UNIVERSIDAD DE CONCEPCIÓN
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

IMPLEMENTACIÓN Y USO DE CLASIFICADORES DE ML PARA MFI EN FIBRAS ÓPTICAS

POR

Alberto Andrés Herrera Saldías

Memoria de Título presentada a la Facultad de Ingeniería de la Universidad de Concepción para optar al título profesional de Ingeniero Civil en Telecomunicaciones

Profesor Guía
Gabriel Saavedra M.

Enero 2025
Concepción (Chile)

"Para Domingo y Leticia"

Agradecimientos.

Sin lugar a duda esta memoria de título, a modo personal, fue el proceso más desafiante y desgastante que me tocó experimentar a lo largo de la difícil época universitaria. Es por esto por lo que quiero expresar mi sincero agradecimiento a todas las personas, familiares y conocidos, que siempre tuvieron una palabra de aliento durante mi estadía en la UdeC, y aquellos quienes desinteresadamente estuvieron dispuestos a ayudar cada vez que lo necesité.

De igual manera, quiero agradecer a todos mis compañeros con quienes compartí tanto en clases como en el auditorio Salvador Gálvez. A Don Migue, que me dió la oportunidad de trabajar en el auditorio y además de toda su ayuda brindada. Y en especial a mi amigo, Diego Paredes, con quien forjé lazos de amistad a lo largo de los años que traspasan lo universitario. También, agradecer al equipo Biomédica FC, equipo de interfacultades que me abrió las puertas y me permitió conocer grandes jugadores, pero por sobre todo grandes personas, en especial a quienes conforman la directiva. Gracias por enseñarme que más allá de la competencia, lo más importante es la unidad y el respeto mutuo. Con momentos de dulce y de agraz, me siento profundamente agradecido y honrado de ser parte de un excelente grupo humano.

Además, un agradecimiento a mi profesor guía, profesor Gabriel Saavedra, por su valiosa orientación, paciencia, compromiso y apoyo constante, los cuales fueron esenciales para poder llevar a cabo esta memoria de título.

No puedo dejar de agradecer a mi pareja, Javiera Cuevas por su incondicional apoyo y ayuda durante toda mi etapa universitaria. Durante todo el proceso su motivación y ánimo fueron fundamentales para superar los desafíos. Agradezco su paciencia y comprensión en los momentos más difíciles, cuando los desafíos parecían interminables, así como su capacidad para celebrar conmigo cada pequeño logro alcanzado. No solo has sido mi apoyo, sino también mi compañera en este viaje académico y personal. Gracias por ser mi fuente de fuerza y motivación y por estar a mi lado en cada paso de este proceso. Además, el agradecimiento a toda su familia, en especial al Team La Avenida (Grace y Vale), a sus padres Ramón y Graciela, a Caro y Seba, y como no, quien no podía faltar, al gran Raymond.

Finalmente, de manera muy especial agradezco a mis padres, Marianela y Alberto, quienes me han formado para ser quien soy hoy, por todo el sacrificio que han realizado para siempre otorgarme más de lo que merecí y por brindarme cosas que ellos no tuvieron la oportunidad de disfrutar. También por su motivación, apoyo y comprensión, a su manera, a lo largo de toda mi vida. Y de igual forma a mi hermana Valeria.

Resumen.

El creciente tráfico de datos y las limitaciones físicas y teóricas en la capacidad de transmisión de las fibras ópticas han motivado el desarrollo de técnicas avanzadas para mejorar la eficiencia de las redes ópticas. En este contexto, la identificación de formatos de modulación (MFI, por sus siglas en inglés) en conjunto con las nuevas técnicas de Machine Learning (ML) desempeña un papel crucial para habilitar redes ópticas autónomas, flexibles y de alta capacidad.

Este trabajo presenta la implementación y evaluación de tres modelos de redes neuronales profundas para la clasificación multiclase de señales ópticas: una red neuronal convolucional (CNN) personalizada, MobileNetV2 y ResNet50.

La base de datos utilizada incluye 1768 imágenes generadas a partir de diagramas de constelación de señales QAM y GS-QAM. Los modelos fueron entrenados utilizando técnicas de aprendizaje profundo en Google Colab, y evaluados con métricas como precisión, pérdida, matriz de confusión y F1-score. Los resultados muestran que la CNN personalizada obtuvo el mejor desempeño, alcanzando una precisión del 93% y una pérdida de 0.30, lo que destaca la eficacia de un diseño adaptado específicamente al conjunto de datos. MobileNetV2 logró un desempeño competitivo, con la misma precisión, pero una pérdida ligeramente mayor (0.32), posicionándose como una alternativa eficiente en términos de transferencia de aprendizaje y computación. Por otro lado, ResNet50 presentó el rendimiento más bajo, con una precisión del 84% y una pérdida de 0.67, lo que sugiere limitaciones en su configuración para este problema. El análisis identificó dificultades en la clasificación de clases QAM similares, atribuidas a las características visuales similares de las señales y a la sensibilidad al ruido.

Este estudio demuestra que las redes neuronales personalizadas son una solución robusta para problemas específicos, mientras que los modelos preentrenados requieren ajustes más profundos para optimizar su desempeño. Los hallazgos ofrecen una base para futuras investigaciones orientadas al desarrollo de clasificadores más precisos y eficientes en sistemas de comunicación óptica.

Abstract.

The growing data traffic and the physical and theoretical limitations in the transmission capacity of optical fibers have driven the development of advanced techniques to improve the efficiency of optical networks. In this context, the identification of modulation formats (MFI, for its acronym in English) along with new Machine Learning (ML) techniques plays a crucial role in enabling autonomous, flexible, and high-capacity optical networks.

This study presents the implementation and evaluation of three deep neural network models for multiclass classification of optical signals: a custom convolutional neural network (CNN), MobileNetV2, and ResNet50.

The database used includes 1,768 images generated from constellation diagrams of QAM and GS-QAM signals. The models were trained using deep learning techniques on Google Colab and evaluated using metrics such as accuracy, loss, confusion matrix, and F1-score. The results show that the custom CNN achieved the best performance, with an accuracy of 93% and a loss of 0.30, highlighting the effectiveness of a design specifically adapted to the dataset. MobileNetV2 achieved competitive performance with the same accuracy but a slightly higher loss (0.32), positioning itself as an efficient alternative in terms of transfer learning and computation. In contrast, ResNet50 showed the lowest performance, with an accuracy of 84% and a loss of 0.67, suggesting limitations in its configuration for this problem. The analysis identified challenges in classifying similar QAM classes, attributed to the visual similarities of the signals and sensitivity to noise.

This study demonstrates that custom neural networks are a robust solution for specific problems, while pre-trained models require further adjustments to optimize their performance. The findings provide a foundation for future research aimed at developing more accurate and efficient classifiers for optical communication systems.

Tabla de Contenidos.

Lista de Tablas.

Lista de Figuras.

1. Introducción.

En la actualidad el tráfico de datos y la cantidad de usuarios aumenta de manera exponencial, y los sistemas de comunicación enfrentan un desafío crítico en términos de capacidad. La creciente necesidad de transmitir grandes cantidades de datos ha dado paso a una constante evolución en las comunicaciones modernas, de manera que ofrezca soluciones que puedan hacer frente con el tráfico de datos requerido [1]. Es por esto, que las redes de fibra óptica se han posicionado como el medio de transmisión más importante para las comunicaciones de alta velocidad, dadas sus características como el ancho de banda y su baja atenuación [2]. Dado esto, la capacidad de las redes ópticas supera, tanto a las de radiofrecuencia, como a las de cables de cobre, por lo que la fibra óptica cumple un rol clave en las futuras tecnologías, como en el IoT, Data centers y el 5G [1].

Las redes ópticas actuales son estáticas con rutas transmisor-receptor fija, sin embargo, se espera que las futuras redes sean dinámicas, sin grillas y reconfigurables. En este contexto, la identificación de formato de modulación (MFI) se convierte en una necesidad para permitir el desarrollo de redes autónomas de próxima generación, puesto que permite construir redes adaptables, eficientes y flexibles donde el ancho de banda y el tipo de señal se determinan en función de las condiciones de la red.

Por esta situación, es que para abordar estos desafíos emerge como piedra angular la aparición y el desarrollo de la inteligencia artificial (AI) que es una rama de las ciencias computacionales encargada de estudiar modelos capaces de realizar actividades propias de los humanos [3]. De aquí, un área específica es el “Machine Learning” (ML), que se enfoca en el desarrollo de algoritmos que permiten a las computadoras aprender y tomar decisiones basadas en datos sin necesidad de ser programadas, es decir reemplaza funciones realizadas a mano mediante la iteración a través de arquitecturas de “Deep Learning” (DL), utilizando técnicas para identificar patrones para hacer predicciones o tomar decisiones. Estas técnicas de ML son soluciones prometedoras para optimizar recursos de la red y para MFI específicamente en redes de alta capacidad [4].

Por lo anterior, se centrará en la implementación de algoritmos de ML enfocados a la clasificación de formato de modulación.

2. Definición del problema

El término “Capacity Crunch” se aplicó por primera vez en el comienzo de la década pasada, y señaló el reconocimiento de que la capacidad de transmisión de una fibra óptica no es ilimitada.

El constante crecimiento de la demanda de tráfico de datos ha sido satisfecho en gran medida por la aparición de nuevas técnicas para aumentar la capacidad de una fibra óptica [27], la que a pesar de que es alta, no es ilimitada dado que enfrenta barreras físicas y teóricas. Este crecimiento en la demanda está impulsado por un número creciente de usuarios y aplicaciones que requieren velocidades de transmisión superiores, capacidades de datos más grandes y una menor latencia.

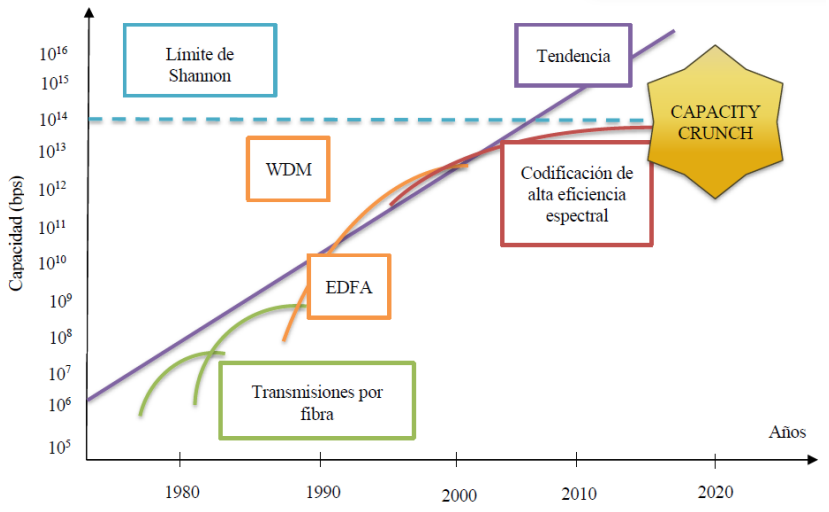


Fig. 1. Ilustración de “Capacity Crunch”. Fuente: [36].

La saturación de la capacidad de las fibras ópticas tiene varias implicaciones. En primer lugar, amenaza con degradar la calidad del servicio (QoS), lo que se traduce en mayores latencias, pérdida de datos, y una experiencia insatisfactoria para los usuarios finales. En segundo lugar, la congestión en las redes también afecta la eficiencia operativa, ya que los proveedores de servicios deben enfrentar costos crecientes para mantener o expandir la infraestructura, un desafío que se agrava en regiones donde los recursos son limitados. Por último, la imposibilidad de satisfacer la creciente demanda de transmisión podría frenar el desarrollo de tecnologías emergentes, como la inteligencia artificial distribuida y las

aplicaciones de realidad aumentada/virtual, que dependen de conexiones de alta velocidad y baja latencia.

Hasta hace unos años, la tasa de error de bits (*BER*), la tasa de error de símbolos (*SER*), entre otras, eran las métricas de rendimiento estándar en la comunidad de comunicaciones ópticas. Sin embargo, recientemente este paradigma ha ido cambiando y las tasas de información alcanzables (*AIR*) se están volviendo más populares. Los AIR indican el número de bits de información por símbolo que se pueden transmitir de manera confiable a través del canal y son el núcleo del célebre concepto de capacidad del canal de Shannon [43].

Más de 100 Tb/s es la capacidad máxima experimental conseguida [28][29]. Sin embargo, esto se considera que es casi la capacidad de una SMF debido a la potencia de entrada y los límites de Shannon no lineal [30]. Si la tasa de crecimiento actual continúa, la capacidad comercial puede alcanzar su límite en la siguiente década, por lo que para satisfacer el aumento de demanda se necesitan nuevas tecnologías que permitan ofrecer una mayor multiplicidad y total compatibilidad con las tecnologías nuevas [11].

Para enfrentar estos desafíos, el desarrollo de tecnologías avanzadas, específicamente de la IA y el ML. Estas técnicas permiten analizar grandes volúmenes de datos y patrones complejos en el comportamiento del canal, optimizando los recursos de red y adaptándose dinámicamente a las condiciones cambiantes. En particular, el ML desempeña un papel clave al trabajar con diagramas de constelación, herramientas visuales que representan la señal modulada y permiten diagnosticar problemas como ruido, dispersión y no linealidades. Los algoritmos de ML pueden analizar estos diagramas automáticamente, identificar patrones anómalos y ajustar parámetros para maximizar la capacidad de transmisión y la eficiencia espectral. Además, clasificadores basados en ML diferencian estados del canal o identifican características específicas de las señales, aplicando correcciones según sea necesario, lo que mejora la calidad del servicio y reduce errores de transmisión. En conjunto, el ML facilita una gestión inteligente y eficiente de las redes ópticas, asegurando su capacidad para satisfacer las demandas de tráfico actuales y futuras de manera sostenible.

2.1 Objetivos

2.2.1 Objetivo general

Implementar y evaluar modelos clasificadores de Machine Learning para la identificación de dos tipos específicos de formato de modulación utilizados en señales ópticas, utilizando técnicas de preprocesamiento de datos, entrenamiento de modelos y análisis de rendimiento.

2.2.2 Objetivos específicos

1. Diseñar una base de datos de interés para entrenar modelos de ML.
2. Implementar una red neuronal convolucional personalizada en Python para clasificar imágenes.
3. Implementar modelos preentrenados para clasificar imágenes.
4. Entrenar los modelos implementados.
5. Analizar los resultados del entrenamiento.

2.3 Alcances y Limitaciones

El alcance de este proyecto es la implementación y evaluación de tres clasificadores de imágenes, adaptados a un conjunto de datos personalizados de 1700 imágenes distribuidas en 8 clases. Además, se realiza una comparación entre ellas en términos de precisión, pérdida, entre otras métricas.

El modelo presenta como limitaciones el conjunto de datos utilizado que es limitado tanto en tamaño (1768 imágenes) como en diversidad, ya que se enfoca únicamente en 2 tipos de formatos de modulación, lo que restringe la capacidad del modelo para generalizar a otros conjuntos de datos o tareas de clasificación más amplias. Además, la dependencia de arquitecturas preentrenadas, sin un ajuste profundo, puede impedir una adaptación óptima a las características específicas del conjunto de datos en cuestión.

2.4 Metodología

La presente investigación se centra en la implementación de modelos de clasificación de imágenes utilizando librerías de código abierto especializadas en aprendizaje automático, como Keras y TensorFlow. La experimentación se realiza en Google Colab, donde se entrena un modelo de clasificación que es evaluado para determinar su desempeño. Los datos recolectados conforman una

base de datos que se utiliza para evaluar el rendimiento del modelo mediante diversas métricas, como la matriz de confusión, la precisión (accuracy) o el f1-score, proporcionando una visión detallada de su rendimiento en la tarea de clasificación. Además, se implementan arquitecturas de modelos preentrenados, de las cuáles se compararán los resultados obtenidos permitiendo un análisis de mayor profundidad de cada modelo en relación con el objetivo planteado en el estudio.

De manera específica con respecto a los objetivos, tenemos que:

1. Diseñar una base de datos para entrenar modelos de ML.

   Para cumplir este objetivo, se diseña una base de datos complementando dos partes de formatos de modulación de interés. En primer lugar, se generan constelaciones QAM utilizando MATLAB. Estas constelaciones representarán diferentes esquemas de modulación, lo que permitirá entrenar modelos de aprendizaje automático para identificar patrones en señales de comunicación óptica. En segundo lugar, se extraen datos de constelaciones de GS-QAM a partir de un paper científico de tal forma que las constelaciones coincidan con la cantidad de bits transmitidos previamente.

2. Implementar una red neuronal convolucional personalizada en Python para clasificar imágenes.

   Para este objetivo, se implementa una arquitectura de red neuronal convolucional (CNN) utilizando Tensorflow y Keras para clasificar imágenes de la base de datos. La arquitectura incluirá capas convolucionales, de agrupamiento y capas densas para la clasificación de imágenes. La implementación también incluirá la elección de hiperparámetros como el número de capas, el tamaño de los filtros, las funciones de activación, y la elección del optimizador.

3. Implementar modelos preentrenados para clasificar imágenes.

   Para este objetivo, se utiliza modelos preentrenados como MobileNetV2 y ResNet50, que son arquitecturas de redes neuronales profundas que ya han sido entrenadas en grandes conjuntos de datos como ImageNet. Se emplea el enfoque de transfer learning, que permite aprovechar el conocimiento adquirido por estos modelos, ajustándolos a las clases específicas de la base de datos. La implementación se realizará cargando los modelos preentrenados, y se adaptan según nuestro interés mediante la adición de capas de salida personalizadas y el ajuste de las últimas capas para que puedan clasificarse correctamente las imágenes de nuestro conjunto de datos. Este enfoque permite aprovechar características ya aprendidas para optimizar el proceso de

entrenamiento y obtener un desempeño superior en la clasificación de las 8 categorías.

4. Entrenar los modelos implementados.

Para entrenar los modelos implementados, se divide el conjunto de datos en dos partes: entrenamiento y validación. También se emplean técnicas para evitar el sobreajuste durante el proceso de entrenamiento. El trabajo práctico se lleva a cabo en Google Colab, aprovechando su entorno basado en la nube para realizar experimentos con diferentes arquitecturas de redes neuronales que incluye el uso de GPUs en sus servidores.

5. Analizar los resultados del entrenamiento.

Luego de entrenar los modelos, se evalúa su desempeño utilizando métricas clave como precisión, pérdida, matriz de confusión y otras métricas relacionadas con la clasificación. Se realiza un análisis de los resultados de cada modelo para comparar su efectividad en las tareas de clasificación, por ejemplo, mediante gráficos de precisión y pérdida.

3. Revisión Bibliográfica

En el área de las comunicaciones ópticas, la identificación precisa de formatos de modulación en las señales ópticas se ha vuelto esencial en un entorno caracterizado por la creciente demanda de ancho de banda y la expansión de aplicaciones de alta velocidad. El avance tecnológico que ha adquirido un protagonismo ineludible en este contexto es la aplicación de la AI y el ML. Esta revisión bibliográfica se sumerge en la intersección de la identificación de formato de modulación en fibra óptica y AI, así como en el papel crucial que desempeña ML este campo en constante evolución.

3.1 Comunicaciones ópticas

Las comunicaciones ópticas se refieren a la transmisión de información a través de señales de luz, generalmente en forma de pulsos de luz. Este enfoque se utiliza principalmente en las telecomunicaciones para transmitir datos a largas distancias a través de fibras ópticas con alta velocidad y capacidad de ancho de banda [5]. Físicamente, la fibra óptica consiste en un núcleo *("core")* cilíndrico y compacto de vidrio, rodeado por un revestimiento *("cladding")*, con la misma estructura, y de un recubrimiento *("buffer")*. La luz avanza por la fibra gracias al cumplimiento de la propiedad de la reflexión total de la luz entre el núcleo y el revestimiento [6].

