSB. En cambio, el valor más alto se vio cuando se transmitió en 4-FSK. La aplicación no guardó los datos de la relación señal-ruido de manera automática. Así que los intervalos para cada tipo de transmisión se calcularon de nuevo, basados en lo que se vio durante las pruebas.

### Tabla 10

Intervalos aproximados de SNR durante las transmisiones OTA

| <b>Modulación</b>       | <b>SNR mínimo aproximado (dB)</b> | <b>SNR máximo aproximado (dB)</b> | <b>Punto medio del intervalo (dB)</b> |
|-------------------------|-----------------------------------|-----------------------------------|---------------------------------------|
| AM-DSB                  | 17,0                              | 22,8                              | 19,9                                  |
| OOK                     | 17,8                              | 23,1                              | 20,5                                  |
| 16-QAM                  | 18,9                              | 23,6                              | 21,3                                  |
| BPSK                    | 17,3                              | 22,4                              | 19,9                                  |
| QPSK                    | 18,0                              | 23,2                              | 20,6                                  |
| 8-PSK                   | 18,5                              | 23,7                              | 21,1                                  |
| 2-FSK                   | 17,6                              | 22,7                              | 20,2                                  |
| 4-FSK                   | 18,7                              | 24,0                              | 21,4                                  |
| GFSK                    | 17,9                              | 23,4                              | 20,7                                  |
| <b>Intervalo global</b> | <b>17,0</b>                       | <b>24,0</b>                       | <b>20,5</b>                           |

Nota. Los límites presentados corresponden a una reconstrucción aproximada basada en los valores visualizados durante las diez rondas. Los decimales no representan estadísticos calculados a partir de un registro exportado. El punto medio corresponde al centro de cada intervalo estimado y no a la media aritmética de todas las ventanas procesadas.

Los puntos medios estimados se encontraron entre 19,9 y 21,4 dB. A pesar de las variaciones observadas, todas las transmisiones fueron clasificadas correctamente después del periodo transitorio. Dentro del intervalo aproximado de 17 a 24 dB no se observó una degradación de la exactitud estabilizada. Este resultado no sustituye una curva de exactitud en función del SNR y no permite establecer el desempeño del modelo en niveles inferiores a 17 dB.

**Tabla 11**

Comportamiento temporal observado durante los cambios de modulación

| <b>Indicador</b>                      | <b>Resultado</b>               |
|---------------------------------------|--------------------------------|
| Rondas completas                      | 10                             |
| Modulaciones por ronda                | 9                              |
| Transmisiones totales                 | 90                             |
| Duración de cada transmisión          | 20 s                           |
| Pausa entre transmisiones             | 10 s                           |
| Periodo transitorio máximo observado  | Aproximadamente 5 s            |
| Predicciones incorrectas transitorias | Ocasionales y no cuantificadas |
| Errores después de la estabilización  | 0                              |
| Confianza estabilizada                | Entre 99,8 % y 100 %           |
| SNR global observado                  | Entre 17 y 24 dB               |
| Exactitud estabilizada                | 100 %                          |

Nota. Las predicciones incorrectas transitorias no fueron contabilizadas como errores de transmisión porque desaparecieron antes del instante de evaluación estabilizada. Su frecuencia de aparición no fue registrada automáticamente y, por tanto, se reporta únicamente de forma cualitativa.

El periodo transitorio no corresponde a un tiempo de inferencia de cinco segundos. La llamada individual al modelo se ejecuta sobre ventanas de 1 024 muestras, mientras que la estabilización observada corresponde al comportamiento del sistema completo después de cambiar la señal transmitida. Durante la transición pueden permanecer predicciones de la modulación anterior dentro del historial utilizado por el voto mayoritario. A medida que ingresan nuevas ventanas, el historial se actualiza y la clase predominante converge hacia la nueva modulación.

#### **4.7.4 Detección de señales WBFM comerciales**

La aplicación también fue utilizada para analizar señales WBFM comerciales dentro de la banda de radiodifusión FM. Al sintonizar frecuencias activas, el sistema detectó la presencia de señales FM y permitió visualizar su comportamiento espectral. Esta prueba resulta relevante porque WBFM no fue generada artificialmente por el transmisor del laboratorio, sino capturada desde estaciones reales.



La detección de huecos espectrales se basa en la identificación de canales con baja actividad o clasificados como “RUIDO/VACÍO”. Esta función permite relacionar la clasificación automática de modulaciones con una aplicación práctica de radio cognitiva, donde resulta necesario reconocer señales activas y posibles espacios disponibles dentro del espectro.



113

Funciones validadas en el *dashboard* C++

| Función del <i>dashboard</i> | Resultado esperado                                  | Evidencia recomendada                                  |
|------------------------------|-----------------------------------------------------|--------------------------------------------------------|
| Captura con RTL-SDR          | Recepción de muestras I/Q en tiempo real            | Captura de pantalla del <i>dashboard</i> activo.       |
| Clasificación AMC            | Identificación de la clase de modulación            | Captura mostrando clase y confianza.                   |
| Visualización PSD            | Representación espectral de la señal recibida       | Captura con gráfica de PSD.                            |
| Visualización I/Q            | Representación de constelación o muestras complejas | Captura con diagrama I/Q.                              |
| Detección WBFM               | Reconocimiento de estaciones FM comerciales         | Captura en banda 88–108 MHz.                           |
| Clase NOISE                  | Identificación de canal vacío o ruido               | Captura en frecuencia sin señal útil.                  |
| Huecos espectrales           | Identificación de frecuencias libres                | Captura de canales marcados como libres o disponibles. |

## 4.8 Caracterización operativa y limitaciones de medición

### 4.8.1 Plataforma de ejecución

La aplicación final se ejecutó en una computadora equipada con un procesador Intel Core i5-2450M con frecuencia base de 2,50 GHz, dos núcleos físicos y cuatro procesadores lógicos. El sistema disponía de aproximadamente 5,9 GB de memoria utilizable. La inferencia se realizó mediante ONNX Runtime utilizando el proveedor de ejecución en CPU. El receptor RTL-SDR operó con una tasa de muestreo de 2 MSPS y el modelo procesó ventanas de 1 024 muestras I/Q. Durante el despliegue no se utilizó una GPU dedicada.