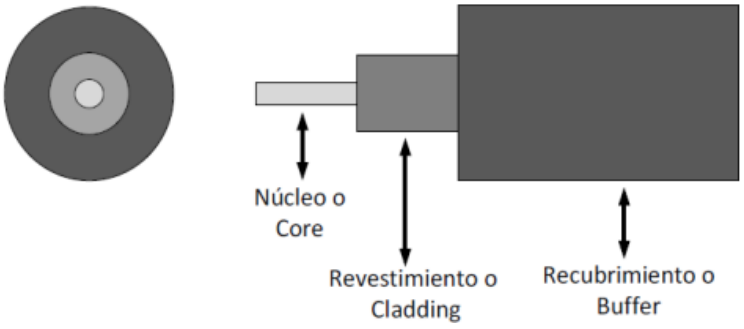


Fig. 2. Esquema fibra óptica. Fuente: [35].

Debido a sus características, en los últimos años la fibra óptica ha adquirido un rol relevante como medio de transmisión, como su gran ancho de banda (en el orden de los [THz]) y su baja atenuación (alrededor de 0.2 dB/km en 1550 [nm]), lo ha convertido en un medio ideal para transmisiones de larga distancia [2].

Estos sistemas se han utilizado a nivel mundial desde la década de los 80's y han revolucionado la tecnología relacionada con las telecomunicaciones [7]. Como cualquier otro sistema de comunicaciones, es necesario disponer de tres elementos fundamentales; un transmisor, un canal y un receptor, pero la diferencia radica en la longitud de onda de la señal empleada [8] ya que a diferencia de los sistemas de microondas (300 [MHz] – 300 [GHz]), la frecuencia de las portadoras ópticas es típicamente alrededor de los 200 [THz] [5]. En este tipo de comunicaciones existen diferentes fenómenos que afectan la señal, como la atenuación, ruido óptico, efectos lineales, como la dispersión cromática o intermodal, y efectos no lineales, como *"Stimulated Raman Scattering"* [9].

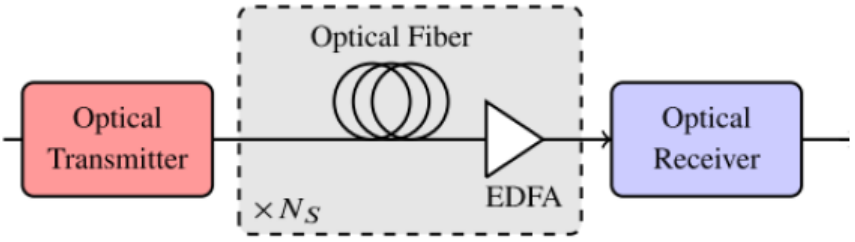


Fig. 3. Sistema de comunicación óptico. Fuente: [9].

La capacidad de un sistema óptico puede ser hasta 10.000 veces mayor que un sistema de microondas, debido al hecho de poseer portadoras de altas frecuencias ya que el ancho de banda de la onda moduladora pueda ser un porcentaje muy pequeño de la onda portadora. El uso de las comunicaciones ópticas se ha desarrollado principalmente para aplicaciones en telecomunicaciones, debido a la influencia de las redes de internet y telefónicas a nivel mundial [7]. Las investigaciones relacionadas con sistemas de comunicación de fibra óptica comenzaron alrededor del año 1975. Los progresos alcanzados en el lapso de 1975-2000 pueden agruparse en diferentes generaciones como se muestra en la Tabla 1.

La capacidad máxima transmisible a través de una SMF ha aumentado en gran cantidad mediante el uso de diversas tecnologías de multiplexación, como la multiplexación por división de tiempo o de longitud de onda (TDM o WDM), y tecnologías digitales coherentes [11]. La evolución de los sistemas WDM consiguen transmitir varios canales en paralelo utilizando cada uno de ellos una portadora diferente, multiplicando el ancho de banda de transmisión, pero requirió el desarrollo de nuevos componentes ópticos, que permitieran la multiplexación/demultiplexación de canales.

TABLA 1 Evolución de generaciones en redes ópticas. Fuente: [10].

Generación	Longitud de onda [nm]	Dispositivo	Característica	Tasa/Distancia
1º Generación	800	Led de GaAs	MMF	50 Mb/s – 10 km
2º Generación	1300	Laser semiconductor InGaAsP	MMF y SMF (0.5 dB/km)	2 Gb/s – 50 km
3º Generación	1550	Laser semiconductor monomodo	SMF Disp. Desplazada (0.2 dB/km)	4 Gb/s – 100 km
4º Generación	1550	EDFA, WDM, detección coherente	SMF de baja dispersión	10 Gb/s – 1500 km
5º Generación	1550	Solitón	Fibras de solitón	2.4 Gb/s – 12 000 km

Los aspectos como los formatos de modulación avanzados, el procesamiento digital de señales (DSP), o la codificación de corrección de errores (FEC) se convierten en los límites de la expansión de la capacidad en los sistemas WDM. Es por esto por lo que, durante la última década, las técnicas de detección coherente y DSP ha sido la piedra angular de los transceptores ópticos en los sistemas de comunicaciones ópticos [12]. Uno de los avances en relación con el transporte de información ha sido el estudio del fenómeno de los “solitones ópticos”, que es un pulso capaz de propagarse sin sufrir ningún tipo de dispersión a través de largas distancias. Con el desarrollo de todo esto, se produce una evolución de los sistemas de comunicaciones ópticas punto a punto a las redes ópticas [8].

Es importante mencionar que, en todos los sistemas de comunicación. la relación señal a ruido (SNR) mide la diferencia entre la potencia de la señal y la potencia del ruido en un canal de comunicación, y se expresa comúnmente en decibelios (dB). Una mayor SNR indica una señal más clara y menos afectada por el ruido. Por ejemplo, una SNR de 20 dB implica que la señal es 100 veces más potente que el ruido. Por otro lado, la tasa de error de bits (BER) cuantifica la cantidad de bits erróneos durante la transmisión, comparando los bits erróneos con el total transmitido. Un BER bajo refleja una mayor fiabilidad en la

transmisión; por ejemplo, un BER de 10^{-6} significa que uno de cada millón de bits está corrompido. Ambas métricas son esenciales para evaluar la calidad de las comunicaciones digitales y optimizar el rendimiento del sistema. [46]

3.2 Redes ópticas

Las redes ópticas se refieren a sistemas de comunicación que utilizan la luz para transmitir información a través de fibras ópticas con el objetivo de lograr velocidades de transmisión extremadamente altas y capacidades de ancho de banda significativas, en donde su objetivo es conseguir la calidad requerida a todos los canales de transmisión sobre una fibra óptica en altas distancias y con un alto grado de eficiencia. Estas redes ópticas, que son un conjunto de nodos interconectadas, aprovechan tecnologías avanzadas para maximizar la transmisión de datos, siendo esenciales para soportar el crecimiento exponencial de la demanda de ancho de banda en la era digital actual.

En investigaciones actuales [13], se han alcanzado capacidades WDM agregadas en SMF de hasta 115 Tbps y en MMF de hasta 10 Pbps. Debido a la gran demanda de los usuarios, por la necesidad de servicios de alta calidad a lo largo de estos años, han surgido oportunidades de desarrollo en las actuales redes de comunicaciones. Por ello, surge la necesidad de profundizar en nuevas redes de alta capacidad que soporten las demandas de ancho de banda y el avance exponencial del transporte de tráfico.

En este contexto, las redes WDM tiene un efecto positivo en los sistemas de telecomunicaciones ya que es apto para aumentar la capacidad de canal, y tienen como característica principal realizar funciones adicionales en el dominio óptico como. Todas estas ampliaciones son gracias a las prestaciones de una nueva capa dentro de la red llamada: capa óptica, que está ubicada entre la capa de acceso y la capa de internet. Además, existe una segunda clase de redes de segunda generación, llamadas las redes PON (Passive Optical Network). Este tipo de redes posee un carácter pasivo de acceso y de distribución, de tipo punto multipunto, es decir, la señal óptica es encaminada desde un punto a varios usuarios distintos.

El concepto de FTTx surge gracias a esta topología ya que se trata de las fibras que llegan hasta el interior de la casa, edificio, postes o hasta la central del proveedor de servicios. [36]

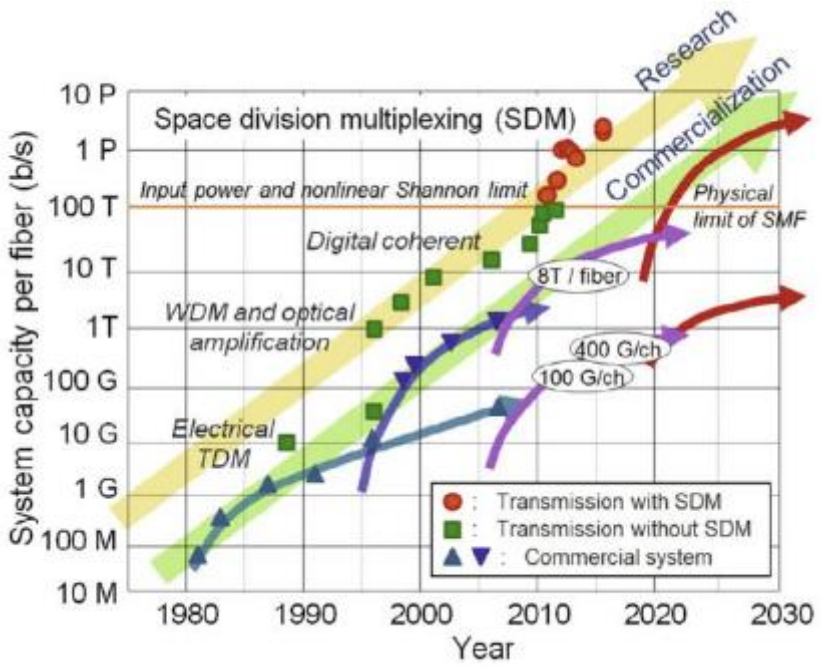


Fig. 4. Evolución de sistemas de comunicaciones ópticas. Fuente: [11].

La detección coherente brinda la capacidad de recuperar el campo óptico completo, la amplitud, la fase y el estado de polarización, lo que permite una alta eficiencia espectral y brinda la capacidad de mitigar las alteraciones de la transmisión mediante el uso de técnicas de DSP avanzadas [13]. DSP desempeña un papel relevante en este tipo de redes al mejorar la calidad y la eficiencia de transmisión, dado la adaptación en la modulación, y la compensación de dispersión y de efectos no lineales.

Por otra parte, un rol fundamental con respecto a este sistema de comunicación son los formatos de modulación utilizados, dentro de los cuales encontramos dos tipos: formatos de detección directa y formatos de detección coherente [1].

3.3 Evolución de las redes ópticas

Uno de los límites en el crecimiento y evolución de las redes ópticas no se encuentra solamente en las limitaciones de los elementos que componen las redes ópticas sino también el carácter dinámico que permitan conducir a modelos más autogestionados y evite desplazamientos de técnicos a las zonas de fallo, con o sin programación de trabajos

que requieran incluso cortes de tráfico. Además, las redes de gran recorrido generalmente emplean una topología de red de tipo malla ("mesh") que requiere un nuevo tipo de ROADM que permita un espectro flexible que pueda cambiarse desde la entrada hasta los puertos de salida. A medida que las demandas de velocidad de datos aumentan constantemente, para incrementar el rendimiento de la red, las redes metropolitanas y regionales también están evolucionando hacia la topología de las redes *"mesh"*, propuestas adaptativas y flexibles. Este tipo de redes flexibles es lo que se conoce como Redes Ópticas Elásticas o "*EON's*", que plantean el uso eficiente de recursos (formatos de modulación, espectro, rutas, entre otros) obteniendo grandes ventajas frente a las redes ópticas WDM.

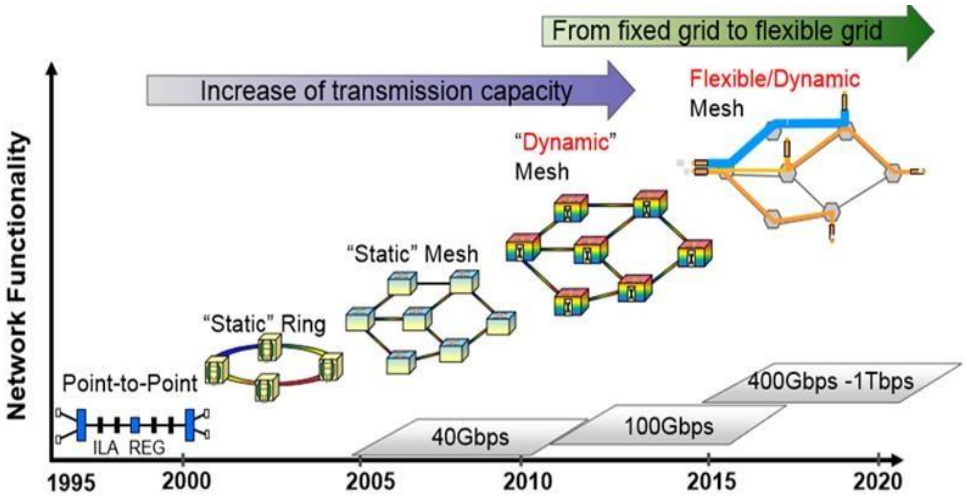


Fig. 5. Evolución de redes ópticas. Fuente: [13].

3.4 Formato de modulación

El formato de modulación es la técnica utilizada para modular la señal que se transmite a través de una fibra óptica. La modulación es el proceso de modificar información que se encuentra en forma de señales eléctricas de alta frecuencia, en señales de luz que se quiere transmitir. En el caso de las fibras ópticas, la modulación se utiliza para codificar la información en forma de pulsos de luz que se transmiten a través de la fibra. Existen diversas técnicas de modulación, como la modulación por intensidad, por fase y por frecuencia, que alteran diferentes propiedades de la luz para transmitir información. La modulación por intensidad ajusta la intensidad de la luz de acuerdo con la señal eléctrica,

mientras que la modulación por fase modifica la fase de la luz para lograr una transmisión más resistente a ruidos e interferencias. La modulación por frecuencia, aunque menos común, también puede utilizarse en sistemas específicos de multiplexación. Estas técnicas son fundamentales para mejorar la eficiencia en la transmisión de datos, permitiendo la comunicación a altas velocidades y largas distancias con mínima pérdida de señal y menor susceptibilidad a interferencias electromagnéticas. [47]

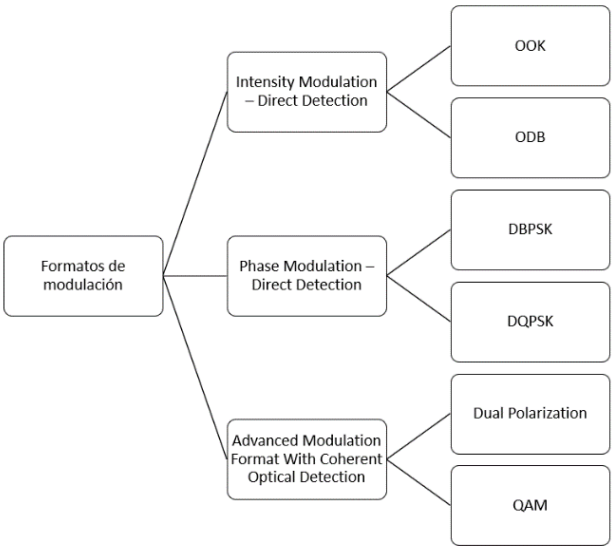


Fig. 6. Formatos de modulación. Fuente: [1].

3.4.1 Formatos de modulación clásicos

Los formatos de modulación clásicos, como la modulación de amplitud (AM), la modulación de frecuencia (FM) y la modulación de fase (PM), son fundamentales en las comunicaciones. En la AM, la amplitud de la señal portadora varía en función de la señal de información, siendo fácil de implementar, pero susceptible al ruido y las interferencias. La FM, que cambia la frecuencia de la portadora, es más resistente a interferencias y ruidos, aunque requiere equipos más complejos. La modulación de fase, por su parte, ajusta la fase de la portadora y es eficaz en comunicaciones digitales, pero exige una sincronización precisa. Estos formatos son esenciales en aplicaciones como la radio, la televisión y las comunicaciones digitales, cada uno con ventajas y desventajas según las necesidades del sistema de transmisión. [48]

Un formato que combina variaciones en la amplitud y la fase de una señal portadora es la modulación en amplitud en cuadratura (QAM), técnica ampliamente utilizada para transmitir

información. A diferencia de los métodos clásicos, QAM permite transmitir múltiples bits por símbolo, lo que aumenta la eficiencia espectral y la capacidad de la transmisión. Se utilizan diferentes combinaciones de amplitud y fase para representar una gran cantidad de símbolos, lo que la convierte en una técnica altamente eficaz en sistemas de comunicación de alta velocidad, como los utilizados en redes de datos y televisión digital. Sin embargo, QAM es sensible al ruido y requiere un procesamiento complejo en el receptor para la correcta decodificación de la señal. [49]

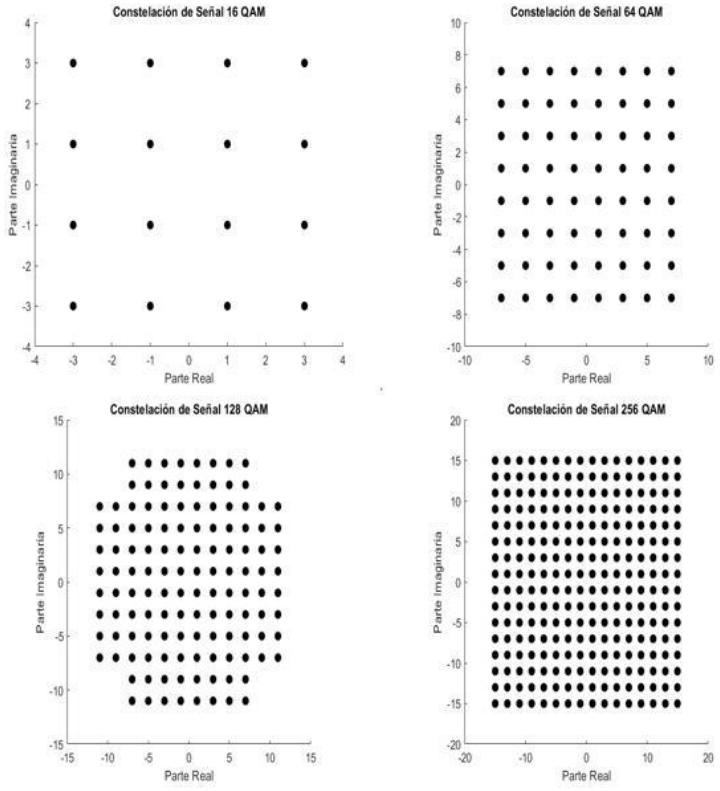


Fig. 7. Constelaciones de formatos clásicos. Fuente: [54].

3.4.2 Formatos de modulación optimizados

En sistemas avanzados de comunicación óptica, los formatos de modulación optimizados deben equilibrar la eficiencia espectral y la sensibilidad del receptor para reducir la interferencia entre canales adyacentes y mejorar la calidad de la señal, especialmente en enlaces de larga distancia. Para lograr esto, se emplean técnicas como moduladores avanzados, como el Mach-Zehnder Modulator, y formas de pulso como Nyquist o OFDM. Este tipo de modulación es esencial para incrementar la capacidad de las redes ópticas de próxima generación, capaces de manejar mayores volúmenes de datos a través de fibras ópticas. [50]

La modulación en amplitud en cuadratura con forma geométrica (GS-QAM) es una técnica avanzada

diseñada para optimizar las constelaciones de señales en canales con ruido blanco gaussiano aditivo (AWGN). A diferencia del QAM convencional, que utiliza una cuadrícula uniforme para los puntos de la constelación, GS-QAM optimiza la disposición de estos puntos, reduciendo el impacto del ruido y mejorando el rendimiento, especialmente en condiciones de baja relación señal-ruido. Esto permite obtener mejores factores Q y mejorar el rendimiento general en las comunicaciones ópticas [51]

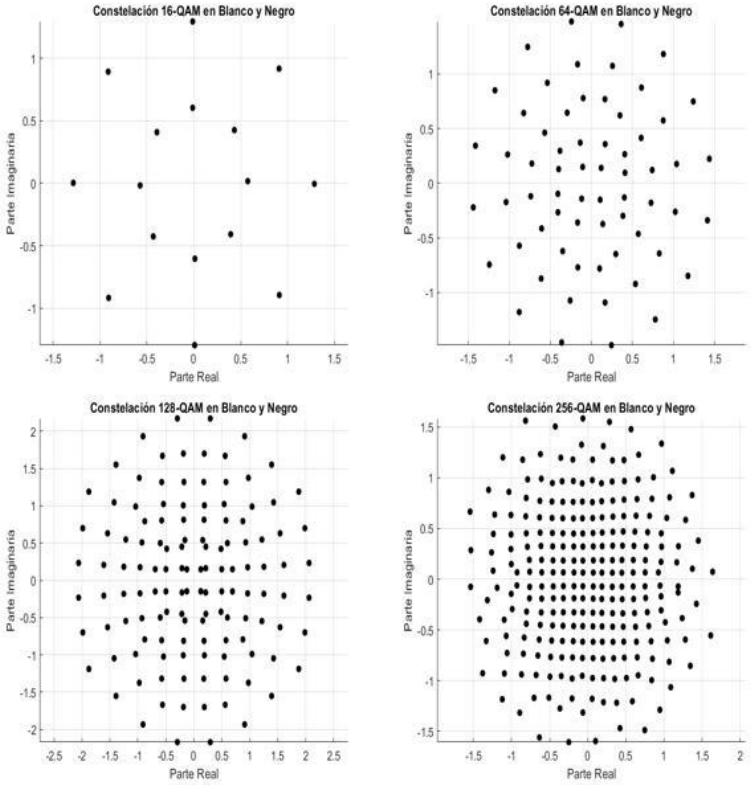


Fig. 8. Constelaciones de formatos optimizados. Fuente: [42].