**Tabla 13**

Características del equipo utilizado para ejecutar la aplicación

| Componente      | Especificación      |
|-----------------|---------------------|
| Procesador      | Intel Core i5-2450M |
| Frecuencia base | 2,50 GHz            |
| Núcleos físicos | 2                   |

|                        |                        |
|------------------------|------------------------|
| Procesadores lógicos   | 4                      |
| Caché L1               | 128 kB                 |
| Caché L2               | 512 kB                 |
| Caché L3               | 3 MB                   |
| Memoria utilizable     | Aproximadamente 5,9 GB |
| Receptor               | RTL-SDR                |
| Transmisor de prueba   | Pluto+ SDR             |
| Tasa de muestreo       | 2 MSPS                 |
| Tamaño de ventana      | 1 024 muestras I/Q     |
| Motor de inferencia    | ONNX Runtime           |
| Proveedor de ejecución | CPU                    |

#### 4.8.2 Uso observado de CPU y memoria

El consumo de recursos fue observado mediante el Administrador de tareas de Windows mientras la aplicación se encontraba en ejecución. En la captura realizada, el proceso amc\_ota utilizó 29,6 % de CPU y 24,1 MB de memoria. La actividad de disco y red atribuida al proceso fue de 0 MB/s y 0 Mbps, respectivamente. En el mismo instante, el sistema completo presentó entre 43 % y 46 % de utilización de CPU y entre 92 % y 93 % de memoria ocupada.

**Tabla 14**

Uso de recursos observado durante la ejecución de amc\_ota

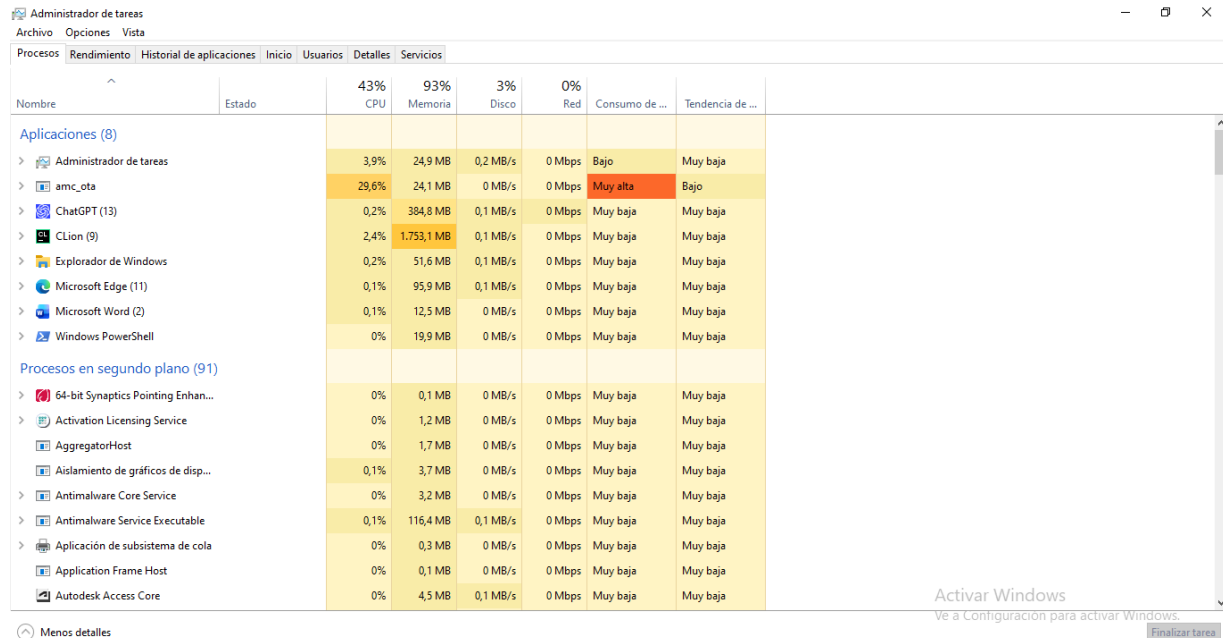
| Métrica                            | Valor observado          |
|------------------------------------|--------------------------|
| CPU instantánea de amc_ota         | 29,6 %                   |
| CPU típica estimada de amc_ota     | Aproximadamente 30 %     |
| Memoria instantánea de amc_ota     | 24,1 MB                  |
| Memoria típica estimada de amc_ota | Aproximadamente 24–25 MB |
| CPU total del sistema              | 43–46 %                  |
| Memoria total utilizada            | 92–93 %                  |
| Actividad de disco de amc_ota      | 0 MB/s                   |
| Actividad de red de amc_ota        | 0 Mbps                   |

Nota. Los valores corresponden a observaciones del Administrador de tareas de Windows.

No representan una media, un máximo o una desviación calculados mediante un registro automático durante toda la ejecución.

Los resultados muestran que la aplicación mantuvo un consumo reducido de memoria y una utilización moderada de CPU en el equipo evaluado. La elevada utilización global de memoria

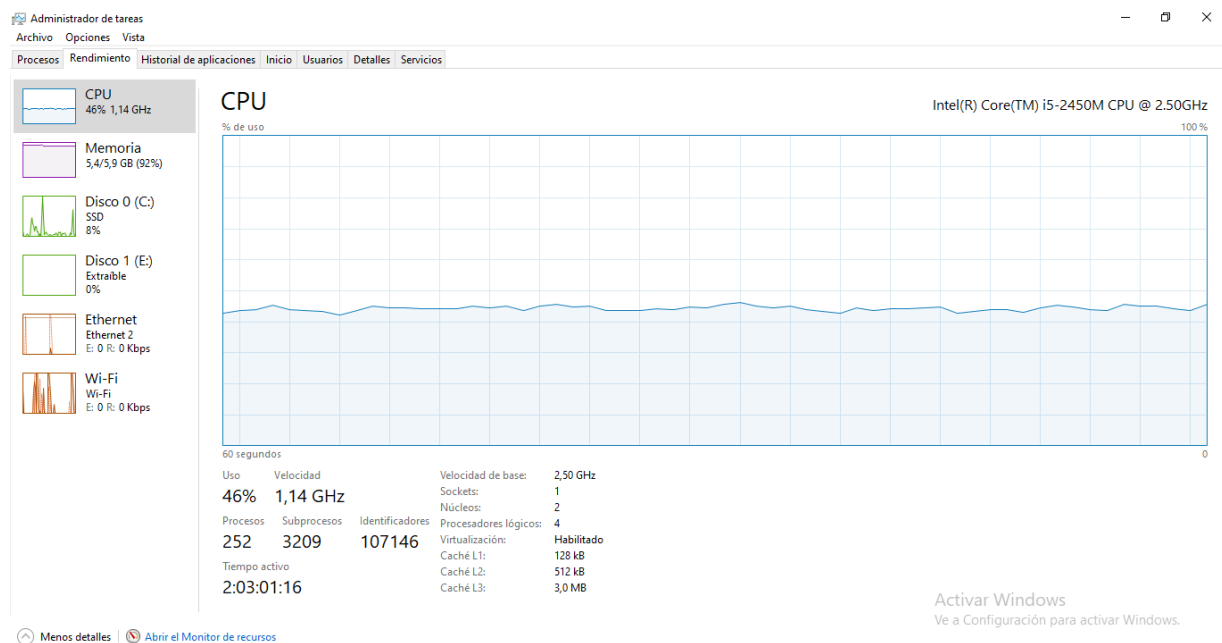
no fue ocasionada principalmente por amc\_ota, debido a que la aplicación utilizaba solamente 24,1 MB. Durante la prueba permanecían abiertos otros programas, entre ellos CLion, que utilizaba aproximadamente 1 753 MB.