3.5 Identificación del formato de modulación

El reciente desarrollo de tecnologías de receptores de comunicaciones ópticas basadas en detección coherente, combinado con digitalización de alta velocidad y DSP, es una revolución en el campo de las comunicaciones ópticas. Es por esto por lo que la identificación precisa del formato de modulación se transforma en un requisito clave, puesto que en un nodo de red cumple la función crítica de indicar algoritmos de demodulación optima, permite optimizar el uso de los recursos espectrales, mejorar la eficiencia de la red óptica, y además sirve como función de monitoreo de línea para la

gestión de asignaciones de canales y ancho de banda dentro del dominio óptico [37]. La Figura 9 proporciona una representación visual de características de distintos tipos de formatos de modulación.

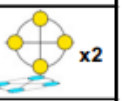
Modulation format	OOK	OOK-VSB	DQPSK	RZ-DPSK-3ASK	PM-DQPSK	OP-FDM-RZ-DQPSK	PM-QPSK	PM-OFDM-QPSK
coh. / noncoh.	noncoh.	noncoh.	noncoh.	noncoh.	noncoh.	noncoh.	coh.	coh.
Bits/symbol	1	1	2	2.5	2x2	2x2	2x2	2x2x2
Symbol Rate (Gbd)	112	112	56	44	28	28	28	14
Constellation								 x2
DWDM Grid (GHz)	200 	100 	100 	50 	50 	100 	50 	50 
Spectral Efficiency (bits/s/Hz)	0.5	1	1	2	2	1	2	2

Fig. 9. Características de formatos de modulación. Fuente: [39].

Las técnicas de AI y específicamente de ML, son soluciones prometedoras para agregar inteligencia a los nodos de la red óptica. ML se puede utilizar para el modelado de canales ópticos [40], especialmente en casos donde el modelado teórico es complejo. Esto permite el uso de formatos de modulación adaptables según las condiciones de transmisión, lo que a su vez requiere de técnicas de MFI adecuadas para identificar el tipo de formato de modulación en el receptor, de tal manera que pueda identificar el tipo de señal y realizar tareas de procesamiento digital de señales como demodulación, ecualización o filtrado. Los principales beneficios de utilizar ML encontramos la adaptabilidad en tiempo real, eficiencia, flexibilidad, reconfigurabilidad y mayor seguridad de red. Dado esto, será necesario transceptores flexibles que admiten múltiples velocidades de datos y múltiples formatos de modulación [41], de tal manera que puedan garantizar una demodulación adecuada. A continuación, se muestra un ejemplo de arquitectura de un receptor coherente con MFI.

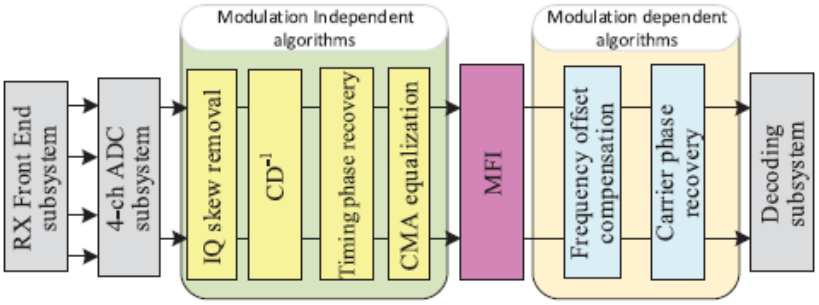


Fig. 10. Ejemplo de receptor coherente con MFI. Fuente: [1].

3.6 Inteligencia Artificial y Machine Learning

La "*Artificial Intelligence*" es una de las ramas de las ciencias computacionales encargada de estudiar modelos de cómputos capaces de realizar actividades propias de los seres humanos en base a dos de sus características primordiales: el razonamiento y la conducta [14]. Por otro lado, "*Machine Learning*" es una rama de la AI que se centra en el desarrollo de algoritmos que con una cantidad adecuada de datos y experiencias previas pueden ser entrenadas y aprender la relación entre la entrada y salida [1]. El proceso a partir del cual operan los algoritmos de ML puede ser divididos en tres partes.

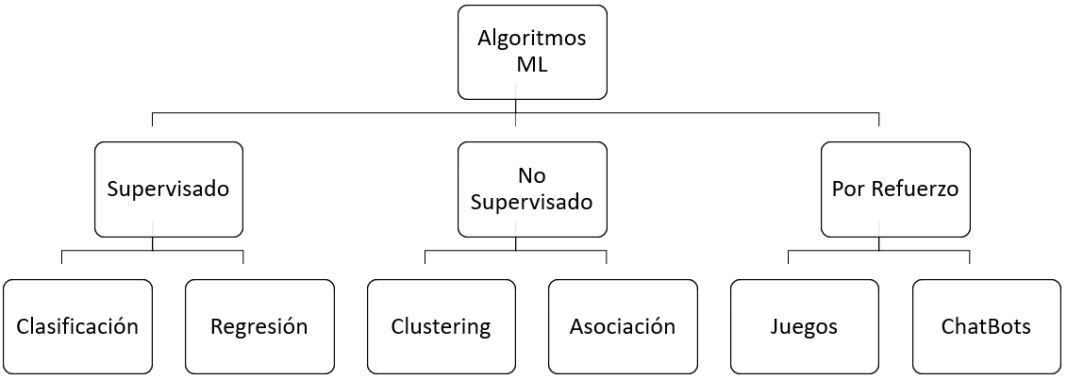


Fig. 11. Tipos y usos de algoritmos de Machine Learning. Fuente: [16].

En primer lugar, a través del proceso de decisión, el algoritmo hace predicciones o clasificaciones sobre un conjunto de datos. En segundo lugar, la función de error juzga el desempeño del proceso de decisión, según si sus predicciones o clasificaciones han sido correctas o incorrectas. Finalmente, el proceso de optimización del modelo se encarga de ajustar los pesos del modelo para reducir las discrepancias entre las estimaciones y los datos reales. Este proceso completo se reitera para obtener mejores resultados [15].

En general, estos algoritmos tienen como objetivo estimar una función desconocida que asigna entradas a salidas, las cuales representan los parámetros de un problema determinado y las soluciones a estos. Cuando pares de entradas y salidas están disponibles se trata de "Aprendizaje Supervisado", es decir, el sistema recibe datos ya clasificados y organizados de tal manera que la respuesta se conoce de antemano y se comparan las respuestas que entrega el algoritmo con el fin de mejorar los resultados. Aquí, existen dos tipos principales; Clasificación, que es utilizado para identificar o diferenciar entidades, y Regresión, que es utilizado para hacer predicciones. Por otra parte, cuando sólo la entrada o, sólo la salida está disponible, se trata de "Aprendizaje No Supervisado". En este caso, los datos no son etiquetados por lo que no pueden ser clasificados de antemano. Aquí, el Clustering, se encarga de encontrar características comunes en los datos y los agrupa, y la Asociación identifica la cercanía de los datos, los cuáles son los dos tipos principales.
También dentro de los algoritmos de ML, hay una clase especial de problemas que requiere una serie de decisiones o funciones para llegar a la solución final se puede aprender utilizando "Aprendizaje por refuerzo". En este aprendizaje luego de ser entrenado el algoritmo, interactúa con su entorno para recibir una evaluación de sus resultados y así mejorar el modelo [16].

Para MFI, que es considerado un problema de clasificación, el uso de técnicas de ML puede aportar muchos beneficios tanto para las redes actuales como para las redes futuras [1]. Los algoritmos de ML en el concepto de aprendizaje "end-to-end" reemplazan la extracción de funciones hechas a mano mediante la iteración a través de arquitecturas de *"Deep Learning"* para aprender automáticamente funciones directamente a partir de datos sin procesar.

El "Aprendizaje Profundo", una subsección de ML, es una familia de algoritmos que pretenden emular el aprendizaje humano con el fin de poder llegar a resolver distintos problemas que no han sido resuelto mediante métodos analíticos y se ha convertido en un tema importante que cuenta con aplicaciones para diversas áreas, como la medicina, ciberseguridad, reconocimiento visual, dentro de las que más destacan [17][18]. Este aprendizaje se basa en el concepto de "Red Neuronal Artificial" (ANN), inspiradas en la estructura biológica de las células neuronales del cerebro humano, las ANN presentan un modelo numérico de una determinada estructura para el aprendizaje y la toma de

decisiones, que comprende algunos componentes de procesamiento individuales, construido entre capas interconectadas [19][20].

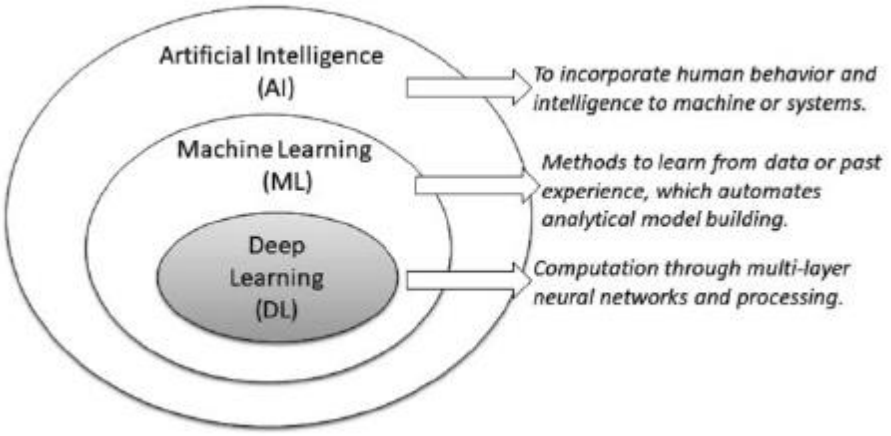


Fig. 12. Ilustración de AI, ML y DL. Fuente: [18].

El componente más básico de una ANN se llama neurona y se caracteriza por estar compuestos de capas de neuronas, que están conectadas en forma de cascada [1] y las cuales están formadas a su vez por pesos, quienes son los encargados de realizar las operaciones determinadas en la fase de entrenamiento [21]. Cada neurona se modela como una función de activación no lineal cuyas entradas se multiplican por pesos y se desplazan mediante coeficientes de sesgo, lo que proporciona al algoritmo ANN propiedades no lineales generales y permite aprender prácticamente cualquier función [1]. Cada capa única comprende neuronas ocultas que se utilizan para transformar los datos de entrada y determinar las salidas a las siguientes neuronas asociadas [22]. Su arquitectura básica es de tres capas; una capa de entrada, que es la encargada de recibir los datos, una capa oculta, en la cual se procesan los datos, y una capa de salida, donde se obtienen los resultados. Se ilustra en la fig. 13.

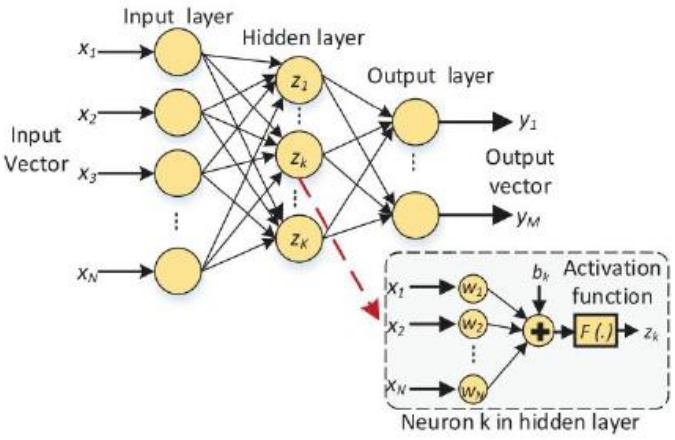


Fig. 13. Esquema de una ANN. Fuente: [1].

De manera similar a las ANN, las "Redes Neuronales Convolucionales" son un tipo de algoritmo donde se utilizan múltiples capas ocultas, pero a diferencia de la red anterior, las entradas suelen ser multidimensionales como las imágenes. Estas presentan la capacidad de que cada parte de la red se entrena para realizar una tarea, lo que reduce la cantidad de capas ocultas y por ende reduce su tiempo de entrenamiento.

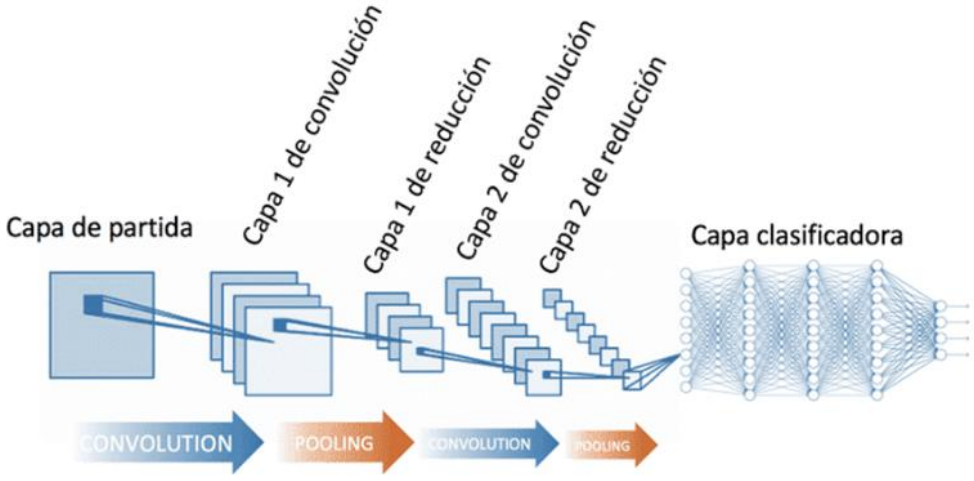


Fig. 14. Esquema de una CNN. Fuente: [21].

Su arquitectura consta de tres tipos de capas; capa de convolución, capa de agrupación y capa clasificadora, como muestra la fig. 14. En la primera capa cada se realizan operaciones de producto y suma entre la capa de entrada y un *"kernel"* o filtro que produce un mapa de características de dimensiones más reducida que la original. Luego, la capa de agrupación o reducción disminuye la dimensión de cada mapa de características utilizando un filtro, como un filtro promedio o un filtro máximo (*"maxpool"*). La salida de las capas anteriores se aplana y alimenta una red neuronal

completamente conectada que realiza las tareas asignadas [22].

Los algoritmos de CNN son útiles tanto para problemas de clasificación como de regresión.

3.7 Funciones de activación

Las funciones de activación son un componente esencial en las redes neuronales convolucionales (CNN), ya que transforman las salidas de las neuronas y determinan cómo se procesará la información a lo largo de las capas de la red. Las funciones de activación desempeñan un papel clave en la extracción de características relevantes y la decisión final de clasificación.

3.7.1 ReLU

La función ReLU es una de las funciones de activación más comunes en redes neuronales profundas debido a su simplicidad y eficiencia computacional. Devuelve un valor de 0 para cualquier entrada negativa, y para valores positivos devuelve el mismo valor de entrada. Esta función introduce no linealidad en la red, ayudando a resolver problemas complejos como el reconocimiento de imágenes. [57]

3.7.2 Max-Pooling

Aunque no es una función de activación como tal, el Max-Pooling es un procedimiento de reducción de dimensionalidad frecuentemente utilizado en las capas convolucionales para extraer características robustas y reducir el tamaño espacial de las representaciones. Este método opera seleccionando el valor máximo dentro de un área definida, reduciendo la resolución de las características mientras mantiene la información más importante. [58]

3.7.3 Softmax

La función Softmax se utiliza principalmente en las capas finales de clasificación de redes neuronales, particularmente cuando se trata de problemas de clasificación multiclase. Esta función convierte los valores de salida de la red en probabilidades normalizadas, escalando las entradas a un rango de 0 a 1. Softmax asegura que la suma de las probabilidades sea igual a 1, permitiendo interpretar la salida como una distribución de probabilidad. [59]

3.8 Redes neuronales profundas

Por otro lado, las Redes Neuronales Profundas (DNN) son una arquitectura de redes neuronales especializadas en el procesamiento de imágenes y visión por computadora. Se utilizan en una amplia variedad de aplicaciones, desde la detección de objetos en imágenes hasta la generación de contenido visual y el análisis de secuencias de video, y han demostrado un rendimiento excepcional en estas tareas. A pesar de esto, no son una tarea sencilla pues no existe un patrón definido para establecer un número de capas [3]. Dentro de las más conocidas encontramos a:

TABLA 2 Redes Neuronales Profundas. Fuente: [23].

Tipo	# Capas
AlexNet	8 capas
VGG	16 capas
GoogleNet	22 capas
MobileNet	52 capas
ResNet	152 capas

3.8.1 MobileNet

MobileNet fue una de las primeras iniciativas para crear arquitecturas de redes neuronales convolucionales (CNN) que pueden desplegarse fácilmente en aplicaciones móviles. Una de las principales innovaciones son las convoluciones separables por profundidad, que se visualizan a continuación. Una convolución separable divide un núcleo de convolución normal en dos núcleos. Por ejemplo, en lugar de un núcleo de 3x3, obtenemos un núcleo de 3x1 y otro de 1x3. Esta separación reduce la cantidad de operaciones necesarias para realizar la convolución, lo que hace que sea mucho más eficiente. Sin embargo, no siempre es posible separar en la dimensión espacial, por lo que es más común separar en la dimensión de profundidad (canales).

La versión 2 de MobileNet (MobileNetV2) introdujo los *residuos invertidos* y los *cuellos de botella lineales* para mejorar la eficiencia del modelo, permitiendo una mayor conservación de la información después de las activaciones (ReLU). Además, se incorporó la activación *ReLU6* para optimizar las activaciones en cálculos de baja precisión y hacer el modelo más adecuado para cuantización [52].

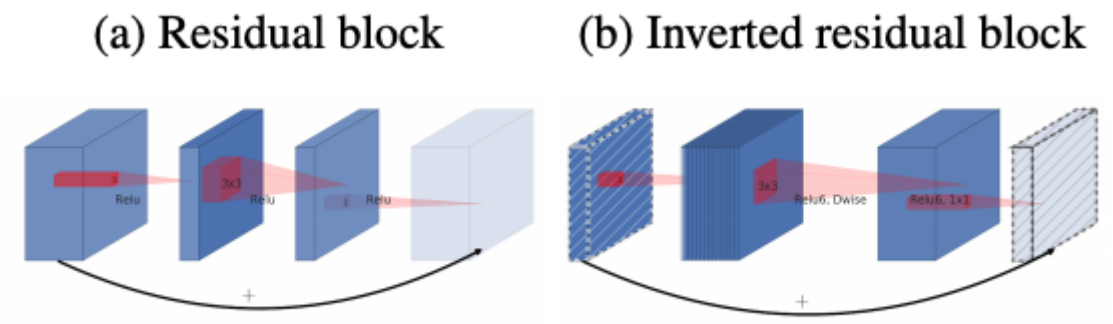


Fig. 15. Arquitectura de MobileNetV2. Fuente: [52].