| Nombre                                | Estado | 43% CPU | 93% Memoria | 3% Disco | 0% Red | Consumo de ... | Tendencia de ... |
|---------------------------------------|--------|---------|-------------|----------|--------|----------------|------------------|
| <b>Aplicaciones (8)</b>               |        |         |             |          |        |                |                  |
| Administrador de tareas               |        | 3,9%    | 24,9 MB     | 0,2 MB/s | 0 Mbps | Bajo           | Muy baja         |
| amc_ota                               |        | 29,6%   | 24,1 MB     | 0 MB/s   | 0 Mbps | Muy alta       | Bajo             |
| ChatGPT (13)                          |        | 0,2%    | 384,8 MB    | 0,1 MB/s | 0 Mbps | Muy baja       | Muy baja         |
| CLion (9)                             |        | 2,4%    | 1.753,1 MB  | 0,1 MB/s | 0 Mbps | Muy baja       | Muy baja         |
| Explorador de Windows                 |        | 0,2%    | 51,6 MB     | 0,1 MB/s | 0 Mbps | Muy baja       | Muy baja         |
| Microsoft Edge (11)                   |        | 0,1%    | 95,9 MB     | 0,1 MB/s | 0 Mbps | Muy baja       | Muy baja         |
| Microsoft Word (2)                    |        | 0,1%    | 12,5 MB     | 0 MB/s   | 0 Mbps | Muy baja       | Muy baja         |
| Windows PowerShell                    |        | 0%      | 19,9 MB     | 0 MB/s   | 0 Mbps | Muy baja       | Muy baja         |
| <b>Procesos en segundo plano (91)</b> |        |         |             |          |        |                |                  |
| 64-bit Synaptics Pointing Enhan...    |        | 0%      | 0,1 MB      | 0 MB/s   | 0 Mbps | Muy baja       | Muy baja         |
| Activation Licensing Service          |        | 0%      | 1,2 MB      | 0 MB/s   | 0 Mbps | Muy baja       | Muy baja         |
| AggregatorHost                        |        | 0%      | 1,7 MB      | 0 MB/s   | 0 Mbps | Muy baja       | Muy baja         |
| Aislamiento de gráficos de disp...    |        | 0,1%    | 3,7 MB      | 0 MB/s   | 0 Mbps | Muy baja       | Muy baja         |
| Antimalware Core Service              |        | 0%      | 3,2 MB      | 0 MB/s   | 0 Mbps | Muy baja       | Muy baja         |
| Antimalware Service Executable        |        | 0,1%    | 116,4 MB    | 0,1 MB/s | 0 Mbps | Muy baja       | Muy baja         |
| Aplicación de subsistema de cola      |        | 0%      | 0,3 MB      | 0 MB/s   | 0 Mbps | Muy baja       | Muy baja         |
| Application Frame Host                |        | 0%      | 0,1 MB      | 0 MB/s   | 0 Mbps | Muy baja       | Muy baja         |
| Autodesk Access Core                  |        | 0%      | 4,5 MB      | 0,1 MB/s | 0 Mbps | Muy baja       | Muy baja         |

**Figura 36. Consumo instantáneo de CPU y memoria del proceso amc\_ota durante la ejecución.**

En la Figura 36 se muestra que el proceso amc\_ota utilizó 29,6 % de CPU y 24,1 MB de memoria durante el instante capturado. La ocupación global de memoria estuvo influida por otros programas abiertos.



**Figura 37. Características del procesador y utilización general del equipo durante la ejecución.**

La Figura 37 presenta las características del procesador Intel Core i5-2450M y la utilización general del equipo durante un intervalo visual de 60 segundos. En el instante final se observó un 46 % de uso total de CPU y una velocidad dinámica de 1,14 GHz, diferente de la frecuencia base nominal de 2,50 GHz.

#### 4.8.3 Operación continua y actualización

Durante las pruebas funcionales, la aplicación realizó captura, procesamiento, clasificación y actualización del *dashboard* de forma continua. La interfaz fue limitada por diseño a aproximadamente 30 cuadros por segundo para controlar el consumo de recursos. Esta verificación confirma el funcionamiento operativo del prototipo, pero no proporciona una medición aislada del tiempo de inferencia ni de la latencia exacta desde la captura hasta la presentación.

#### 4.8.4 Continuidad de adquisición y pérdida de muestras

La aplicación incorpora una cola acotada para la recepción asíncrona y un contador interno de bytes descartados cuando el consumidor no procesa los datos con suficiente rapidez. En la versión utilizada para las pruebas, este contador no fue expuesto en el *dashboard* ni almacenado en un archivo de registro; por ello, no se dispone de una medición cuantitativa de pérdidas de muestras. Durante las pruebas no se observaron interrupciones evidentes en la interfaz ni en la reproducción de audio WBFM, pero esta observación cualitativa no permite afirmar que la pérdida haya sido exactamente cero.

#### **4.8.5 Limitaciones de la medición temporal**

El alcance del presente trabajo se definió en torno al diseño, la implementación y la validación funcional del prototipo de clasificación. En coherencia con dicho alcance, la aplicación desarrollada se orientó a la captura, la clasificación y la visualización en tiempo real, y no incorporó una capa de instrumentación temporal dedicada. Por este motivo, la implementación no incluyó marcas de tiempo internas para medir de forma aislada la inferencia ONNX, el preprocesamiento, la actualización del estado y la presentación gráfica, y no se reportan tiempos medios, desviaciones estándar, ventanas procesadas por segundo ni latencia total desde la captura hasta la presentación.

No obstante, el comportamiento observado permite establecer una cota cualitativa del desempeño temporal. La interfaz se mantuvo fluida y limitada por diseño a aproximadamente 30 cuadros por segundo, la reproducción de audio WBFM se realizó de forma continua y no se apreciaron bloqueos ni acumulaciones perceptibles durante las pruebas prolongadas. Estos indicios son consistentes con un procesamiento por ventana suficientemente rápido para la operación en tiempo real, aunque no sustituyen una caracterización temporal instrumentada. La incorporación

de telemetría persistente para registrar estas métricas de forma cuantitativa se plantea como trabajo futuro, en concordancia con las recomendaciones formuladas al final de este trabajo.