3.8.2 ResNet

ResNet es una arquitectura de red neuronal convolucional profunda que se caracteriza por su uso de conexiones residuales, permitiendo el aprendizaje de arquitecturas más profundas sin enfrentar el problema de los gradientes desvanecidos. Cuenta con diferentes profundidades, pero en su variante ResNet50, cuenta con 50 capas, su estructura se divide en capas convolucionales para extraer características de las imágenes, seguidas de bloques de identidad y convolucionales que procesan y transforman estas características. Además, incluye capas totalmente conectadas que realizan la clasificación final utilizando una función de activación softmax. Esta red es especialmente eficaz para la clasificación de imágenes en grandes conjuntos de datos gracias a su capacidad para aprender funciones residuales y mejorar el rendimiento en tareas complejas. [53].

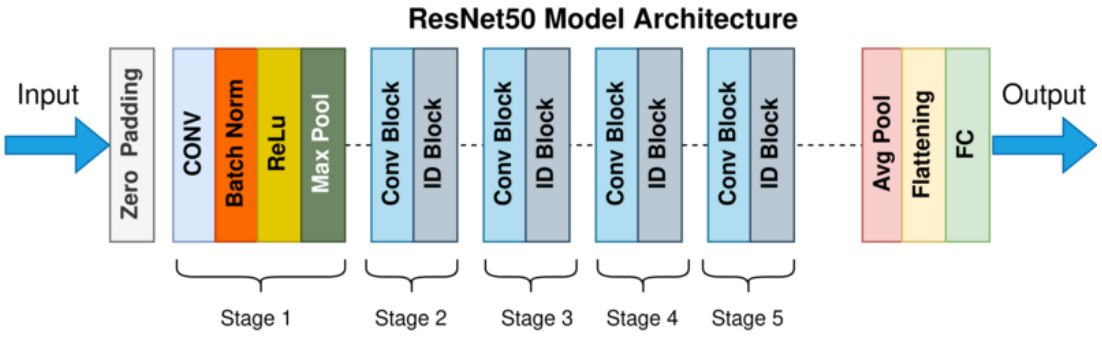


Fig. 16 Arquitectura de ResNet50. Fuente: [53].

3.9. Trabajos Previos

Dentro de los métodos actuales para MFI, hay diversos documentos publicados con métodos validos dentro de los cuales podemos encontrar el algoritmo de los “k-nearest neighbors”. Por otra parte, también el algoritmo de k-means es considerado un método

válido [4]. En [24] publicado en 2015, los autores demuestran al algoritmo "k-means" como un método valido para MFI en comunicaciones ópticas, permitiendo la agrupación de constelaciones recibidas, lo que en función de la cantidad y posición de los "clusters" permite la identificación del formato de modulación, proponiendo un nuevo método que no requiere de parámetros iniciales. Por otra parte, en [25] publicado en 2016, los autores demuestran que las *"DNN"* son capaces de identificar eficazmente el formato de modulación transmitido para diferentes relaciones de OSNR, incluso en presencia de otros fenómenos degradantes como desviaciones de fase y frecuencia.

También, en [26] publicado en 2019, los autores describen un método para MFI basado en el algoritmo de "k-nearest neighbors", empleando como entradas los histogramas de las componentes real e imaginaria de la señal óptica recibida muestreada en una muestra por símbolo. De igual manera, en [1] publicado en 2020, los autores plantean dos tipos de métodos; métodos clásicos y métodos de ML.

En los métodos clásicos se dividen en dos tipos; métodos en base a probabilidades y métodos en base a características.

Con respecto a los métodos de ML encontramos técnicas que son clasificadas en función del tipo de detección de señal, ya sea directa o coherente. A pesar de que la detección directa tiene la ventaja de reducir el costo de la solución propuesta, la mayoría de las técnicas de MFI utilizan detección coherente debido a la dificultad para recuperar la información de la fase en receptores de detección directa. Aquí encontramos algoritmos supervisados, no supervisados y de refuerzo. Dentro de los supervisados encontramos CNN, ANN y SVM. Por otro lado, en los no supervisados encontramos algoritmos del tipo *"clustering".*

La mayoría de los métodos basados en ML utilizan detección coherente debido a la capacidad de recuperar información de fase, un aspecto crítico en los sistemas de comunicación óptica moderna. Si bien los métodos de detección directa son menos costosos, no alcanzan la precisión requerida en escenarios complejos.

Estudios recientes han explorado además combinaciones de métodos clásicos y ML, buscando aprovechar las ventajas de ambos enfoques.

A continuación, se presentan ventajas y desventajas tanto de los métodos clásicos como métodos de ML.

TABLA 3 Ventajas y desventajas de técnicas para MFI. Fuente: [1].

Tipo de método	Algoritmo	Ventajas	Desventajas
Clásico	k-NN	Fácil de implementar. Admite funciones de mapeo altamente no lineales.	Sensibles a la calidad de los datos y valores atípicos. Requiere alta memoria de almacenamiento.
Clásico	k-means	Eficiencia computacional.	Sensible a ruido y valores atípicos Forma fija de "clusters".
ML	CNN	Escalable y robusto. Proporciona un rendimiento de última generación con entradas de dos dimensiones.	Alta complejidad computacional.
ML	SVM	Efectivo en espacios dimensionales grandes.	La selección del kernel es heurístico.
ML	Clustering on partition	Rápido y fácil de implementar.	El número de "clusters" suele ser heurístico.

4. Desarrollo de metodología

En esta sección se describe el procedimiento implementado para implementar un clasificador de imágenes multiclase utilizando redes neuronales convolucionales. Las CNN son especialmente adecuadas para el reconocimiento y clasificación de imágenes debido a su capacidad para extraer automáticamente características relevantes, como bordes, texturas y patrones espaciales, a través de capas convolucionales. Esta capacidad de aprender jerárquicamente características de bajo a alto nivel permite a estas redes identificar patrones complejos con gran precisión.

Como parte de la implementación del clasificador, inicialmente se opta por implementar y entrenar una CNN personalizada, aprovechando su capacidad para aprender patrones relevantes directamente desde los datos. Sin embargo, para evaluar el rendimiento del modelo de clasificación y comparar distintos enfoques, también se implementaron dos modelos previamente entrenados que son ampliamente utilizados en la clasificación de imágenes: "MobileNetV2 y ResNet".

Ambos modelos fueron preentrenados en grandes conjuntos de datos, como "ImageNet", que contiene más de 14 millones de imágenes etiquetadas, lo que les permite aprovechar el aprendizaje transferido de características generales y adaptarse de manera eficiente a nuevas tareas de clasificación, como la clasificación multiclase de este caso.

4.1 Base de datos

La base de datos utilizada para el algoritmo consta de dos partes; la primera fue simulada mediante un script de Matlab donde se generan imágenes de diagramas de constelación que representan diferentes tipos de modulaciones en amplitud de cuadratura (*QAM*), en este caso se utilizan 4 formatos de interés los cuales se muestran en la tabla 4.

Por otro lado, la segunda parte del dataset fue extraída desde "*Achievable Information Rates for Fiber Optics: Applications and Computations"* [42], de donde se seleccionan 4 formato de interés de modulaciones de amplitud en cuadratura geométricamente optimizada (*GS-QAM*), de tal manera que coincida la cantidad de bits por símbolo que se transmite con los formatos previamente seleccionados.

El formato QAM es ampliamente utilizado en comunicaciones digitales dado que combina la modulación de amplitud y fase, permitiendo transmitir múltiples bits por símbolos. Por otro lado, GS-QAM ajusta la disposición de los puntos de constelación en el diagrama de modulación, optimizando la eficiencia espectral y mejorando la resistencia al ruido.

TABLA 4 Formatos de modulación utilizados. Fuente: [54].

Bits por símbolo	QAM	Geometrically Shaped
4 bits por símbolo	16 QAM	2D-16 QAM
6 bits por símbolo	64 QAM	2D-64 QAM
7 bits por símbolo	128 QAM	2D-128 QAM
8 bits por símbolo	256 QAM	2D-256 QAM

Para cada conjunto de símbolos modulados, la señal fue pasada por un canal AWGN para simular el impacto del ruido en la transmisión. La relación señal a ruido (SNR) se estableció en un rango de 10 a 20 [dB], incrementando en pasos de a 1 [dB]. El objetivo de agregar ruido con diferentes niveles de SNR es simular condiciones de comunicación más realistas, dado que los sistemas de comunicación están sujetos a interferencias y ruidos que afectan la calidad de la señal. Para cada valor de SNR, la señal modulada se transmite a través de un canal AWGN en donde se obtiene la señal recibida que incorpora el efecto del ruido. Estas variaciones permiten evaluar la capacidad de los algoritmos de clasificación para adaptarse y reconocer patrones incluso cuando las señales están degradadas por interferencias de ruido. Mediante el comando “scatterplot” es posible graficar visualmente la constelación, lo que permite ver la distribución y relación entre las variables.

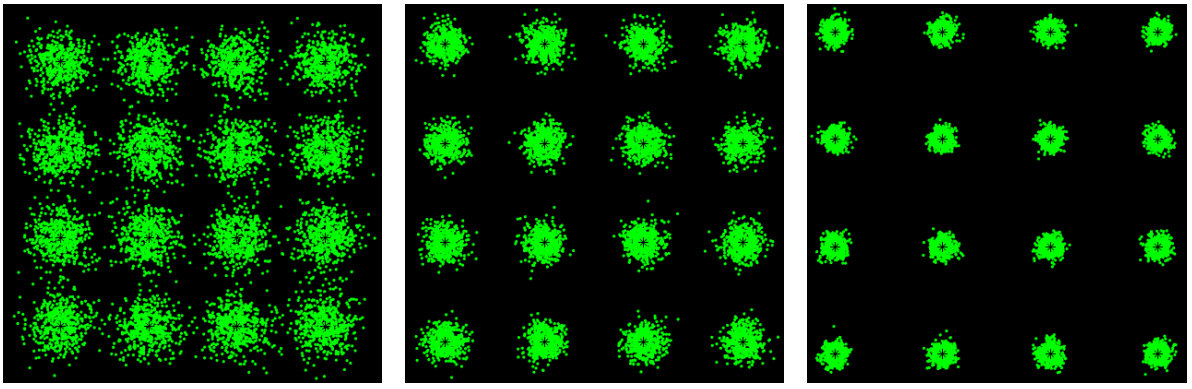


Fig. 17. 16-QAM con diferentes SNR. Fuente: [54].

Este proceso se repite para cada tipo de formato de modulación y para cada valor de SNR en el rango mencionado, de tal manera que se obtiene una base de datos de 221 imágenes por cada clase, las cuales son almacenadas en Google Drive. Este conjunto de imágenes variado proporciona una

amplia gama de condiciones de canal, lo que es crucial para entrenar y evaluar modelos de clasificación capaces de manejar diversos escenarios del mundo real.

4.2 Preprocesamiento de las imágenes

El dataset utilizado contiene 1,768 imágenes distribuidas en 8 clases distintas, organizadas en carpetas, donde cada carpeta representa una clase específica. Para uniformizar la entrada al modelo, cada imagen fue redimensionada a 224 x 224 píxeles, tamaño que balancea adecuadamente calidad de imagen y velocidad de procesamiento. Además, se normalizaron los valores de los píxeles dividiéndolos entre 255, lo que ajustó el rango de cada pixel a [0,1], mejorando la eficiencia del entrenamiento y facilitando la convergencia. Para gestionar los datos, se empleó la clase "ImageDataGenerator", que permitió dividir automáticamente el dataset en un 80% para entrenamiento y un 20% para validación, sin necesidad de organizar manualmente carpetas específicas para cada conjunto.

Se implementa generador de datos de entrenamiento que ha sido configurado utilizando "train_datagen" para cargar imágenes desde el directorio raíz especificado. Las imágenes se cargan en formato RGB, lo que permite trabajar con imágenes a color, y se establece el tipo de clasificación como categórica para abordar un problema de clasificación multiclase. Dada la cantidad de imágenes del dataset, el tamaño de lote se define en 32, un valor empleado para equilibrar la eficiencia y la velocidad de entrenamiento, y se activa el barajado de datos para evitar que el modelo aprenda patrones específicos en el orden de las imágenes. Este generador está configurado para manejar exclusivamente los datos de entrenamiento.

Por otra parte, se implementa un generador de datos de validación que utiliza los mismos parámetros de preprocesamiento y redimensionamiento que en el generador de datos de entrenamiento, con la diferencia que se configura para cargar exclusivamente el 20% de las imágenes reservadas para la validación. Además, el barajado de datos se desactiva para asegurar que las imágenes de validación se procesen en el mismo orden en cada iteración. El resultado de este proceso se utiliza para todos los modelos clasificadores que se implementan a continuación.

4.3 Definición del modelo

El modelo de clasificación de imágenes se implementa mediante el uso de una arquitectura CNN que está diseñada para extraer características relevantes de las imágenes, comenzando con capas convolucionales que capturan características básicas, y avanzando hacia capas más profundas que

identifican patrones más abstractos. Para la implementación de este modelo de clasificación de imágenes, se emplean diversas bibliotecas de Python que facilitan tanto el procesamiento de datos como la construcción y evaluación del modelo. TensorFlow y Keras son las herramientas principales, proporcionando estructuras para diseñar redes neuronales profundas y emplear técnicas como el aprendizaje por transferencia utilizando modelos preentrenados, como MobileNetV2 o ResNet50.

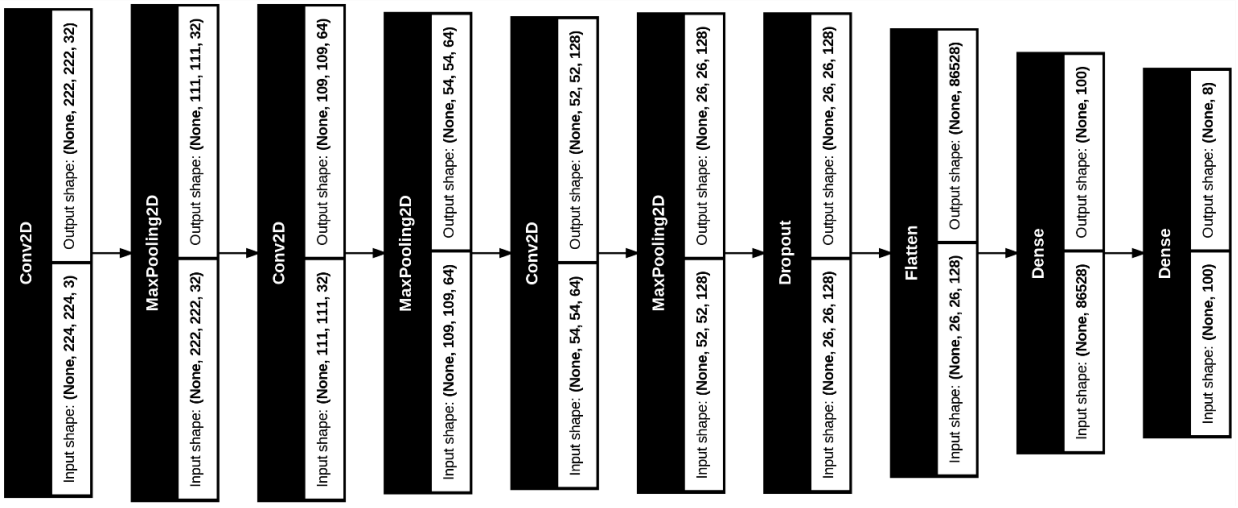


Fig. 18. Ilustración de capas de CNN implementada. Fuente: [54].

4.4Arquitectura del modelo

Se utiliza la clase “Sequencial” de Keras, que permite apilar capas de manera secuencial en donde la salida de cada capa se pasa directamente como entrada a la siguiente. La primera capa es una capa de convolución, con 32 kernels de tamaño 3x3 que recorren cada imagen por alto y ancho, y acompañada de una función de activación ReLU, diseñada para extraer características locales de las imágenes de entrada. A continuación, una capa de Max-Pooling con una ventana de 2x2 reduce la dimensionalidad de las características extraídas. Posteriormente, el modelo incluye dos bloques adicionales de capas de convolución de 64 y 128 kernels de igual tamaño que la primera y ambas capas con una función ReLU, seguidas cada una de ellas por una capa de Max-Pooling, para capturar patrones más complejos y seguir reduciendo aún más las dimensiones de la imagen. Para prevenir el sobreajuste, se incorpora una capa de Dropout con una tasa del 50%, que desactiva aleatoriamente el 50% de las unidades durante el entrenamiento.

Las características extraídas por las capas convolucionales y de pooling se aplanan mediante una capa de Flatten, convirtiéndolas en un vector unidimensional. Este vector se pasa a una capa densa

de 100 neuronas con activación ReLU, que permite al modelo aprender representaciones más abstractas. Finalmente, la capa de salida cuenta con un número de neuronas igual al número de clases en el conjunto de datos, es decir 8 neuronas, que utiliza una función de activación "softmax", que convierte las salidas en probabilidades de pertenencia a cada clase.

TABLA 5 Resumen modelo CNN implementada. Fuente: [54].

Layer	Output Shape	# Param
conv2d	(None, 222, 222, 32)	896
max_pooling2d	(None, 111, 111, 32)	0
conv2d_1	(None, 109, 109, 64)	18.496
max_pooling2d_1	(None, 54, 54, 64)	0
conv2d_2	(None, 52, 52, 128)	76.856
max_pooling2d_2	(None, 26, 26, 128)	0
Dropout	(None, 26, 26, 128)	0
flatten	(None, 86528)	0
dense	(None, 100)	8.652.900
dense_1	(None, 8)	808

Una vez definido el modelo, se procedió a su compilación utilizando el optimizador "Adam", que es un optimizador adaptativo basado en gradientes que destaca por su eficiencia y capacidad de convergencia en tareas de aprendizaje profundo. La función de pérdida seleccionada fue "categorical_crossentropy", adecuada para problemas de clasificación multi-clase, mientras que la métrica de evaluación utilizada fue "accuracy", que permite medir el rendimiento del modelo en términos de su capacidad para clasificar correctamente las imágenes.

Además, se implementaron tres callbacks para optimizar el proceso de entrenamiento y prevenir el sobreajuste: "ModelCheckpoint", que guarda el modelo en su mejor versión según el desempeño en el conjunto de validación; "EarlyStopping", que interrumpe el entrenamiento si no se observa una mejora en la pérdida de validación durante un número específico de épocas, restaurando los pesos del modelo a la versión de mejor rendimiento; y "ReduceLROnPlateau", que ajusta la tasa de aprendizaje reduciéndola a la mitad si no hay mejoras en la pérdida de validación durante tres épocas consecutivas, permitiendo al modelo realizar ajustes más finos y evitar el estancamiento.

4.5 Entrenamiento del modelo

En el proceso de entrenamiento del modelo, se utiliza el método “fit”, que es un método que ajusta iterativamente los parámetros internos de la CNN para minimizar la función de pérdida y optimizar la precisión en la clasificación de imágenes. El modelo se entrena inicialmente durante un máximo de 10 épocas a modo de tener un bosquejo del rendimiento de la red. Sin embargo, con fines de optar a mejores resultados se utiliza un máximo de 30 épocas, cada una recorriendo todo el conjunto de datos de entrenamiento en lotes de imágenes, lo que mejora la eficiencia y estabilidad en el ajuste de los parámetros.

Para optimizar dinámicamente el entrenamiento, se incluyen los tres callbacks previamente mencionados. Durante el entrenamiento, el rendimiento del modelo se evalúa en un conjunto de validación después de cada época, calculando métricas como la precisión y la pérdida para verificar su capacidad de generalización. Adicionalmente, se registra el historial del entrenamiento, lo cual permite analizar la evolución de las métricas en cada época.

4.6 Implementación de MobileNet

En esta sección, se implementó un modelo de clasificación de imágenes utilizando MobileNetV2, una red neuronal convolucional preentrenada sobre el conjunto de datos ImageNet. El modelo fue cargado excluyendo las capas finales de clasificación, lo que permite aprovechar las representaciones aprendidas previamente en una amplia variedad de tareas de clasificación general. Este enfoque de transfer learning evita la necesidad de entrenar la red desde cero, y permite focalizar el entrenamiento únicamente en las capas personalizadas añadidas. El modelo base de MobileNetV2 fue congelado, de manera que sus pesos permanecieron fijos durante el proceso de entrenamiento. Esto asegura que las capas preentrenadas no se vean alteradas, y el ajuste de los parámetros únicamente ocurra en las nuevas capas añadidas para adaptar la red al conjunto de datos específico.

Sobre esta arquitectura base, se añadió una capa de GlobalAveragePooling2D, la cual reduce los mapas de características a un único vector, facilitando su procesamiento posterior. Seguidamente, se incorporó una capa densa con 1024 neuronas y activación ReLU, diseñada para aprender representaciones específicas del nuevo conjunto de datos. Para prevenir el sobreajuste durante el entrenamiento, se incluyó una capa de Dropout con una tasa del 50%. Finalmente, se añadió una capa de salida con función de activación softmax, ajustada al número de clases presentes en la tarea de clasificación. El modelo fue compilado y entrenado utilizando los mismos parámetros que en el caso de la red convolucional anterior.

4.7 Implementación de ResNet

A continuación, se implementó un modelo de clasificación de imágenes basado en ResNet50, una red neuronal profunda que utiliza conexiones residuales para facilitar el aprendizaje de representaciones jerárquicas complejas. Similar a la implementación de MobileNetV2, se cargaron los pesos preentrenados de ResNet50, excluyendo las capas de clasificación finales, lo que permitió aprovechar el conocimiento aprendido por la red en la tarea de clasificación sobre ImageNet. Al utilizar ResNet50 como extractor de características, se adaptó la red a la tarea específica de clasificación mediante la adición de nuevas capas.

Primero, se incluyó una capa de GlobalAveragePooling2D para reducir los mapas de características a un único vector. A continuación, se añadió una capa densa con 1024 neuronas y activación ReLU para aprender representaciones relevantes del conjunto de datos en cuestión. Para mitigar el riesgo de sobreajuste, se incorporó una capa de Dropout con una tasa del 50%. La red concluye con una capa de salida que utiliza la función de activación softmax para realizar la clasificación multiclase, ajustada al número de clases del conjunto de datos utilizado.

Al igual que con el modelo de MobileNetV2, el modelo de ResNet50 fue compilado y entrenado utilizando parámetros idénticos a los utilizados en la primera red convolucional implementada.

4.8 Evaluación de los modelos

En el ámbito de la inteligencia artificial y el aprendizaje automático, evaluar la efectividad de un modelo es fundamental para asegurar su rendimiento en tareas específicas, como la clasificación de datos; y, para ello, se emplean diversas métricas que permiten medir la precisión y la capacidad predictiva del modelo en función de los resultados obtenidos, ya sea en términos de probabilidad o de clasificación binaria o múltiple. Entre las métricas más utilizadas en los problemas de clasificación, se destacan la precisión, el recall, el F1-score y la precisión global o accuracy, las cuales, aunque estén estrechamente relacionadas, ofrecen perspectivas distintas sobre el desempeño del modelo que resulta crucial para obtener una evaluación precisa y completa.

4.8.1 Matriz de confusión

La matriz de confusión es una herramienta de evaluación que permite analizar visualmente el desempeño de un modelo de clasificación, mostrando en una matriz cuadrada la relación entre las predicciones del modelo y las etiquetas reales para cada clase. Esta matriz está compuesta por cuatro elementos clave: los verdaderos positivos (TP), que representan las instancias correctamente predichas como positivas; los falsos positivos (FP), que son las instancias incorrectamente clasificadas como positivas; los verdaderos negativos (TN), correspondientes a las instancias correctamente predichas como negativas; y los falsos negativos (FN), que indican las instancias incorrectamente predichas como negativas. [3]

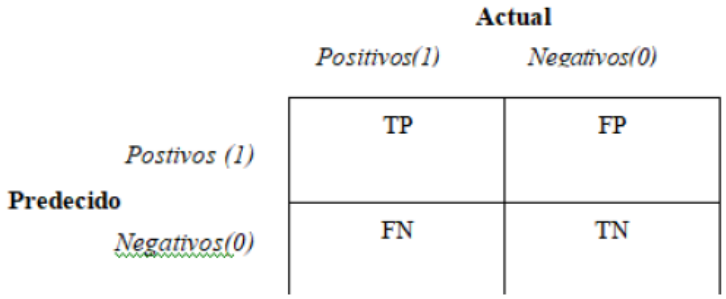


Fig. 19. Ilustración Matriz de confusión. Fuente: [3].

El uso de la matriz de confusión permite identificar patrones específicos de error y proporciona un análisis detallado del comportamiento del modelo para cada clase. Esta información es valiosa para realizar ajustes en el modelo, ya que facilita la detección de clases problemáticas o de tendencias de clasificación incorrectas. Al ofrecer una visión completa del rendimiento del modelo en función de cada categoría, la matriz de confusión es fundamental para corregir la precisión del sistema.

4.8.2 Accuracy

"Accuracy", o precisión global, representa el porcentaje de predicciones correctas entre el total de predicciones realizadas. Es particularmente útil cuando las clases están equilibradas en el conjunto de datos, ya que permite medir la tasa de aciertos sin una tendencia hacia alguna clase específica. En tales casos, una alta precisión suele ser un indicativo confiable del buen funcionamiento del modelo. Sin embargo, en casos donde las clases están desbalanceadas, la precisión puede ser una métrica poco representativa dado que un modelo que predice mayormente la clase con mayor frecuencia podría alcanzar una alta precisión sin ser efectivo en identificar correctamente las clases minoritarias.

4.8.3 Recall

El recall mide la capacidad del modelo para identificar correctamente todas las etiquetas positivas dentro de un conjunto de datos. Esta métrica se calcula como la proporción de verdaderos positivos (TP) sobre el total de instancias que realmente pertenecen a la clase positiva. Es especialmente relevante en situaciones donde es crucial detectar todos los casos positivos, aunque esto pueda aumentar el número de falsos positivos, dado que no detectar un caso positivo tendría consecuencias significativas.

4.8.4 Precision

"Precision", o la precisión de predicción de clase, es la métrica que evalúa la proporción de verdaderos positivos entre todas las instancias que el modelo ha clasificado como positivas. Esta medida es particularmente útil cuando el objetivo es minimizar los falsos positivos, es decir, cuando resulta esencial evitar clasificar incorrectamente instancias negativas como positivas. En términos prácticos, la precisión indica la exactitud del modelo al identificar casos positivos, ofreciendo una evaluación de su confiabilidad en este aspecto.

4.8.5 F1 score

El F1-score es una métrica que combina la precisión y el recall mediante su media armónica, proporcionando una evaluación equilibrada del rendimiento de un modelo de clasificación. Esta métrica es especialmente útil en conjuntos de datos desbalanceados, ya que pondera tanto la capacidad del modelo para identificar correctamente las instancias positivas como su habilidad para minimizar los falsos positivos. De esta forma, el F1-score permite obtener una visión más integral del desempeño del modelo en situaciones donde tanto la precisión como el recall son relevantes para evaluar su efectividad.

5 Resultados

Durante la evaluación de los modelos, se analizan diversas métricas de rendimiento tanto en el conjunto de entrenamiento como en el de validación para interpretar su precisión y capacidad de generalización. Se utilizan curvas de entrenamiento para visualizar la evolución de la precisión y la pérdida a lo largo de las épocas, lo que permite identificar posibles problemas como el sobreajuste si la precisión de validación deja de mejorar. Además, se evalúa el rendimiento del modelo en el conjunto de validación utilizando el método “model.evaluate”, lo que proporciona una medida final de la precisión y la pérdida. Para un análisis más detallado, se emplea la matriz de confusión, que permite observar los errores de clasificación entre las diferentes clases.

Finalmente, se genera un informe de clasificación utilizando que incluye métricas clave como precisión, recall y F1 score, permitiendo evaluar el desempeño del modelo en cada clase individual y el balance entre precisión y sensibilidad en la clasificación. Estos pasos permiten obtener una visión completa sobre la efectividad del modelo.

5.1 Clasificador CNN

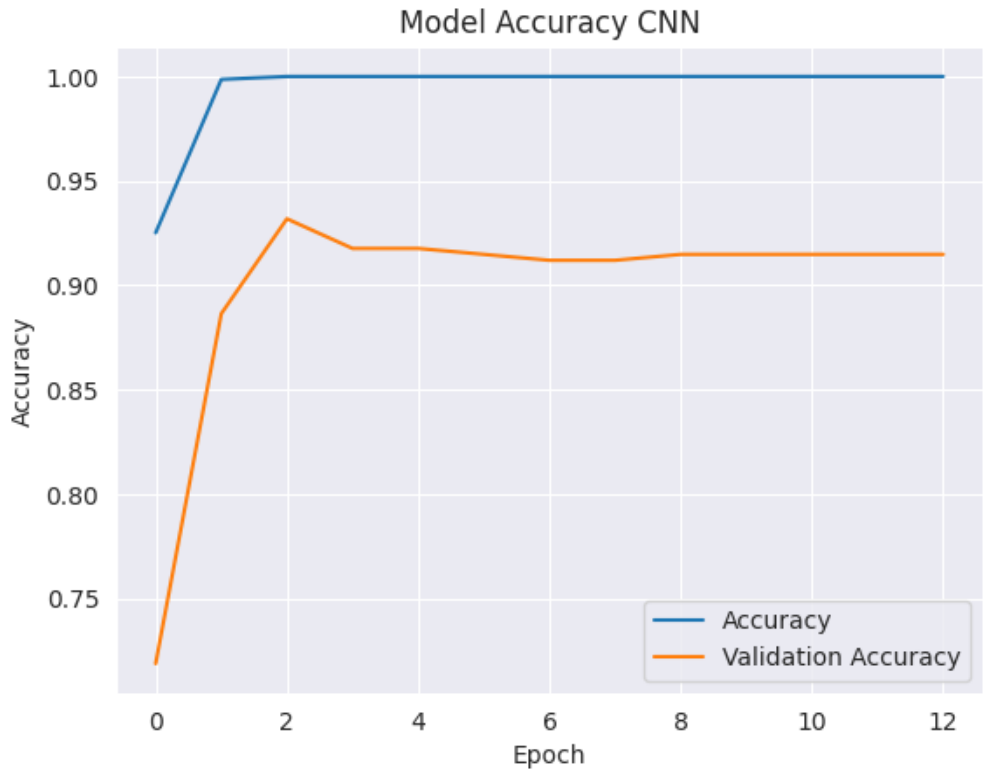


Fig. 20. Precisión de CNN implementada. Fuente: [54].

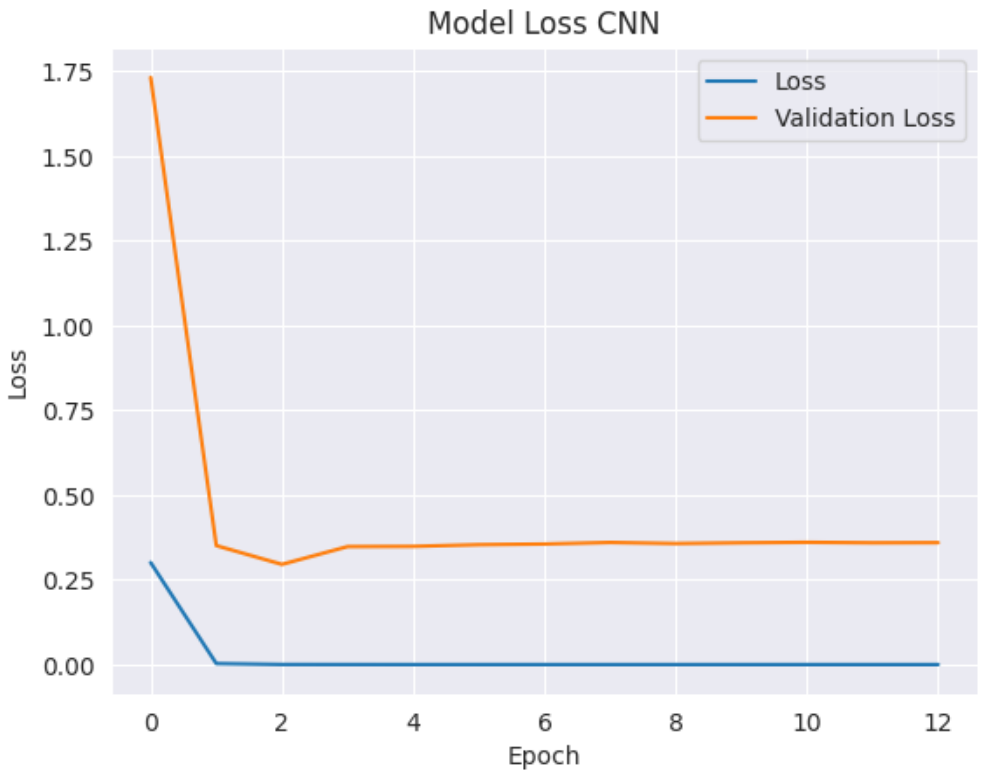


Fig. 21. Pérdida de CNN implementada. Fuente: [54].

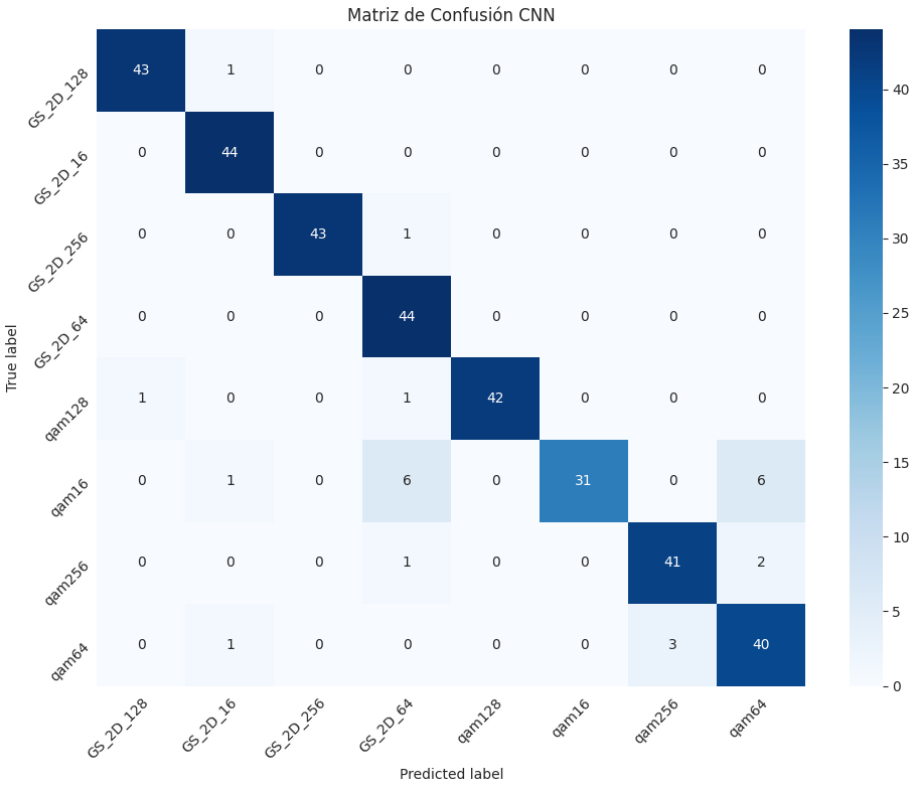


Fig. 22. Matriz de confusión de CNN implementada. Fuente: [54].

TABLA 6 Informe de clasificación CNN. Fuente: [54].

Clase	Precision	Recall	F1-score
16-QAM	1.00	0.70	0.83
64-QAM	0.83	0.91	0.87
128-QAM	1.00	0.95	0.98
256-QAM	0.93	0.93	0.93
2D-GS-16	0.94	1.00	0.97
2D-GS-64	0.83	1.00	0.91
2D-GS-128	0.98	0.98	0.98
2D-GS-256	1.00	0.98	0.99

El modelo de la CNN personalizada implementada logra un desempeño global sólido, alcanzando una precisión del 93% en el conjunto de validación. Las métricas promedio ponderadas, como la precisión, el recall y el F1-score, se mantuvieron en un rango de 93-94%, indicando un equilibrio generalizado en la clasificación de las 8 clases. Se observan clases las cuales mostraron un excelente desempeño, con un F1-score cercano a 1.00.

Sin embargo, se identificaron áreas de mejora en clases específicas. Por ejemplo, la clase 16-QAM presentó un recall reducido (0.70), a pesar de su alta precisión (1.00), lo que indica una tendencia a no identificar correctamente todas las muestras reales de esta clase. De manera similar, la clase 64-QAM mostró confusión con otras categorías, obteniendo un F1-score de 0.87, atribuido a una precisión menor (0.83) y un recall de 0.91. Estas observaciones fueron confirmadas en la matriz de confusión, que evidenció frecuentes errores entre las clases 16,64 y 256 QAM, debido a las similitudes en sus características.

En general, el modelo muestra un desempeño sólido y adecuado para el caso de interés.

5.2 Clasificador MobileNet

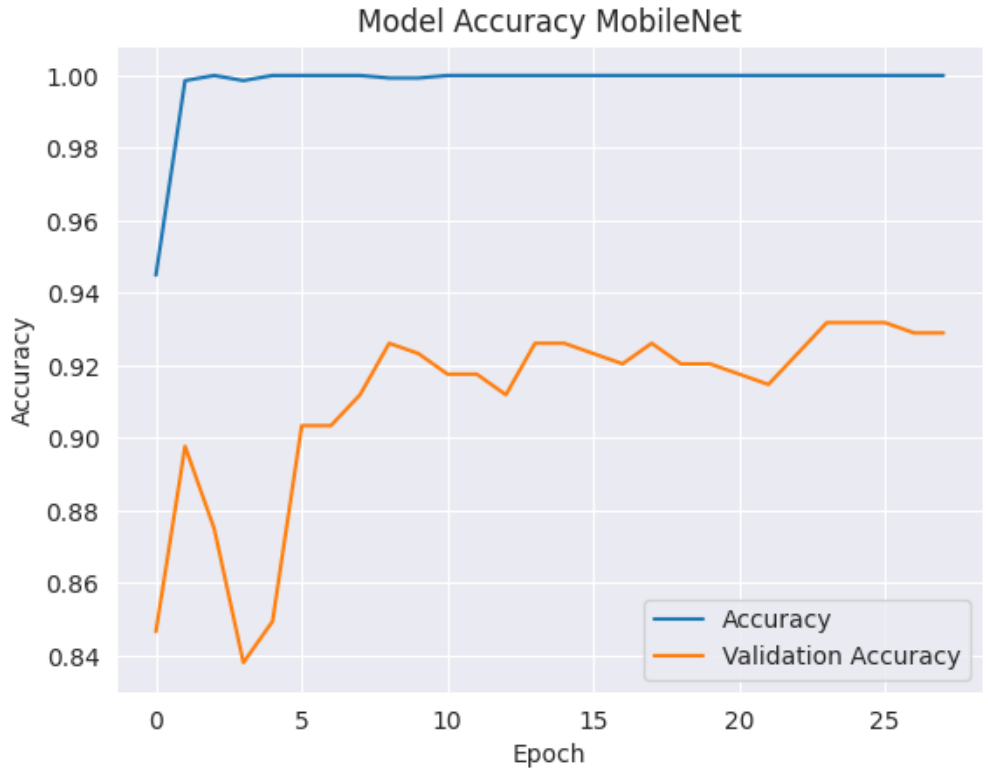


Fig. 23. Precisión de MobileNet implementada. Fuente: [54].