#### **4.9 Discusión general de resultados**

Los resultados indican que la red neuronal convolucional de una dimensión residual, llamada CNN-1D, es capaz de aprender características muy útiles de las ventanas I/Q que se encuentran en el conjunto de datos. La matriz de confusión muestra que la precisión es perfecta cuando se prueba con el conjunto interno de datos de prueba. Sin embargo, se debe ser cuidadoso al interpretar este resultado. Esto se debe a que la división de los datos se realizó por ventanas y es posible que existan relaciones entre ventanas que provienen de la misma captura de datos.

La integración del modelo en C++ permitió comprobar funcionalmente la captura, el procesamiento, la clasificación y la visualización. No obstante, la ausencia de instrumentación temporal impide presentar valores concluyentes de latencia, capacidad de procesamiento y pérdida de muestras.

El aumento de datos incluyó perturbaciones de ruido entre 5 y 30 dB. La curva de exactitud en función del SNR (sección 4.6.2) mostró un buen desempeño cuando el SNR estaba por encima de 5 dB. Pero cuando el SNR era menor, la exactitud disminuyó considerablemente. El umbral de 0,65 se estableció de forma heurística. No se usó una optimización formal para definirlo. Estas condiciones limitan las afirmaciones sobre robustez y sobre el reconocimiento en conjunto abierto.

Los resultados demuestran que el prototipo es efectivo en los laboratorios, pero no se puede afirmar que funcione del mismo modo en otros aparatos o entornos sin realizar más ensayos.

Un profesional puede identificar patrones en las gráficas de la señal. El sistema proporciona asistencia en aspectos como la automatización y la supervisión de los datos: analiza continuamente

la señal, estima su nivel de confianza, compara canales e identifica posibles huecos espectrales. El sistema no sustituye la opinión del profesional, sino que puede ayudar tanto a personas sin conocimientos técnicos como a expertos.

Se realizaron diez pruebas adicionales que demostraron los resultados iniciales: se transmitieron 90 señales y se recibieron 90 respuestas correctas luego de un ajuste inicial, con una precisión del 100 % y una confianza alta (más de 0,998). En cuanto al diseño, las pruebas utilizaron señales diferentes, aunque con el mismo *hardware* y ambiente de laboratorio. Esta serie probó el funcionamiento del dispositivo bajo estas condiciones, pero no es adecuada para otros dispositivos o ambientes.

Se encontró un error de predicción durante los primeros 5 segundos. La clasificación se estabilizó posteriormente, sin hallarse fallos persistentes. El hallazgo indica que hay que diferenciar el comportamiento de una ventana temporal de la conclusión definitiva obtenida con base en el voto mayoritario.

El SNR se mantuvo aproximadamente entre 17 y 24 dB. Dentro de este intervalo no se observó una reducción de la exactitud estabilizada.

La aplicación utilizó el 29,6 % de la capacidad de procesamiento de la computadora y ocupó 24,1 megabytes de memoria cuando se la observó. Esto muestra que la aplicación puede funcionar bien en una computadora con un procesador de dos núcleos que no tiene una unidad de procesamiento gráfico propia. Sin embargo, estos valores no nos dicen todo sobre cómo funciona la aplicación con el tiempo.

#### **4.10 Limitaciones observadas**

Los resultados son buenos, pero se debe tener en cuenta que el rendimiento del sistema puede variar mucho dependiendo de dónde se realicen las pruebas. En este caso, las pruebas se

llevaron a cabo en una habitación especial con condiciones muy controladas y con muy poca distancia entre el transmisor y el receptor. Entonces, los resultados que se obtuvieron solo son válidos para ese lugar y el sistema puede funcionar de manera diferente en otros lugares.

Un lugar abierto y elevado reduce las reflexiones y hace que el camino directo sea el más importante. Sin embargo, en una ciudad con edificios, tráfico y personas en movimiento, hay muchos caminos posibles y se producen interferencias externas. Si se está cerca del mar, también se produce la reflexión sobre el agua.

Las señales se generaron con un transmisor y se capturaron con un receptor. Esto no significa que el sistema solo funcione con estos dispositivos. La arquitectura utilizada puede aprender patrones en las señales, independientemente de las características del *hardware* utilizado. Sin embargo, no se puede descartar completamente que existan variaciones al utilizar otros dispositivos SDR, diferentes ganancias, antenas, niveles de potencia o condiciones de canal. Esta posibilidad debe considerarse como una limitación no comprobada experimentalmente, por lo que sería necesario realizar pruebas adicionales con otros equipos y escenarios para confirmar el grado de generalización del modelo.

Asimismo, la división entre entrenamiento, validación y prueba se realizó a nivel de ventana y no a nivel de captura. Dado que ventanas adyacentes provenientes de una misma captura o transmisión están altamente correlacionadas, las métricas obtenidas sobre el conjunto de prueba deben interpretarse como una cota optimista del desempeño esperable frente a capturas completamente nuevas.

Otra limitación es que el sistema no reconoce bien señales que no conoce. Aunque se usó un umbral basado en la confianza máxima de la distribución softmax, esto es solo un primer paso para reconocer señales que no están en la lista. Para mejorar esto, se podrían probar métodos más

seguros, como ajustar la confianza, la energía de los logits o entrenar con clases que no están en la lista.

Aunque se hicieron diez pruebas con 90 transmisiones, siempre se usó el mismo transmisor, receptor, frecuencia, lugar y configuración. Los resultados muestran que el sistema funciona bien en este entorno, pero no se puede decir que funcione perfectamente en todas las condiciones.

Se observaron la confianza y la relación señal-ruido en pantalla, pero no se guardaban automáticamente después de cada transmisión. La confianza media y los rangos de relación señal-ruido para cada clase se calcularon a partir de lo que se observó en pantalla.

Tampoco se contabilizaron de forma automática las predicciones incorrectas del periodo transitorio. Se conoce que ocurrieron ocasionalmente y que desaparecieron dentro de aproximadamente cinco segundos, pero no se dispone de su frecuencia exacta.

#### **4.10.1 Caso de estudio: falsos positivos WBFM fuera de la banda comercial**

Durante las pruebas de barrido en bandas superiores, particularmente en rangos asociados a servicios móviles/IMT, se observó que algunas señales fuera del conjunto de entrenamiento eran clasificadas como WBFM con alta confianza. Sin embargo, estas señales no producían audio útil al ser demoduladas como FM y no correspondían al comportamiento esperado de una emisión WBFM.

Este comportamiento evidenció una limitación del clasificador de conjunto cerrado. Aunque el sistema incorpora un umbral de confianza para declarar señales desconocidas, dicho mecanismo no es suficiente cuando una señal fuera de distribución genera una salida softmax elevada para una clase conocida.