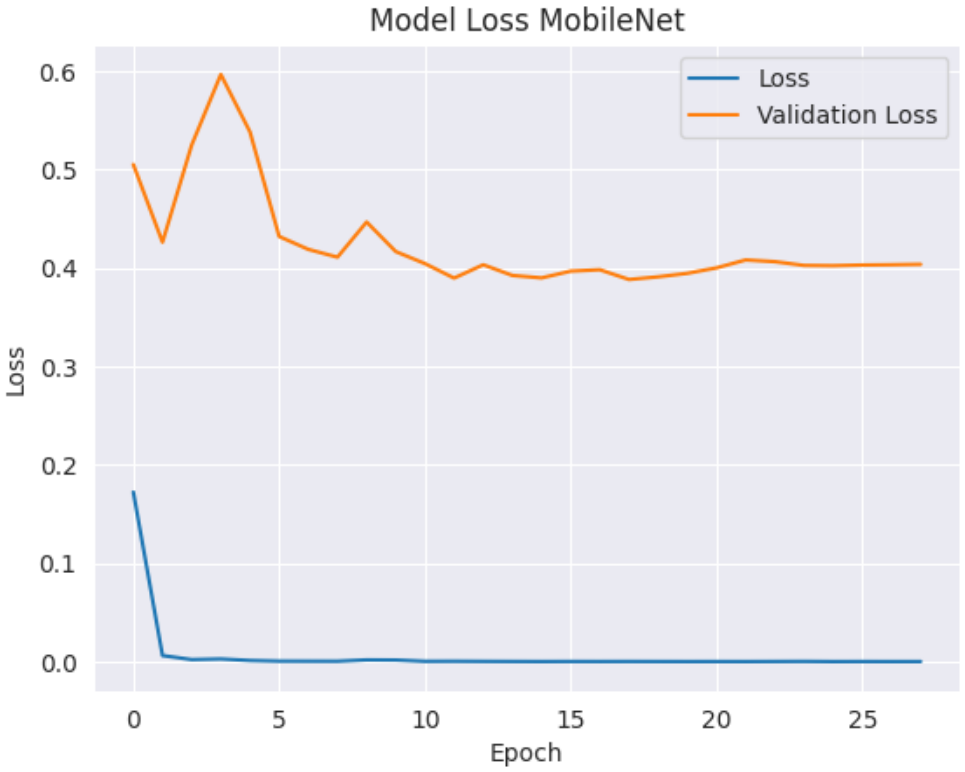


Fig. 24. Pérdida de MobileNet implementada. Fuente: [54].

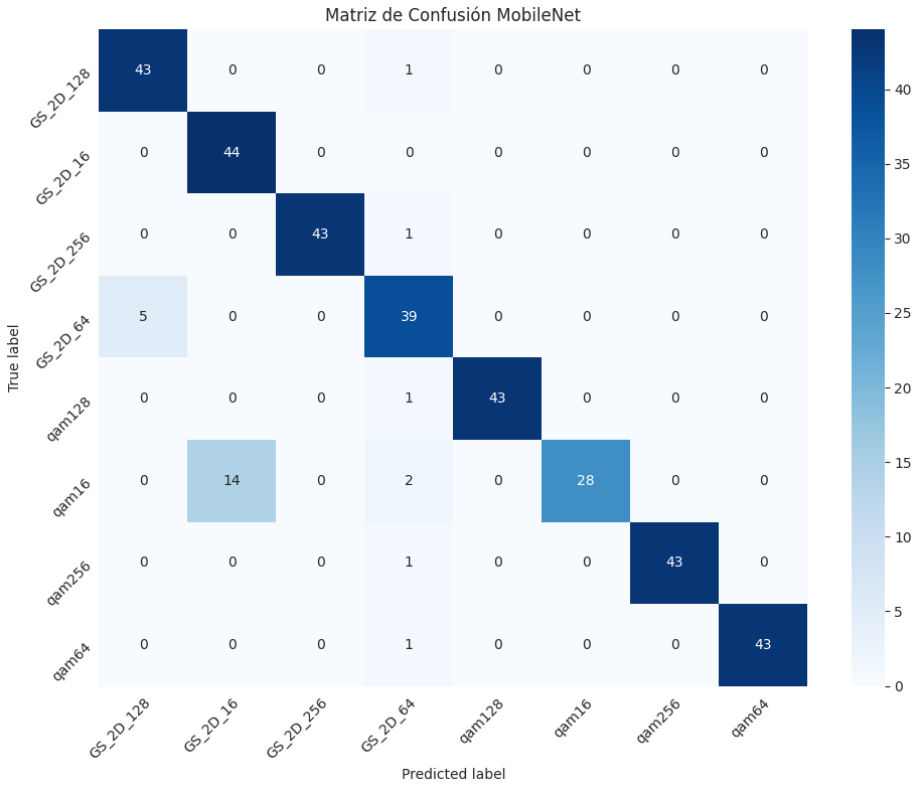


Fig. 25. Matriz de confusión de MobileNet implementada. Fuente: [54].

TABLA 7 Informe de clasificación MobileNet. Fuente: [54].

Clase	Precision	Recall	F1-score
16-QAM	1.00	0.64	0.78
64-QAM	1.00	0.98	0.99
128-QAM	1.00	0.98	0.99
256-QAM	1.00	0.98	0.99
2D-GS-16	0.76	1.00	0.86
2D-GS-64	0.85	0.89	0.87
2D-GS-128	0.90	0.98	0.93
2D-GS-256	1.00	0.98	0.99

El modelo MobileNet mostró un excelente desempeño en la tarea de clasificación, alcanzando una precisión casi perfecta en el conjunto de entrenamiento (superior al 99%) desde las primeras épocas. Sin embargo, en el conjunto de validación, la precisión se estabilizó alrededor del 92%, lo que sugiere un leve sobreajuste. Este comportamiento también se reflejó en las métricas de pérdida,

donde la pérdida del conjunto de entrenamiento se redujo casi a cero, mientras que la pérdida de validación se estabilizó alrededor de 0.4. Esto indica que el modelo optimizó excesivamente para los datos de entrenamiento, lo que puede limitar su capacidad de generalización.

Mediante la matriz de confusión y las métricas de clasificación se confirma un desempeño sobresaliente en la mayoría de las clases. Clases como 64, 128 y 256-QAM, y GS-256-QAM lograron métricas de precisión, recall y F1-score cercanas al 100%, evidenciando una clasificación precisa.

Sin embargo, se identificaron dificultades en las clases 16-QAM y GS-16-QAM, con F1-scores de 0.78 y 0.86, respectivamente, debido a confusiones con otras clases similares, como se observa en la matriz de confusión. Este comportamiento sugiere que podría ser necesario mejorar la extracción de características o aplicar técnicas adicionales para diferenciar mejor estas categorías.

En general, el modelo MobileNet se considera adecuado para la tarea de clasificación de imágenes, con resultados positivos en la mayoría de las clases.

5.3 Clasificador ResNet

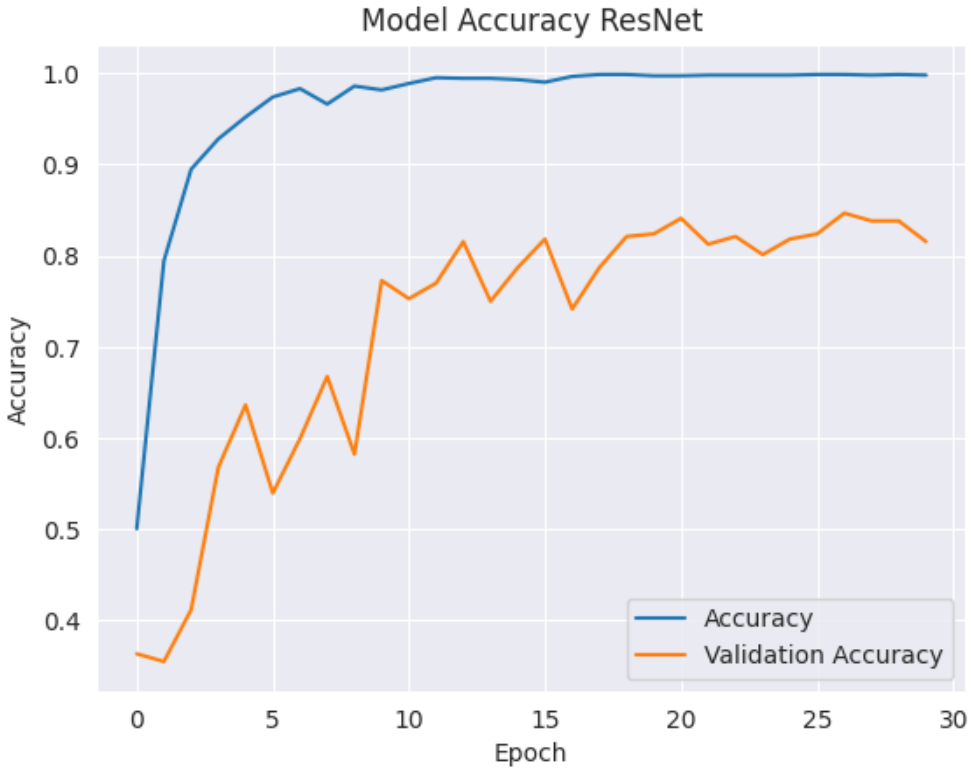


Fig. 26. Precisión de ResNet implementada. Fuente: [54].

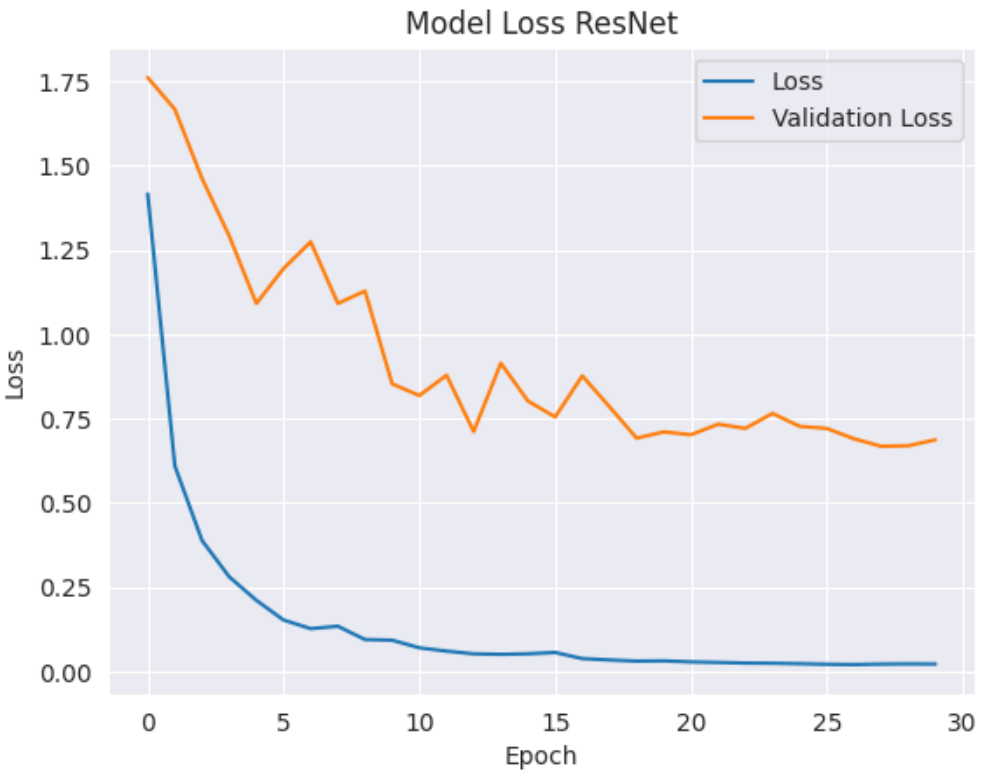


Fig. 27. Pérdida de ResNet implementada. Fuente: [54].

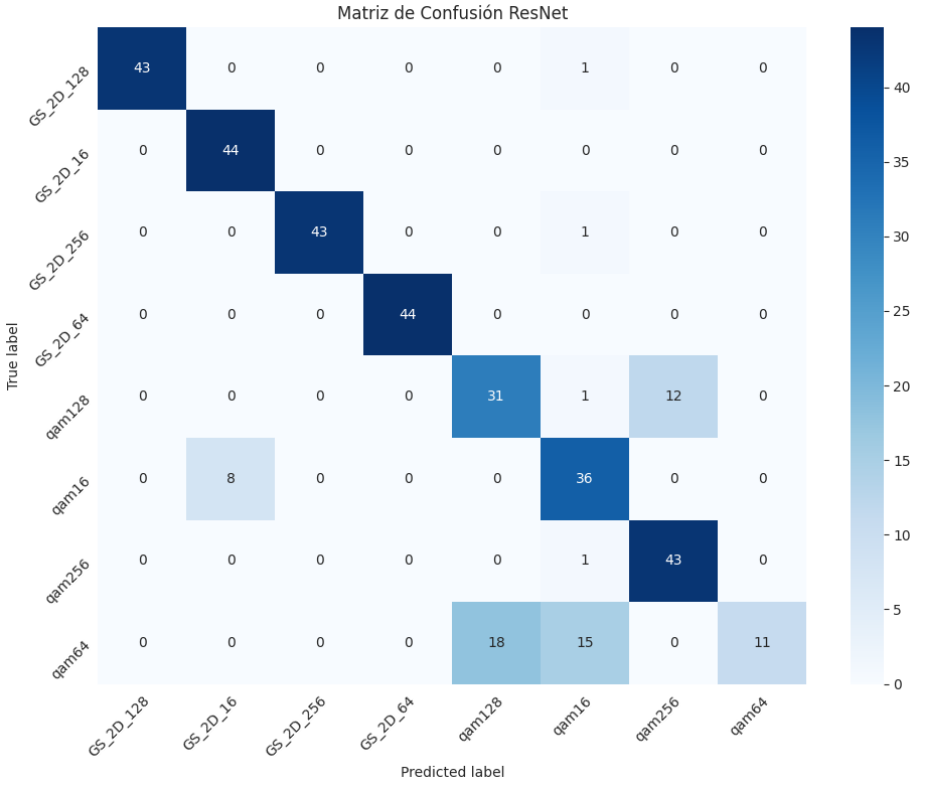


Fig. 28. Matriz de confusión de ResNet implementada. Fuente: [54].

TABLA 8 Informe de clasificación Resnet. Fuente: [54].

Clase	Precision	Recall	F1-score
16-QAM	0.65	0.82	0.73
64-QAM	1.00	0.25	0.40
128-QAM	0.63	0.70	0.67
256-QAM	0.78	0.98	0.87
2D-GS-16	0.85	1.00	0.92
2D-GS-64	1.00	1.00	1.00
2D-GS-128	1.00	0.98	0.99
2D-GS-256	1.00	0.98	0.99

El análisis del desempeño del modelo ResNet evidencia una rápida convergencia de la precisión en el conjunto de entrenamiento hacia valores elevados, lo que indica que el modelo aprende adecuadamente las características presentes en los datos de entrenamiento. Sin embargo, la precisión en el conjunto de validación muestra fluctuaciones a lo largo de las épocas, lo que sugiere posibles problemas de sobreajuste o inestabilidad en la generalización hacia datos no vistos. De manera consistente, la pérdida en el conjunto de entrenamiento decrece rápidamente hacia valores cercanos a cero, mientras que la pérdida en el conjunto de validación se estabiliza en un nivel más alto, reflejando un menor ajuste a los datos de validación en comparación con los datos de entrenamiento. La matriz de confusión destaca un desempeño sobresaliente en la clasificación de las clases GS-QAM, con altas tasas de acierto y un número reducido de errores fuera de la diagonal principal. En contraste, las clases 64 y 128-QAM presentan mayores niveles de confusión, lo que podría atribuirse a similitudes en características representativas de estas clases. Las métricas de evaluación muestran que las clases GS-QAM alcanzan valores casi perfectos en precisión, recall y F1-score, reflejando un alto nivel de desempeño para estas clases. Por otro lado, las clases QAM evidencian un desempeño más variable. Por ejemplo, la clase 64-QAM obtiene una precisión del 100%, pero un recall del 25%, lo que indica que el modelo tiene dificultades para identificar todas las muestras reales de esta clase, a pesar de clasificar correctamente las pocas muestras identificadas como pertenecientes a esta clase.

En términos generales, el modelo demuestra un excelente desempeño en la clasificación de las clases GS-QAM, mientras que en las clases QAM, el rendimiento es menos consistente.

5.4Análisis resultados

TABLA 9 Resultados de modelos. Fuente: [54].

Modelo	Precisión	Pérdida
CNN	0.93	0.30
MobileNet	0.93	0.32
Resnet	0.84	0.67

Como se muestra en la tabla 9, los resultados obtenidos destacan el desempeño sobresaliente de la CNN personalizada, que logra tanto la mayor precisión como la menor pérdida entre los modelos evaluados. Esto sugiere que el diseño de una arquitectura adaptada específicamente a las características del dataset permite capturar de manera más eficiente las relaciones entre las características de las imágenes. Esta ventaja se traduce en un modelo más confiable y ajustado al problema planteado.

Por otro lado, MobileNet también ofrece un desempeño competitivo, con una precisión equiparable a la de la CNN y una pérdida ligeramente mayor. Esto lo convierte en una alternativa sólida, especialmente considerando su diseño optimizado para aplicaciones prácticas, como dispositivos móviles o entornos con recursos computacionales limitados. Este equilibrio entre rendimiento y eficiencia computacional lo posiciona como una opción versátil para escenarios donde se busque un balance entre precisión y portabilidad.

En contraste, ResNet presenta el menor rendimiento de los tres modelos, con una precisión significativamente más baja y una pérdida mayor. Esto podría deberse a la mayor profundidad de su arquitectura, que podría estar sobreajustando el modelo o no aprovechando plenamente las características del dataset, dada su naturaleza específica. Este desempeño subóptimo sugiere que para lograr una mejora sería necesario ajustar hiperparámetros, realizar una selección más cuidadosa de las capas a congelar, o ampliar el dataset con técnicas de aumento de datos.

Los resultados obtenidos refuerzan que, para este problema específico, una CNN personalizada representa la opción más robusta en términos de rendimiento, mientras que MobileNet ofrece una solución práctica y eficiente. Por su parte, ResNet podría requerir ajustes adicionales para optimizar su desempeño en este contexto, destacando la importancia de adaptar los modelos preentrenados a las características particulares de cada dataset.

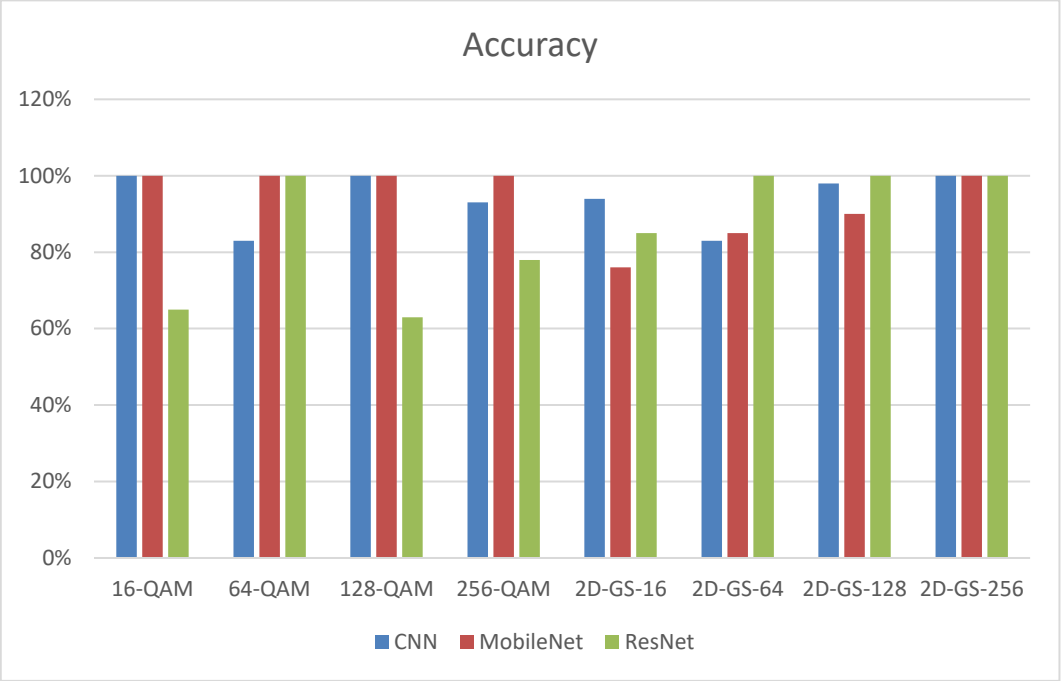


Fig. 29. Accuracy de modelo por clase. Fuente: [54].