Para mitigar este problema se aplicó la validación DSP descrita en la sección 3.5.5. La corrección no consistió en prohibir WBFM fuera del rango comercial, ya que esto habría

introducido una regla artificial basada únicamente en frecuencia. En su lugar, se exigió evidencia física adicional mediante el discriminador FM, SNR, ancho ocupado y PAPR.

Después de incorporar esta validación, las predicciones WBFM fuera de la banda comercial que no presentaban comportamiento compatible con FM fueron reportadas como “SEÑAL DESCONOCIDA”. En cambio, dentro de la banda comercial FM se mantuvo la detección normal de estaciones reales, evitando afectar el funcionamiento principal del sistema.

#### **4.11 Trabajo futuro**

Aunque la validación DSP mejora el comportamiento frente a falsos positivos WBFM, el sistema sigue teniendo una limitación importante: el modelo fue entrenado con un conjunto finito de clases. Por tanto, señales no incluidas en el entrenamiento, como OFDM u otras formas de onda de comunicaciones modernas, pueden producir respuestas de alta confianza en clases conocidas.

Para mejorar esta solución de manera más efectiva, se debe trabajar con ejemplos desconocidos, introducir señales de modulación por desplazamiento de frecuencia con cuadratura y modificar el modo de selección del índice que tenga el mayor valor de softmax por otro modo de selección como los logits u OpenMax. Además, resultaría beneficioso implementar un control para monitorear la ejecución de la aplicación: el tiempo dedicado por ventana a una tarea, con su promedio y desviación estándar, el número de ventanas procesadas por cada segundo, el tiempo desde la captura hasta el inicio del proceso, el uso de la unidad de procesamiento central y la memoria disponible, y el número de muestras eliminadas. En cuanto al umbral, es necesario establecerlo mediante los niveles de rechazo de falsos positivos y falsos negativos.

El caso se documenta como observación experimental: el sistema se evaluó también con señales reales ajenas al conjunto de entrenamiento, lo que permitió identificar y corregir una debilidad del clasificador inicial.

## 4.12 Consideraciones finales del capítulo

En este capítulo se presentan los resultados de los experimentos del sistema que clasifica automáticamente las modulaciones OTA. Se descubrió que las señales que se capturan tienen características que se pueden distinguir tanto en el plano I/Q como en el dominio de la frecuencia. También el modelo llamado CNN-1D residual logró un buen desempeño en la prueba, lo que se ve en la matriz de confusión y en la distribución de confianza.

El modelo se integró en una aplicación de C++ que se utiliza para el monitoreo del espectro y se enlazó con la etapa de entrenamiento. Esto permitió comprobar cómo funciona el sistema cuando está en operación continua. Los resultados muestran que este enfoque es viable para clasificar las modulaciones que se capturan con un receptor de radio definida por *software* y aplicar esa clasificación al análisis de cómo se ocupa el espectro radioeléctrico.

## CONCLUSIONES

Se creó un entorno experimental SDR. Se usó para generar señales, transmitir las, capturarlas y comprobar su funcionamiento. El proceso dio como resultado un conjunto equilibrado de ventanas I/Q. Se incluyeron los efectos del *hardware* y del canal de transmisión.

Se empleó una arquitectura CNN-1D residual que procesa bloques de muestras I/Q de 1024 elementos. El modelo obtuvo un 100 % de precisión en la prueba interna. Esta cifra puede resultar bastante elevada: la partición consistió en bloques de ventanas en lugar de sesiones independientes de captura.

El modelo fue convertido a formato ONNX e integrado en un programa C++. La aplicación incorpora la adquisición de datos mediante RTL-SDR, el análisis espectral, la clasificación, la representación gráfica y una serie de funcionalidades de sensado. Esta combinación permitió probar la integridad de la solución en un entorno de laboratorio sin interrupciones.

La validación incluyó diez rondas completas y 90 transmisiones OTA. Después de un período de unos cinco segundos, todas las transmisiones fueron clasificadas correctamente. Se logró una exactitud estable del 100 % y una confianza de casi 99,9 % en condiciones de SNR entre 17 y 24 dB.

El entrenamiento usó ruido entre 5 y 30 dB. La evaluación de precisión según el SNR indica que el umbral de fiabilidad práctico está cerca de 5 dB. Por encima de ese valor, el clasificador supera el 99 %. Por debajo, baja a 40,98 % a 0 dB y a 3,16 % a -5 dB. Los dos niveles más bajos no fueron cubiertos por el aumento de datos.

El umbral de confianza de 0,65 es un criterio que se usa para rechazar ciertos casos. Se comprobó su compatibilidad con las clases conocidas en el conjunto de prueba cerrado. No se encontró que sea el mejor valor para señales que no estén en la distribución prevista.

La aplicación se usó en un procesador Intel Core i5-2450M con dos núcleos. No había GPU. El uso fue del 29,6 % de la CPU y 24,1 MB de memoria. Estos valores corresponden a capturas puntuales, no a promedios de toda la ejecución.

En las condiciones establecidas, se logró el objetivo principal de crear un modelo que clasifica modulaciones OTA de forma automática. Este modelo se basa en una red neuronal convolucional unidimensional residual y en receptores SDR. El objetivo era apoyar tareas de sensado espectral en un ambiente académico controlado.

Hubo predicciones incorrectas por un tiempo, pero con baja confianza, durante algunos cambios de modulación. Esas predicciones dejaron de ocurrir cuando se actualizaron los votos mayoritarios con las nuevas señales. Esto significa que el 100 % de los resultados muestran la clasificación final y no quiere decir que todas las ventanas iniciales estuvieran clasificadas con precisión.

## RECOMENDACIONES

Realizar futuras divisiones del *dataset* por sesión o transmisión completa para evitar que ventanas correlacionadas de una misma captura se distribuyan entre entrenamiento y prueba.

Para mejorar el modelo, es necesario probarlo con diferentes transmisores, receptores, antenas, distancias y situaciones de propagación. Esto ayudará a determinar si el modelo puede funcionar bien en diferentes escenarios.

También es importante ver cómo funciona el modelo en situaciones con señales de mala calidad. Se puede hacer registrando señales en condiciones reales y comparándolas con las que se obtienen con ruido artificial.

Además, se necesita crear un conjunto de señales desconocidas y probar métodos de rechazo más robustos. Se pueden incluir la calibración de confianza, la energía de logits, OpenMax o la exposición a valores atípicos. Esto ayudará a determinar si el modelo puede manejar señales que no ha visto antes y si el modelo puede rechazar señales que no son confiables.

Incorporar telemetría persistente para registrar el tiempo de inferencia, la latencia total, el uso de recursos, las ventanas procesadas y la pérdida de muestras.

Automatizar el registro de las transmisiones OTA para calcular los aciertos, los errores, la exactitud y la confianza a partir de repeticiones independientes.

## REFERENCIAS

- [1] Dobre, O., Abdi, A., Bar-Ness, Y., & Su, W. (2007). Survey of automatic modulation classification techniques: classical approaches and new trends. *IET Communications*, 1(2), 137-156. <https://doi.org/10.1049/iet-com:20050176>
- [2] Hameed, F., Dobre, O., & Popescu, D. (2009). On the likelihood-based approach to modulation classification. *IEEE Transactions On Wireless Communications*, 8(12), 5884-5892. <https://doi.org/10.1109/twc.2009.12.080883>
- [3] Abd-Elaziz, O. F., Abdalla, M., & Elsayed, R. A. (2023). Deep Learning-Based Automatic Modulation Classification Using Robust CNN Architecture for Cognitive Radio Networks. *Sensors*, 23(23), 9467. <https://doi.org/10.3390/s23239467>
- [4] Xiao, W., Luo, Z., & Hu, Q. (2022). A Review of Research on Signal Modulation Recognition Based on Deep Learning. *Electronics*, 11(17), 2764. <https://doi.org/10.3390/electronics11172764>
- [5] Wu, P., Sun, B., Su, S., Wei, J., Zhao, J., & Wen, X. (2020). Automatic Modulation Classification Based on Deep Learning for Software-Defined Radio. *Mathematical Problems In Engineering*, 2020, 1-13. <https://doi.org/10.1155/2020/2678310>
- [6] Liu, K., Gao, W., & Huang, Q. (2021). Automatic Modulation Recognition Based on a DCN-BiLSTM Network. *Sensors*, 21(5), 1577. <https://doi.org/10.3390/s21051577>
- [7] O'Shea, T. J., Roy, T., & Clancy, T. C. (2018). Over-the-Air Deep Learning based radio signal classification. *IEEE Journal Of Selected Topics In Signal Processing*, 12(1), 168-179. <https://doi.org/10.1109/jstsp.2018.2797022>

- [8] Clancy, C., Hecker, J., Stuntebeck, E., & O'Shea, T. (2007). Applications of Machine Learning to Cognitive Radio Networks. *IEEE Wireless Communications*, 14(4), 47-52. <https://doi.org/10.1109/mwc.2007.4300983>
- [9] Carrera, E. V., Gutiérrez, C., & Salvador, F. (2025). Modulation Classification in Wireless Communication Systems via Deep Learning. En *Lecture notes in networks and systems* (pp. 112-123). [https://doi.org/10.1007/978-3-031-87704-9\\_8](https://doi.org/10.1007/978-3-031-87704-9_8)
- [10] O'Shea, T. J., Corgan, J., & Clancy, T. C. (2016). Convolutional Radio modulation Recognition networks. *arXiv* (Cornell University). <https://doi.org/10.48550/arxiv.1602.04105>
- [11] O'Shea, T., & Hoydis, J. (2017). An Introduction to Deep Learning for the Physical Layer. *IEEE Transactions On Cognitive Communications And Networking*, 3(4), 563-575. <https://doi.org/10.1109/tccn.2017.2758370>
- [12] Saber, M., El Rharras, A., Saadane, R., Chehri, A., Hakem, N., & Kharraz, H. A. (2020). Spectrum Sensing for Smart Embedded Devices in Cognitive Networks using Machine Learning Algorithms. *Procedia Computer Science*, 176, 2404-2413. <https://doi.org/10.1016/j.procs.2020.09.311>
- [13] Le, Q. N., Huynh, T. Q., Ta, H. Q., Tan, P. V., & Nguyen, L. L. (2024). Modified receiver architecture in software-defined radio for real-time modulation classification. *EURASIP Journal On Advances In Signal Processing*, 2024(1). <https://doi.org/10.1186/s13634-024-01176-6>

- [14] Harper, C. A., Thornton, M. A., & Larson, E. C. (2023). Automatic Modulation Classification with Deep Neural Networks. *Electronics*, 12(18), 3962. <https://doi.org/10.3390/electronics12183962>
- [15] Huang, L., Pan, W., Zhang, Y., Qian, L., Gao, N., & Wu, Y. (2019). Data Augmentation for Deep Learning-based Radio Modulation Classification. *arXiv (Cornell University)*. <https://doi.org/10.48550/arxiv.1912.03026>
- [16] Bahuguna, A. S., Kumar, K., Pundir, Y. P., Alaknanda, N., & Bijalwan, V. (2021). A Review of Various Digital Modulation Schemes Used in Wireless Communications. *Algorithms For Intelligent Systems*, 561-570. [https://doi.org/10.1007/978-981-33-6307-6\\_56](https://doi.org/10.1007/978-981-33-6307-6_56)
- [17] Rojas, A. J., & Garcés, H. O. (2022). Signal-to-Noise Ratio Based Fault Detection and Identification. *Frontiers In Control Engineering*, 3. <https://doi.org/10.3389/fcteg.2022.806558>
- [18] GeeksforGeeks. (2025, 23 de julio). BPSK Binary Phase Shift Keying. GeeksforGeeks. <https://www.geeksforgeeks.org/electronics-engineering/bpsk-binary-phase-shift-keying>
- [19] Keim, R. (2019, 21 de enero). Understanding Quadrature Phase Shift Keying (QPSK) Modulation. *Technical Articles*. <https://www.allaboutcircuits.com/technical-articles/quadrature-phase-shift-keying-qpsk-modulation>
- [20] Link-Pp. (2025, 16 de septiembre). What is FSK Modulation and How Does It Work. <https://resources.l-p.com/glossary/fsk-modulation-what-it-is-how-fsk-works-digital-communication>