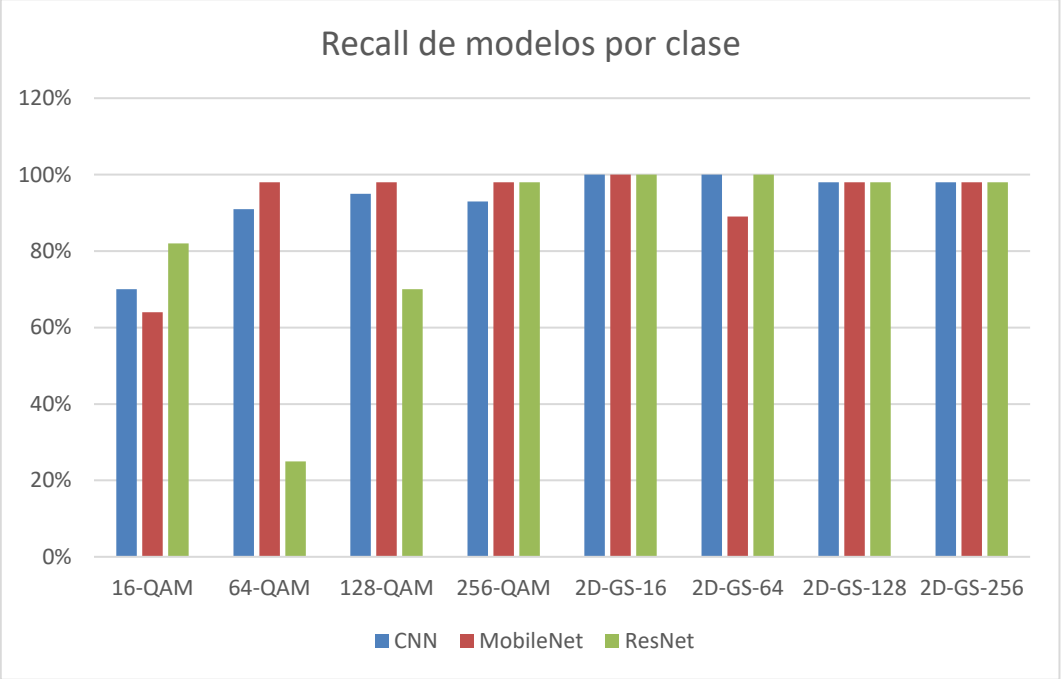


Fig. 30. Recall de modelo por clase. Fuente: [54].

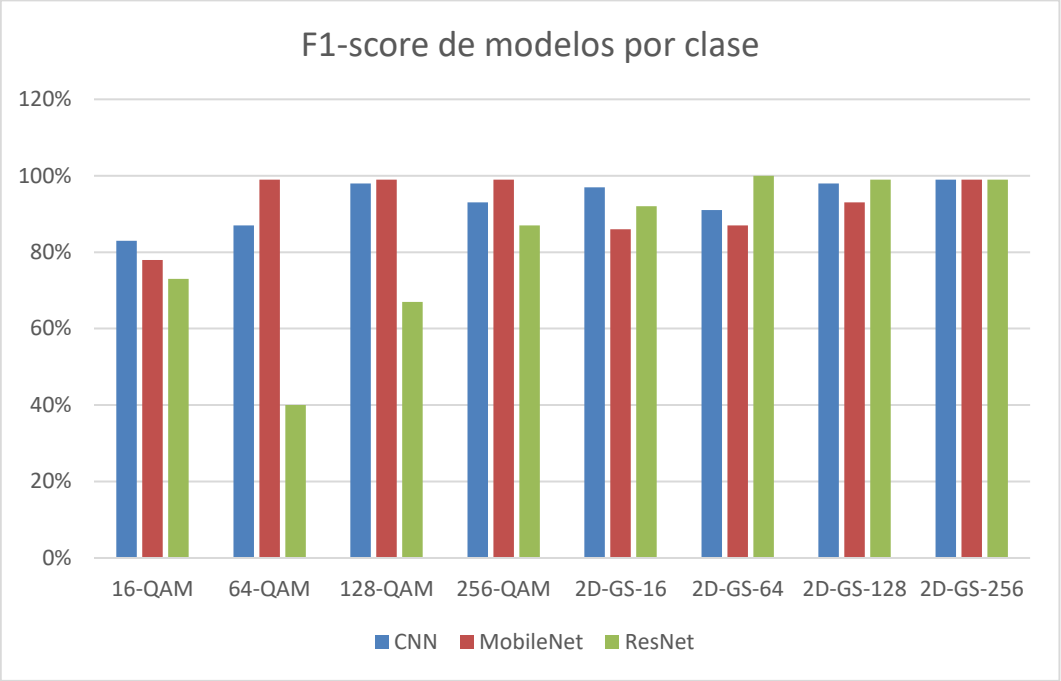


Fig. 31. F1-score de modelo por clase. Fuente: [54].

6. Discusión

Realizar una correcta clasificación de diagramas de constelaciones es fundamental en sistemas de comunicación digital, ya que permite diagnósticos más precisos sobre el desempeño del sistema y contribuye a la optimización de redes, especialmente en el contexto de tecnologías modernas.

En este proyecto se implementa y evalúan tres modelos de redes neuronales convolucionales para la clasificación multiclase de imágenes de diagrama de constelaciones: una CNN personalizada, MobileNetV2 y ResNet50. Los resultados de sus métricas, matriz de confusión y comportamiento durante el entrenamiento y validación permiten realizar un análisis comparativo entre los tres modelos implementados.

En general, el modelo de CNN personalizada mostró un rendimiento destacado, con una precisión de validación del 93% y una pérdida menor en comparación con MobileNet y ResNet. Esto resalta el buen funcionamiento de un modelo propio para adaptarse a las características específicas del conjunto de datos, lo que sugiere que diseñar arquitecturas personalizadas puede ser una estrategia eficiente para problemas específicos.

En contraste, MobileNet presentó un equilibrio sólido entre precisión (93%) y pérdida (0.32), destacándose como una alternativa preentrenada que es fácil de adaptar y entrenar para nuevas tareas. Sin embargo, los resultados muestran un ligero sobreajuste, evidenciado por una pérdida significativamente baja en el conjunto de entrenamiento, pero más alta en validación. Este comportamiento indica que, aunque MobileNet es eficiente en términos computacionales y de ajuste, podría beneficiarse de ajustes adicionales.

Por otra parte, ResNet, mostró el desempeño más bajo entre los modelos evaluados, con una precisión de validación del 84% y una pérdida elevada (0.67). Si bien el modelo logró un excelente rendimiento en las clases GS-QAM, las clases QAM presentaron desafíos significativos. Este resultado sugiere que ResNet, aunque es una arquitectura potente, no se adapta de manera correcta a este problema en su configuración actual. Por lo que una opción es implementar mayor personalización a la arquitectura, o utilizar un enfoque diferente en el preprocesamiento de los datos para mejorar su desempeño.

Se puede observar que los modelos evidenciaron fortalezas notables en la clasificación de las clases GS-QAM, con métricas de desempeño consistentemente altas. Esto sugiere que las características de estas clases son fácilmente diferenciables, lo que facilita su clasificación.

Por otro lado, una observación recurrente en todos los modelos fue la dificultad para clasificar clases QAM con características similares. Estas confusiones, confirmadas por la matriz de confusión y métricas como recall y F1-score, reflejan la necesidad de refinar la extracción de características distintivas entre estas clases. Por lo que, a pesar del buen desempeño general, hay áreas de mejora identificadas, particularmente en clases con confusión significativa, lo cual abren oportunidades para futuras investigaciones.

El desempeño superior observado en la CNN personalizada puede atribuirse a su diseño optimizado para las características específicas del conjunto de datos empleado. Este modelo no solo logra capturar de manera eficiente las características clave presentes en las imágenes, sino que también minimiza el impacto de elementos irrelevantes o ruidosos que podrían ser priorizados por arquitecturas generalistas como los modelos preentrenados utilizados. Por otra parte, aunque MobileNet y ResNet representan arquitecturas más complejas y robustas, se evidenció que su tiempo de entrenamiento y consumo de recursos computacionales fue significativamente mayor en comparación con la CNN personalizada. Este aspecto es crítico en aplicaciones donde la eficiencia computacional y la rapidez son factores determinantes para la implementación práctica de los modelos.

Además, el nivel de relación señal a ruido juega un papel clave en el desempeño de los modelos. Un SNR alto facilita la clasificación al resaltar patrones distintivos en las imágenes, mientras que un SNR bajo introduce ruido que puede confundir a los modelos, especialmente en clases visualmente similares como QAM. Esto fue evidente en las métricas como recall y F1-score, que revelaron dificultades para separar ciertas clases. Las confusiones observadas en la matriz de confusión reflejan la necesidad de mejorar la discriminación entre modulaciones con características similares. De igual forma, notamos que niveles bajos de SNR exponen las limitaciones de los modelos preentrenados, como MobileNet y ResNet, que suelen estar diseñados para tareas generales y podrían no estar completamente optimizados para trabajar con datos ruidosos. En contraste, la CNN personalizada, debido a su diseño específico para este conjunto de datos, podría mostrar mayor resiliencia frente al ruido, adaptándose mejor a las características del problema.

Técnicas como el aumento de datos (introducir perturbaciones específicas para resaltar diferencias), el uso de arquitecturas más profundas o explorar el uso de modelos híbridos que combinen

características preentrenadas con capas personalizadas diseñadas específicamente para el problema podrían ser soluciones efectivas para mitigar estas limitaciones.

En conclusión, la selección del modelo ideal depende del contexto. Si el objetivo es un modelo personalizado que equilibre precisión, pérdida y eficiencia computacional, la CNN personalizada es la mejor opción. Por otro lado, MobileNet ofrece una alternativa práctica para soluciones rápidas y transferibles. Sin embargo, los resultados también destacan áreas de mejora, particularmente en la clasificación de clases similares y el ajuste de modelos más complejos como ResNet. Estas observaciones abren camino a futuras investigaciones orientadas a optimizar modelos de clasificación en sistemas de comunicación digital.

7. Conclusiones.

Los avances en las redes ópticas, como el uso de formatos de modulación avanzados y técnicas de acceso en la capa física, han incrementado tanto la velocidad de datos como la complejidad de las redes, afectando la calidad de la señal. En este contexto, el monitoreo de rendimiento óptico y la identificación del formato de modulación se han vuelto cruciales. OPM permite estimar las degradaciones de la señal sin interrumpir el tráfico, mejorando la confiabilidad de la red. MFI, por su parte, permite a los receptores adaptarse a las señales para optimizar la demodulación, evitando interferencias y mejorando el rendimiento del sistema. Juntas, estas tecnologías facilitan la creación de redes ópticas autónomas, capaces de identificar el tipo de señal y realizar tareas de procesamiento sin información previa, lo que es clave para redes dinámicas y reconfigurables como las EON, que requieren nodos ópticos inteligentes.

En el presente estudio, se evalúa el desempeño de tres modelos de redes neuronales profundas para la clasificación multiclase de señales ópticas: una CNN personalizada, MobileNetV2 y ResNet50. Los resultados obtenidos muestran que la arquitectura de la CNN personalizada fue la más efectiva para este problema específico, destacándose por su capacidad para adaptarse mejor a las características del conjunto de datos y superar las limitaciones inherentes de los modelos preentrenados. Aunque MobileNetV2 mostró un rendimiento sólido en varias clases, presentó signos de sobreajuste, lo que afectó su capacidad de generalización. Por su parte, ResNet50, a pesar de su robustez arquitectónica, mostró fluctuaciones y un desempeño inconsistente, especialmente en las clases con señales similares, como las señales QAM.

Estos hallazgos enfatizan la importancia de adaptar las arquitecturas de red a las características específicas de los datos y del problema, y subrayan la relevancia de un ajuste meticuloso de los hiperparámetros. Aunque los tres modelos mostraron un desempeño positivo, se identificaron áreas de mejora, particularmente en la diferenciación de clases con patrones visuales similares, lo que abre nuevas líneas de investigación para optimizar estos modelos.

Además, el análisis evidenció que la relación señal a ruido juega un papel crucial en la calidad de las imágenes de los diagramas de constelaciones, lo que impacta directamente el desempeño de los modelos de clasificación. La resiliencia frente al ruido, la adaptabilidad de los modelos y la capacidad para manejar clases con características similares son factores determinantes para el éxito de la

clasificación de señales. Esto resalta la necesidad de futuras optimizaciones que permitan desarrollar sistemas más robustos y precisos, capaces de operar en condiciones de ruido adversas en contextos reales de comunicación óptica.

A partir de lo anterior, se puede indicar que una arquitectura personalizada, como la CNN diseñada, puede superar a los modelos preentrenados en tareas específicas de clasificación, destacando la importancia de soluciones adaptadas a problemas concretos. La combinación de enfoques personalizados, junto con el uso de modelos preentrenados y técnicas avanzadas de regularización y preprocesamiento, proporcionará el camino hacia sistemas más precisos y robustos, con un gran potencial para la clasificación de señales en aplicaciones reales, especialmente en el contexto de redes ópticas autónomas y adaptativas.

8. Glosario.

AI, IA : Inteligencia Artificial.

AIR : Tasas de información alcanzables.

ANN : Red Neuronal Artificial.

AM : Modulación en amplitud.

AWGN : Ruido blanco Gaussiano aditivo

BER : Tasa de error de bits.

CNN : Red Neuronal Convolucional.

DL : Aprendizaje Profundo.

DNN : Red Neuronal Profunda.

DSP : Procesamiento digital de señales.

DT : Arbol de Decisión.

EON : Red Ópticas Elásticas.

EDFA : Amplificador de fibra dopado con Erbio

FEC : Corrección de Errores hacia Adelante.

FM : Modulación en frecuencia.

FTTx : Fibra hasta la …x.

GPU : Unidad de procesamiento gráfico.

IoT : Internet de las Cosas.

MFI : Identificación de Formato de Modulación

ML. : Aprendizaje por Maquina (Machine Learning)

MMF : Fibra Multi Modo.

OFDM : Multiplexación por división de frecuencia ortogonal.

OPM : Monitoreo de Rendimiento Óptico.

OSNR : Razón Señal a Ruido Óptico.

PM : Modulación en fase.

PON : Red Óptica Pasiva.

QAM : Modulación en amplitud en cuadratura.

GS-QAM : Modulación en amplitud en cuadratura con forma geométrica.

QOS : Calidad de servicio

ReLu : Unidad Lineal Rectificada.

RGB : Rojo Verde y Azul.

SER : Tasa de error de símbolos.

SMF : Fibra Mono Modo.

SNR : Razón Señal a Ruido.

SVM : Máquina de Vectores de Soporte.

TDM : Multiplexación por División de Tiempo.

WDM : Multiplexación por División de Longitud de Onda.

9. Referencias.

[1] W. S. Saif, M. A. Esmail, A. M. Ragheb, T. A. Alshawi and S. A. Alshebeili, *"Machine Learning Techniques for Optical Performance Monitoring and Modulation Format Identification: A Survey,"* in IEEE Communications Surveys & Tutorials, vol. 22, no. 4, pp. 2839-2882, Fourthquarter 2020.

[2] F. Saavedra, D, Leonelli, and A, Lamas. *"Propagación de los pulsos ópticos a través de amplificadores de fibra dopada con erbio (EDFA)".* Revista Facultad de Ingeniería - Universidad de Tarapacá, 13(3), pp 82-88.

[3] Á. Artola, "*Clasificación de imágenes usando redes neuronales convolucionales en Python*", Trabajo Fin de Grado, Grado en Ingeniería de las Tecnologías de Telecomunicación, 2019, Departamento de Teoría de la Señal y Comunicaciones, Escuela Técnica Superior de Ingeniería, Universidad de Sevilla.

[4] A.L. Nunes de Souza, T. Sutili and J. Hélio da Cruz Júnior. "*Artificial Intelligence Techniques Evaluation for Modulation Format Identification in Optical Networks". J. Microw. Optoelectron. Electromagn. Appl.* 2021. Vol. 20(4):689-701. 2005.

[5] G. P. Agrawal, "*Fiber-Optic Communication Systems,"* 3rd ed. Wiley, USA, 2002.

[6] J.M. Fàbrega, S. Graells, *"Sistemas de comunicaciones ópticos*", UOC, Barcelona 2013.

[7] A Guauxochitl M. "*Capítulo 1 Comunicaciones por fibra óptica*", UNAM, 2012.

[8] *Capítulo 1. INTRODUCCIÓN Estudio de Redes de Difracción de Bragg en Fibra Óptica*,https://biblus.us.es/bibing/proyectos/abreproy/70225/fichero/Volumen+1%252FCapitulo1.pdf , accedido: 05 de diciembre 2023.

[9] G. Saavedra, *"Capítulo 1, Comunicaciones Ópticas"*, UdeC, Concepción, 2023.

[10] R. Olivares, *"Comunicaciones por fibra óptica (Elo-357)"*, UTFSM, Depto. Electronica 2007.

[11] Mizuno, T., & Miyamoto, Y. (2017). "*High-capacity dense space division*

multiplexing transmisión". Optical Fiber Technology, 35, 108-117.

[12] F. N. Khan, Q. Fan, C. Lu and A. P. T. Lau, "*An Optical Communication's Perspective on Machine Learning and Its Applications,"* in Journal of Lightwave Technology, vol. 37, no. 2, pp. 493-516, 15 Jan.15, 2019, doi: 10.1109/JLT.2019.2897313.

[13] J. Haro G. (2020). Análisis de Factores Limitantes en la Escalabilidad de Redes ópticas legadas y de nueva generación: propuesta de soluciones basadas en la gestión de la polarización. Tesis (Doctoral), E.T.S.I. Telecomunicación (UPM) <https://oa.upm.es/view/institution/Telecomunicacion/>.https://doi.org/10.20868/UPM.thesis.65781

[14] B. Lopéz Takeyas, "*Introducción a la Inteligencia Artificial*", Instituto Tecnológico de Nuevo Laredo, Nuevo Laredo, 2007

[15] Á. Serrahima, J. Bengoechea "*AVANCES Y DESAFÍOS DE LA INTELIGENCIA ARTIFICIAL*", Escuela técnica superior de ingeniería (ICAI), Madrir, 2022.

[16] Blog Ed Team, https://ed.team/blog/que-es-machine-learning-y-deep-learning-el-corazon-de-la-ia, Accedido: 07 de diciembre 2023.

[17] SmartPanel, *¿Qué es el Deep-Learning?*, https://www.smartpanel.com/que-es-deep-learning , Accedido: 07 de diciembre 2023.

[18] Sarker, I.H. "*Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions*". SN COMPUT. SCI. 2, 420 (2021).

[19] S.A. Khaparde, P.B. Kale and S.H. Agarwal, "*Application of artificial neural network in protective relaying of transmission lines*", Proceedings of the First International Forum on Applications of Neural Networks to Power Systems, 1991.

[20] L. Caroprese, P. Veltri, E. Vocaturo & E. Zumpano, "*Deep Learning Techniques for Electronic Health Record Analysis*", International Conference on Information, Intelligence, Systems and Applications (IISA), 2018.

[21] Á. Marín V., "*Localización de fuentes sonoras en tiempo real, mediantes redes neuronales, y agrupaciones de micrófonos*", Trabajo Fin de Grado, 2020, Escuela de Ingeniería y Arquitectura, Universidad de Zaragoza.

[22] Diego Calvo. *Red Neuronal Convolucional CNN*. https://www.diegocalvo.es/
red-neuronal-convolucional/, July 2017. Accedido: 01 de diciembre 2023.

[23] M. Pak and S. Kim, "*A review of deep learning in image recognition*," 2017 4th International Conference on Computer Applications and Information Processing Technology (CAIPT), Kuta Bali, Indonesia, 2017, pp. 1-3, doi: 10.1109/CAIPT.2017.8320684.

[24] R. Boada, R. Borkowski, and I. T. Monroy, "Clustering algorithms for Stokes space modulation format recognition," Opt. Express, vol. 23, no. 12, pp. 15 521–15 531, Jun 2015, [doi:10.1364/OE.23.015521].

[25] F. N. Khan, K. Zhong, W. H. Al-Arashi, C. Yu, C. Lu, and A. P. T. Lau, "Modulation format identification in coherent receivers using deep machine learning," IEEE Photonics Technology Letters, vol. 28, no. 17, pp. 1886–1889, Sep. 2016, [doi:10.1109/LPT.2016.2574800].