- [21] Jstraugh. (2021, 8 de diciembre). Do BPSK vs QPSK have the same performance? Signal Processing Stack Exchange. <https://dsp.stackexchange.com/questions/79490/do-bpsk-vs-qpsk-have-the-same-performance>
- [22] Libretexts. (2022, 1 de noviembre). 2.6: Modulación por desplazamiento de frecuencia, FSK. LibreTexts Español. [https://espanol.libretexts.org/Bookshelves/Ingenieria/Diseno\\_de\\_microondas\\_y\\_RF\\_I\\_-\\_Sistemas\\_de\\_radio\\_%28Steer%29/02%3A\\_Modulaci%C3%B3n/2.06%3A\\_Modulaci%C3%B3n\\_por\\_desplazamiento\\_de\\_frecuencia%2C\\_FSK](https://espanol.libretexts.org/Bookshelves/Ingenieria/Diseno_de_microondas_y_RF_I_-_Sistemas_de_radio_%28Steer%29/02%3A_Modulaci%C3%B3n/2.06%3A_Modulaci%C3%B3n_por_desplazamiento_de_frecuencia%2C_FSK)
- [23] Scholl, S. (2022). RF Signal Classification with Synthetic Training Data and its Real-World Performance. arXiv (Cornell University). <https://doi.org/10.48550/arxiv.2206.12967>
- [24] Wei, N. W., & Mendel, J. (2000). Maximum-likelihood classification for digital amplitude-phase modulations. IEEE Transactions On Communications, 48(2), 189-193. <https://doi.org/10.1109/26.823550>
- [25] Swami, A., & Sadler, B. (2000). Hierarchical digital modulation classification using cumulants. IEEE Transactions On Communications, 48(3), 416-429. <https://doi.org/10.1109/26.837045>
- [26] Wang, X., Liu, D., Zhang, Y., Li, Y., & Wu, S. (2022). A spatiotemporal multi-stream learning framework based on attention mechanism for automatic modulation recognition. Digital Signal Processing, 130, 103703. <https://doi.org/10.1016/j.dsp.2022.103703>
- [27] Sun, Y., & Ball, E. (2022). Automatic Modulation Classification Based on Machine Learning. En Mechanisms and machine science (pp. 53-67). [https://doi.org/10.1007/978-3-031-09909-0\\_5](https://doi.org/10.1007/978-3-031-09909-0_5)

- [28] E-Academy. (2023, 9 de octubre). Random Forest + Logistic regression + KNN + Naive Bayes + SVM with Python | DAY:24 [Video]. YouTube. <https://www.youtube.com/watch?v=120yivtrfnQ>
- [29] Liu, Z., Zhang, B., Luo, H., & He, H. (2025). Differentiated Embedded Pilot Assisted Automatic Modulation Classification for OTFS System: A Multi-Domain Fusion Approach. *Sensors*, 25(14), 4393. <https://doi.org/10.3390/s25144393>
- [30] Abdel-Moneim, M. A., El-Shafai, W., Abdel-Salam, N., El-Rabaie, E. M., & El-Samie, F. E. A. (2021). A survey of traditional and advanced automatic modulation classification techniques, challenges, and some novel trends. *International Journal Of Communication Systems*, 34(10). <https://doi.org/10.1002/dac.4762>
- [31] Zhang, Z., Luo, H., Wang, C., Gan, C., & Xiang, Y. (2020). Automatic modulation classification using CNN-LSTM based Dual-Stream structure. *IEEE Transactions On Vehicular Technology*, 69(11), 13521-13531. <https://doi.org/10.1109/tvt.2020.3030018>
- [32] Ramos, M. A., Camacho, R., Buitrago, P. A., Urda, R. D., & Restrepo, J. P. (2024). Software Defined Radio, a perspective from education. *Frontiers In Education*, 8. <https://doi.org/10.3389/feduc.2023.1228610>
- [33] Chen, C. Y. (2010). Reconfigurable Software Defined Radio and Its Applications. *Journal of Applied Science and Engineering*, 13(1), 29-38. <https://doi.org/10.6180/jase.2010.13.1.04>
- [34] Zheng, Q., Tian, X., Yu, L., Elhanashi, A., & Saponara, S. (2025). Recent Advances in Automatic Modulation Classification Technology: Methods, Results, and Prospects. *International Journal Of Intelligent Systems*, 2025(1). <https://doi.org/10.1155/int/4067323>

- [35] Meng, F., Chen, P., Wu, L., & Wang, X. (2018). Automatic Modulation Classification: a deep learning enabled approach. *IEEE Transactions On Vehicular Technology*, 67(11), 10760-10772. <https://doi.org/10.1109/tvt.2018.2868698>
- [36] CUAS Hub. (2023, 15 de agosto). Electronic Warfare- the Forgotten discipline - Cuashub.com. <https://cuashub.com/es/contenido/guerra-electronica-la-disciplina-olvidada/>

# ANEXO A. MANUAL DE INSTALACIÓN, CONFIGURACIÓN Y OPERACIÓN

## A.1 Objetivo

El manual describe la preparación, la compilación y la operación de la aplicación de clasificación automática de modulaciones OTA. Incluye la solución de errores frecuentes y las advertencias de seguridad en el uso del RTL-SDR y del Pluto+ SDR.

La aplicación está desarrollada en C++ y clasifica señales I/Q recibidas por RTL-SDR mediante un modelo CNN-1D residual exportado a ONNX. Es un prototipo académico, no un instrumento certificado de medición del espectro.

## A.2 Requisitos

### A.2.1 *Hardware*

- Computadora con sistema operativo Windows.
- Receptor RTL-SDR y una antena compatible con las frecuencias analizadas.
- Pluto+ SDR para las transmisiones controladas de prueba.
- Cables USB de datos en buen estado.

### A.2.2 *Software*

- CLion con un *toolchain* C++ y CMake configurados.
- ONNX Runtime y bibliotecas requeridas por el proyecto.
- Controlador compatible con RTL-SDR.
- Python, NumPy y PyADI-IIO para ejecutar el transmisor de prueba.

El Pluto+ SDR y Python solamente son necesarios para realizar transmisiones controladas. La aplicación receptora puede utilizarse sin el transmisor para observar señales presentes en el espectro.

### A.3 Archivos principales

```
C:\Tesis\  
├── main.cpp  
├── CMakeLists.txt  
├── test_transmitter.py  
├── amc_common.py  
├── rtl_sdr.dll  
├── pthreadVC2.dll  
├── amc_dataset\models\  
│   ├── amc_model.onnx  
│   └── amc_model.onnx.data
```

Los archivos `amc_model.onnx` y `amc_model.onnx.data` deben permanecer en la misma carpeta. El archivo con extensión `.data` contiene los pesos externos requeridos por el modelo.

### A.4 Compilación y ejecución mediante CLion

1. Abrir CLion y seleccionar Open.
2. Abrir la carpeta raíz `C:\Tesis`.
3. Esperar a que CLion detecte y cargue `CMakeLists.txt`.
4. Verificar que el *toolchain* de C++ esté configurado correctamente.
5. Seleccionar el objetivo `amc_ota` y el perfil de compilación utilizado durante el desarrollo.
6. Ejecutar Build Project y comprobar que la compilación termine sin errores.
7. Abrir la configuración de ejecución de `amc_ota`.
8. Verificar que los argumentos incluyan la ruta de `amc_model.onnx`.
9. Conectar el RTL-SDR y ejecutar `amc_ota` desde CLion.

Las bibliotecas dinámicas requeridas, incluyendo las asociadas con RTL-SDR y ONNX Runtime, deben estar disponibles en la carpeta del ejecutable o en una ubicación reconocida por Windows.

## A.5 Preparación del RTL-SDR

1. Conectar la antena al RTL-SDR.
2. Conectar el dispositivo a un puerto USB del equipo.
3. Esperar a que Windows reconozca el receptor.
4. Cerrar otros programas que puedan estar utilizando el RTL-SDR.
5. Ejecutar `amc_ota` desde CLion.