[26] Q. Zhang, H. Zhou, Y. Jiang, B. Cao, Y. Li, Y. Song, J. Chen, J. Zhang, and M. Wang, "A simple joint modulation format identification and OSNR monitoring scheme for IMDD OOFDM transceivers using K-nearest neighbor algorithm," Applied Sciences, vol. 9, no. 18, p. 3892, 2019, [doi:10.3390/app9183892]

[27] A. D, Ellis, N.M, Suibhne, D. Saad and D.N, Payne. "*Communication networks beyond the capacity crunch*". Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 374(2062). 2016.

[28] A. Sano, T. Kobayashi, S. Yamanaka, A. Matsuura, H. Kawakami, Y. Miyamoto, K. Ishihara, H. Masuda, *102.3-Tb/s (224 × 548-Gb/s) C- and extended L-band all-Raman transmission over 240 km using PDM-64QAM single carrier FDM with digital pilot tone,* in: Proc. OFC/NFOEC, Los Angeles, USA, 2012, Postdeadline paper PDP5C.3.

[29] D. Qian, M.-F. Huang, E. Ip, Y.-K. Huang, Y. Shao, J. Hu, T. Wang, *101.7-Tb/s (370 × 294-Gb/s) PDM-128QAM-OFDM transmission over 3 × 55-km SSMF using pilot-based phase noise mitigation,* in: Proc. OFC/NFOEC, Los Angeles, USA, 2011, Postdeadline paper PDPB5.

[30] R.-J. Essiambre, G. Kramer, P.J. Winzer, G.J. Foschini, B. Goebel, *"Capacity limits of optical fiber networks"*, IEEE J. Lightwv. Technol., 28 (4)

(2010), pp. 662-701.

[31] Google developer, https://developers.google.com/?hl=es-419, accedido 10 de mayo 2024.

[32] Á. Serrahima de Bedoya. "*Avances y desafíos de la inteligencia artificial*". Escuela técnica superior de ingeniería. Madrid. pp. 75. 2022.

[33] M. Haenlein, and A. Kaplan. "*A Brief History of Artificial Intelligence: On the Past, Present, and Future of Artificial Intelligence*". California Management Review. 2019.

[34] Z. He, "*Deep Learning in Image Classification: A Survey Report*," 2020 2nd International Conference on Information Technology and Computer Application (ITCA), Guangzhou, China, 2020, pp. 174-177, doi: 10.1109/ITCA52113.2020.00043.

[35] Nieto P., A. J. & Carriel S., R. P. (2020). "*La fibra óptica como medio para el desarrollo de las telecomunicaciones en Ecuador*". E-IDEA Journal of Engineering Science, 2(5), 12-29.

[36] C. Gómez G., "*Capacity crunch: Análisis de la capacidad de canal en Redes Ópticas y evaluación de posibles soluciones*", Trabajo Fin de Grado, Grado en Ingeniería de las Tecnologías de Telecomunicación, 2020, Departamento de Ingeniería Electrónica, Escuela Técnica Superior de Ingeniería, Universidad de Sevilla.

[37] Centro de formación técnica para la industra, *"Redes de Fibra Óptica: todo lo que necesitas saber",* https://www.cursosaula21.com/que-son-las-redes-de-fibra-optica/#:~:text=Los%20dos%20tipos%20principales%20de,monomodo%20y%20la%20fibra%20multimodo.&text=La%20fibra%20monomodo%20se%20utiliza,la%20intensidad%20de%20la%20señal., accedido el 10 de mayo de 2024.

[38] M. J. O'Mahony, C. Politi, D. Klonidis, R. Nejabati, and D. Simeonidou, *"Future optical networks,"* J. Lightw. Technol., vol. 24, no. 12, pp. 4684–4696, Dec. 9, 2006.

[39] Lach, E., & Idler, W. "*Modulation formats for 100 G and beyond*.", 2012.

[40] C. Häger and H. D. Pfister, *"Deep learning of the nonlinear Schrödinger equation in fiber-optic communications,"* in Proc. IEEE Int. Symp. Inf. Theory (ISIT), 2018, pp. 1590–1594.

[41] K. Roberts and C. Laperle, "*Flexible transceivers*," in Proc. Eur. Conf. Exhibit. Opt. Commun., 2012, pp. 1–3.

[42] A. Alvarado, T. Fehenberger, B. Chen and F. M. J. Willems, "Achievable Information Rates for Fiber Optics: Applications and Computations," in Journal of Lightwave Technology, vol. 36, no. 2, pp. 424-439, 15 Jan.15, 2018, doi: 10.1109/JLT.2017.2786351.

[43] C. E. Shannon, "A mathematical theory of communications," *Bell Syst. Tech. J.*, vol. 27, Jul. 1948.

[44] Y. Christiansen, "*Simulación de redes ópticas elásticas en multibanda*", Memoria de Título, 2023, Departamento de Ingeniería Eléctrica, Facultad de Ingeniería, Universidad de Concepción.

[45] «Python» [En línea] Available: https://docs.python.org. Accedido el: 10/05/2024.

[46] LinkedIn. [En línea]. *How do you measure a communication system?*. Accedido el 10 de noviembre de 2024, de https://es.linkedin.com/advice/3/how-do-you-measure-communication-system?lang=es

[47] StudySmarter. [En línea]. *Modulación óptica: Principios & Teoría*. StudySmarter. Accedido el 05 de noviembre de 2024, de https://www.studysmarter.es/resumenes/ingenieria/ingenieria-de-telecomunicaciones-ingenieria/modulacion-optica/

[48] Algor Education. [En línea]. *Fundamentos de modulación en comunicaciones*. Algor Education. Accedido el 10 de noviembre de 2024, de https://cards.algoreducation.com/es/content/8DvSzann/fundamentos-modulacion-comunicaciones

[49] Transistores.info. [En línea]. *Modulación: Tipos y aplicaciones en comunicaciones*. Accedido el 10 de noviembre de 2024, de https://transistores.info/guia-completa-modulacion-tipos-aplicaciones-en-comunicaciones/

[50] Altabas, Jose & Arribas, Paula & Sotelo, Felix & Izquierdo, David & Lazaro, Jose & Garcés, Ignacio. (2015). Análisis de Formatos Coherentes de Modulación Avanzados para Redes Ópticas Metropolitanas y de Acceso. 3. 37-38. 10.26754/jji-i3a.201501589.

[51] Heide, Sjoerd & Luis, Ruben & Goossens, Sebastiaan & Puttnam, Benjamin & Rademacher, Georg & Koonen, Ton & Shinada, Satoshi & Awaji, Yoshinari & Alvarado, Alex & Furukawa, Hideaki & Okonkwo, Chigo. (2022). Real-time transmission of geometrically-shaped signals using a software-defined GPU-based optical receiver. Optics Express. 30. 10.1364/OE.450514.

[52] Lepelaars, C. (2024). *The evolution of mobile CNN architectures* (Report ID: 204484). Weights & Biases. https://wandb.ai/carlolepelaars/mobile_architectures/reports/The-Evolution-Of-Mobile-CNN-Architectures--VmlldzoyMDQ0ODQ#final-thoughts

[53] Kundu, N. (2020, May 11). *Exploring ResNet50: An in-depth look at the model architecture and code implementation*. Medium. https://medium.com/@nitishkundu1993/exploring-resnet50-an-in-depth-look-at-the-model-architecture-and-code-implementation-d8d8fa67e46f

[54] Elaboración propia.

[55] Dataiteam. [En línea] *Convolutional Neural Network (CNN) Tutorial* [Código fuente]. Kaggle. Accedido el 10 de octubre de 2024 en https://www.kaggle.com/code/kanncaa1/convolutional-neural-network-cnn-tutorial

[56] Pourmoradi, N. [En línea] *Fruits and Vegetables Image - MobileNetV2*. Kaggle. Accedido el 11 de octubre de 2024 en https://www.kaggle.com/code/nimapourmoradi/fruits-and-vegetables-image-mobilenetv2

[57] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

[58] Scherer, D., Müller, A., & Behnke, S. (2010). *Evaluation of pooling operations in convolutional architectures for object recognition*. International Conference on Artificial Neural Networks.

[59] Sibi, P., Jones, S. A., & Siddarth, P. (2013). Analysis of different activation functions using back propagation neural networks. Journal of Theoretical and Applied Information Technology.

[60] Bin Chen, Zhiwei Liang and Alex Alvarado, "Binary Labeling for 2D and 4D constellations", https://github.com/TUe-ICTLab/Binary-Labeling-for-2D-and-4D-constellations

10. Anexos.

10.1 Código utilizado.

Én esta sección se muestra el código utilizado.

10.1.1 CNN personalizada

```
import os
import cv2
import random
import numpy as np
import pandas as pd
import seaborn as sns
import tensorflow as tf
import matplotlib.pyplot as plt
from tensorflow.keras import models, layers
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint, ReduceLROnPlateau
from tensorflow.keras.utils import plot_model
from sklearn.metrics import confusion_matrix, classification_report, ConfusionMatrixDisplay
from google.colab import drive

# Configuración de estilo para gráficos
sns.set_style('darkgrid')

# Montar Google Drive
drive.mount('/content/drive')
base_dir = '/content/drive/My Drive/MDT'

# Preparar generadores de datos con ImageDataGenerator
train_datagen = ImageDataGenerator(
    rescale=1./255,
    fill_mode='nearest',
    validation_split=0.2
```

```
)

train_generator = train_datagen.flow_from_directory(
    base_dir,
    target_size=(224, 224),
    color_mode='rgb',
    class_mode='categorical'
    batch_size=32,
    shuffle=True,
    seed=42,
    subset='training'
)

data_gen_pruebas = train_datagen.flow_from_directory(
    base_dir,
    target_size=(224, 224),
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32,
    shuffle=False,
    seed=42,
    subset='validation'
)

# Definir el modelo CNN
model = models.Sequential([
    layers.Conv2D(32, (3, 3), activation='relu', input_shape=(224, 224, 3)),
    layers.MaxPooling2D(2, 2),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D(2, 2),
    layers.Conv2D(128, (3, 3), activation='relu'),
    layers.MaxPooling2D(2, 2),
```

```
    layers.Dropout(0.5),
    layers.Flatten(),
    layers.Dense(100, activation='relu'),
    layers.Dense(train_generator.num_classes, activation="softmax")
])

# Compilación del modelo
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])

# Resumen del modelo
model.summary()

checkpoint_cb = ModelCheckpoint('MyModelmdtfinal.keras', save_best_only=True)
earlystop_cb = EarlyStopping(patience=10, restore_best_weights=True)
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=3, min_lr=1e-6)

# Entrenamiento del modelo
history = model.fit(
    train_generator,
    epochs=5,
    validation_data=data_gen_pruebas,
    callbacks=[checkpoint_cb, earlystop_cb, reduce_lr]
)

# Graficar accuracy y loss
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'], label='Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.title('Model Accuracy')
```

```
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()

plt.subplot(1, 2, 2)
plt.plot(history.history['loss'], label='Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title('Model Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()

plt.tight_layout()
plt.show()

# Evaluación del modelo
loss, accuracy = model.evaluate(data_gen_pruebas)
print(f'Accuracy: {accuracy:.2f}')
print(f'Loss: {loss:.2f}')

# Predicciones
data_gen_pruebas.reset()
predictions = model.predict(data_gen_pruebas)
predicted_classes = np.argmax(predictions, axis=1)

# Etiquetas verdaderas y las predicciones
true_classes = data_gen_pruebas.classes
class_labels = list(data_gen_pruebas.class_indices.keys())

# Matriz de confusión
conf_matrix = confusion_matrix(true_classes, predicted_classes)
disp = ConfusionMatrixDisplay(confusion_matrix=conf_matrix, display_labels=class_labels)
```

```
disp.plot(cmap=plt.cm.Blues)
plt.title('Matriz de Confusión')
plt.show()

# Informe de clasificación
report = classification_report(true_classes, predicted_classes, target_names=class_labels)
print("Informe de Clasificación:")
print(report)

# Etiquetas
labels = class_labels

plt.figure(figsize=(10, 8))
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap="Blues", xticklabels=labels,
yticklabels=labels)
plt.xlabel("Predicted label")
plt.ylabel("True label")
plt.title("Matriz de Confusión")

plt.xticks(rotation=45, ha="right", rotation_mode="anchor")
plt.yticks(rotation=45, ha="right", rotation_mode="anchor")
plt.tight_layout()
plt.show()
```

10.1.2 MobileNet

```
import os
import cv2
import numpy as np
import pandas as pd
import seaborn as sns
import tensorflow as tf
import matplotlib.pyplot as plt
```

```
from sklearn.metrics import confusion_matrix, classification_report, ConfusionMatrixDisplay
from tensorflow.keras import models, layers
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint, ReduceLROnPlateau
from tensorflow.keras.utils import plot_model
from termcolor import colored

# Configuración de visualización y estilo
sns.set_style('darkgrid')

from google.colab import drive
drive.mount('/content/drive')

base_dir = '/content/drive/My Drive/MDT'

# Configuración del generador de imágenes para entrenamiento y validación
train_datagen = ImageDataGenerator(
    rescale=1./255,
    validation_split=0.2  # División de datos (80% entrenamiento, 20% validación)
)

# Generador de datos para entrenamiento
train_generator = train_datagen.flow_from_directory(
    base_dir,
    target_size=(224, 224),
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32,
    shuffle=True,
    seed=42,
    subset='training'
```

```
)

# Generador de datos para validación
data_gen_pruebas = train_datagen.flow_from_directory(
    base_dir,
    target_size=(224, 224),
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32,
    shuffle=False,
    seed=42,
    subset='validation'
)

# Cargar el modelo MobileNetV2 preentrenado (sin las capas finales)
base_model = MobileNetV2(weights='imagenet', include_top=False, input_shape=(224, 224, 3))

# Congelar el modelo base para no entrenar sus pesos
base_model.trainable = False

# Construir el modelo con nuestras propias capas encima de MobileNetV2
model = models.Sequential([
    base_model,
    layers.GlobalAveragePooling2D(),
    layers.Dense(1024, activation='relu'),
    layers.Dropout(0.5),
    layers.Dense(train_generator.num_classes, activation='softmax')  # Capa de salida
])

# Compilar el modelo
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

```
# Callbacks
checkpoint_cb = ModelCheckpoint('MobileNetV2_best_model.keras', save_best_only=True)
earlystop_cb = EarlyStopping(patience=10, restore_best_weights=True)
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=3, min_lr=1e-6)

# Entrenamiento del modelo
history = model.fit(
    train_generator,
    epochs=5,
    validation_data=data_gen_pruebas,
    callbacks=[checkpoint_cb, earlystop_cb, reduce_lr]
)

# Graficar accuracy y loss
plt.plot(history.history['accuracy'], label='Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.title('Model Accuracy MobileNet')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend()
plt.show()

plt.plot(history.history['loss'], label='Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title('Model Loss MobileNet')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend()
plt.show()

# Evaluación del modelo
```

```
loss, accuracy = model.evaluate(data_gen_pruebas)
print(f'Accuracy: {accuracy:.2f}')
print(f'Loss: {loss:.2f}')

# Predicciones
data_gen_pruebas.reset()  # Resetea el generador
predictions = model.predict(data_gen_pruebas)
predicted_classes = np.argmax(predictions, axis=1)

# Obtener las etiquetas verdaderas
true_classes = data_gen_pruebas.classes
class_labels = list(data_gen_pruebas.class_indices.keys())

# Matriz de confusión
conf_matrix = confusion_matrix(true_classes, predicted_classes)
disp = ConfusionMatrixDisplay(confusion_matrix=conf_matrix, display_labels=class_labels)
disp.plot(cmap=plt.cm.Blues)
plt.title('Matriz de Confusión MobileNet')
plt.show()

# Informe de clasificación
report = classification_report(true_classes, predicted_classes, target_names=class_labels)
print("Informe de Clasificación:")
print(report)

# Graficar la matriz de confusión
plt.figure(figsize=(10, 8))
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap="Blues", xticklabels=class_labels,
yticklabels=class_labels)
plt.xlabel("Predicted label")
plt.ylabel("True label")
plt.title("Matriz de Confusión MobileNet")
```

```
plt.xticks(rotation=45, ha="right")
plt.yticks(rotation=45, ha="right")
plt.tight_layout()
plt.show()
```

10.1.3 ResNet

```
import os
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import classification_report, confusion_matrix, ConfusionMatrixDisplay
from tensorflow.keras import models, layers
from tensorflow.keras.applications import ResNet50
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint, ReduceLROnPlateau
from tensorflow.keras.optimizers import Adam
from google.colab import drive

# Montar Google Drive
drive.mount('/content/drive')

# Directorios
base_dir = '/content/drive/My Drive/MDT'

# 1. Configuración de ImageDataGenerator para las imágenes de entrenamiento y validación
train_datagen = ImageDataGenerator(rescale=1./255, fill_mode='nearest', validation_split=0.2)

train_generator = train_datagen.flow_from_directory(
    base_dir,
    target_size=(224, 224),
```

```
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32,
    shuffle=True,
    seed=42,
    subset='training'
)

data_gen_pruebas = train_datagen.flow_from_directory(
    base_dir,
    target_size=(224, 224),
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32,
    shuffle=False,
    seed=42,
    subset='validation'
)

# 2. Cargar el modelo preentrenado ResNet50 sin las capas finales
base_model = ResNet50(weights='imagenet', include_top=False, input_shape=(224, 224, 3))

# 3. Congelar el modelo base para no actualizar sus pesos
base_model.trainable = False

# 4. Construir el modelo con nuevas capas encima de ResNet50
model = models.Sequential([
    base_model,  # ResNet50 preentrenado
    layers.GlobalAveragePooling2D(),  # Capa de pooling global
    layers.Dense(1024, activation='relu'),  # Capa densa
    layers.Dropout(0.5),  # Regularización con Dropout
    layers.Dense(train_generator.num_classes, activation='softmax')  # Capa de salida
```

```
])

# 5. Compilar el modelo
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])

# 6. Definir callbacks
checkpoint_cb = ModelCheckpoint('ResNet50_best_model.keras', save_best_only=True)
earlystop_cb = EarlyStopping(patience=10, restore_best_weights=True)
reduce_lr_cb = ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=3, min_lr=1e-6)

# 7. Entrenar el modelo
history = model.fit(
    train_generator,
    epochs=5,
    validation_data=data_gen_pruebas,
    callbacks=[checkpoint_cb, earlystop_cb, reduce_lr_cb]
)

# 8. Graficar precisión y pérdida
plt.figure(figsize=(12, 5))

# Precisión
plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'], label='Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.title('Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend()

# Pérdida
plt.subplot(1, 2, 2)
```

```
plt.plot(history.history['loss'], label='Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title('Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend()

plt.tight_layout()
plt.show()

# 9. Evaluar el modelo
loss, accuracy = model.evaluate(data_gen_pruebas)
print(f'Accuracy: {accuracy:.2f}')
print(f'Loss: {loss:.2f}')

# 10. Calcular la matriz de confusión y el reporte de clasificación
data_gen_pruebas.reset()
predictions = model.predict(data_gen_pruebas)
predicted_classes = np.argmax(predictions, axis=1)

true_classes = data_gen_pruebas.classes
class_labels = list(data_gen_pruebas.class_indices.keys())

# Matriz de confusión
cm = confusion_matrix(true_classes, predicted_classes)
print("Matriz de Confusión:")
print(cm)

# Reporte de clasificación
report = classification_report(true_classes, predicted_classes)
print("Informe de Clasificación:")
print(report)
```

```
Graficar la matriz de confusión
plt.figure(figsize=(10, 8))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=class_labels,
yticklabels=class_labels)

#Ejes
plt.xlabel("Predicted label")
plt.ylabel("True label")
plt.title("Matriz de Confusión")

plt.xticks(rotation=45, ha="right", rotation_mode="anchor")
plt.yticks(rotation=45, ha="right", rotation_mode="anchor")

plt.tight_layout()
plt.show()
```

11. Resumen FI.

UNIVERSIDAD DE CONCEPCION – FACULTAD DE INGENIERIA

RESUMEN DE MEMORIA DE TITULO

Departamento **:** Departamento de Ingeniería …

Carrera **:** Ingeniería Civil …

Nombre del memorista **:**

Título de la memoria **:**

Fecha de la presentación oral **:**

Profesor(es) Guía **:**

Profesor(es) Revisor(es) **:**

Concepto **:**

Calificación **:**

Resumen (máximo 200 palabras)