Si la aplicación indica que no puede abrir el RTL-SDR, se debe comprobar que el dispositivo esté conectado, que otro programa no lo tenga abierto, que el controlador sea compatible y que `rtlsdr.dll` esté disponible para el ejecutable.

## A.6 Configuración desde la interfaz

Después de iniciar la aplicación, el usuario puede modificar seis parámetros desde el panel lateral. Al presionar APLICAR, la aplicación actualiza la configuración del barrido y limpia el historial asociado a los parámetros anteriores.

**Tabla A1**

Parámetros configurables desde el *dashboard*

| Parámetro          | Unidad | Función                                 |
|--------------------|--------|-----------------------------------------|
| Frecuencia inicial | MHz    | Límite inferior del barrido             |
| Frecuencia final   | MHz    | Límite superior del barrido             |
| Paso               | MHz    | Separación entre frecuencias analizadas |
| Ganancia           | dB     | Ganancia aplicada al receptor RTL-SDR   |

|                  |    |                                                  |
|------------------|----|--------------------------------------------------|
| Sensibilidad SNR | dB | Umbral utilizado para detectar actividad         |
| Estancia         | s  | Tiempo configurado de observación por frecuencia |

Un paso menor incrementa la cantidad de frecuencias visitadas y la duración del ciclo. Una ganancia excesiva puede saturar el receptor, mientras que una ganancia insuficiente puede impedir la detección de señales débiles. La sensibilidad SNR debe ajustarse de acuerdo con el nivel de ruido del entorno.

La corrección PPM, la tasa de muestreo, el tamaño de ventana, el umbral softmax, el voto mayoritario y los parámetros internos de procesamiento no pueden modificarse desde el *dashboard*.

## **A.7 Operación del *dashboard***

### **A.7.1 Modo de barrido**

1. Ingresar la frecuencia inicial y final.
2. Establecer el paso entre canales.
3. Ajustar la ganancia y la sensibilidad SNR.
4. Definir el tiempo de estancia.
5. Presionar APLICAR.
6. Esperar varios ciclos antes de interpretar estados persistentes o huecos espectrales.

Para cada canal, la aplicación presenta la frecuencia, clase actual, clase estabilizada, confianza, SNR estimado, estado de ocupación y rol asignado.

### **A.7.2 Modo detalle**

Al seleccionar una frecuencia de la tabla se accede al modo detalle, donde se muestran la densidad espectral de potencia, la constelación I/Q, la clase detectada, la confianza y el SNR. Si una señal WBFM cumple las validaciones correspondientes, también puede reproducirse su audio.

Después de cambiar de frecuencia o modulación puede existir un periodo transitorio. Durante las pruebas, la clasificación se estabilizó dentro de aproximadamente cinco segundos. Una confianza elevada no garantiza que la señal pertenezca realmente a una clase conocida.

## **A.8 Ejecución del transmisor de prueba**

El transmisor se ejecuta desde PowerShell mientras la aplicación receptora permanece abierta y sintonizada en la misma frecuencia.

1. Conectar el Pluto+ SDR.
2. Abrir PowerShell en la carpeta C:\Tesis.
3. Ejecutar el comando de prueba.

```
python test_transmitter.py --rounds 10 --tx-time 20 --pause 10 --freq 408
```

El comando ejecuta diez rondas, nueve modulaciones por ronda y 90 transmisiones totales.

Cada modulación se transmite durante 20 segundos y es seguida por una pausa de 10 segundos.

Entre rondas se aplica un periodo adicional de enfriamiento de 40 segundos.

Para evaluar una transmisión se deben ignorar los primeros cinco segundos y utilizar posteriormente la clase estabilizada mediante voto mayoritario. Este intervalo evita contabilizar como error una predicción transitoria producida durante el cambio de señal.

## **A.9 Solución de errores frecuentes**

### **Tabla A2**

Errores frecuentes y acciones recomendadas

| Problema                       | Posible causa                      | Acción recomendada                           |
|--------------------------------|------------------------------------|----------------------------------------------|
| No abre el RTL-SDR             | Dispositivo desconectado u ocupado | Reconectar y cerrar otros programas SDR      |
| No se encuentra el modelo      | Ruta incorrecta                    | Revisar los argumentos de ejecución en CLion |
| Error al cargar ONNX           | Falta amc_model.onnx.data          | Colocar ambos archivos del modelo juntos     |
| La aplicación no inicia        | Biblioteca dinámica ausente        | Verificar las DLL requeridas                 |
| No aparece la transmisión      | Frecuencias TX y RX diferentes     | Configurar la misma frecuencia               |
| Siempre muestra ruido          | Señal débil o ganancia baja        | Revisar antena, ganancia y sensibilidad      |
| Señal saturada                 | Potencia o ganancia excesiva       | Reducir ganancia o aumentar la separación    |
| Clase errónea al cambiar señal | Periodo transitorio                | Esperar aproximadamente cinco segundos       |
| Pluto+ no conecta              | Dirección IP o conexión incorrecta | Verificar la conexión y URI del dispositivo  |
| Pluto+ se calienta             | Transmisión prolongada             | Detener la prueba y permitir enfriamiento    |
| WBFM sin audio                 | Señal no válida o error de salida  | Revisar frecuencia y dispositivo de audio    |
| La interfaz no abre            | Problema de OpenGL                 | Revisar el controlador gráfico               |

## A.10 Seguridad y advertencias

- Usar cables USB y conectores RF en buen estado.
- No hacer movimientos que puedan dañar los conectores del RTL-SDR ni del Pluto+ SDR.
- No cubrir el Pluto+ SDR cuando se está transmitiendo durante mucho tiempo.
- Detener la prueba si hay un calentamiento inusual.
- Distanciar las antenas lo suficiente para no sobrecargar el receptor.
- Detener la transmisión antes de quitar la antena o el Pluto+ SDR.

Las transmisiones se realizan con fines académicos y en condiciones autorizadas. Antes de activar el Pluto+ SDR conviene verificar frecuencia, potencia, ancho de banda y ubicación, y emplear la menor potencia que permita la captura, sin interferir servicios autorizados.

Identificar una frecuencia como hueco espectral no autoriza a transmitir en ella. La información de la interfaz es referencial y no sustituye la normativa ni las autorizaciones de la autoridad competente.

### **A.11 Cierre del sistema**

1. Finalizar test\_transmitter.py si continúa activo.
2. Confirmar que la transmisión haya terminado.
3. Cerrar la aplicación amc\_ota.
4. Desconectar el Pluto+ SDR.
5. Desconectar el RTL-SDR.
6. Guardar los equipos en un lugar seco y ventilado.